Acta Technica *62* No. 2B/2017, 41–52

Research on high-dimensional clustering algorithm based on information entropy change trend

Zhang Yunfeng[1], Fang Haoshuai[1]

Abstract. The rapid development of computers and network technology has caused the explosive growth of scale and dimension of data in the Internet. How to cluster, analysis and mine massive high-dimensional data automatically is a research hotspot in the area of information retrieval and data mining. In order to meet the need for high-dimensional data clustering algorithm at present preset parameters, the computational complexity is high, according to the information retrieval system in data distribution characteristics of proposed high-dimensional clustering algorithm based on edge distribution entropy and the change trend. The algorithm firstly, the samples are mapped to each dimension, according to the principle of edge differential entropy decrease monotonically separate hierarchical clustering, and then use the single dimensional clustering results to maximize the edge of differential entropy and the principle of sample data are divided according to the, to get the final clustering result.

Key words. Clustering algorithm, high dimension, information entropy, scale out, query suggestion, relevancy evaluation.

1. Introduction

With the development of computer and Internet technologies, the amount of social information has taken on an explosive growth tendency so information that users can obtain is dramatically increasing. In order to better screen, store, index, retrieve and assess the massive data, the information retrieval system came into being [1]. Clustering analysis and application of massive high dimensional data is a challenge in the present field of data mining. Researches on theories and methods suitable for high-dimensional clustering are of great significance to improvement of data mining theories and extension of clustering application [2].

[1]North China Institute of Aerospace Engineering Computer and Remote Sensing Information Technology Institute Hebei, Langfang, 065000, China

2. Clustering algorithm model based on changing tendency of information entropy

2.1. Relative definitions

The attribute set $A = \{A\_1, A_2, \cdots, A_d\}$ is composed of d continuous real number fields and S is a d-dimensional hyperspace $S = A\_1 \times A_2 \times \cdots \times A_d$. The input is a d-dimensional data point set $V = \{v_1, v_2, \cdots, v_n\}$, $v_j = \langle v_{j1}, v_{j2}, \cdots, v_{jd}\rangle$, where the ith component u_i is the value of A_i in S. The component values of different data points in each dimension can be the same. Suppose that u_i are different values of $v_{i1}, v_{i2}, \cdots, v_{iu_i}$ in the ith dimension, respectively. The lengths between values are, respectively, $l_{i1}, l_{i2}, \cdots, l_{iu_i}$, where $l_{ix} = v_{i(x+1)} - v_{ix}$. The number of data points $v_{i1}, v_{i2}, \cdots, v_{iu_i}$ is $k_{i1}, k_{i2}, \cdots, k_{iu_i}$. When $j = u_i$, it can be considered that $l_{i(j+1)} = \infty$, where

$$\sum_{x_1=1}^{u_1} k_{1x_1} = \cdots = \sum_{x_i=1}^{u_i} k_{ix_i} = \cdots = \sum_{x_d=1}^{u_d} k_{dx_d} = n\,. \tag{1}$$

When applying the single dimensional clustering algorithm to the data, each point will move in the positive or negative direction of the coordinate axis. We use s_{ij} to indicate the moving direction of v_{ij}: 1 for positive direction and –1 for negative direction. Symbol H_i indicates the edge differential entropy of the data in the ith dimension and $H_{i[v_{ij},\, v_{i(j+1)}]}$ indicates the edge differential entropy calculated within the interval $[v_{ij},\, v_{i(j+1)}]$ in the ith dimension.

2.2. Clustering algorithm

The clustering algorithm based on changing tendency of information entropy can be divided into three steps.

Step 1: Map the data sample to each dimension and sort the data in each dimension.

Step 2: Move the data points according to Theorem 1 in each dimension. When two points meet, merge them and recalculate k (the k value of the new point is the sum of the k values of the merged points), l and H_i and re-determine the moving direction s of each data point. Move in this way until only one point is left in each dimension. Thus we can obtain d-hierarchical cluster trees $CT = \{CT_1, CT_2, \cdots, CT_d\}$, as shown in Fig. 1.

