A Novel Incorporate Algorithm of Concept Lattice

Xi Gong
School of Software Engineering, University of Science and Technology LiaoNing, Anshan, 114051, China
E-mail: askdjy05gx@163.com

Abstract With the expansion of the research field, the research object of some original seemingly unrelated properties have been studied together. At this time, the number of attribute in formal context has changed. For the increased attributes, we need to construct a new concept lattice. The existing incremental building algorithms of concept lattice need the original formal context as the basis, with single attribute or a set of attribute of the object to rebuild the concept lattice. They can't effectively utilize these existing concept lattice that have not relation in attributes. Here, the paper presents one new algorithm for incorporating concept lattice based on the existed concept lattices. We can directly build the "together lattice" from bottom to top by direct product operation on the existed concept lattices and the mapping relation between the direct product lattice of two existed concept lattices and the "together lattice". Formal contexts that attribute sets have no intersection are fit for this algorithm.

Keywords-component: formal context; concept lattice; incorporating; direct product

I. INTRODUCTION

Since the formal concept analysis theory was put forward by German Wille[1] professor in 1982, with 30 years' development, the formal concept analysis and its core data structure concept lattice was proved to be an effective tool of knowledge discovery and data analysis .It has been widely used in the field of knowledge discovery, software engineering, information retrieval, information filtering etc[2-6]. How to improve the efficiency of constructing concept lattice is a hotspot problem. The existing algorithm to structure concept lattice can be divided into two categories: batch algorithm [7,8] and incremental algorithm [9-10]. Along with the information storage increase gradually, it often need a lot of formal context merge together to form larger formal context. The space and time complexity of concept lattice grows exponentially with the scale of the increase of the formal context. If we use the existing incremental algorithms to reconstruct the concept lattice of the new formal context, the method consumed a mass of memory and time. This algorithm directly utilizes two existing formal context corresponding to the original concept lattices to construct the total concept lattice that corresponding to the merged formal context and solves the problem of low utilization ratio of the original concept lattice.

II. BASIC CONCEPTS AND BASIC THEOREMS[11]

Definition 1 A formal context is composed of two sets O and A and a relation I between O and A. The elements of O are called the objects and the elements of A are called the attributes. In order to express that object o is in relation I with attribute a, we write $(o,a)\in I$ or oIa.

Definition 2 Given a set of objects $X\subseteq O$ and a set of attributes $Y\subseteq A$ from the formal context (O,A,I), two operators can be defined as:

$$f(X)=\{a\in A|\forall x\in X, xIa\} \quad g(Y)=\{o\in O|\forall y\in Y, oIy\}$$

Definition 3 For $x\subseteq O$, $y\subseteq A$, a pair (x,y) such that $f(x)=y, g(y)=x$, is called a (formal) concept. For a concept (x,y) the set x is called the extent and the set y the intent of the concept.

Definition 4 If $(x_1,y_1),(x_2,y_2)$ are two concepts of formal context (O,A,I) with $x_1\subseteq x_2$, then (x_2,y_2) is called the super concept of (x_1,y_1), written as $(x_1,y_1)\leq(x_2,y_2)$.If $(x_1,y_1)\leq (x_2, y_2)$, $(x_1,y_1)\neq(x_2,y_2)$ then written as $(x_1,y_1)<(x_2,y_2)$.If there is no (x_3,y_3) such that $(x_1,y_1)<(x_3,y_3)<(x_2,y_2)$ then (x_2,y_2) is called as the father concept of (x_1,y_1) and (x_1,y_1) is called as the child concept of (x_2,y_2). With respect to this partial order, the set of all formal concepts forms a complete lattice written as $L(O,A,I)$. $L(O, A, I) = (L(O,A,I),\leq)$ is called the concept lattice of the formal context (O,A,I).

Definition 5 The direct product of two partial order sets $(M_1,\leq)$ and $(M_2,\leq)$ is defined as a partial order set $(M_1\times M_2,\leq)$.the " $\leq$ " means that $(x_1,x_2)\leq(y_1,y_2):\Leftrightarrow x_1\leq y_1$ and $x_2\leq y_2$.

