TELKOMNIKA, Vol. 11, No. 8, August 2013, pp. 4477~4483
e-ISSN: 2087-278X

An Algorithm on Generating Lattice based on Layered Concept Lattice

Zhang Chang-sheng[1,2,3], Ruan Jing[4], Huang Hai-long*[3], Li long-chang [3], Yang Bing-ru[1,2]
[1]School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing, 100083, China
[2]Beijing Key Laboratory of Knowledge Engineering for Materials Science, Beijing 100083, China
[3]College of Physics & Electronic Information Engineering, Wenzhou University, Zhejiang, 325035, China
[4]Wenzhou Vocational & Technical College, Zhejiang, 325035, China
Corresponding author, e-mail:jsj_zcs@126.com, ruanjing1979@126.com, 156732998@qq.com, jsj_llc@wzu.edu.cn, bryang_kd@126.com

Abstract

Concept lattice is an effective tool for data analysis and rule extraction, a bottleneck factor on impacting the applications of concept lattice is how to generate lattice efficiently. In this paper, an algorithm LCLG on generating lattice in batch processing based on layered concept lattice is developed, this algorithm is based on layered concept lattice, the lattice is generated downward layer by layer through concept nodes and provisional nodes in current layer; the concept nodes are found parent-child relationships upward layer by layer, then the Hasse diagram of inter-layer connection is generated; in the generated process of the lattice nodes in each layer, we do the pruning operations dynamically according to relevant properties, and delete some unnecessary nodes, such that the generating speed is improved greatly; the experimental results demonstrate that the proposed algorithm has good performance.

Keywords*: concept lattice batch processing, layering, hasse diagram, CONCEPT LATtice*

1. Introduction

In the applications of the concept lattice, the first solution is to construct the lattice. Therefore, it is important to study the generating lattice algorithms of concept lattice. There are many literatures on the construction algorithms of concept lattice. The generating methods of concept lattice can be divided into two categories: batch processing algorithms and incremental construction algorithm. Many classical algorithms on batch processing, such as: Bordat [1], Osham [2], Chein [3], Ganter [4], Nourine [5] and other batch processing algorithms. Scholars have done some improvements of the above algorithms and done a wide range of applications, such as in the literature [6-10], there are some improvements on classical algorithms, the authors in the literatures [11-13] have proposed the association rules algorithms based on concept lattice, some classification and clustering algorithms based on concept lattice are proposed in the literatures [14-15], and the links between the concept lattice and rough set are studied in the literatures [16-17].

Algorithm Bordat is a typical effective algorithm on batch processing, in this algorithm, concept lattice L is constructed from the top node, which is generated all of its child nodes for each node, and complete the connection between the child nodes to the parent node, then call for the recursive process of each child node. The basic idea of the algorithm is: if the current node is (O_1, D_1), D is a set of attributes. We need to find the subset of attributes $D_2 \subseteq D\text{-}D_1$, such that D_2 is able to maintain the characteristics of the complete binary group, i.e. it is the maximum expansion. Then $D_1 \cup D_2$ constitutes the connotation of a child node for the current node. The algorithm is simple, intuitive, and easy to parallelize. Its disadvantage is that it repeatedly generates the same node. It can be observed that each node is generated his father so many times.

From the construction process of Bordat algorithm, it can see that although Bordat algorithm is also relates to the hierarchy of concept, Bordat algorithm does not have a clear hierarchical division, it just constructs the concept lattice in a top-down process, using the manner of generating child nodes to build lattice. In Bordat algorithm, it can not confirm that

Received January 24, 2013; Revised May 14, 2013; Accepted May 24, 2013

what level of the concept, which concept belongs to the same level, and it just rely on the parent-child relationship between the nodes to confirm the upper-lower relationships between some particular nodes. In Algorithm Bordat, the generated sequence of concept is similar to the depth-traversal order of the tree, such that it can only determine the direct precursor-successor relationship between the nodes in level. Therefore, its level is localized.

