International Journal of Database Theory and Application
Vol.8, No.1 (2015), pp.87-94
http://dx.doi.org/10.14257/ijdta.2015.8.1.10

# Improvement of Thinking Theme Discovery Algorithm on Density-Based Clustering

Xuedong Gao[1], Lei Zou[1] and Zengju Li[2]

[1]*University of Science and Technology Beijing, Beijing, P.R.China, 100083*
[2]*Bank Card Test Center, Beijing, P.R.China, 100070*
*gaoxuedong@manage.ustb.edu.cn; zouleino_1@126.com; liecas@sina.com*

***Abstract***

*In traditional data mining process, the definition of mining objects and analysis tasks are all decided artificially based on the analysts' knowledge and experience. To achieve intelligent data analysis, a method called thinking theme discovery technology is proposed to imitate humans' thinking models. Since traditional thinking theme discovery algorithm is based on hierarchical clustering, the efficiency of which is far from acceptable with the increasing of data amounts. This paper improves the efficiency of the algorithm on density-based clustering method. With five complex network datasets and one commercial theme dataset, the experimental results show that both the effectiveness and efficiency of the algorithm are improved.*

***Keywords:*** *thinking theme discovery, analysis question clustering, density-based clustering, hierarchical clustering*

## 1. Introduction

With the rapid development of information technology, the explosively increasing data make humans get into the embarrassment of rich data but lack of information. To obtain useful and potential information and knowledge, large amounts of excellent algorithms in artificial intelligence, machine learning, data warehouse, statistics and data mining are proposed. Among traditional data mining technology, the definition of mining objects and analysis tasks are all decided artificially according to the analysts' knowledge and experience, and the whole data mining process from problems discovering to problems solving has to be involved by humans, which have great effect on the mining results. The development of intelligent mining experts and analysts' potential knowledge and experiences has become a necessity and a hot research focus [5], which will reduce the data analysis technologies' reliabilities on humans, and also the artificial effect on the mining results.

Some researchers have proposed a thinking process discovery method to make it possible to imitate human's thinking process and mine human's thinking models. In the thinking mining process, classical clustering methods [6] consist of partition clustering method, hierarchical clustering method, density-based clustering method, grid-based clustering method, model-based clustering method and method for high-dimensional clustering [1]. Traditional thinking theme discovery algorithm based on analysis questions clustering adopts to hierarchical clustering method. With the increasing of the data amount, the efficiency of hierarchical clustering method [2] is very low, which of density-based clustering method is relatively high, and the shapes of clustered classes are random. The distribution of objects in the class is also random and will not be influenced by outliers [3, 7]. The paper improved the thinking theme discovery algorithm on density-based clustering method. The experiment results show that both the effectiveness and efficiency of the algorithm were improved, while the definition of the parameters [8] in the algorithm needs further research.

ISSN: 2005-4270 IJDTA

## 2. Thinking Theme Discovery Algorithm Based on Hierarchical Clustering

The key problems of thinking process mining include thinking sequences construction, thinking theme discovery-updating and frequent thinking sequence pattern discovery [4]. The research focus of the paper is thinking theme discovery algorithm.

### 2.1. Basic Concepts

Definition 1: Analysis Question(*AQ*)

Analysis question is the recognition on a problem. Analysis question is the preliminary clustering of thinking sequences. *AQ* consists of four elements: thinking sequence set, analysis question count set, analysis question feature, analysis question feature vector. It is defined as *AQ*=<*AQTS*, *AQCS*, *AQF*, *AQFV*>. While *AQTS* is short for analysis question thinking sequence set, AQCS short for analysis question count set, *AQF* short for analysis question feature, *AQFV* short for analysis question feature vector.

Definition 2: Analysis Question Thinking Sequence Set(*AQTS*)

*AQTS* is the set of all thinking sequences making up of analysis question. *AQTS* can continuously include new thinking sequence or thinking sequence set.

Definition 3: Analysis Question Count Set(*AQCS*)

*AQCS* records all concepts frequencies in thinking sequences of *AQTS*. When new thinking sequence is included in *AQ*, the concept frequency of related concepts in *AQCS* adds one. Add the new concept to *AQCS* and set the concept frequency of new concept to one. If many analysis questions are combined as one analysis question, the combined *AQCS* is the corresponding concepts summation.