Theorem 1 concepts lattice $L(O,A,I)$ is a complete lattice. The infimum and supremum are defined as:

$$\bigwedge_{t\in T}(A_t,B_t)=(\bigcap_{t\in T}A_t, f(\bigcap_{t\in T}A_t)) \quad \bigvee_{t\in T}(A_t,B_t)=(g(\bigcap_{t\in T}B_t), \bigcap_{t\in T}B_t),$$

T is a index set.

III. BASIC PRINCIPLE OF THE ALGORITHM

In this paper, the original formal context is (O,A_1,I_1). The concept lattice expressed by L_1.The additional attribute set in the formal context is A_2. Its corresponding formal context is (O,A_2,I_2), the concept lattice expressed by L_2. $L_X=L_1\times L_2$ is the direct product of L_1 and L_2. L_X is a complete lattice. (c_1,c_2) is the element of L_X. c_1 is the concept of I_1 as well as c_2 is the concept of I_2. The set expressed by C_X. The total concept lattice is expressed by L. C expresses the set of all elements of L.When $A_1\cap A_2=\varnothing$, this algorithm is applicative.

Definition 6 φ: $C \to C_X$ is a mapping that associates a concept of the total lattice L to a concept of the direct product lattice L_X.

$$\varphi((x,y))=((g(y\cap A_1),y\cap A_1),(g(y\cap A_2),y\cap A_2))$$

Proposition 1 The mapping φ is an order embedded that associates L to L_X and injective.

Proof: Consider $(x_1,y_1),(x_2,y_2)\in C$ with $(x_1,y_1)\le(x_2,y_2)$ then

$$\varphi((x_1,y_1))=((g(y_1\cap A_1),y_1\cap A_1),(g(y_1\cap A_2),y_1\cap A_2))$$
$$\varphi((x_2,y_2))=((g(y_2\cap A_1),y_2\cap A_1),(g(y_2\cap A_2),y_2\cap A_2))$$

For $(x_1,y_1)\le(x_2,y_2) \Rightarrow y_2\subseteq y_1 \Rightarrow y_2\cap A_1\subseteq y_1\cap A_1$, $y_2\cap A_2\subseteq y_1\cap A_2 \Rightarrow (g(y_1\cap A_1),y_1\cap A_1)\le(g(y_2\cap A_1),y_2\cap A_1)$ and $(g(y_1\cap A_2),y_1\cap A_2)\le(g(y_2\cap A_2),y_2\cap A_2)$ then $\varphi((x_1,y_1)) \le\varphi((x_2,y_2))$. Obviously, the mapping φ is order-preserving.

Consider $\varphi((x_1,y_1))\le\varphi((x_2,y_2))$ with $(x_1,y_1),(x_2,y_2)\in C$ then $(g(y_1\cap A_1),y_1\cap A_1)\le(g(y_2\cap A_1),y_2\cap A_1)\Rightarrow y_2\cap A_1\subseteq y_1\cap A_1$ and $(g(y_1\cap A_2),y_1\cap A_2)\le(g(y_2\cap A_2),y_2\cap A_2)\Rightarrow y_2\cap A_2\subseteq y_1\cap A_2$ For $(y_2\cap A_1)\cup(y_2\cap A_2)\subseteq(y_1\cap A_1)\cup(y_1\cap A_2)\Rightarrow y_2\cap(A_1\cup A_2)\subseteq y_1\cap(A_1\cup A_2)\Rightarrow y_2\cap A\subseteq y_1\cap A$. For $y_1\subseteq A$ and $y_2\subseteq A\Rightarrow y_2\subseteq y_1\Rightarrow(x_1,y_1)\le(x_2,y_2)$. So, the mapping φ is an order embedded.

Next, we proof the mapping φ is injective.