In CT_i, the data point $V = \{v_1, v_2, \cdots, v_d\}$ is only located in the leaf node and the clustering node $C_i = \{c_{iLayer_yNoZ} \,|y \in [1, n]\,, z \in [1, n]\}$ is only located in the non-leaf node. All clustering nodes in the path from the root to each leaf node are the category of this leaf node. The leaf nodes with the same ancestor belong to the category indicated by this ancestor [3].

Each hierarchy of the hierarchical cluster tree indicates the merging process of one clustering. Only two clusters are merged in each clustering, so for each hierarchy except the bottom hierarchy, only one clustering node has two child nodes. Other

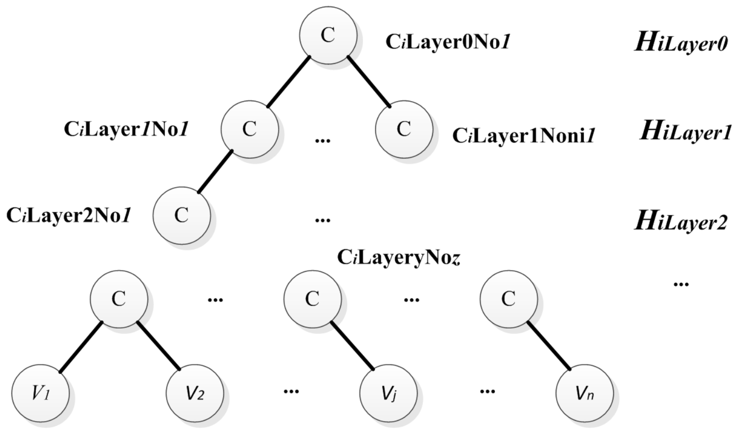

Fig. 1. Single dimensional hierarchical cluster tree

clusters with only one cluster merely change their positions rather than be merged with other clustering points during this clustering process [4].

Step 3: Each time, re-partition of the already-made data clustering results by choosing one from the d hierarchical cluster trees to maximize the increment of the sum of edge entropies. Each hierarchy of the hierarchical cluster tree indicates the partition method of a data sample. Partition is defined as follows.

Suppose the data sample has a partition $p_{i_1} = \{c_{i_1 1}, c_{i_1 2}, \cdots, c_{i_1 m}\}$ in the i_1th dimension and has another partition $p_{i_2} = \{c_{i_2 1}, c_{i_2 2}, \cdots, c_{i_2 n}\}$ in the i_2th dimension, where $i_1 \neq i_2$. Define $p_{i_1 i_2} = p_{i_1} \times p_{i_2}$ as the result by re-partitioning the p_{i_1} partition result with p_{i_2}. The calculation method is as follows:

$$p_{i_1 i_2} = p_{i_1} \times p_{i_2} =$$

$$= \left\{ c_z \middle| c_z = c_x \bigcap c_y,\ x \in [1, i_1 m],\ y \in [1, i_2 n],\ z \in [1, i_1 m \times i_2 n] \right\}. \qquad (2)$$

2.3. Time complexity and space complexity of the algorithm

The algorithm complexity can usually be classified into time complexity and space complexity. Time complexity can be expressed as CPU cost and I/O cost while space complexity can be expressed as memory cost [5]

First we will analyze CPU cost. This algorithm can be divided into three steps: initialization, single dimensional clustering and overall partition. The complexity of each step is shown in Table 1.

Secondly we will analyze I/O cost. The process of initialization requires reading-in of all data of the data points and then, respectively, writing-into d files indicating data of different dimensions. The cost of this process is $2nd$. The process of single dimensional clustering requires reading-in of single dimensional data and writing the merged result into the result file after each merging. This cost is n, so the general

I/O cost of d dimensional merging is nd [6]. Each partition requires reading-in of a clustering result so a maximum of $c-1$ clustering results are needed to produce c clusters. This cost is $c-1$. The I/O cost of the export is 1. Therefore, the general I/O cost of this algorithm is$3nd+c$.

