THE NEW SOFTWARE PACKAGE FOR DYNAMIC HIERARCHICAL CLUSTERING FOR CIRCLES TYPES OF SHAPES

Tetyana Shatovska, Tetiana Safonova, Iurii Tarasov

Abstract: In data mining, efforts have focused on finding methods for efficient and effective cluster analysis in large databases. Active themes of research focus on the scalability of clustering methods, the effectiveness of methods for clustering complex shapes and types of data, high-dimensional clustering techniques, and methods for clustering mixed numerical and categorical data in large databases. One of the most accuracy approach based on dynamic modeling of cluster similarity is called Chameleon. In this paper we present a modified hierarchical clustering algorithm that used the main idea of Chameleon and the effectiveness of suggested approach will be demonstrated by the experimental results.

Keywords: Chameleon, clustering, hypergraph partitioning, coarsening hypergraph.

ACM Classification Keywords F.2.1 Numerical Algorithms and Problems

Introduction

The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering. A cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters. A cluster of data objects can be treated collectively as one group in many applications. Data clustering is under vigorous development. Contributing areas of research include data mining, statistics, machine learning, spatial database technology, biology, and marketing. Owing to the huge amounts of data collected in databases, cluster analysis has recently become a highly active topic in data mining research. As a branch of statistics, cluster analysis has been studied extensively for many years, focusing mainly on distance-based cluster analysis. Active themes of research focus on the scalability of clustering methods, the effectiveness of methods for clustering complex shapes and types of data. Chameleon is a clustering algorithm that explores dynamic modeling in hierarchical clustering. In its clustering process, two clusters are merged if the interconnectivity and closeness between two clusters are highly related to the internal interconnectivity and closeness of objects within the clusters. The merge process based on the dynamic model facilitates the discovery of natural and homogeneous clusters and applies to all types of data as long as a similarity function is specified. Chameleon is derived based on the observation of the weakness of two hierarchical clustering algorithms: CURE and ROCK. CURE and related schemes ignore information about the aggregate interconnectivity of objects in two different clusters, whereas ROCK and related schemes ignore information about the closeness of two clusters while emphasizing their interconnectivity. In this paper, we present our experiments with hierarchical clustering algorithm CHAMELEON for circles cluster shapes with different densities using hMETIS program that used multilevel k-way partitioning for hypergraphs and a Clustering Toolkit package that merges clusters based on a dynamic model. In CHAMELEON two clusters are merged only if the inter-connectivity and closeness between two clusters are comparable to the internal inter-connectivity of the clusters and closeness of items within the clusters. The methodology of dynamic modeling of clusters is applicable to all types of data as long as a similarity matrix can be constructed. We present a modified hierarchical clustering algorithm that measures the similarity of two clusters based on a new dynamic model with different shapes and densities. The merging process using the dynamic model presented in this paper facilitates discovery of natural and homogeneous not only circles cluster shapes.

1 Related work

In this section, we give a brief description of existing clustering algorithms.

A hierarchical method creates a hierarchical decomposition of the given set of data objects. A hierarchical method can be classified as being either agglomerative or divisive, based on how the hierarchical decomposition is formed. The agglomerative approach, also called the bottom-up approach, starts with each object forming a separate group. It successively merges the objects or groups close to one another, until all of the groups are merged into one, or until a termination condition holds. The divisive approach, also called the top-down approach, starts with all the objects in the same cluster. In each successive iteration, a cluster is spitted up into smaller clusters, until eventually each object is in one cluster, or until a termination condition holds.

Hierarchical methods suffer from the fact that once a step is done, it can never be undone. This rigidity is useful in that it leads to smaller computation costs by not worrying about a combinatorial number of different choices. However, a major problem of such techniques is that they cannot correct erroneous decisions. There are two approaches to improving the quality of hierarchical clustering: (1) perform careful analysis of object "linkages" at each hierarchical partitioning, such as in CURE and Chameleon, or (2) integrate hierarchical agglomeration and iterative relocation by first using a hierarchical agglomerative algorithm and then refining the result using iterative relocation, as in BIRCH [Zhang, 1996].