Consider $\varphi((x_1,y_1))=\varphi((x_2,y_2))$ with $(x_1,y_1),(x_2,y_2)\in C$ then

$(g(y_1\cap A_1),y_1\cap A_1)=(g(y_2\cap A_1),y_2\cap A_1)\Rightarrow y_1\cap A_1=y_2\cap A_1$

$(g(y_1\cap A_2),y_1\cap A_2)=(g(y_2\cap A_2),y_2\cap A_2)\Rightarrow y_1\cap A_2=y_2\cap A_2$

For $(y_1\cap A_1)\cup(y_1\cap A_2)=(y_2\cap A_1)\cup(y_2\cap A_2)\Rightarrow y_1\cap(A_1\cup A_2)=y_2\cap(A_1\cup A_2)\Rightarrow y_2\cap A=y_1\cap A$. For $y_1\subseteq A$ and $y_2\subseteq A\Rightarrow y_1=y_2\Rightarrow(x_1,y_1)=(x_2,y_2)$. So, the mapping φ is injective.

Proposition 2 The mapping φ is supremum-preserving.

Proof: Let $X=\{(x_t,y_t)|T$ is an index set, $\forall t\in T$, (x_t,y_t) is a concept of total lattice $L\}\subseteq C$, so X is a subset of C, then $\vee X=(g(\cap_{t\in T} y_t),\cap_{t\in T} y_t)$, $\varphi(\vee X)=\varphi(g(\cap_{t\in T} y_t),\cap_{t\in T} y_t)=((g(\cap_{t\in T} y_t\cap A_1),\cap_{t\in T} y_t\cap A_1),(g(\cap_{t\in T} y_t\cap A_2),\cap_{t\in T} y_t\cap A_2))$. $\varphi(X)=\{(g(y_t\cap A_1),y_t\cap A_1),(g(y_t\cap A_2),y_t\cap A_2)\,|T$ is an index set, $\forall t\in T,(x_t,y_t)$ is a concept of total lattice $L\}$.

$$\vee\varphi(X)=((g(\cap_{t\in T}(y_t\cap A_1)),\cap_{t\in T}(y_t\cap A_1)),(g(\cap_{t\in T}(y_t\cap A_2)),\cap_{t\in T}(y_t\cap A_2)))$$
$$=((g(\cap_{t\in T} y_t\cap A_1),\cap_{t\in T} y_t\cap A_1),(g(\cap_{t\in T} y_t\cap A_2),\cap_{t\in T} y_t\cap A_2))=\varphi(\vee X).$$

So mapping φ is supremum-preserving.

Definition 7 ψ: $C_X\to C$ is a mapping that associates a concept of the direct product lattice L_X to a concept of the total lattice L.

$$\psi((x_1,y_1),(x_2,y_2))=(x_1\cap x_2, f(x_1\cap x_2))$$

Proposition 3 The mapping ψ is order-preserving and infimum-preserving.

Proof: If $((x_1,y_1),(x_2,y_2)),((x_3,y_3),(x_4,y_4))\in C_X$ and $((x_1,y_1),(x_2,y_2))\le((x_3,y_3),(x_4,y_4))$ then:

$$\psi((x_1,y_1),(x_2,y_2))=(x_1\cap x_2, f(x_1\cap x_2)),$$
$$\psi((x_3,y_3),(x_4,y_4))=(x_3\cap x_4, f(x_3\cap x_4)).$$

$((x_1,y_1),(x_2,y_2))\le((x_3,y_3),(x_4,y_4))\Rightarrow x_1\subseteq x_3, x_2\subseteq x_4\Rightarrow x_1\cap x_2 \subseteq x_3\cap x_4 \Rightarrow (x_1\cap x_2, f(x_1\cap x_2))\le(x_3\cap x_4, f(x_3\cap x_4)) \Rightarrow \psi((x_1,y_1),(x_2,y_2))\le\psi((x_3,y_3),(x_4,y_4))$. So mapping ψ is order-preserving.

Next, we proof the mapping ψ is infimum-preserving.