Table 1. Algorithm complexity

Step	Operating complexity	Description
Data partition	$nd+dn\log n$	Partition the data points to the single dimensional data and sort them in ascending order.
Single dimensional clustering	$dn(n+1)$	Cluster in each dimension until one point is obtained.
Data set partition	$c(n\log n+n)$	Partition the data set to obtain c clusters.
General complexity	$dn^2+(d+c)n\log n+2nd+cn$	General computational complexity of the algorithm.

The memory occupied by the algorithm or "space complexity" is also one of the important performance indexes. When moving the single dimensional data, we should load data of a certain dimension of all data points into the memory and then move them until one merged point is obtained, so the memory cost is $O(n)$. When merging data points, we should keep on inquiring serial numbers of data points in a certain interval in a certain dimension (if after sorting the d-dimensional data of the data points in ascending order, we merge the points in the $[x_1, x_2]$ interval with the points in the $[x_3, x_4]$ interval, then during the partition process, we need to inquire which points are included in $[x_1, x_2]$ and which points are included in $[x_3, x_4]$ in order to obtain the intersection with the current data set.). This requires loading the values and serial numbers of all points in all dimensions in ascending order into the memory for inquiries. However, we needn't operate these data so we can load them into several servers according to interval and dimension and start the inquiry service program for the partition program to inquire. Therefore, theoretically, it can meet the data set with infinite n and d. So, the limitation of this algorithm on the memory exists in the merging stage. The required memory or space complexity is $O(n)$.

3. Experiment design & result analysis

3.1. Experiment assessment indexes

This study tests efficiency and accuracy of this algorithm with random and public data:

1. Efficiency: Test relations between operating time, data quantity, dimension and cluster quantity:

2. Accuracy: Test discrepancy between the clustering result derived from this algorithm and the standard result and compare it with other algorithms.

3.2. Generation method of synthetic data

The methods for randomly generating data sets in this experiment include uniform distribution and normal distribution. Parameters in the generation program for uniform distribution data sets can be seen in Table 2, which includes coordinates $\langle c_{i1}, \cdots, c_{id}\rangle$ of k d-dimensional core points $core_1$, $core_2$, $\ldots$, $core_k$, $core_i$, data point quantity n_k produced by $core_i$ and generation radii $< ri_1, ri_2, \cdots, ri_d >$ of $core_i$. The method of generating data points: choose the core points in sequence to produce data points of the designated quantity. The coordinate of a certain data point in a certain dimension j equals to the coordinate c_{ij} of $core_i$ plus a value randomly and uniformly produced from $[-r_{ij}, \cdots, r_{ij}]$. Thus the data point produced by each core point is located within a hyper-rectangle cube with the core point as the center.

The generation program for normal distribution data sets is to randomly produce data samples in each dimension according to normal distribution. The number of parameters needed is the same as the number of parameters needed in the generation program for uniform distribution data sets, but c_{ij} indicates the average μ_{ij} of the ith normal distribution in the jth dimension, r_{ij} indicates the standard deviation σ_{ij} and n_i indicates the data point quantity of the ith normal distribution.

Table 2. Generation parameter of data samples

	Coordinate	Radius	Quantity
$core_1$	$\langle c_{11}, \cdots, c_{1d}\rangle$	$\langle r_{11}, \cdots, r_{1d}\rangle$	n_1
$core_2$	$\langle c_{21}, \cdots, c_{21d}\rangle$	$\langle r_{21}, \cdots, r_{2d}\rangle$	n_2
$\cdots$	$\cdots$	$\cdots$	$\cdots$
$core_k$	$\langle c_{k1}, \cdots, c_{k1d}\rangle$	$\langle r_{k1}, \cdots, r_{kd}\rangle$	n_k

3.3. Experiment result and analysis of synthetic data

3.3.1. Test generation of data set. In order to test the performance of this algorithm, this research has tested the performance and quality of this algorithm with the randomly generated data samples derived with the generation method of synthetic data sets. The performance test includes relations between running time, data points, dimension quantity and cluster quantity. Each experiment will record total running time of clustering, reading/writing time of the hardware I/O and reading/writing & computing time of the program in the memory. Accuracy test includes clustering quality assessment of this algorithm and comparison of clustering quality with other clustering algorithms [7].

