A Pruning Based Incremental Construction Algorithm of Concept Lattice*

Zhang Ji-Fu[1,2], Hu Li-Hua[1], and Zhang Su-Lan[1]

[1] School of Computer Science and Technology, Tai-Yuan University of Science and Technology, Tai-Yuan 030024, P.R. China
[2] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100080, P.R. China
jifuzh@sina.com

Abstract. The concept lattice has played an important role in knowledge discovery. However due to inevitable occurrence of redundant information in the construction process of concept lattice, the low construction efficiency has been a main concern in the literature. In this work, an improved incremental construction algorithm of concept lattice over the traditional Godin algorithm, called the pruning based incremental algorithm is proposed, which uses a pruning process to detect and eliminate possible redundant information during the construction. Our pruning based construction algorithm is in nature superior to the Godin algorithm. It can achieve the same structure with the Godin algorithm but with less computational complexity. In addition, our pruning based algorithm is also experimentally validated by taking the star spectra from the LAMOST project as the formal context.

Keywords: concept lattice, pruning, redundant information, incremental construction algorithm, star spectra.

1 Introduction

From a philosophical point of view, a concept is a unit of thoughts consisting of two parts, the extension and the intension. Based on the philosophical understanding of concept, the formal concept analysis [1] was introduced by Wille.R in 1982, and later used to detect, sort and display of concepts. Based on the formal concept analysis, the extension covers all objects belonging to this concept and the intension comprises all attributes valid for all those objects, by which the philosophical understanding of concept was realized. By nature, concept lattice describes the relationship between objects and attributes, indicates the relationship of generation and specialization between concepts. Besides, its Hasse diagram is an effective tool of data visualization. Thanks to its straightness, simplicity and completeness of knowledge expressing, the concept lattice has been widely applied in software engineer, knowledge engineer, knowledge discovery and so on [2], [3], [11], etc.

* This paper is supported by the National Natural Science Foundation of P.R.China (60573075).

At present, broadly speaking, there are two kinds of concept lattice construction algorithms: The incremental algorithm [4], [5], [6] and the patch algorithm [8]. The basic idea of the patch algorithm is to generate all concepts at first, then according to the relationship of generation and specialization, to generates edges, then form concept lattice. Such algorithms include Bordat algorithm, OSHAM algorithm, Chein algorithm, Ganter algorithm, Nourine algorithm and so on [8]. The basic idea of the incremental construction algorithm is to initialize a null concept at first, then gradually form concept lattice by adopting different suitable operations based on the intersection difference between the attributes of a newly added object with the intension of the original concept lattice nodes. Such algorithms include Godin, Gapineto and T.B.Ho algorithm [2], [8]. Many researchers have proposed some improvements on the above algorithms, such as the fast incremental algorithm for building concept lattice [9] and so on. Lots of experiments show that the incremental construction algorithm is a promising one, and the Godin algorithm is a typical incremental construction algorithm.

In many cases, concept lattice construction uses mass、high-dimensional data as formal context. For the analysis of mass data, usually too many nodes are generated due to the completeness requirement of concept lattice, which in turn causes large storage and low construction efficiency during the incremental construction process, because the attributes of a newly added object must be compared with the intension of the original concept lattice nodes one by one. As a result, with more added objects, the updating efficiency of concept lattice becomes worse. In reality, in the process of incremental construction, much redundant information is generated, which unnecessarily increases the comparing times of concept lattice intension but has no effect on the resulting structure. Hence how to eliminate or reduce the redundant information in the concept lattice construction process is a key issue to increase its construction efficiency. To this end, we propose a technique, coined as "pruning", to eliminate possible redundant information in this work, and the proposed algorithm is shown to work satisfactorily. In particular, our experiments show that our proposed algorithm (PCL) could improve the construction efficiency by above 15 % than the Godin algorithm, a popular algorithm in the literature.

2 Basic Concept of the General Concept Lattice and Its Incremental Construction

Definition 1. A formal context is defined as a triplet K=(O, D R), where O is a set of objects, D is a set of attributes and R is a binary relation between O and D, which describes the inherent lattice structure and defines the natural groupings and relationships between the objects and their attributes. This structure is known as a concept lattice or Galois lattice L.

