International Journal of Database Theory and Application
Vol.9, No.9 (2016), pp.83-96
http://dx.doi.org/10.14257/ijdta.2016.9.9.08

Research on Clustering Algorithm Based on Grid Density on Uncertain Data Stream

Tang Xianghong, Yang Quanwei and Zheng Yang

Key Laboratory of Advanced Manufacturing Technology (Guizhou University), Ministry of Education, Guiyang, China
txhwuhan@163.com, 825537796@qq.com, 403370748@qq.com

Abstract

To solve the clustering algorithm based on grid density on uncertain data stream in adjustment cycle for clustering omissions, the paper proposed an algorithm, named GCUDS, to cluster uncertain data steam using grid structure. The concept of the data trend degree was defined to describe the grade of a data point belonging to some grid unit and the defect of information loss around grid units was removed in the GCUDS algorithm. The GCUDS algorithm obtained better results of clustering and higher time efficiency than other algorithms over uncertain data stream, through improving the traditional online clustering framework and maintaining three buffers of micro-cluster. Experimental results showed that the GCUDS algorithm could effectively cluster in different shape database and outperform existing methods in clustering quality and efficiency.

Keywords: *uncertain data stream; clustering; grid; grid structure; density*

1. Introduction

In recent years, with continuous development of information technology, data stream models characterized by high speed and large-scale, are widely found in many applications such as internet applications, sensor networks cloud computing and so on[1-2]. Due to the unreproducible characteristic, algorithms over data stream only scan data in data stream for single pass [3-4]. In addition, original data are not accurate and there is some deviation [4-5], where uncertainty is generated by the data itself, measure errors and so on [6]. Therefore, the analysis and mining of uncertain data stream has become a hot research topic recently [7].

This paper focuses on clustering over uncertain data stream. Traditional clustering methods over static data cannot be suitable for data streams, especially uncertain data stream. At present clustering algorithms for uncertain data stream are still rare. The UMicro algorithm put forward by Aggarwal [8] is the typical one in clustering algorithms for uncertain data stream. It expands the CF structure into the ECF structure and enhances the description of uncertain part. But the maximal number of clusters that this algorithm can specifies cannot exceed n_{micro} and the size of data of this algorithm is limited. At the same time, the algorithm may miss some important information in the distance calculation. Zhang *et al.* proposed a kind of algorithm [9] based on information entropy to measure uncertainty information of tuples. But analysis of the distribution characteristics of recent data in the process of clustering of the algorithm is not enough. The P-streams algorithm [10] is propound by Dai *et al.* to do clustering in data streams. But more information is lost by the algorithm, when there are many outliers and the candidate cluster window is wholly updated. The EMicro algorithm was given by Zhang *et al.* [2] to improve the quality of clustering by considering uncertain level and distance between tuples and tuples. But the way to generate the initial clusters of the algorithm is too random and these outliers which repress entinitial clusters will be directly deleted to affect the quality of clustering. Moreover, Thanh *et al.* [11] put forward a kind of CLARO model based on natural extraction of continuous random variables to support the uncertainty data stream. Kriegel *et al.* proposed the FDBSCAN clustering algorithm [12].Calculation accuracy and time of the algorithm cannot be guaranteed and the clustering results may be affected, because calculating objects of the algorithm are discrete. The FDCUS algorithm [13] was proposed by Tu *et al.* But the algorithm lacks the processing of edge grid cells to lose edge information of clustering and clustering results will be affected.

However, most of the above clustering algorithms are based on distance measure and there are many short comings. For example, the number of clustering should be predefined and it is not suitable to find a non-convex shape cluster. What's more, it is sensitive to noise and outliers and so on[14].Some of the discussed clustering algorithms can mine clustering of any shape, but the clustering accuracy is not enough and efficiency is not high.

Therefore, this article proposed an uncertain data stream clustering algorithm based on grid density to solve defects of above methods by improving the method of grid density. Main work includes following aspects: first, the data trend degree of an object is defined to describe the approach distance of the object to the grid cell which the object will fall in and the adjacent grid cells. Second, a new grid clustering framework structure is designed in the GCUDS algorithm. In the new structure, a small number of grid cells march a real-time clustering along with arrival of those new data and clustering results in most of grid cells are updated by fixed time interval. Better clustering results and high performance of real-time will be achieved by the structure. Finally, the GCUDS reduces the unnecessary calculation of grid processing through maintaining three micro cluster buffers at the same time.

2. The Related Model and Definitions

2.1. Uncertain Data Stream Model

In the uncertain data stream, uncertainty of tuples can be classified into two kinds: uncertainty of tuple existence and uncertainty of tuple attribute. In this paper, the stream model in which attribute values of tuples are determinate and existence of tuples is uncertain is mentioned. In addition, the model is widely applied in such as fields of the RFID, wireless sensor and so on. In the model, if a tuple can be expressed as $<\overrightarrow{X_j}, p_j>$, where $<\overrightarrow{X_j}, p_j>=\{(x_j^1, x_j^2, ..., x_j^d), p_j\}$, $\overrightarrow{X_j}$ is the jth tuple and p_j is the probability of the existence of $\overrightarrow{X_j}$, $0 \le p_j \le 1$, the data stream will be described as $L=(<\overrightarrow{X_1}, p_1>, <\overrightarrow{X_2}, p_2>, ..., <\overrightarrow{X_m}, p_m>)$, where m is the total number of tuples.