The following is the performance result derived from synthetic data. The testing machine used is Pentium 4CPU 2.66 GHz with memory of 1.49 GB.

Figure 2 displays the running time for data points to increase from 1000 to 100000.

The data sample is composed of 10 dimensions, including 10 clustering centers. The algorithm presets the cluster quantity to be 128. It can be seen that the curves for total running time and computing time in the memory have shown quadratic characters while the reading/writing time of the hardware I/O are linearly related to data point quantity. All conform to the description of algorithm complexity.

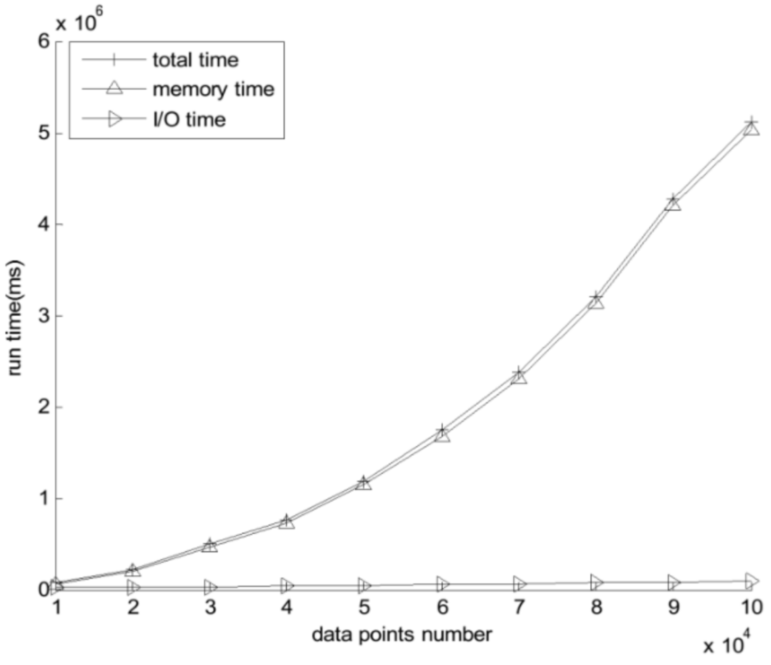

Fig. 2. Relationship between data point quantity and running time

Figure 3 displays the running time for data dimension to increase from 10 to 100. The data sample is composed of 10000 data points, including 10 clustering centers. The algorithm presets the cluster quantity to be 128. It can be seen that the total running time, the computing time in the memory and the reading/writing time of the hardware I/O are all linearly related to dimension quantity. All conform to the description of algorithm complexity.

3.3.2. Cluster quantity. Figure 4 displays the running time for cluster quantity to increase from 2 to 1000. The data space is composed of 10000 data points, including 10 dimensions. It can be seen that the total running time and the computing time in the memory have shown quadratic characters while the reading/writing time of the hardware I/O remains basically unchanged. All conform to the description of algorithm complexity.

3.3.3. Accuracy. In order to test clustering quality, this research adopts the Davies–Bouldin index method as the parameter for internal assessment. The external assessment will adopt the Rand measure (William M. Rand 1971) method and at the