Most clustering algorithms either favor clusters with spherical shape and similar sizes, or are fragile in the presence of outliers. CURE overcomes the problem of favoring clusters with spherical shape and similar sizes and is more robust with respect to outliers. CURE employs a novel hierarchical clustering algorithm that adopts a middle ground between centroid-based and representative-object-based approaches. Instead of using a single centroid or object to represent a cluster, a fixed number of representative points in space are chosen. The representative points of a cluster are generated by first selecting well-scattered objects for the cluster and then "shrinking" or moving them toward the cluster center by a specified fraction, or shrinking factor. At each step of the algorithm, the two clusters with the closest pair of representative points (where each point in the pair is from a different cluster) are merged. ROCK is an alternative agglomerative hierarchical clustering algorithm that is suited for clustering categorical attributes. It measures the similarity of two clusters by comparing the aggregate interconnectivity of two clusters against a user-specified static interconnectivity model, where the interconnectivity of two clusters is defined by the number of cross links between the two clusters, and link is the number of common neighbors between two points. In other words, cluster similarity is based on the number of points from different clusters who have neighbors in common [Guha, 1999].

ROCK first constructs a sparse graph from a given data similarity matrix using a similarity threshold and the concept of shared neighbors. It then performs a hierarchical clustering algorithm on the sparse graph.

There are two major limitations of the agglomerative mechanisms used in existing schemes. First, these schemes do not make use of information about the nature of individual clusters being merged. Second, one set of schemes (CURE and related schemes) ignore the information about the aggregate interconnectivity of items in two clusters, whereas the other set of schemes ignore information about the closeness of two clusters as defined by the similarity of the closest items across two clusters.

2 Overview of CHAMELEON: Clustering Using Dynamic Modeling

Chameleon is a clustering algorithm that explores dynamic modeling in hierarchical clustering [Karypis, 1999a]. Chameleon represents its objects based on the commonly used k-nearest neighbor graph approach. This graph representation of the data set allows CHAMELEON to scale to large data sets. Each vertex of the k-nearest neighbor graph represents a data object, and there exists an edge between two objects if one object is among the k-most similar objects of the other. The k-nearest neighbor graph captures the concept that neighborhood radius of an object is determined by the density of the region in which this object resides [Mitchell, 1997].

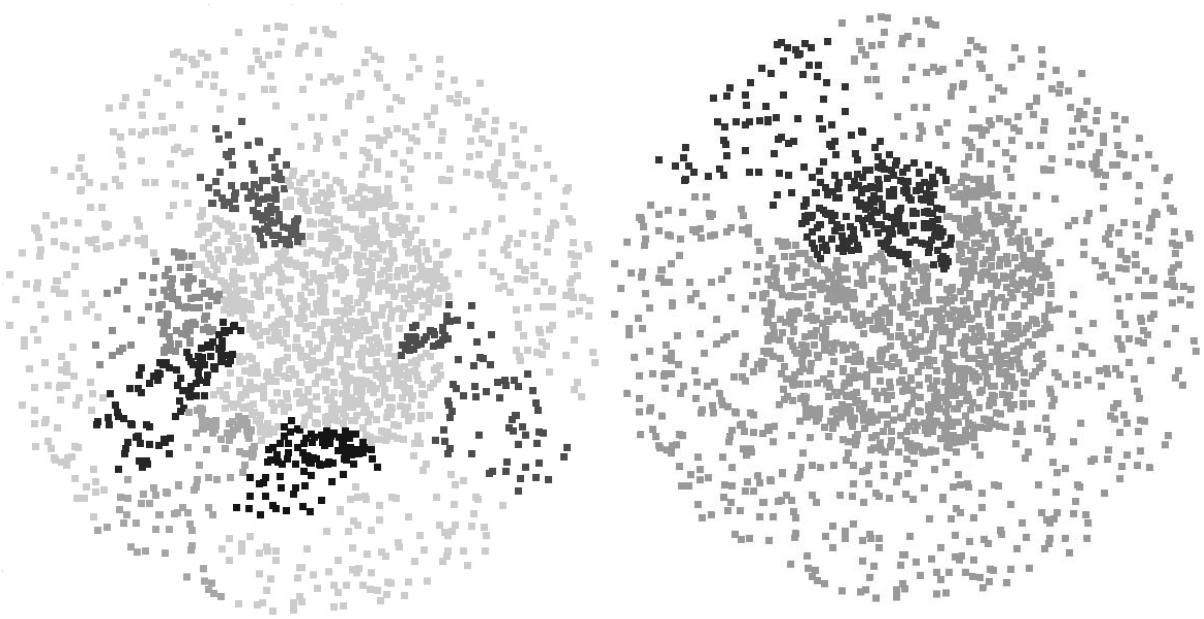
During the next step a sequence of successively smaller hypergraphs are constructed – Coarsening Phase. Two primary schemes have been developed for selecting what groups of vertices will be merged together to form single vertices in the next level coarse hypergraphs. The first scheme called edge-coarsening (EC) [Alpert, 1997] selects the groups by funding a maximal set of pairs of vertices (i.e., matching) that belong in many hyperedges. The second scheme that is called hyperedge-coarsening (HEC) [Karypis, 1997] finds a maximal independent set of hyperedges, and the sets of vertices that belong to each hyperedge becomes a group of vertices to be merged together. At each coarsening level, the coarsening scheme stop as soon as the size of the resulting coarse graph has been reduced by a factor of 1.7 [Karypis, 1999b]. The third phase of the algorithm is to compute a k-way partitioning of the coarsest hypergraph such that the balancing constraint is satisfied and the partitioning function as mincut is optimized. During the uncoarsening phase, a partitioning of the coarser hypergraph is projected to the next level finer hypergraph, and a partitioning refinement algorithm is used to optimize the objective function without violating the partitioning balancing constraints. At the final iteration of algorithm CHAMELEON determines the similarity between each pair of clusters by taking into account both at their relative inter-connectivity and their relative closeness. It selects to merge clusters that are well inter-connected as well as close together with respect to the internal inter-connectivity and closeness of the clusters. By selecting clusters based on both of these criteria, CHAMELEON overcomes the limitations of existing algorithms that look either at the absolute inter-connectivity or absolute closeness.