2.2. Network Model

A dimensional data space is expressed as $S = S_1 \times S_2 \times ... S_i \times ... \times S_d \, (i=1,2,...,d)$ where S_i is the ith dimension. S is divided into a grid with $N=\prod_{i=1}^{d} p_i$ grid cells, when each dimension S_i is evenly divided into l_i segments and is denoted as $S_i = S_{i,1} \cup S_{i,2} \cup \ldots \cup S_{i,li}$. Each grid cell consists of $S_{1,u} \times S_{2,u_2} \times ... S_{2,u_i} \times \ldots S_{d,u_d} \left(u_i = 1,2,\ldots,l_i\right)$ and is expressed as $g=\left(u_1,u_2,\ldots,u_d\right)$. A data item $\vec{X}=\left(\left(x_1,x_2,\ldots,x_d\right), \;\; p, \;\; t\right)$ whose arrival time is t will be mapped to the corresponding grid cell $g(x)=\left(u_1,u_2,\ldots,u_d\right)$ according to its data mark coordinates $\vec{X}=\left(x_1,x_2,\ldots,x_d\right)$, where $x_i \in S_{i,u_i}$.

2.3. Related Definition of Grid Processing Mechanism

Definition 1: the grid density

Suppose that a grid g contains data points $X_1, X_2, \ldots, X_n$ at t moment and corresponding weights of these data points are $\mathrm{D}(X_1, \mathrm{t}), \mathrm{D}(X_2, \mathrm{t}), \mathrm{D}(X_n, \mathrm{t})$, the density of the grid cell g is equal to the sum of all weights of data points that fall into the grid cell g, namely $\mathrm{D}(\mathrm{g}, \mathrm{t}) = \sum \mathrm{D}(\mathrm{X}, \mathrm{t})$.

Definition2: the data trend degree

The data trend degree refers to degree or trend how close an uncertain data point is to some grid cell and adjacent grid cells and it is denoted as TD.

When an uncertain data point P arrives and is mapped into the grid cell B2, P need be relocated. In other words, the data trend degree of P to P's grid cell B2 and adjacent grid cells need be calculated. By calculation it can be known that the smaller the distance between P and the centroid of the grid cell is, the closer P is to the cluster of the grid cell. It leads to the conclusion that the data trend degree of a data point is inversely proportional to the distance between the data point and the centroid of the grid cell which the data point belongs to.

Moreover, the higher the distribution density between P and the centroid of the grid cell which is P belongs to or the adjacent cell is , the closer P is to be the cluster of the grid cell. So the data trend and degree is proportional to the grid density $\mathrm{D}(\mathrm{g}, \mathrm{t})$ and the data trend degree is defined as $TD = {}^{\mathrm{D}(\mathrm{g}, \mathrm{t})}\!/_{d}$ (d is the distance between P and the centroid of the grid cell) after lots of trial and error. When a new data point arrives, it will be mapped in the grid cell to which the data trend degree of P is the biggest. For example, if the data trend degree of P to the grid cell C1 is bigger than to the gird cell B2, P will be mapped in the grid cell C1.The result of the clustering based on the data trend degree will be more accurate than previous methods.

Definition 3: **s**parse grid cell, transitional grid cell, dense grid cell

If $\mathrm{D}(\mathrm{g}, \mathrm{t})$ is the density of the grid cell g at the moment of t and $\mathrm{D}(\mathrm{g}, \mathrm{t}) \le \frac{C_l}{N(1-e^{-\lambda})} = D_l$, the grid cell g will be called the sparse grid, where C_l($0 \le C_l \le 1$) called the sparse coefficient is a constant and λ is the attenuation coefficient.

If $\mathrm{D}(\mathrm{g}, \mathrm{t})$ is the density of the grid cell g at the moment of t and $\frac{C_m}{N(1-e^{-\lambda})} \ge \mathrm{D}(\mathrm{g}, \mathrm{t}) \ge \frac{C_l}{N(1-e^{-\lambda})} = D_l$, the grid cell g will be called the transitional grid cell, where C_m called the dense coefficient is a content and $0 \le C_m \le N$.

If $\mathrm{D}(\mathrm{g}, \mathrm{t})$ is the density of the grid cell g at the moment of t and $\mathrm{D}(\mathrm{g}, \mathrm{t}) \ge \frac{C_m}{N(1-e^{-\lambda})} = D_m$, the grid cell g will be called the dense grid.