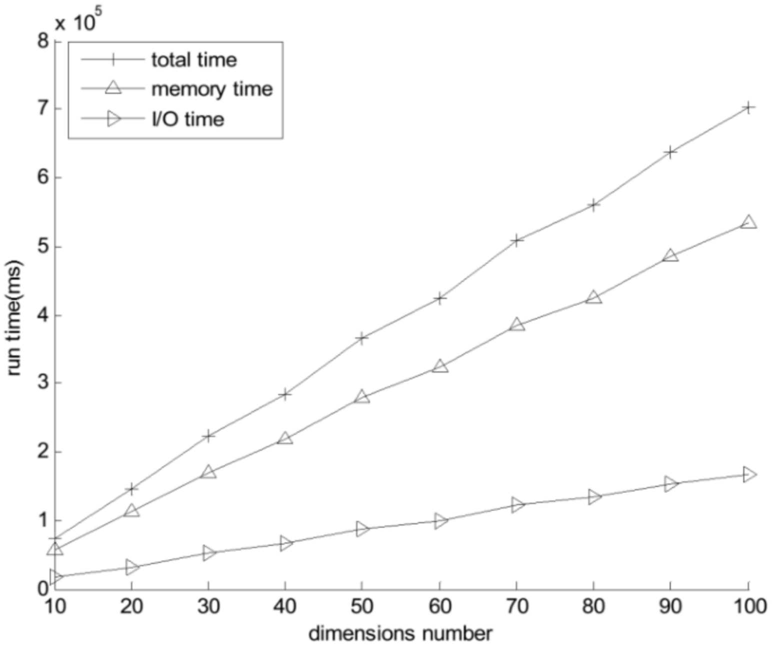

Fig. 3. Relationship between dimension and running time

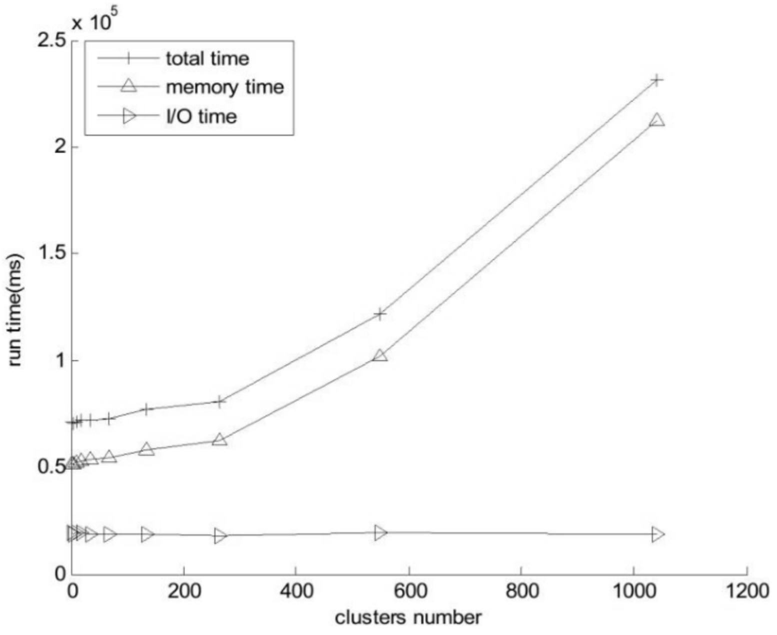

Fig. 4. Relationship between cluster quantity and running time

same time test the two clustering quality parameters from algorithms of CLIQUE[70] and DBSCAN[57] as a horizontal comparison.

This research mainly focuses on overall accuracy and stability of the algorithm. The formula for calculating Rand Index is

$$R = \frac{TP + TN}{TP + FP + FN + TN}, \tag{3}$$

where TP refers to the quantity of pairs which are in one category in the X clustering method and also in one category in the Y clustering method, TN refers to the quantity of pairs which are not in one category in the X clustering method and not in one category in the Y clustering method, FP refers to the quantity of pairs which are in one category in the X clustering method but not in one category in the Y clustering method, and FN refers to the quantity of pairs which are not in one category in the X clustering method but in one category in the Y clustering method.

The value range of Rand Index is [0,1]. The closer it is to 1, the better is the clustering quality.

This research tests accuracy and stability of DBSCAN, CLIQUE and this algorithm with the same synthetic data. DBSCAN represents the method based on density and CLIQUE represents the high dimensional subspace clustering method. Use randomly chosen radii, core points and data point quantity to produce 8 data sample generation program and then use these programs. Each program can randomly produce 100 samples whereas range of core point quantity is $[2, 32]$, range of dimension quantity is $[1, 20]$ and range of core point coordinate is [-500, 500]. The value range of radius is two times the range of core point coordinate divided by core point quantity. The value range of the data sample quantity is $[50, 5000]$. CLUQUE & DBSCAN clustering algorithm requires presetting of parameters. For every 100 data samples, this experiment will use optimal parameters derived from previous experiments of the same dimension and the same data sample quantity.