Let $Y=\{(x_t,y_t)|T$ is an index set, $\forall t\in T,(x_t,y_t)$ is a pair of concepts in direct product lattice L_X. $x_t=(p_t,q_t),y_t=(r_t,s_t)\}$, so Y is a subset of C_X, then $\wedge Y=(\cap_{t\in T} x_t, f(\cap_{t\in T} x_t))$.

$$\psi(\wedge Y)=\psi(\cap_{t\in T} x_t, f(\cap_{t\in T} x_t))=(\cap_{t\in T} p_t\cap(\cap_{t\in T} r_t), f(\cap_{t\in T} p_t\cap(\cap_{t\in T} r_t)))$$

$\psi(Y)=\psi(x_t,y_t)=\{(p_t\cap r_t, f(p_t\cap r_t))|T$ is an index set, $\forall t\in T$, (x_t,y_t) is a pair of concepts in direct product lattice $L_X\}$

$$\wedge\psi(Y)=(\cap_{t\in T}(p_t\cap r_t), f(\cap_{t\in T}(p_t\cap r_t)))=(\cap_{t\in T} p_t\cap(\cap_{t\in T} r_t), f(\cap_{t\in T} p_t\cap(\cap_{t\in T} r_t)))=\psi(\wedge Y).$$

So, mapping φ is infimum-preserving.

According to the above definition, theorem and proposition, may safely draw the following conclusion.The total lattice L and the image on L_X that are produced by mapping φ are isomorphic. With this conclusion, to produce the total lattice L by one sublattice of the direct product lattice L_X whom satisfied some condition become possible.

Properties 1 $\forall c\in C$, $\varphi(c)=\min(\psi^{-1}(c))$

Proof: Let $((x_1,y_1),(x_2,y_2))=\min(\psi^{-1}(c)),c=(p,q)\in C$. For any $((s_1,t_1),(s_2,t_2))\in(\psi^{-1}(c))$ such that $((x_1,y_1),(x_2,y_2))\le((s_1,t_1),(s_2,t_2))$.

①if $\left|\psi^{-1}(c)\right|=1$ then $((x_1,y_1),(x_2,y_2))=((s_1,t_1),(s_2,t_2))$. So properties 1 is tenable.

②if $\left|\psi^{-1}(c)\right|\neq1$, then $((x_1,y_1),(x_2,y_2))<((s_1,t_1),(s_2,t_2))$.

Due to the" $\le$ "relation in $L_X\Rightarrow y_1\supseteq t_1, y_2\supseteq t_2$. The mapping φ is injective $\Rightarrow q\cap A_1=q\cap y_1, q\cap A_2=q\cap y_2$. So properties 1 is tenable.

Definition 8 φ':

$$\forall c\in C,\ \varphi'(c)=\{(c_i,c_j)|Extent(c_i)\cap Extent(c_j)=Extent(c)\}$$

$Extent(c)$ expresses the extent of concept c. The concepts of the total lattice L can be computed easily by the image of the mapping φ'.

Properties 2 For any concept $c=(x,y)\in C$, if $\varphi(c)=((x_1,y_1),(x_2,y_2))$ then $x=x_1\cap x_2$ and $y=y_1\cup y_2$.

Proof : Due to the define of $\varphi(c)$, $y_1=y\cap A_1, y_2=y\cap A_2$ $\because A_1\cap A_2=\varnothing$, $A_1\cup A_2=A$, $\therefore y=y_1\cup y_2$. $g(y_1)=x_1, g(y_2)=x_2$ then $g(y)=g(y_1\cup y_2)=g(y_1)\cap g(y_2)=x_1\cap x_2$. So properties 2 is tenable.

$children(c)$ expresses the set of the child concepts of concept c. It's elements are determined by the child concepts of $\varphi(c)$ whom is the image of concept c in direct product lattice L_X. We get the images of $\varphi(c)$'s child

concepts by mapping ψ . The maximum of them form the set $children(c)$.

Definition 9 $children(c)=\max(\{\psi(\overline{c})|\overline{c}\in children(\varphi(c))\})$

If we want to build the total concept lattice with the original lattice and the additional lattice there are three questions need process.

(1)To find the update concepts from the direct product lattice that can produce new concepts.

(2)To compute the extent and the intent of these concepts.

(3)To define the father-son relation between these concepts.

(1)(2)(3) are the main steps of this algorithm and the above properties form the theory foundation. This algorithm can produce the total concept lattice only through the direct product operation on the original lattice and the additional lattice.

Input: direct product lattice L_X ,the result of direct product operation on The original lattice L_1 and the additional lattice L_2 .