The type of the grid cell is decided by the density of the grid cell. According to types of grid cells, the grid space is divided into three levels, namely strong, middle and lower. The grid units in different levels will be treated differently and it can avoid the waste of computation caused by non-discrimination against all elements in the grid. In addition, through the analysis of many experiments, it is observed that levels of grid units can be changed slowly but are unlikely to skip a grade ,with the inflow of new data and the decline of old ones. In the process of clustering, the combination of the two grid cell must follow the rule that one grid cell can only be merged with adjacent grid cells of the same level or lower level. But mergence of clusters needs not less than two adjacent grid cells of the highest level which cells clusters belong to.

Definition 4: the grid gravity [4]

For the grid cell g, if h is its adjacent grid and $k_1 = 2\sigma^2, \quad k_2 = \alpha k$, the grid gravity of g to h is defined as

$$\mathrm{F}(\mathrm{g},\mathrm{h},\mathrm{t}) = \ln \frac{k_1 \cdot \ln\left[k_2 \cdot D(g,t) D(h,t)\right]}{dist(g,h,t)^2}$$

Where k$_1$and k$_2$are the gravity coefficient, k (k>0)is the direct proportion coefficient, σ is the distance parameter. α(α>0)is the direct proportion coefficient.

Definition 5: the gravity threshold[13]

If the current moment is t_c and the moment when the last data were read is t_g, the gravity threshold of the grid cell on the ith dimension is expressed as the following formula:

$$F_{\min}(i,t_g,t_c) = \ln[\frac{k_1}{l_i^2} \bullet \ln \frac{k_2 C_l C_m (1-e^{-\lambda(t_c-t_g+1)})}{N^2(1-e^{-\lambda})^2}]$$

Where l_i is the center distance between the adjacent grid cells on the ith dimension.

Definition 6: the grid feature vector

The feature vector of the grid cell g is expressed as $(\mathrm{t}, \mathrm{D}, p_c, X_{\mathrm{centre}}, \mathrm{type}, \mathrm{label})$,where t is the moment when g is updated recently, D is the latest density of g, p_c is the sum of the initial weights of the grid data, X_{centre} is the weighted-centroid of the grid g which is recently updated, type is the name of the class which the grid g belongs to, label is the serial number of the cluster which the grid g belongs to.

In deterministic data stream management, the method of clustering is the procedure that the system will map them into the corresponding grid cells according to their coordinates and then do clustering based on statistical analysis of the grid density, when tuples flow into the stream. But the method does not run well over the uncertain data stream because of the uncertainty and attenuation of tuples in the uncertain data stream. To handle the uncertainty and attenuation, the following assumptions about the grid processing mechanism are made in this paper:

Assuming that the probability of each new datum in the data stream is its initial weight and is still expressed as p, the weight of each data declines with time and the attenuation function of weight is $\mathrm{f}(\mathrm{t}) = \mathrm{pe}^{-\lambda \mathrm{t}} (\lambda > 0)$, where t is the current time. If a data element X is read at the t$_c$ moment, its current weight is $\mathrm{D}(\mathrm{X},\mathrm{t}) = \mathrm{f}(\mathrm{t} - t_c) = \mathrm{pe}^{-\lambda(\mathrm{t}-t_c)}$.

According to literature [7], when a new datum is mapped into the grid cell g at the current moment t and the grid cell g last received a new element at the $t_g (t_g < t)$ moment, the density of the grid cell g can be incrementally updated by the following formula:

$$\mathrm{D}(\mathrm{g},\mathrm{t}) = \mathrm{f}(\mathrm{t} - t_g) \cdot \mathrm{D}(\mathrm{g}, t_g) + \mathrm{p}$$

When new data arrive, the grid density will be changed over time. But it is not necessary to update the grid density over time because it can meet the requirement of grid density statistics that the update of the grid density is done only when new data or queries arrive.

In addition, the relevant analysis shows that the centroid of the grid cell not only relates to the coordinate of the data point but also to the current weight value. The centroid in relation with the coordinate and current weight values of data points is called weighted centroid in this paper. Assuming that the grid cell g contains the data points $X_1, X_2, \ldots, X_n$ at t moment and the current weights of these data respectively are $\mathrm{D}(X_1, \mathrm{t}), \mathrm{D}(X_2, \mathrm{t}), \mathrm{D}(X_n, \mathrm{t})$, the weighted centroid of the grid cell g is defined as

$$X_{\text{centre}}(g,t) = \frac{\sum_{i=1}^{n} D(x_1,t)\,x_i}{D(g,t)}$$

Based on literature [4], [13], the incremental update formula of the weighted centroid of the grid cell is:

$$X_{centre}(g,t) = \frac{\left(D(g,t) - D(x_n,t)\right)□X_{centre}(g,t-1) + D(x_n,t)□x_n}{D(g,t)}$$

In summary, the new uncertain data point will be located in the grid cell whose grid trend degree to the new uncertain data point is largest according to Definition 2, when a new uncertain data point arrives. This will help the clustering results be more accurate. The information of the grid density and centroid in the grid feature vector can be incrementally updated directly. In addition, only one data point can be read in each time unit, when new points are put in the stream. The incremental update of the grid cells' information is small proportion and has features of obvious change and strong randomness. Most of information update is an attenuation type whose change follows the distribution of $e^{-\lambda}$ and whose change range is small in a short time. Therefore, a new grid clustering framework is designed in this paper based on above characteristics of information update. In the new grid clustering framework, real-time clustering is executed in the grid cells with incremental update, while clustering results are modified in ones with attenuation update by a time interval. The new framework can lead to an accuracy and efficient clustering result.