Cluster each sample with the above three clustering methods and calculate Rand Index of each clustering result. Average the experiment results produced by 100 samples using the same program. Figure 5 is the RandIndex histogram of the results of each clustering algorithm.

In perspective of average and variance, we can see that this method is not only superior to CLIQUE and DBSCAN methods in average accuracy but also remarkably enhances stability. The significant difference results derived by testing this algorithm and the other two algorithms with F-test are 0.009542 and 0.001241. This shows that there is a significant difference in RandIndex variance between this algorithm and CLIQUE or DBSCAN algorithm, that is, stability of this algorithm is superior to that of CLUQUE or DBSCAN algorithm.

According to the display of Rand Index, this method is more stable and accurate in performance while DBSCAN or CLIQUE methods will cause big fluctuations due to poor matching of its preset parameters with the data. This is caused by limitation of the method itself. This is because other algorithms require presetting of some thresholds but the optimal thresholds are usually hard to be set for a new

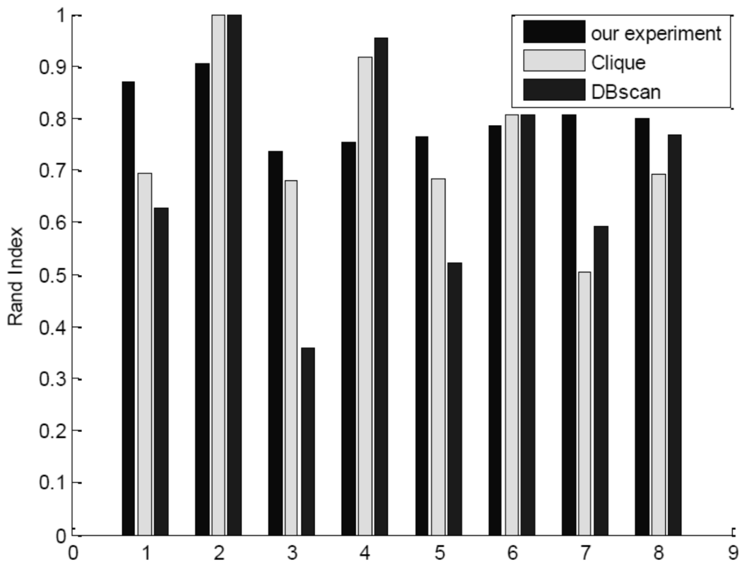

Fig. 5. Rand index of each clustering algorithm

data sample. Sometimes DBSCAN presents very low accuracy especially in handling high dimensional data. Sometimes this algorithm also has lowered accuracy for the clusters of these data samples cannot be divided by the axial hyperplane.

This research adopts the publicly downloaded data sets for specially testing clustering algorithms provided by School of Computing, University of Eastern Finland [1]. The data set is Aggregation, including 788 data points and 7 pre-labeled clusters.

Table 3 displays parameters corresponding to cluster quantity. It can be seen that when cluster quantity is 12 or 16, the sum of single dimensional entropy tends to be stable; when cluster quantity is 12, RandIndex reaches its optimal value 0.897855. So a good clustering result is achieved.