Output: L , the total concept lattice L

Procedure $Constrct-lattice$ (Input ； L_X ,Output ： L)

Begin

$L\leftarrow\varnothing$;

For each (c_i,c_j) in $C_1\times C_2$ do

$E\leftarrow Extent(c_i)\cap Extent(c_j)$

$Judgevalue\leftarrow\psi(children(c_i,c_j))$

// If this concept is an update concept

If $Findnew(E,Judgevalue)$ then

//produce a new concept c . Its extent is E and intent is the union of c_i 's intent and c_j 's intent.

$c\leftarrow(E,Intent(c_i)\cup Intent(c_j))$

For each $\overline{c}$ in $\max(Judgevalue)$ do

$Link(c,\overline{c})$ // $\overline{c}$ is c 's child concept. Link $\overline{c}$ to c

$L\leftarrow L\cup\{c\}$ //Add the new concept c to the L

End

//To judge whether (c_i,c_j) is updated concept. If the return value is true then (c_i,c_j) is updated concept.

Produce $Findnew(E,Judgevalue)$

Begin

$flag\leftarrow false$ // signal variable

For each $\overline{c}$ in $Judgevalue$ do

If $Extent(\overline{c})\neq E$ then

$flag\leftarrow true$

Else begin

$flag\leftarrow false$

//if $Extent(\overline{c})=E$ then break the circulation and return the $flag$'s value.

return $flag$

end

return $flag$

End

// the algorithm get the child concepts of (c_i,c_j)

Procedure $children(c_i,c_j)$

Begin

$child\leftarrow\varnothing$

For each (c_m,c_n) in $C_1\times C_2$ do

If $(c_m,c_n)<(c_i,c_j)$ and has no (c_k,c_r) such that $(c_m,c_n)<(c_k,c_r)<(c_i,c_j)$ then

// (c_m,c_n) is child concept of (c_i,c_j)

$child\leftarrow child\cup\{(c_m,c_n)\}$

$children\leftarrow child$

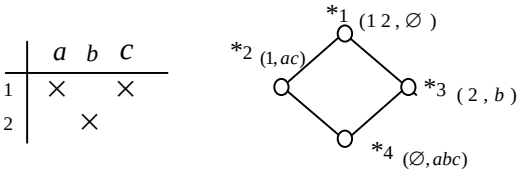
End

IV. EXAMPLE

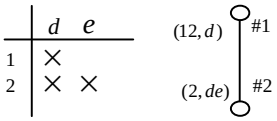
Here, we use a specific example to demonstrate the algorithm. The original formal context (O,A_1,I_1) .The additional formal context (O,A_2,I_2) .Let O={1,2} , A_1={a,b,c} , A_2={d,e} .These formal contexts and their concept lattices are shown as Figure 1.The total formal context (O,A,I) is shown as Figure 2.

The lattice generated by direct product operation on two of the original formal context concept lattice is shown on the left in Figure 3.

Using the above algorithm on this lattice, firstly find the update concepts, secondly according to the properties 2 calculated the concept C of the combined total lattice. Finally, according to the definition 9 found child nodes of concept c and connecting them. It constructed a combined total concept lattice (such as the right side of Figure 3 shows).



（a）The original formal context and its concept lattice



（b）The additonal formal context and its concept lattice

Figure 1.The original formal context, the additional formal context and their corresponding concept lattice

	a	b	c	d	e
1	×		×	×	
2		×		×	×

Figure 2.The incorporative formal context

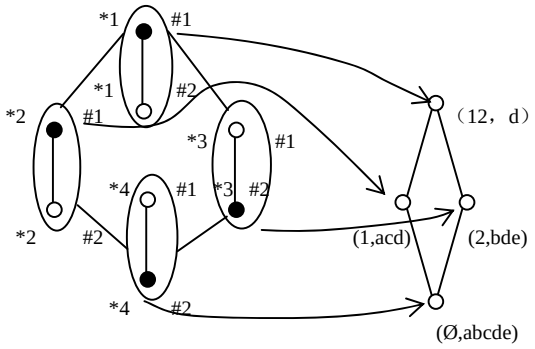


Figure 3 The direct product lattice L_x、mapping ψ and the incorporative concept lattice L

In order to verify the correctness of the algorithm, we compared this algorithm and the algorithm that produces concept lattice by formal context with experiments. The latter's input is the total formal context and the output is the total concept lattice. The program use VB6.0 as programming language. Experiment data use random data.