3. Clustering Process of Uncertain Data Stream

A detailed description of the algorithm in this paper:

With the arrival of new data points, the data trend degree of the new data to its grid cell and its adjacent grid cells need be calculated and they will be mapped into the corresponding multidimensional space grid according to the data trend degree. Then the feature vector of the grid will be updated. If the grid cell is an independent grid cell (there is only one high-level grid cell in the cluster and the rest are low-level grid cells), the new data points will create a new cluster based on the category of the grid cell which it belongs to, and then the new cluster will be put into the corresponding cluster buffer. If the grid cell is not an independent grid cell, the new data point will be classified into the cluster of the grid cell which the data point belongs to. The micro clusters in the buffer are entirely updated and regulated every a time interval of T length, where T is the time cycle parameter. In a time interval of T length, minor modification is used for the clustering results by real-time incremental update because new arriving data points have a great effect on the clustering results but the impact caused by old data attenuation on the clustering results is minimal in a relatively short time.

4.1 The frame of the algorithm

Algorithm (data stream: DS, the total number of the grid: N, attenuation factor: λ, Minimum density threshold: C_l, Maximum density threshold: C_m, gravitational coefficients: k_1, k_2)

```
1) Begin
2) While (The data stream DS does not end) do
3) t=0
4) while (the new arriving data point $\vec{X}$ is received)
```

```
5)  Update Grid(Grid L is t , g);    // the new datum is mapped into the grid cell g
6)  if ( the grid cell g is dense and isolated)
      A core micro-cluster which only contains the grid cell g is created and is added into the core micro-cluster buffer BUF
7)  else if (the grid g is dense but not isolated)
          BUF is traversed to find the cluster of the grid cell g
          FindOptimalCluster();
8)      else if (the grid g is transitional and isolated)
      A fake core micro-cluster which only contains the grid cell g is created and is added into the fake core micro-cluster buffer FUF
9)  else if (the grid g is transitional but not isolated)
          FUF is traversed to find the cluster of the grid cell g
    FindOptimalCluster();
10) else if (the grid g is sparse and isolated)
    It is judged whether the grid density is more than the threshold or not. If the grid density is more than the threshold, a micro-cluster which only contains the grid cell g will be created and is added into the outlier micro-cluster buffer DUF. If not, the grid cell g will be deleted.
11) else if (the grid g is sparse but not isolated )
          DUF is traversed to find the cluster of the grid cell g
                    FindOptimalCluster();
12) t++;
13) if (t MOD T==0)
14) CkeckClustersProcess();    // adjust and maintain the micro-cluster
```

In the algorithm, the data $\left(\vec{X},p\right)$ recently received update the grid feature vector by the function UpdateGrid() (step 5). If the grid cell g is dense and isolated, the core micro-cluster which only contains the grid cell g will be created and be added into the core micro-cluster buffer BUF (step 6). If the grid cell g is dense but not isolated, the BUF should be traversed to find the cluster of the grid cell g and the function FindOptimalCluster() is called to determine whether the grid is absorbed by the existing cluster (step 7).If the grid g is transitional and isolated, a pseudo core micro-cluster which only contains the grid cell g should be created and be added into the pseudo core micro-cluster buffer BUF (step 8). If the grid g is transitional but not isolated, the FUF should be traversed to find the cluster of the grid g and the function FindOptimalCluster()can be called to determine whether the grid is absorbed by the existing cluster (step 9). If the grid cell g is sparse and isolated, the algorithm needs to compare the grid density with the threshold. If the grid density is more than the threshold, the micro-cluster which only contains the grid cell g will be created, and be added into the micro-cluster buffer DUF. Otherwise, the grid cell g will be deleted (step 10). If the grid g is sparse but not isolated, the algorithm will traverse the FUF to find the cluster of the grid cell g and call the function FindOptimalCluster() to determine whether the grid is absorbed by the existing cluster (step 11). Finally, the algorithm calls the function CkeckClustersProcess() to adjust the micro cluster buff (step 12).

Meanwhile, the grid cells in the core micro-cluster buffer usually compose a cluster which considers the dense grid cells as the centers and includes their adjacent sparse grid cells and transition grid cells. The grid cells in the pseudo core micro-cluster buffer usually compose a cluster which considers the transition grid cells as the centers and contains their adjacent sparse grid cells. The outlier micro-cluster buffer contains the clusters which consider the sparse grid cells as the centers and the clusters are generally derived from the clusters created by the new not empty grid cells or the clusters moved from the fake core micro-cluster buffer for attenuation.