But in the practical applications, the spatial and temporal complexities of many generating lattice algorithms based on concept lattice need to be improved, in this paper, on the basis of Algorithm Bordat, an algorithm LCLG on generating lattice in batch processing based on layered concept lattice is developed, this algorithm is generating lattice in batch processing, on the layer basis of the cardinalities of the attributes in concept lattice, do the pairwise operations between the concept nodes in current layer or the concept nodes and the provisional nodes to generate the nodes in next layer, and do the pruning operations dynamically according to relevant properties, and delete some unnecessary nodes, such that the generating time is reduced greatly. The experimental results demonstrate that the efficiency of the proposed algorithm is prior to Algorithm Bordat.

2. Related Definitions

Definition 1. A formal context is a triple K =(G ,M ,*I*), where G is a set of objects, M is a set of attributes, I is a binary relation between the G and M, i.e. $I \subseteq G \times M$, g*I*m denotes it exists a relationship *I*.

Definition 2. In the formal context K, a binary group (A,B) from G×M exists the following two properties:

(1) $B = f(A)$, where $f(A) = \{m:(m \in M) \wedge (\forall g \in A \subseteq G, gIm)\}$;

(2) $A = g(B)$, where $g(B) = \{g :(g \in G) \wedge (\forall m \in B \subseteq M, gIm)\}$.

In the formal context K, (A, B) is called as a concept, where B is referred to as intent of the concept (Intent), and A is called the extension of the concept (Extent). Each concept is a node of concept lattice, the maximum element of lattice is (G,f(G)), the minimum element of lattice is (g(M),M).

Definition 3. A partial order relation between the concept nodes is established. Given $C_1=(A_1, B_1)$ and $C_2=(A_2,B_2)$,then $C_1>C_2 \Leftrightarrow B_1 \subset B_2 \Leftrightarrow A_1 \supset A_2$, the leading order means C_1 is the parent node of C_2 or the generalization. If concepts $C_1=(A_1, B_1)$ and $C_2=(A_2, B_2)$ satisfy $A_2 \subset A_1$, and there does not exist the concepts (A,B) such that $A_2 \subset A \subset A_1$; then C_1 is called the direct super-concept of C_2, C_2 is a direct sub concept of C_1, referred to as $(A_1, B_1)>(A_2, B_2)$. The linear diagram of concept lattice is generated based on the partial order relation, which is Hasse diagram.

Definition 4. In the form context K=(G,M,*I*), the relationship between the object $g \in G$ and the property $m \in M$ is $(g,m) \in I$ or $(g,m) \notin I$. A limited form background can be represented by a matrix, if $(g,m) \in I$, we use digita 1 to represent in the matrix; if $(g,m) \notin I$, we use digita 1 to represent in the matrix, this matrix is said to be a transaction matrix T in formal context K.

Definition 5. Let K = (G, M, *I*) be a form context, for an attribute $y \in M$, in the matrix of K, if y corresponding to the columns has numbers of n equal to 1, we call the rank of the attribute is n, denoted as r(y)=n. Denoted by $m=\max\{r(y)|y \in M\}$, m is the rank of form context.

Definition 6. In the concept lattice, if B in the concept (A, B) contains the number of attributes is K, called (A, B) is the lattice nodes of the K-th layer, the layer is referred to as the L_k.

Definition 7. Redundant node: suppose there exists the node C in the L-th layer, the set of objects is A, and the set of attributes is B. If there exists the node C'=(A',B') in the generated process from the L layer to the L+1 layer node, such that $C \neq C'$, and A=A'or B=B', then the node C is the redundant node in the L-th layer.

Definition 8. Contained node: suppose there exists the node C in the L-th layer, the set of objects is A, and the set of attributes is B. If there exists the node C'=(A',B') in the generated process from the L layer to the L+1 layer node, such that $B' \subseteq B$, then the node C is the contained node in the L-th layer.

The following properties can be obtained from the above definitions.

Property 1. Any one lattice nodes in the N-th layer, which at least is covered by one of the lattice nodes in the N-1-th layer.