Definition 4: Analysis Question Feature(*AQF*)

*AQF* is the sequence of feature concepts which represent the main contexts of an analysis question. It is defined as $AQF=<c_1,c_2,\ldots,c_n>$, while $c_i$: $1 \le i \le n$ meets the condition of $\text{frequency}(c_i) = (AQCS_{c_i}/m) \ge \alpha$. $\alpha$ is the threshold of analysis question feature, and m is the number of thinking sequences in an *AQTS*.

Definition 5: Analysis Question Feature Vector(*AQFV*)

In the analysis question feature vector, every component represents the different influences of the concept on analysis question feature. The specific computation method can refer to reference [5].

Definition 6: Thinking Theme(*TKT*)

*TKT* is the abstract and summarization of problems on the basis of analysis questions. It's the clustering results of analysis questions. *TKT* is defined as *TKT*=<*AQS*, *AQFS*, *AQFVS*>, while *AQS* represents analysis question set, *AQFS* represents analysis question feature set and *AQFVS* represents analysis question feature vector set.

### 2.2. Similarity Computation

The key of thinking theme discovery algorithm based on analysis question is the computation of similarity. According to the basic idea of thinking theme discovery, three similarity computation equations are given as follows.

(1)Thinking Sequence Similarity(*TSSIM*)

*TSSIM* refers to the similarity between thinking sequence *TS* and analysis question *AQF*. It is used to judge whether a thinking sequence *TS* fits the analysis question feature

 

*AQF* and whether the thinking sequence *TS* can be divided into the analysis question *AQ*. The equation is defined as follows:

$$TSSIM(TS, AQF) = \frac{|TS \cap AQF|}{|TS|} \quad (1)$$

Where $|TS|$ means the number of concepts included in thinking sequence *TS*, and $|TS \cap AQF|$ means the number of common concepts in thinking sequence *TS* and analysis question *AQF*.

(2)Analysis Question Similarity(*AQSIM*)

*AQSIM* measures the similarity between two analysis questions. Analysis question *AQ* is represented with analysis question feature vector. *AQSIM* can be defined as the similarity between analysis question feature vectors. Since the analysis question feature vector has been normalized, the similarity between $AQ_i$ and $AQ_j$ can be defined as:

$$AQSIM(AQ_i, AQ_j) = feature\_zero_{AQ_i} \cdot feature\_zero_{AQ_j} \quad (2)$$

Where $feature\_zero_{AQ_i}$ represents the normalized feature vector of analysis question $AQ_i$, and $feature\_zero_{AQ_j}$ represents the normalized feature vector of analysis question $AQ_j$. The specific computation method can refer to reference [5].

(3)Thinking Theme Similarity(*TKTSIM*)

*TKTSIM* measures the similarity between two thinking themes. The similarity between $TKT_i$ and $TKT_j$ can be defined as:

$$TKTSIM(TKT_i, TKT_j) = max_{AQ_m \in TKT_i, AQ_n \in TKT_j}\{AQSIM(AQ_m, AQ_n)\} \quad (3)$$

The equation shows that the similarity between two thinking themes is the maximum of *AQSIM* among two thinking themes.

### 2.3. Algorithm Introduction

The thinking theme discovery algorithm based on analysis questions includes two sections: analysis question generation and thinking theme discovery. The details of the two algorithms will be introduced as follows. Our research focus is the second section that is the improve-ment of thinking theme discovery algorithm.

(1) Analysis Question Generation Algorithm

Analysis question generation algorithm contrast the thinking sequence to be clustered with every analysis question feature in analysis question set. Find out the best matching analysis question and put the thinking sequence into the analysis question set. Update the analysis question count set and analysis question feature to finish a circle. After finishing analysis questions generation, compute all analysis question feature vectors to prepare for the discovery of thinking themes. The details of analysis question generation algorithm are as follows:

Inputs: thinking sequence set *TSS*, concept hierarchy weight table *CHWT*, the threshold of concept feature $\alpha$, the threshold of analysis question similarity $\beta$.

Outputs: analysis question set *AQ*.