3.2. Evolutionary Function of the Cluster-CkeckClustersProcess()

```
Algorithm 2
  1) while（a micro-cluster is not detected in buffer）
      a) while（the weight of some grid cell in the micro-cluster is lower than ρ）
         The grid cell is removed from the cluster;
         If（the cluster is not continuous）
             The cluster is divided into two clusters;
      b) while（some cluster in a buffer does not belong to this buffer according to
         the features of the cluster）
         The cluster is moved from the buffer it is in to the corresponding buffer the
         features of the cluster；
```

The function CkeckClustersProcess contains three steps. Firstly, the micro-cluster grid cells are checked in each buffer and old micro-cluster grid cells are removed from the micro-cluster. Whether micro-cluster grid cells are old or not is based on their weight ratio parameter. As mentioned as before, the probability sum of a grid cell is p_c and the weight ratio of the grid cell can be defined as $\mathrm{D}(g,t)/p_c$ which belongs to [0,1]. If the weight ratio of a grid cell is lower than ρ(ρ is a parameter predefined by users.),it will indicate most data in the grid cell are old and the grid cell should be removed from its micro-cluster. Secondly, as new data continually arrive continuously and are mapped into the grid cells and the grid cells are constantly evolved or removed, it may happen that the micro-clusters are disconnected .When a disconnected micro-cluster is found, it should be divided into two clusters by the function. Finally, if a buffer contains a cluster which does not belong to the buffer according to the features of the cluster, the cluster should be moved to the corresponding buffer.

3.3. The Processing Function of New Coming Data--UpdateGrid()

```
Description of UpdateGrid () :
Input：Data in an uncertain data stream
Output: Real-time information of the grid cells
  1) Read the data in a data stream

  2) If (the adjacent dense grid cell of the new data element is found);// Computation
     of the data trend degree of the sparse grid cell can affect the boundary of the
     cluster
  3) The distance $d_i$ of the data element to the centroid of its grid cell and its adjacent
     dense grid cell should be calculated respectively.
  4) Verify $d_i$,
  5) If ($d_i$ is valuable)
      a) Calculate the trend degree of the data element to the corresponding grid
         cells;
      b) Classify the data element into the grid cell with the highest trend degree.
      c) Update the feature vector of the grid cell
  6) Else the data element is directly mapped to the grid cell which it belongs to and
     the information of the grid is updated;
```

3.4. Cluster Selection Function-FindOptimalCluster()

Description of FindOptimalCluster():

```
1)  If（the grid cell g is changed to a sparse grid cell）
        while（all adjacent grid cells of the gird cell g）
2)      Calculate the grid gravity of the grid cell g and its adjacent grid cells
3)      If（The biggest grid gravity is less than the gravity threshold）
4)              delete the grid cell g；
5)      Else the grid cell g will be merged into the cluster which contains the
    adjacent grid cell with the biggest gravity.
        If（The cluster of the grid cell g is not connected）
            The cluster will be divided into two clusters;
6)  Else  if（The grid cell g is changed to a dense grid cell）
        Select the grid cell h which is adjacent with the grid cell g and has the
    biggest grid gravity;
7)      If（the gird cell h is dense）
          Merge the cluster of the grid cell g and the cluster of the grid cell h;
        Else if(the gird cell h is  transitional )
            If(the grid cell g is a dense grid cell and have the biggest grid gravity in
    the adjacent grid cells of  the grid cell h.)
            Map  the grid cell h into the cluster of the grid cell g；
8)  Else if（the gird cell g is changed to a transitional grid cell）
                If( there are the dense grid cells adjacent to the grid cell g)
        Find the cluster which contains the grid cell which have the biggest grid
    gravity in the adjacent grid cells of the grid cell g and map the grid cell g into the
    cluster.
9)      end if
10) end
```

In the function FindOptimalCluster(), it is important to determine the level of the grid cell g. If the grid cell g is a sparse grid cell and the maximal grid gravity from all its adjacent density grid cells is less than the minimal grid gravity from the adjacent boundary grid cells of the cluster, the grid cell g will be deleted. Otherwise, the grid cell g should be merged into the cluster. If the cluster is disconnected, it will be divided into two different clusters. If g is a density grid cell and the gravity of the grid cell h is the biggest in the adjacent grid cells of g, the cluster of grid g will be merged with the cluster of the grid cell h. If the grid cell h is a dense and transitional grid cell and the gravity of grid g is the biggest in the adjacent density gird cells of the grid cell h, the grid cell h will be merged in to the cluster of the grid cell g. If the grid cell g is changed to a transitional grid cell and there are the dense grid cells adjacent to the grid cell g, the cluster which contains the grid cell which have the biggest grid gravity in the adjacent grid cells of the grid cell g will be found and the grid cell g will be mapped into the cluster.