Property 2. The concept nodes in L_k layer and other concept nodes or temporary nodes intersect in pair, and generate L_{k+1} layer nodes, where the non-concept nodes are called temporary nodes.

Property 3. For C_1=(a,b) and C_2=(a',b'), if it satisfies a'$\subseteq$a, b'$\subseteq$b, then directly discard C_2.

Property 4. For C_1=(a,b) and C_2=(a',b'), if it satisfies a'=a, b'$\neq$b, then directly discard C_2 after C_1=(a, b$\cup$b').

Property 5. For C_1=(a,b) and C_2=(a',b'), if it satisfies a'$\neq$a, b'=b, then directly discard C_2 after C_1=(a$\cup$a',b).

3. LCLG Algorithm

3.1. The Idea of Algorithm LCLG

In this paper, the designed algorithm LCLG draws the characteristics of Bordat algorithm, it is simple, intuitive and easy-to-parallel computing, and the core designed idea of this algorithm is as follows. For the concept nodes in current k layer and other concept nodes or the temporary nodes in k layer, do the operations as follows: construct the intersection operation of the extension and the combination operation of the connotation for every two nodes, and the results are taken as the extension and the connotation of the newly generated node for k+1 layer; for a new generated node, we do the pruning operation according to Property 3~5 and mark the non-concept nodes as provisional nodes C*; and for each newly generated node, we connect the new generated nodes to the upper concept parent nodes to generate the parent-child relationship, in this step, if the parent node is a temporary node, we continue to look for the parent concept of concept nodes and do the connections; after generating all the nodes in k+1 layer, we delete all temporary node in k layer to free the space up, until finish generating k+1 layer and establish a father and son connection between nodes. Repeat the above steps until it does not to build a new layer, the complete concept lattice is generated, at the same time, the methods of seeking the parent-child relationship of concept nodes up to layer by layer, such that this algorithm can generate the Hasse diagram of inter-layer connection.

3.2. LCLG Algorithm

Input: A formal context K

Output: A corresponding Hasse diagram of the concept lattice L

Step 1: Let formal context K be changed into transaction matrix T', sort in a descending order according to r(m)=n, and generate matrix T(the rows denote transaction records, and the columns denote attributes), and let maximum element C_{max}=(G,ϕ);

Step 2: Generate all the nodes in $L_{k=0}$ layer: for each attribute m$\in$M, to generate the nodes C_m=(g(m),m), directly marked as provisional nodes C_m*, the provisional nodes C_m* and C_{max} are composed of $L_{k=0}$ layer;

Step 3: Generate all the nodes in $L_{k=1}$ layer after the operations of the nodes in $L_{k=0}$ layer: do the pairwise operations between the concept nodes (note that the only concept node in k=0 layer is C_{max}) and other provisional nodes in k=0 layer. The operations are as follows: construct the intersection operation of the extension and the combination operation of the connotation for every two nodes, and the results are taken as the extension and the connotation of the newly generated node for the next layer; for a new generated node at each operation, we do the pruning operations and delete the repeat nodes and the contained nodes in the process of combination; mark the non-concept nodes as provisional nodes C*; and construct the connections for the nodes to C_{max}; to judge the newly generated nodes in $L_{k=1}$ layer whether there exists a node C' and if it satisfies |C'.Extent|=|G|, if it exists, then delete C_{max} and the connections, and let C' be a maximum element, delete all the provisional nodes and relevant connections in k=0 layer, the k=1 layer is completed.

Step 4: Continue to generate all the nodes in k=k+1 layer: do the pairwise operations for the concept nodes in k=k+1 layer or between the concept nodes and the provisional nodes in current layer to generate the extension and the connotation of the nodes in k+1 layer; at the same time, for a new generated node we do the pruning operations and delete the repeat nodes and the contained nodes in the process of combination; mark the non-concept nodes or the nodes with the cardinality of the connotations |C.Intent|>k+1 as provisional nodes; we connect the new generated nodes to the upper concept parent nodes to generate the parent-child

relationship, in this step, if the parent node in k layer is a temporary node, we continue to look for the parent node of concept nodes in k-1 layer and do the connections; after generating all the nodes in k+1 layer, delete all the provisional nodes and the relevant connections in k layer until the k+1 layer is completed and the parent-child connections are constructed.