3 Performance Analysis

The overall computational complexity of CHAMELEON depends on the amount of time it requires to construct the K – nearest neighbors graph and the amount of time it requires to perform the two phases of the clustering algorithm. In [Karypis, 1999a] was shown that CHAMELEON is not very sensitive of values k for computing the k-nearest neighbor graph, of the value of MINSIZE for the phase I of the algorithm, and of scheme for combining relative inter-connectivity and relative closeness and associated parameters, and it was able to discover the correct clusters for all of these combinations of values for k and MINSIZE. In this section, we present experimental evaluation of clustering using hMETIS hypergraph partitioning package for k-way partitioning of hypergraph and for recursive bisection [Karypis, 1998] and CLUTO 2.1.1– A Clustering Toolkit [Karypis, 2003].

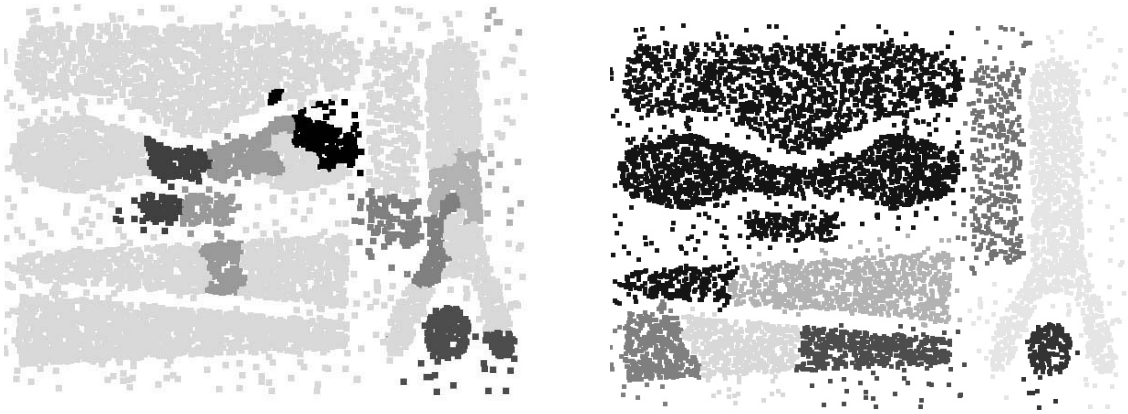
We experimented with five different data sets containing points in two dimensions: “disk in disk”, t4.8k, t5.8k, t8.8k, t7.10k [Karypis lab.].The first data set, has a particularly challenging feature that two clusters are very close to each other and they have different densities and circles shapes. We choose the number of neighbors k=5, 15, 40, MINSIZE = 5%. Looking at Fig. 1, a) we can see the results of the k-way partitioning of hypergraph by hMETIS package [Karypis lab.] and b) merging process by CLUTO package [Karypis lab.] with k=5 nearest neighbors. Looking at Fig.1 we can see that in both cases we have not correctly identified the genuine clusters.