3.5. Time Complexity

The time complexity of the algorithm GCUDS cost mainly consists of two parts. First part is the time complexity caused by dividing each dimension and mapping new data to the corresponding grid cell. The time complexity of first part is no more than O(m*2d), where 2d is the time complexity to calculate the data trend degree of each data point , d is the number of dimensions of data space and m is the size of data set. Second part is the time complexity of clustering. The time f clustering is mainly spent on choosing the clusters, classifying the types of the grid cells and evolving the clusters. The time complexity of choosing the clusters is O(m*2d).The worst time complexity in classifying the types of the grid cells is no more than O(m). The time consumption of evolving the clusters is less than O(N) where N is the number of grid cells. So the time complexity of the algorithm is O（N+2m*2d+m）=O(N+m*d).Unlike other clustering algorithms such as FDCUS, the algorithm GCUDS introduces the data trend degree to get more accurate clustering results, though calculation of the data trend degree may increases the time complexity to some extent. Besides, it adopts a new clustering framework structure. In the new clustering framework structure, the grid cells are divided into a few levels and are added into different buffers where the micro-clusters are regularly adjusted. It helps to reduce some unnecessary calculation and to lower the time complexity.

4. Experimental Results and Analysis

In this paper, the experimental platform is a PC which has AMD Phenom(tm) Ⅱ X4 810 Processor, 2.60GHz dominant frequency, 2GB internal storage and the operating system of PC Windows 7. MATLAB 2010b is used as the programming tool to implement clustering algorithms.

4.1. Test Data Design and Parameter Setting

In the following experiments, simulation data sets are used as test data sets. The simulation data sets are transformed from some deterministic data sets. Specific work is as follows:

Simulation data set 1: firstly, a deterministic data set of size 100000 to 500000 is randomly generated. The data set contains a number of clusters of not convex shape and has class drifting over time unit. The number range of dimensions of the data set is between 2 to 40. For each data point in the data set, a decimal number between 0 and 1 is randomly created as the probability of the existence of the data point. Through above actions, the deterministic data set is transformed to the uncertain data set.

Simulation data set 2: a series of centers of the cluster and the corresponding radius of the cluster are arbitrarily selected. The radii of clusters are adjusted by plus or minus ε to simulate drifting of the clusters. In the cluster, the data distribution roughly follows the Gaussian distribution. 600000 records are totally generated and are transformed to the uncertain data set by the above method.

The parameters of the algorithm GCUDS are set as follows: $D_m=3$, $D_l=1$, $\lambda=3$, the gravitational coefficients $k_1=1.0$, $k_2=1.0$.

4.2. Analysis of Clustering Effect

To evaluate the performance of the algorithm proposed by this paper, FDCUS and Emicro are chosen as algorithms of comparison. Since the scopes of applicability of the two algorithms are different, different algorithms use different test data sets. For FDCUS, the simulation data sets 1 of 60k which contain 5000 abnormal points and 4 not convex shape of clustering is used to measure the clustering performance of FDCUS and GCUDS. The simulation data set 2 is selected to test the clustering effect of Emicro and GCUDS, because Emicro algorithm is based on the algorithm K-mean and cannot mine the cluster of not convex shape.

The clustering quality evaluation function Q [13] is used to evaluate the clustering quality of the process stages of FDCUS and GCUDS over data stream .The clustering quality is shown in Figure 1 where the horizontal axis represents the progress of the data stream and the vertical axis indicates the clustering quality. From Figure 1, the clustering quality of two algorithms is steadily growing with the data stream going forward and the performance of GCUDS is better than FDCUS. FDCUS neglects data points around the grid cell and classification of the data points is not accurate. In addition, it is too simple to handle the boundary grid cells in FDCUS and the clustering effect is negatively affected. GCUDS can use the trend data degree to describe the data points around the grid and employthe grid gravity to maintain the marginal grid cells. Therefore, GCUDS performs better than FDCUS.

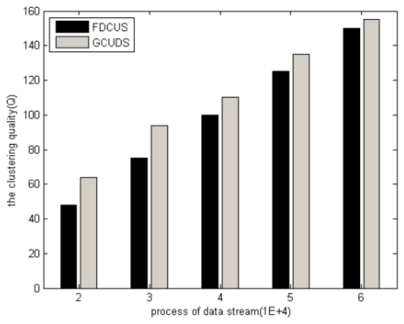

Figure 1. The Clustering Quality of FDCUS and GCUDS

Secondly, the clustering quality of Emicro cannot be assessed by the above quality evaluation criteria and the sum of the squares of distance (SSQ) [8] which is a method to describe the clustering quality cannot evaluate the cluster of existence probability.Therefore, the average cluster quality (AQ) [2] is employed as the measurement method to compare Emicro and GCUDS in this paper. The experiment result is shown in Figure 2 where the horizontal axis represents the progress of the data stream and the vertical axis indicates the average quality. In Figure 2, GCUDS in the paper has better clustering quality than Emicro. GCUDS maps the new arriving data points to the grid cells according to the trend data degree and it helps to improve the distribution precision of data points. GCUDS also introduces the grid gravity to handle the marginal grid cells and evolves the clusters to update the clustering results at regular intervals. The measures that GCUDS takes are beneficial to the clustering quality. However, for Emicro the initial clustering directly affects the clustering results and deletion of global outliers also has a negative effect on the clustering quality.