Step 5: k=k+1;

Step 6:Repeat Step 4-5, until it does not to generate a new node in L_K layer, then generate minimum element and construct the parent-child connections.

Step 7: Output the corresponding Hasse diagram of the concept lattice L.

3.3. Instance of Verification

The formal context K is determined by the front eight things in the transaction database D as shown in Table 1, the fixed number of attributes is 6.

Table 1. Formal Context

	a	b	c	d	e	f
1	1	0	1	1	0	1
2	1	1	1	1	0	1
3	1	0	1	0	0	0
4	1	1	0	0	0	0
5	1	1	0	0	0	0
6	1	1	1	0	0	0
7	1	1	0	1	0	1
8	1	0	1	1	1	0

Example:

Input: the formal context K shown in Table 1

Output: the corresponding Hasse diagram of concept lattice L.

Step 1: Let formal context K be changed into transaction matrix, sort in a descending order according to r(m)=n, and generate matrix T; D=8;C_{max}=(12345678,ϕ); For each m∈M, generate C*=(g(m),m), we can get the set of the generated nodes as follows: {C_{01}*=(12345678,a),C_{02}*=(24567,b),C_{03}*=(12368,c), C_{04}*= (1278,d), C_{05}*= (127,f), C_{06}*= (8,e)}; the provisional nodes C_m* and C_{max} are composed of the nodes in $L_{k=0}$ layer.

	a	b	c	d	f	e
1	1	0	1	1	1	0
2	1	1	1	1	1	0
3	1	0	1	0	0	0
4	1	1	0	0	0	0
5	1	1	0	0	0	0
6	1	1	1	0	0	0
7	1	1	0	1	1	0
8	1	0	1	1	0	1

Figure 1. The Transaction Matrix T after Ordering

Step 2: Generate all the nodes in $L_{k=1}$ layer after the operations of the nodes in $L_{k=0}$ layer as follows: for the concept nodes C_{max} and other temporary nodes C*=(g(m),m) in the current layer $L_{k=0,}$ construct the intersection operation of the extension G∩g(m) and the combination operation of the connotation $\phi \cup$m, and generate new nodes in $L_{k=1}$ layer, and for each newly generated node, we do the pruning operation according to Property 3~5; and judge the node whether is concept node after pruning, the non-concept nodes are marked as provisional nodes. In this step, the set of the generated nodes are as follows:{C_{11}=(12345678,a), C_{12}*=(24567,b), C_{13}*=(12368,c), C_{14}*=(1278,d), C_{15}*= (127,f), C_{16}*= (8,e)}, because C_{11} satisfies |g(a)|=|G|, thus C_{max}=C_{11}; and delete all the provisional nodes and the relevant connections in $L_{k=0}$ layer.

Step 3: Generate all the nodes in $L_{k=2}$ layer after the operations of the nodes in $L_{k=1}$ layer as follows: from the unique concept node and other provisional nodes in $L_{k=1}$ layer, we can get that {C_{21}=(24567,ab), C_{22}=(12368,ac), C_{23}= (1278,ad), C_{24}*= (127,af), C_{25}*= (8,ae)}. Because it does not need to prune nodes in this step, construct the parent-child connections to C_{11}; and delete all the provisional nodes and the relevant connections in k=1 layer.