The data set t8.8k has eight clusters of different shapes, size and orientation, some of which are inside the space enclosed by other clusters. Moreover, it also contains random noise such as a collection of points forming vertical streaks. Looking at Fig. 2 with k=5 nearest neighbors we can see that hMETIS also compute k-way partitioning of hypergraph with mistakes closer to the border of two classes and CLUTO can not effectively merge clusters for such type of dataset using asymmetric k-NN, with k=5. It means that algorithm of the partitioning phase is very sensitive to the value of k for spherical shapes of clusters and to the types of k-NN graph (symmetric and asymmetric). It is very important to choose an optimal value of k, because with k=16 and more, and only for symmetric k-NN with weights of edges equal to the number of common neighbors we obtain final clustering with minimum percentages of errors.



a) k-way partitioning by hMETIS b) final clusters by CLUTO

Fig. 1 Data set “disk in disk” with k=5 nearest neighbors and asymmetric k-NN



a) k-way partitioning by hMETIS b) final clusters by CLUTO

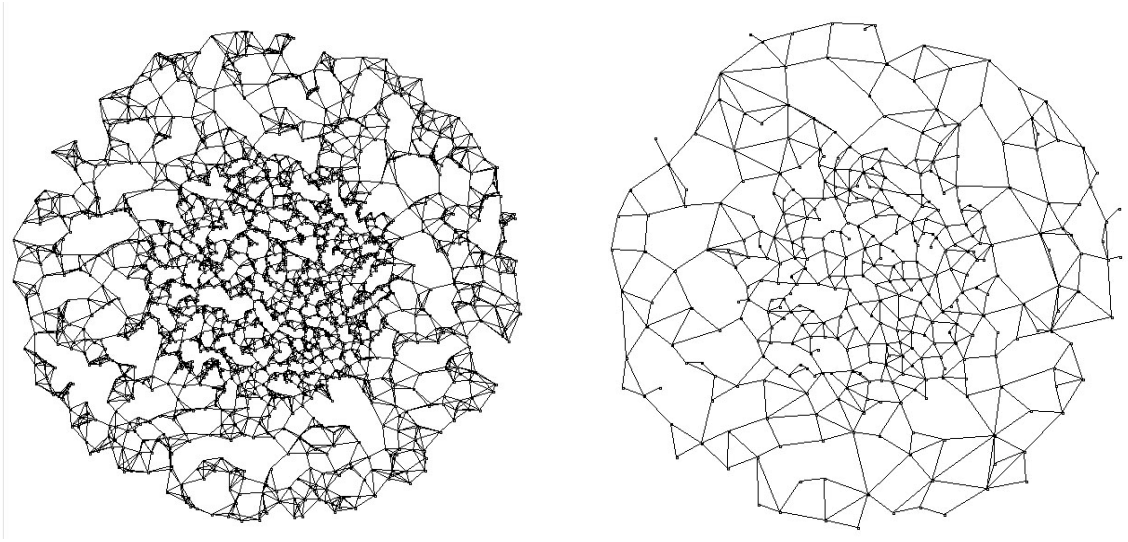
Fig. 2 Data set “t8.8k” with k=5 nearest neighbors and asymmetric k-NN

4 Modeling the cluster similarity

As we remark above the CHAMELEON operates on a sparse graph in which nodes represent data items and weighted edges represent similarities among the data item (symmetric graph) [Karypis, 1999a]. In our algorithm during first phase we construct an asymmetric k-NN graph and there exists an edge between two points if for one of it there exist closest neighbor among all existing neighbors according to the value of k. Note that the weight of an edge connecting two objects in the k-NN graph is a similarity measure between them, as usual a simple distance measure (or inversely related to their distance).