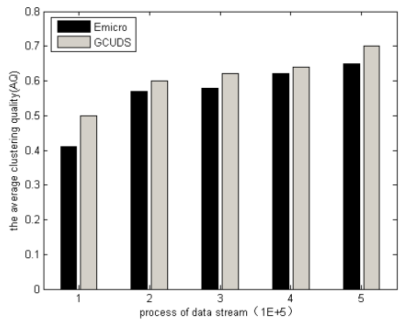

Figure 2. Comparison Emicro with GCUDS on the Clustering Quality

4.3. Analysis of Clustering Time

In order to further assess the performance of GCUDS in this paper, two kinds of data streams of different shapes is used to compare the running time of the algorithms.

The clustering time of FDCUS with GCUDS over two data streams is shown in Figure 3 where the horizontal axis represents the amount of data and the vertical axis indicates the clustering processing time. The experiment result from Figure3 indicates that the clustering time of the two algorithms has a linear growth with the growth of the amount of data and FDCUS takes longer time than GCUDS. FDCUS needs to update a large number of grid cells in the periodical adjustment and spends a great deal of time. But GCUDS takes different strategies of micro-cluster adjustment to update different buffers by the grid hierarchy and reduces some unnecessary computation time. Therefore, the clustering time of GCUDS is super to that of FDCUS.

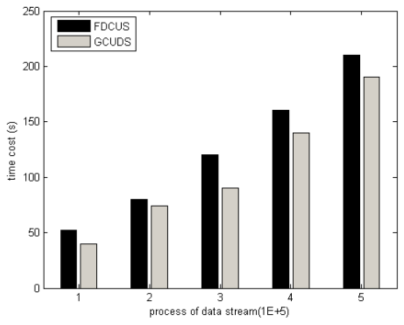

Figure 3. The Clustering Time of FDCUS and GCUDS

Figure 4 is comparison of the clustering time of Emicro and GCUDS. As shown as Figure 4, the clustering time of the two algorithms has a linear growth with the amount of data and GCUDS takes shorter time than Emicro. Emicro is the clustering algorithm based on K-mean and needs to spend a lot of time in calculating distance. GCUDS is the clustering algorithm based on the grid and does not require the time cost of calculating distance. Therefore, the clustering time of GCUDS is lower than Emicro.

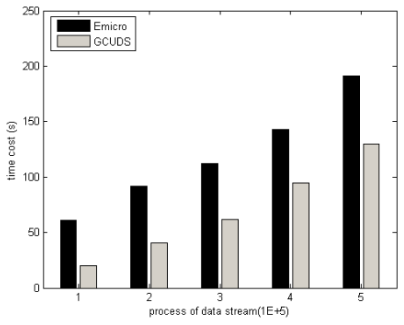

Figure 4. The Clustering Time of Emicro and GCUDS

4.4. The Scalability of GCUDS

Figure 5 shows the scalability of GCUDS on simulation data set 1which is the data set with not convex shapes. In Figure 5, the horizontal axis represents dimensions and the vertical axis indicates the clustering processing time. It is shown in Figure 6 that the scalability of GCUDS on simulation data set 2, which is a data set with ball shapes. The horizontal axis of Figure 6 stands for dimensions and the vertical axis expresses the clustering processing time.

In Figure 5, the size of simulation data set 1 changes from 100,000 to 500,000 and the dimension range is from 10 to 50. It is seen in Figure 5 that the time of GCUDS smoothly grows when the size and dimensions of the data set increase. Therefore, GCUDS has a good scalability in clustering on the data set with non- ball shapes.

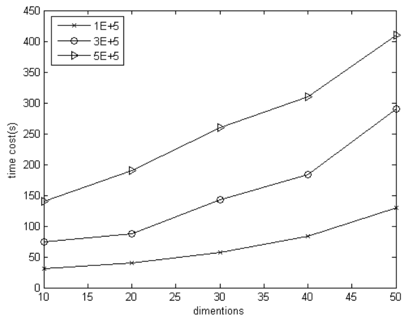

Figure 5. The Scalability of GCUDS on Simulation Data Set 1

In Figure 6, the size of simulation data set 2 changes from 100,000 to 500,000 and the dimensions range is from 10 to 50. As shown in Figure 6, the time of the algorithm in this paper has steady growth, with the increase of the size of the data set. And with the increase of the dimensions, the processing time has steady growth too. This algorithm also has good scalability in ball shape data sets clustering.