Step 4: Generate all the nodes in $L_{k=3}$ layer after the operations of the nodes in $L_{k=2}$ layer as follows: from the concept nodes in $L_{k=2}$ layer and other concept nodes or provisional nodes, we can get the set of the nodes and the parent-child connections, where the set of the generated nodes are as follows: {C_{31}=(26,abc),C_{32}*=(27,abd), C_{33}*= (27,abf), C_{34}= (128,acd), C_{35}*= (12,acf), C_{36}*= (8,ace), C_{37}= (127,adf), C_{38}*= (8,ade)}; according to the properties, we combine C_{32}* and C_{33}* be C_{32}*=(27,abdf), though C_{32}* is a concept node, while |C_{32}*.Intent|>3, thus it is not the concept node in current layer, the mark it as provisional node; Similarly, C_{36}*= (8,acde); finally, the set of the nodes in $L_{k=2}$ layer is {C_{31}=(26,abc),C_{32}*=(27,abdf), C_{34}= (128,acd), C_{35}*= (12,acf), C_{36}*= (8,acde), C_{37}= (127,adf)}; and delete all the provisional nodes and the relevant connections in k=2 layer.

Step 5: Generate all the nodes in $L_{k=4}$ layer after the operations of the nodes in $L_{k=3}$ layer as follows. From the concept nodes in $L_{k=3}$ layer and other concept nodes or provisional nodes, we can get the set of the nodes and the parent-child connections, where the set of the generated nodes are as follows: {C_{41}=(12,acdf),C_{42}=(8,acde), C_{43}= (27,abdf), C_{44}*= (2,abcdf)}; delete all the provisional nodes and the relevant connections in k=3 layer.In this step, from the concept node C_{37}=(127,adf) and the provisional node C_{32}*=(27,abdf)*, we generate the concept node C_{43}= (27,abdf), when C_{43} is connected with two parent nodes, we find that the parent node C_{32}* is a provisional node, then continue to find the parent concept node from the above layer, C_{21}=(24567,ab) is constructed the parent-child connections, which is shown as Figure 2.

Step 6: Generate all the nodes in $L_{k=5}$ layer after the operations of the nodes in $L_{k=4}$ layer as follows. From the concept nodes in $L_{k=4}$ layer and other concept nodes or provisional nodes, we can get that {C_{51}= (2,abcdf)}; and delete all the provisional nodes and the relevant connections in k=4 layer.

Step 7: Because it is impossible to generate the nodes for the next layer from the $L_{k=5}$ layer, thus the process of generating lattice is completed, construct the minimum element C_{min}=(ϕ, abcdef), and construct the connections;

Step 8: Output the corresponding Hasse diagram of the concept lattice L.

The corresponding Hasse diagram of the obtained concept lattice is shown in Figure 3.

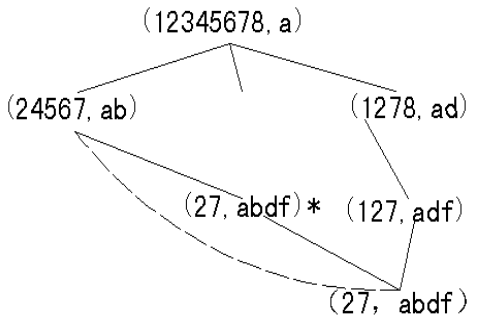

Figure 2. Find the Upper Parent Concept Nodes

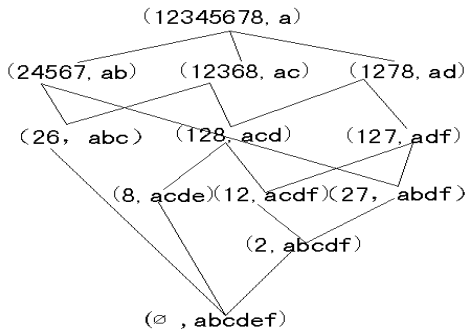

Figure 3. The Corresponding Hasse Diagram of the Concept Lattice L

4. Experimental Analysis

To test the performance of the proposed algorithm, we do three groups experiments to make a comparative analysis for the algorithms Bordat and LCLG. The experimental environment is a Pc machine of a Windows Server 2003 operating system, I7 2.0G 64-bit quad-core eight lines with the processor, 4G memory, and the program runs on the environment of JAVA SDK1.4.2. In this experiment, we use “attribute-attribute with no relevance probability” to describe the relevancy extent between attributes and attributes.