Step 1: Import thinking sequence set *TS* and concept hierarchy weight table *CHWT*;

Step 2: Initialize analysis question set *AQ*;

Step 3: Begin with the thinking sequence $TS_i$ in thinking sequence table. Compute the similarities $TSSIM(ts_i, AQF_j), j = 1,2,\dots,|AQ|$; If the maximum of $TSSIM(ts_i, AQF_j)$ is bigger than the threshold $\beta$, then put the thinking sequence $TS_i$ into the corresponding analysis question. Update the analysis question count set and analysis question feature. Otherwise define a new analysis question and put the analysis question into analysis question set. Repeat step 3 until all the thinking sequences in thinking sequence set *TSS* are clustered into analysis question set *AQ*.

Step 4: Compute every analysis question feature vector *AQFV*.

Step 5: Output the analysis question set *AQ*.

(2)Thinking Theme Discovery Algorithm

First regard every analysis question as a thinking theme. According to the analysis question feature vector obtained from analysis question generation algorithm, compute the thinking theme similarity with cosine similarity. Then, cluster the analysis questions into thinking themes with hierarchical clustering method. The details of thinking theme discovery algorithm based on hierarchical clustering method are as follows:

Inputs: analysis question set *AQ*, the threshold of thinking theme similarity $\mu$

Outputs: thinking theme set *TKT*.

Step 1: Initialize thinking theme set *TKT*. Define every analysis question $AQ_i$ as a thinking theme. Put all the thinking themes into thinking theme set *TKT*.

Step 2: Compute the thinking theme similarity matrix.

Step 3: Find out the maximum of thinking theme similarity. If the thinking theme similarity is bigger than the threshold $\mu$, then combine the two thinking themes whose similarity is the greatest into one thinking theme and update the combined thinking theme.

Step 4: Repeat step 2,3 until all the thinking themes are combined as one thinking theme or the greatest thinking theme similarity between thinking themes is less than the threshold $\mu$.

Step 5: Output thinking theme set *TKT*.

Above thinking theme discovery algorithm is based on hierarchical clustering. The time efficiency of the algorithm is very low, especially when the analysis question number is very large. To improve the efficiency of the algorithm, the thinking theme discovery algorithm based on density is proposed, which is called *TTDD* for short in the following.

## 3. Thinking Theme Discovery Algorithm on Density-Based Clustering

Traditional density-based clustering method has two parameters. So one of the parameters of *TTDD* algorithm is the radius ε, which represents the similarity threshold between the thinking themes, and the other one is the minimal number of neighbors *MinPts*, which means if the number of analysis questions whose similarities with some analysis question is not less than ε is at least *MinPts*, then the analysis question is called a core analysis question. The specific descriptions of *TTDD* algorithm are as follows:

Step 1: Initialize thinking theme sets.

Step 2: Compute analysis question similarity matrix and confirm the core analysis question set according to the definition of the parameter *MinPts*.

Step 3: Start clustering with the first element of core analysis question set.

Step 4:

 

Step 4.1: Cluster the core analysis question with other analysis questions whose similarities with the core analysis question is not less than ε into one class, and delete-update the core analysis question set. Then, traverse all the analysis questions whose similarity with the core analysis question is not less than ε.

Step 4.2: If the traversed analysis question is a core analysis question, then jump to step 4.1; If the analysis question is not a core analysis question, then traverse all the analysis questions whose similarities with it is not less than ε. If the traversed analysis question is a core analysis question, then jump to step 4.1 until all the traversed analysis questions are not core analysis questions.

Step 5: Repeat step 3,4 until the core analysis question set is empty.

Step 6: Output the thinking theme sets.

## 4. Experimental Results Analysis

### 4.1. Experimental Data and Environment

In the experiment, we use six datasets: *karate, dolphins, football, netscience, adjnoun and commodity*. All the datasets are complex network datasets except the *commodity* dataset. Although these complex network datasets in fact are completely different from the mining of thinking process, their data structures conform to analysis questions data very well. These datasets can be used for the algorithm test. The commodity dataset is obtained by interviewing an expert to extract his concept pairs. The experiment was finished on PC configuration as: CPU, Intel(R) Core(TM) i3-3110M CPU @ 2.40GHZ 2.40GHZ; RAM, 4G; OS, Windows 7; IDE, Eclipse-SDK-3.7.2-win32 JDK 1.7.0.