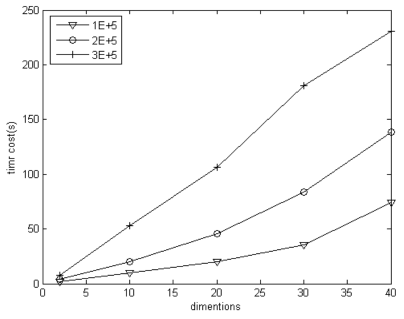

Figure 6. The Scalability of GCUDS on the Simulation Data Set 2

The above experimental results show that GCUDS algorithm has good scalability both in the ball shape data set and in the non-ball shape data set. GCUDS is the clustering algorithm based on the grid density and saves time because it need not spend a lot of time in calculating the distance between the tuples. It also adopts the grid structure where the data space can be easily redrawn and data mapping is not affected. As a result, GCUDS has good performance in scalability.

5. Conclusion

In recent years, with the generation of uncertain data streams, the research on mining technology of uncertain data stream is becoming more and more important. In this paper, the algorithm GCUDS is proposed to solve clustering problems of uncertain data stream. GCUDS is based on the grid to find clusters of arbitrary shapes in the data stream. GCUDS creates the data trend degree to refine on the defects of traditional algorithms based on the grid density in which the data points' weights in the grid cell are treated as the density of the grid cell. In addition, GCUDS provides a new online clustering framework and three micro cluster buffers and they help to improve the accuracy of clustering. Compared with the existing clustering algorithms, experiment results show that GCUDS not only has outstanding performance in clustering result and scalability, but also can handle the high dimensional data streams. In the future research work, adjustment strategy of parameters, clustering quality improvement will be studied.

Acknowledgements

This paper is supported by the Guizhou Science and Technology Major Project (2013)6019), Key Project of Guizhou Province Programs for Fundamental Research and Development (No. [2014] 2001), Key Science and Technology Program of Guizhou Province (No. (2014)3004) and Guizhou Natural Science Foundation (No. [2011] 2196).

References

[1] B. Babcock, S. Babu, M. Datar, R. Motwani and J. Widom, "Models and issues data stream systems", Proceedings of the 21st ACM SIGACT- SIGMOD-SIGART Symp, Madison, **(2002)**, pp. 1-16.
[2] M. Guo, S. Yang, H. Yan, L. Kan and B. Yang, "Review of Computer Engineering Studies", vol. 2, no. 1, **(2014)**.
[3] A. Zhou, F. Cao and W. Qian, "Knowledge and Information Systems", vol. 2, no. 15, **(2008)**.
[4] C. Xing and P. Wen, "Application Research of Computers", vol. 1, **(2015)**.
[5] Y. Peng, Q. Luo and X. Peng, "Chinese Journal of Scientific Instrument", vol. 1, no. 31, **(2010)**.
[6] R. Jin, L. Hong, C. Wang, L. Wu and W. Si, "Review of Computer Engineering Studies", vol. 3, no. 2, **(2015)**.
[7] Z. Liu, Y. Yang and J. Zhang, "Journal of Integrative Plant Biology", vol. 11, no. 51, **(2014)**.
[8] C. C. Aggarwal and P. S. Yu, "A Framework for Clustering Uncertain Data Streams", Proceedings of the 2008 IEEE 24th International Conference on Data, USA, **(2008)**, pp. 150-159.
[9] C. Zhang, M. Gao and A. Zhou, "Tracking High Quality Clusters over Uncertain Data Streams", Proceedings of ICDE'09.IEEE25th International Conference on, shanghai, **(2009)**, pp. 1641-1648.
[10] D. Dai, G. Zhao and S. Sun, "Journal of Software", vol. 5, no. 20, **(2009)**.
[11] T. L. Thanh, L. Peng and Y. Diao, "Journal-the International Journal on Very Large Data Bases", vol. 5, no. 21, **(2012)**.
[12] H. P. Kriegel and M. Pfeifle, "Density-Based Clustering of Uncertain Data", Proceedings of the Fifth IEEE International Conference on Data Mining, Houston, **(2005)**.
[13] T. Li, M. Wu and L. Yang, "Clustering Algorithm on Uncertain Stream based on Time-fading Model", Journal of Chinese Computer Systems, vol. 9, no. 35, **(2014)**.
[14] H. Xu and G. Li, "Density-Based Probabilistic Clustering of Uncertain Data", Proceedings of the 2008 International Conference on Computer Science and Software Engineering, Washington, **(2008)**.
[15] A. Zhou, C. Jin, G. Wang and J. Li, "Journal of computers", vol. 1, mo. 32, **(2009)**.

Authors

Tang Xianghong, received his M.S and Ph.D. degrees in computer science from Huazhong University of Science and Technology, China. He is currently an associate professor in Key Laboratory of Advanced Manufacturing Technology of Ministry of Education at Guizhou University, China. His research interests include intelligent manufacturing, data mining and machine learning.

Yang Quanwei, received his B.S. degrees in mechanical engineering from Wuhan Bioengineering Institute, China. Now he is a master candidate in mechanical engineering at Guizhou University, China. His research interests include intelligent manufacturing and data mining.

Zheng Yang, received his B.S. degrees in mechanical engineering from University of South China, China. Now he is a master candidate in mechanical engineering at Guizhou University, China. His research interests include intelligent manufacturing and machine learning.