The first experiment: set 10 formal contexts, the numbers of attributes and objects are both fixed as 20, the attribute-attribute with no relevance probability increases from 40% to 85%, the results are shown in Figure 4. We can find that the attribute-attribute with no relevance probability has no impact on Bordat through experimental comparisons, in contrast to Algorithm LCLG; the running efficiency has obvious advantages when the attribute-attribute with no relevance probability is low.

The second experiment: set 10 formal contexts, the attribute-attribute with no relevance probability is fixed as 80%, the numbers of objects is fixed as 20 in each formal context, the numbers of attributes increases from 5 gradually to 50, and the results are shown in Figure 4. From the trend of the figure, when the numbers of attributes is more than 25, the time increases quickly for Algorithm Bordat, while the time slows for Algorithm LCLG.

The third experiment: set 10 formal contexts, the attribute-attribute with no relevance probability is fixed as 80%, the numbers of attributes is fixed as 20 in each formal context, the numbers of objects increases from 5 gradually to 50, and the results are shown in Figure 6. We can find that Algorithm LCLG keep stable advantages in contrast to Algorithm Bordat, especially as the increase of the numbers of objects, the running speed has remarkable advantages. From the experiments, we know that Algorithm LCLG greatly reduces the impact on the increase of attributes and object affected to the time complexity of the algorithm. Compared with Algorithm Bordat, it is more adapted to the more objects in the formal context, and when the attribute-attribute with no relevance probability is low, the advantages of this algorithm is relatively more apparent.

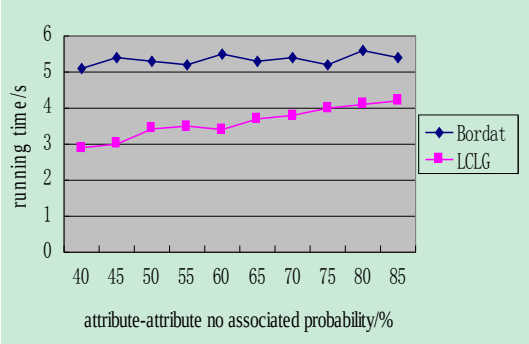

Figure 4. The Trend of the Efficiency of the Algorithm with the Increase of attribute-attribute with No Relevance Probability

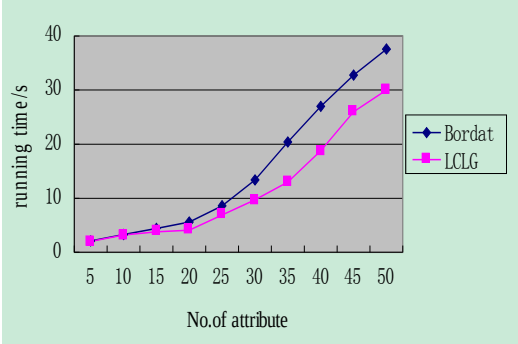

Figure 5. The Trend of the Efficiency of the Algorithm with the Increase of Attributes

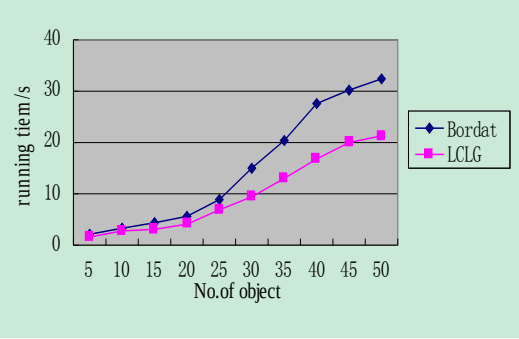

Figure 6. The Trend of the Efficiency of the Algorithm with the Increase of Objects