The number of objects is set to 30. The number of attributes is changing, from the beginning of 20, incrementing by 20 until 200. Each object has the same number of attributes. Context concentration is 40%. The experimental results prove the validity of the algorithm.

Algorithm features: Some of the existing incremental construction algorithms of concept lattice, their core idea is with a concept lattice as the basis, then every increase a attribute to traverse the concept lattice one time[12,13].

According to the additional attributes to determine concept category, and carries on the corresponding operation to form the new concept lattice until you add all attributes to generate the final result.

This algorithm provides a new method with the direct product for constructing concept lattice. Firstly, we produce the direct product lattice. Using the original concept lattice and direct product lattice isomorphic relation, can easily determine the father-son relationship of concepts in the direct product concept lattice. Concepts are only divided into two categories: update and non-update. The operation is simple.

The method only needs to traverse the direct product lattice one time. In the traverse, it adds all additional attributes and generates the concept lattice from the bottom to the top. It does not traverse and operate the concept lattice when there is a new attribute is added like the existing incremental algorithms. So, it is more efficient than the incremental algorithm.

V. CONCLUSION

Concept lattice generation problem is the key steps of knowledge extracting from database. To overcome the shortcomings of the present incremental building algorithms of concept lattice, we propose a new method based on direct product. It is applicable nicely when a large number attributes were added to database or more than two database were merged. The experiment proved that the algorithm is correct, but the application scope is limited. The next improvements include reducing the limitations of the algorithm, and to expand the scope of application of the algorithm, to reduce the search range of the update concept, as well as to improve the association rules extraction algorithm in data mining.

REFERENCES

[1] Wille R. Restructuring lattice theory: an approach based on hierarchies of concepts. Rival I(ed).Ordered Sets. Dordrecht-Boston: Reidel Publishing Company, 1982. 445～470.

[2] R Godin, G Mineau, R Missaoui, et al, "Applying concept formation methods to software reuse",International Journal of Knowledge Engineering and Software Engineering, 1995, 5(1): 119～142.

[3] G W Mineau, R Godin, "Automatic structuring of knowledge bases by conceptual clustering", IEEE Transaction on Knowledge and Data Engineering, 1995, 7(5): 824～828.

[4] R Cole, P Eklund, "Scalability in formal concept analysis", Computational Intelligence, 1999,15 (1): 11～27.

[5] C Carpineto, G Romano, "A lattice conceptual clustering system and its application to browsing retrieval",Machine Learing, 1996, 24(2): 95～122.minutes

[6] R Godin, R Missaoui, "An incremental concept formation approach for learning from databases",Theoretical Computer Science, 1994, 133(5): 387～419.

[7] J P Bordat, "Calcul pratique du trellis de galois d'une correspondence", Mathematiques et Sciences, 1986, 96: 31～47.

[8] Lhouari Nourine, Olivier Raynaud, "A fast algorithm for building lattices",Information Processing Letters, 1999, 71(5-6): 199～204.

[9] R Godin, "Incremental concept formation algorithms based on Galois (concept) lattices",Computational Intelligence, 1995, 11(2): 246～267.

[10] Z.P. Xie, L.Z. Tian,"A Fast Incremental Algorithm for Building Concept Lattice", Chinese Journal of Computers, 2002, 25(5): 490～496.

[11] Ganter B, Wille R. Formal concept analysis: mathematical foundations. Berlin: Springer, 1999.

[12] S.Q. Jian, X.G. Hu, M.H. Jiang, "Incremental Algorithms of Extended Concept Lattice ", Computer Engineering and Applications 2001, 36(15): 132～134.

[13] X.J. Shen, D.J. Han, Z.T. Liu, Jun Ma,"Improvement on Constructing Algorithm of Concept Lattices", Computer Engineering and Applications, 2004, 40(24): 100～104.