In our algorithm the weight of an edge we compute as weighted distance between objects. Fig. 3 represents the k-NN graph for data set “disk in disk” with k=5. During coarsening phase the set of smaller hypergraphs is constructed. In the first stage of coarsening process we choose the set of vertices with maximum degrees and matched it with a random neighbour. On the other stages we visit each vertex in a random order and matched it with adjacent vertex via heaviest edge. Note that usually the weight of an edge connecting two nodes in a

coarsened version of the graph is the number of edges in the original graph that connect the two sets of original nodes collapsed into the two coarse nodes. In our case we compute the weight of the hyperedge as the sum of the weights of all edges that collapse on each other during coarsening step. We stop the coarsening process at each level as soon as the number of multivertices of the resulting coarse hypergraph has been reduced by a constant less than 2 (Fig. 3).



a) k-NN graph with k=5 b) hypergraph after third level of coarsening

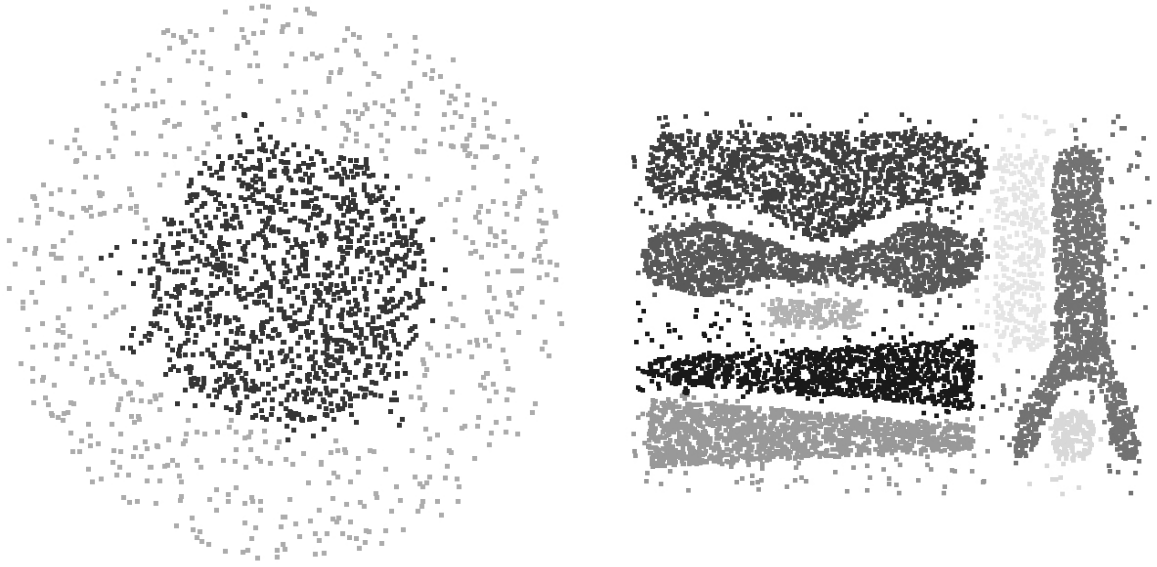
Fig. 3 Data set “disk in disk”

On the next level of algorithm we produce a set of small hypergraphs using k-way multilevel paradigm [Karypis, 1999b]. We start the process of partitioning by choosing k most heavier multivertices, where k = 8, 16, 32. After that we gathering one by one all neighbors from each chosen vertex and obtain the initial partitioning w.r.t. to the balancing constant. The problem of computing an optimal bisection of a hypergraph is NP-hard. One of the most commonly used objective function is to minimize the hyperedge-cut of the partitioning; i.e., the total number of hyperedges that span multiple partitions [Karypis, 1999b]. One of the most accuracy algorithm of partitioning the hypergraph is Kernighan-Lin / Fiduccia – Mattheyses algorithm, in which during each pass, the algorithm repeatedly finds a pair of vertices, one from each of the subdomains, and swaps their subdomains. The pairs are selected so as to give the maximum improvement in the quality of the partitioning even if this improvement is negative. Once a pair of vertices has been moved, neither are considered for movement in the rest of the pass. When all of the vertices have been moved, the pass ends. At this point, the state of the bisection at which the minimum edge-cut was achieved is restored. In our experiments we use a greedy refinement algorithm developed by George Karypis [Karypis, 1999b], but as the gain function for each vertex we compute the differences between the sum of the weights of edges incident on vertex that go to the other partition and the sum of edges weights that stay within the partition. We choose the vertex with maximum positive gain and move it if it result in a positive gain, so we work only with boundary vertices.