To testify our algorithm *TTDD*, we did two groups of experiments. The first experiment focused on the clustering effectiveness analysis. Since the commodity dataset was based on practical commercial theme concept pairs, the experimental results of the dataset can be used to explain the significance of clustering results. The other experiment focused on the efficiency comparison between traditional thinking theme discovery algorithm and *TTDD* algorithm.

Table 1 shows the datasets which are sorted by orders of number of analysis questions. The two algorithms are contrasted on two indexes of number of thinking themes which reflects the algorithm effectiveness and time of thinking theme discovery which reflects the algorithm efficiency. The two parameters of the experiment is set as follows: $\varepsilon$=0.3, *MinPts*=3. The details of the experimental results will be discussed in the following.

**Table 1. Number of Analysis Questions**

| Dataset | Number of Analysis Questions |
|---|---|
| *karate* | 13 |
| *commodity* | 18 |
| *dolphins* | 61 |
| *netscience* | 800 |
| *football* | 1002 |
| *adjnoun* | 1478 |

### 4.2. Experimental Effectiveness Analysis

Table 2 shows the experimental results of traditional hierarchical clustering algorithm and *TTDD* algorithm. We choose the practical commercial theme dataset commodity to

explain the significance of the clustering results. The clustering result of *TTDD* algorithm is shown in Figure 1. As can be seen from Table 2, eventually we got 11 clustered thinking themes. Taking only one thinking theme for example, as can be seen from Figure 1, we cluster different analysis questions containing concepts cost, material cost, transport cost and so on into one thinking theme. In fact, the concepts have relative meanings in real life, so we cluster them into one thinking theme named "cost" is reasonable with our *TTCD* algorithm. The clustering result of commodity dataset proves the effectiveness of our *TTCD* algorithm.

Furthermore, the *TTDD* algorithm obtains more thinking themes than traditional algorithm. That's because original algorithm clusters the analysis questions into one class as long as the maximum of similarities between the analysis question with analysis questions in existing thinking theme is not less than the threshold, while the *TTDD* algorithm clusters the analysis questions into one class only if all the similarities between the core analysis question and other analysis questions are not less than the threshold $\varepsilon$. The *TTDD* algorithm is more reasonable than original algorithm, so it usually generates more thinking themes than the original one.

**Table 2. Number of Thinking Themes Constructed by Thinking Theme Discovery Algorithm**

| Number of Thinking Themes / Datasets | Traditional Algorithm | *TTDD* Algorithm |
|---|---|---|
| karate | 8 | 11 |
| commodity | 11 | 11 |
| dolphins | 19 | 38 |
| football | 5 | 27 |
| adjnoun | 13 | 34 |
| netscience | 394 | 506 |

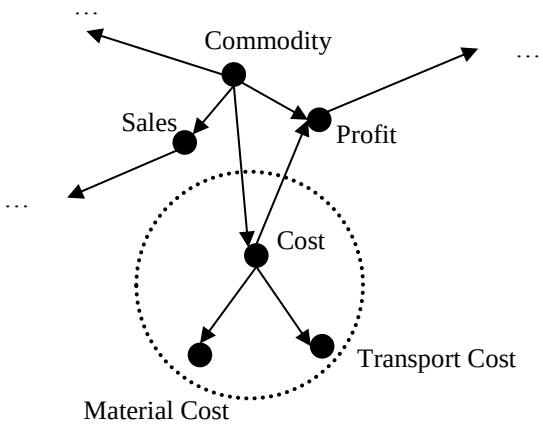


**Figure 1. The Clustering Result of Commodity Dataset**

### 4.3. Experimental Efficiency Analysis

Figure 2 shows the efficiency of two algorithms with the increasing of number of analysis questions. The diamond point line represents the efficiency of traditional algorithm, while the square point line represents that of improved *TTDD* algorithm. As

 

can be seen from the figure, the efficiency of thinking theme discovery is highly improved with *TTDD* algorithm. With the increasing of number of analysis questions, the improvement of the algorithm efficiency is more apparent.