Table 3. Experimental results of actual data

Cluster Number	Sum of entropy	DB index	Rand index	Wk Index
2	0.114169	1.282283	0.250901	3109.942
3	0.26294	4.400401	0.303001	3012.798
4	1.580926	3.78385	0.719845	2364.641
5	1.625217	3.177317	0.720025	2334.227
9	1.731556	2.75222	0.727604	2241.779
12	3.387532	2.528866	0.897855	1217.795
16	3.763777	1.844235	0.890344	898.7305
20	4.001007	1.497016	0.887028	767.9188
23	3.964119	1.417218	0.880108	753.0749
29	4.046071	1.48137	0.865389	712.2116
35	4.3863	1.727795	0.840908	664.814

3.4. Summary of algorithm

Through the above analyses, we can see the following advantages of this algorithm:

1. Stable. The results of many clustering algorithms such as K-MEANS and DBSCAN have a lot to do with preset parameters. Some algorithms, BIRCH for example, are sensitive to input sequences of data. But in this algorithm, the output for a definite input is also definite. Moreover, there is no need to set parameters so it requires no knowledge from any particular fields. This is of great significance to applications in different fields.

2. Parallel process in a distributed way. The data can be split into single dimensional data for clustering separately, so the single dimensional data can be sent to distributed servers for parallel process.

3. It has overcome the problem of intervals between data points of high dimensional data tending to be equal. This algorithm does not adopt intervals to calculate similarities between data points so it can overcome this problem.

4. Discussion

High dimensional data clustering is a very complicate problems so there are still many areas to be further studied and perfected. Though the method proposed in this paper has made some improvements in computational complexity and stability, the long-time study finds that the essence of this study is to divide the data set with a plane parallel to the coordinate axis of the hyperspace. For data sets that cannot be divided by a plane parallel to the coordinate axis, this method cannot achieve good results. This research proposes to first reduce dimensionality with the principal component analysis (PCA) and then employ this method for clustering.

5. Conclusion

Taking reduction of algorithm complexity and dependence on preset parameters as the target, this research proposes a clustering algorithm based on changing tendency of the sum of edge differential entropy according to basic principles of information theory. It first performs separate hierarchical clustering to the data sample mapped to each dimension in the principle of monotone decrease of information distribution entropy and then obtains the final results by partitioning the single dimensional clustering results in the principle of maximizing the sum of single dimensional information entropy. The research verifies this algorithm with synthetic data and actual data.

The study shows that this algorithm, by adopting the method of first performing clustering in a single dimension and then repartitioning the data samples with the single dimensional clustering results in the whole data space, can remarkably enhance its computational efficiency compared with other existing algorithms such as CLIQUE ad DBSCAN. In additional, both experimental results derived from synthetic data and actual data samples show that this method is not only superior to other existing algorithms in accuracy but also has remarkably increased its stability.

References

[1] H. Sun, Y. Du, D. Jiang: *ESCHCD: Entropy-based algorithm for subspace clustering with high dimensional categorical datasets.* Journal of Shandong University (Engineering Science) *41* (2011), No. 5, 37–45.

[2] L. Tang: *Text feature selection method based on information entropy and dynamic clustering.* Computer Engineering and Applications *51* (2015), No. 19, 152–157.

[3] C. Bellenguez, A. Strange, C. Freeman, P. Donnelly, C. C. Spencer: *A robust clustering algorithm for identifying problematic samples in genome-wide association studies.* Bioinformatics *28* (2012), No. 1, 134–135.

[4] M. Mutwil, B. Usadel, M. Schütte, A. Loraine, O. Ebenhöh, S. Persson: *Assembly of an interactive correlation network for the Arabidopsis genome using a novel heuristic clustering algorithm.* Plant physiology *152* (2010), No. 1, 29–43.

[5] L. Cao, H. Zhang, Z. Wang: *EB-SVM: Support vector machine based data pruning with informatior entropy.* Journal of Shandong University (Natural Science) *47* (2012), No. 5, 59–62.

[6] P. Jiang, M. Singh: *SPICi: A fast clustering algorithm for large biological networks.* Bioinformatics *26* (2010), No. 8, 1105–1111.

[7] U. Doraszelski, K. L. Judd: *Avoiding the curse of dimensionality in dynamic stochastic games.* Quantitative Economics *3* (2012), 53–93.

[8] H. Ouyang, X. Dai, Z. Wang, M. Wang: *Rough K-prototypes clustering algorithm based on entropy.* Computer Engineering and Design (2015), No. 5, 1239–1243.

Received August 7, 2017