After the partitioning of hypergraph into the large number of small parts we start to merge the pair of clusters for which both relative inter-connectivity and their relative closeness are high [Karypis, 1999a]. In our research we use George Karypis formula to compute the similarity between sub-clusters. Looking at the Fig.1 b) we can see that for data set “disk in disk” was obtained not correct clustering results. Thus we suggest to modified the above mentioned expression by change the relative inter-connectivity to a new expression that estimates the average weights of edges in each sub-graph and the number of edges that connect two partitions to the number of edges that stay within the smallest partition.

Looking at the Fig. 4 we can see the correct clustering results for the same data set “disk in disk” using our suggested expression. For another above mentioned data sets we obtain as well accuracy results. In all

experiment we use k=5 and in our approach the correctness of classification really doesn't depend of the value of k and of the k-NN type.



a) Data set "disk in disk" b) Data set "t8.8k.txt"

Fig 4. Clustering results using a new approach to the sub-clusters merging, k=5

Conclusion

In this paper, we present our experiments with hierarchical clustering algorithm CHAMELEON for circles cluster shapes with different densities using hMETIS program that used multilevel k-way partitioning for hypergraphs and the Clustering Toolkit package that merges clusters based on a dynamic model. In CHAMELEON two clusters are merged only if the inter-connectivity and closeness between two clusters are comparable to the internal inter-connectivity of the clusters and closeness of items within the clusters. The methodology of dynamic modeling of clusters is applicable to all types of data as long as a similarity matrix can be constructed.

Experimental results showed that hMETIS computes k-way partitioning of hypergraph with mistakes closer to the border of two classes and CLUTO can not effectively merge clusters using asymmetric k-NN, with k=5.

We present a modified hierarchical clustering algorithm that measures the similarity of two clusters based on a new dynamic model with different shapes and densities. The merging process using the dynamic model presented in this paper facilitates discovery of natural and homogeneous not only circles cluster shapes.

Experimental results showed that this method is not sensitive to the value of k and doesn't need a specific k-nearest neighbor graph creating.

Bibliography

[Alpert, 1997] C. J. Alpert, J. H. Huang and A. B. Kahng, Multilevel circuit partitioning. In: Proc. of the 34th ACM/IEEE Design Automation Conference. 1997.

[Guha, 1999] S. Guha, R. Rastogi, K. Shim ROCK: Robust Clustering using linKs, (ICDE'99).

[Karypis, 1997] G. Karypis , R. Aggarwal, V. Kumar and Sh. Shekhar, Multilevel hypergraph partitioning: Application in VLSI domain. In: Proceedings of the Design and Automation Conference. 1997.

[Karypis, 1998] G. Karypis, and V. Kumar , hMETIS 1.5.3: A hypergraph partitioning package. Technical report. Department of Computer Science, University of Minnesota, 1998

[Karypis, 1999a] G. Karypis, E.-H. Han, and V. Kumar. CHAMELEON: A Hierarchical Clustering Algorithms Using Dynamic Modeling. IEEE Computer, 32(8):68–75, 1999.

[Karypis, 1999b] G. Karypis and V. Kumar. Multilevel k-way hypergraph partitioning. In Proceedings of the Design and Automation Conference, 1999.

[Karypis, 2003] G. Karypis, CLUTO 2.1.1. A Clustering Toolkit. Technical report. Department of Computer Science, University of Minnesota, 2003

[Karypis lab.] http://www.cs.umn.edu/~karypis.

[Mitchell, 1997] T. M. Mitchell. Machine Learning. McGraw Hill, 1997

[Zhang, 1996] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH : an efficient data clustering method for very large databases. SIGMOD'96.

Authors' Information

Shatovska Tetyana – Kharkiv National University of Radio Electronics, Computer Science department, P.O.Box: Kharkiv-116, Lenin av. 14, Ukraine; e-mail: tanita_uk@mail.ru

Safonova Tetyina - Kharkiv National University of Radio Electronics, Computer Science department, P.O.Box: Kharkiv-116, Lenin av. 14, Ukraine; e-mail: safonovatm@mail.ru

Tarasov Iuril - Kharkiv National University of Radio Electronics, Computer Science department, P.O.Box: Kharkiv-116, Lenin av. 14, Ukraine; e-mail: ytarasovmail@rambler.ru