The experiment results show that the efficiency of *TTDD* algorithm is higher than the traditional algorithm. Furthermore, since the parameters are decided artificially, *TTDD* algorithm has one less parameter than the original algorithm, which will reduce the effect of artificial factors. At the same time, the *TTDD* algorithm also doesn't need to compute the similarity of thinking themes, which will also improve the efficiency of thinking theme discovery.

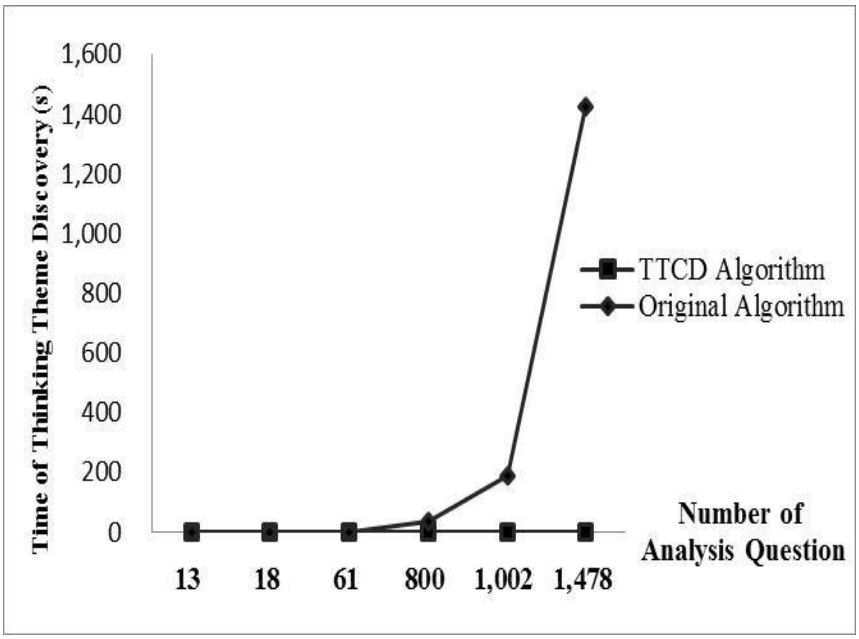


**Figure 2. Time of Thinking Theme Discovery with Two Algorithms**

## 5. Conclusion

This paper improved the thinking theme discovery algorithm on density-based clustering method. The experimental results show that the clustering results of the *TTDD* algorithm are more reasonable, which means the effectiveness of the thinking theme discovery is improved. Furthermore, the efficiency of the algorithm for thinking theme discovery is also improved obviously, especially with the increasing of number of analysis questions. While subject to the datasets and the *TTDD* algorithm, the parameters are very small, and the two parameters which need to be set artificially have obvious effect on the results of clustering. Further research on the set of parameters will be need. Also the explanations of the clustering results of abstract big datasets will still be a problem.

## References

[1] R. Xu and C. Donald, "Survey of Clustering Algorithms", IEEE T. Neural Network, vol. 16, no. 3 **(2005)**.
[2] C. Stephen, "Hierarchical Clustering Schemes", Psychometrika, vol. 32, no. 3, **(1967)**.
[3] Y. Zhao, Y. Cao, J. Zhang and W. Lu, "Cluster of Nucleic Acid Sequences Based on Density K-medoids Method", Computer Engineering, vol. 32, no. 19, **(2006)**.
[4] X. C. Chen, "Thinking Theme Discovery Method Based on Thinking Sequence Clustering", China Management Informationization, vol. 15, no. 12, **(2012)**.
[5] X. C. Chen, "Technology of Thinking Processes Discovery for Data Mining Application", University of Science and Technology Beijing, Beijing, **(2012)**.

[6] X. Tang, "Study on Clustering Algorithm and Its Application", University of Electronic Science and Technology of China, Sichuan, **(2010)**.
[7] S. Wu, X. D. Gao and M. Bastian, "Data Warehousing and Data Mining", Metallurgical Industry Press, Beijing, **(2003)**.
[8] H. Bao, "A Dynamic Parameter Selection Solution in Density Based Clustering Algorithm", Shandong University, Shandong, **(2005)**.

 