5. Conclusion

In this paper, the algorithm LCLG on generating lattice in batch processing based on layered concept lattice is developed. The nodes layered in the concept lattice are introduced,

the lattice is generated by width-prior method, do the pairwise operations between the concept nodes in each layer or the concept nodes and the provisional nodes to generate the nodes in next layer, and do the pruning operations dynamically to delete some unnecessary nodes, such that the generating speed is improved greatly, the concept nodes are constructed the parent-child connections upward layer by layer, then the Hasse diagram of inter-layer connection is generated. From the comparative analysis with Algorithm Bordat in the experiments, we know that Algorithm LCLG greatly reduces the impact on the increase of attributes and object affected to the time complexity of the algorithm. And it is more adapted to the situation that the more objects are in the formal context, when the attribute-attribute with no relevance probability is low, the advantages of Algorithm LCLG is relatively more apparent.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (No.60875029, No.61175048); The Project of Beijing Key Laboratory of Knowledge Engineering for Materials Science(No.Z121101002812005).

References

[1] Bordat JP. Calcul pratique du treillis de Galois d'une correspondence. *Match. Sci. Hum.* 1996; 96: 31-47.
[2] Ho TB. An approach to concept formation based on formal concept analysis. *IEICE Trans Information and Systems*. 1995; 78(5): 553-559.
[3] M Chein. Algorithm de recherche des sous-matrices premieres dune matrices. *Bull. Math. Soc. Sci. R. S. Romaine*. 1996; 13(61): 21-25.
[4] Ganter RE. Bowman C M. *Digital Libraries, Conceptual Knowledge Systems and the Nebula Interface*. University of Arkansas. 1995; 125-131.
[5] Nourine L, Raynaud. *A fast algorithm for building lattices*. Workshop on Computational Graph Theory and Combinatorics(CGTC). Victoria. Canada. 1999; 76-84.
[6] R Godin, R Missaoui. Incremental concept formation for learning from Databases. *Theoretical Computer Science*. 1994; (133): 387-419.
[7] Zhi Hui-lai, Zhi Dong-jie, Liu Zong-tian. Theory and Algorithm of Concept Lattice Union. *Acta Electronica Sinica*. 2010; 38(2): 455-459.
[8] Shen Jin-biao, Liu Yue-jin. A novel building algorithm of concept lattice. *Journal of Hefei University of Technology (Natural Science)*. 2010; 33(2):301-307.
[9] Gu Chun-sheng. Fully Homomorphic Encryption, Approximate Lattice Problem and LWE. *International Journal of Cloud Computing and Services Science*. 2013; 2(1): 1-15.
[10] Qu Wen-jian, Tan Guang-xing, Qiu Tao-rong. New Algorithm of Generating Concept Lattice Using Universal Matrix. J*ournal of Chinese Computer Systems*. 2012; 33(3): 558-564.
[11] Wang Hui, Wang Jing. Non-redundant association rules extraction of frequent concept lattice based on FP-tree. *Computer Engineering and Applications*. 2012; 48(15): 12-14.
[12] Chen Xiang, Wu Yue. Association Rule Mining Algorithm Based on Base Set and Concept Lattice. *Computer Engineering*. 2010; 36(19): 34-36.
[13] Yang Kai, Jin Yong-long, He Zhi-jun, Ma Yuan. An Approach for Briefest Rules Extraction Based On Compact Dependencies. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2013; 11(2): 941-947.
[14] Hu Xue-Gang, Chen Hui, Ma Feng. *The Mining of Classification Rules Based on Multiple Extended Concept Lattices*. Proceedings of the Fourth International Conference on Machine Learning and Cybernetics(MLC). Guangzhou. China. 2005; 2063-2066.
[15] Zhao Xu-jun, Zhang Ji-fu, Ma Yang, Cai Jiang-hui. Novel Classification Rule Acquisition Algorithm. *Journal of Chinese Computer Systems*. 2012; 33(5): 1126-1130.
[16] Lv Yue-jin, Liu Hong-mei. Improved algorithm for attribute reduction on concept lattice. *Computer Engineering and Applications*. 2011; 47(8): 146-148.
[17] Wang De-xing, Hu Xue-gang, Wang Hao. Algorithm of mining association rules based on Quantitative Concept Lattice. *Journal of Hefei University of Technology (Natural Scienc*e). 2002; 25(5): 678-682.