Definition 2. Given a concept lattice L constructed from formal context K, each one of its nodes is a couple, denoted as C (A, B), where A$\in$P (O) is called the extension of concept, B$\in$P (D) called the intension of concept. P(O) and P(D) are power sets of O and D respectively.

Definition 3. Concept lattice L must be a complete couple with respect to R. that means for each node C (A, B), following two conditions are both satisfied:

(1) $A=B'=\{a\in O \mid \forall b\in B, a\,R\,b\}$
(2) $B=A'=\{b\in D \mid \forall a\in A, a\,R\,b\}$

Definition 4. In the concept lattice L, if a node Ci (Ai, Bi) satisfies the following condition, it is defined as the supremum of this node, denoted as Sup(Ci). J is the alphabetical order set of concept lattice L.

$$\bigvee_{i\in J}(A_i,B_i)=\left(\left(\bigcap_{i\in J}B_i\right)',\bigcap_{i\in J}B_i\right)$$

Definition 5. If C1=(A1, B1) and C2=(A2, B2) are two different nodes, then $C1<C2 \Leftrightarrow A1\subset A2 \Leftrightarrow B2\subset B1$. If there does not exist other node C3=(A3,B3) in the lattice such that C1<C3<C2, we say C1 is the sub(child) concept of C2 , denoted as C1=child(C2) ; C2 is the super (parent) concept of C1, denoted as C2=father (C1).

In general, the formal context of concept lattice is represented by a table as shown in Table 1, where the rows represent objects, and the column represent attributes. As an example, the corresponding concept lattice of Table 1 is shown in Fig. 1.

Table 1. Formal Context

O \ D	A	B	C	D	E
1	√	√			
2		√		√	√
3		√	√		
4	√			√	
5			√		

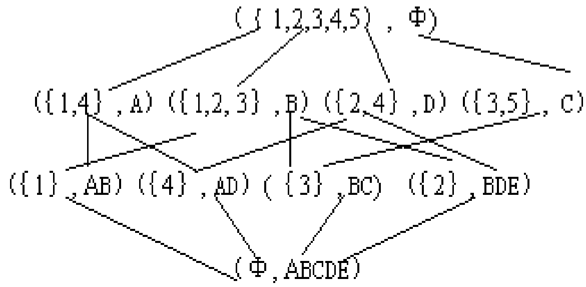

Fig. 1. Hasse Figure of General Concept Lattice

Given a concept lattice L, constructed from the formal context K=(O, D, R), the incremental construction is such a process that while a new object x is added (S is the set of attributes of x), the concept lattice L is modified according to the relationship between the attributes of the object x and the intension of the original concept lattice nodes at the new formal context $K'=(O\cup\{x\}, D, R)$. In the process of incremental construction, according to the attribute set S of the newly added object and the intension of the original concept lattice nodes, concept lattice nodes can be classified into three cases: old node, modified node and newly added node. Their definitions are:

Definition 6. Let C (A, B) be a node of concept lattice L, if the intersection between B and S is NULL, denoted as $B\cap S=\Phi$, then C is called an old node.

Definition 7. Let C (A, B) be a node of concept lattice L, if B is a subset of S, i.e., $B\subseteq S$, then C is called a modified node.

Definition 8. Let C (A, B) be a node of concept lattice L, if the intersection between B and S, $H=B\cap S$, is not NULL, denoted as $H\neq\Phi$, and then C is called a generated node. Additionally if the following two conditions are both satisfied:

(1) H is not equal to the intension of any nodes of concept lattice L.
(2) The intersection of S with any super node C_1 of C is not equal to H, i.e., $B_1\cap S\neq H$,

Then $C_1=(A\cup\{x\}, H)$ is called a newly added node of concept lattice L.

Fig2 shows the old nodes, modified nodes, and newly added nodes by the incremental construction algorithm. Fig. 2. is generated from Fig .1. by adding a new object x (6, {A, E}). The node ({2,6}, E) is a new node; all the other nodes are old ones.

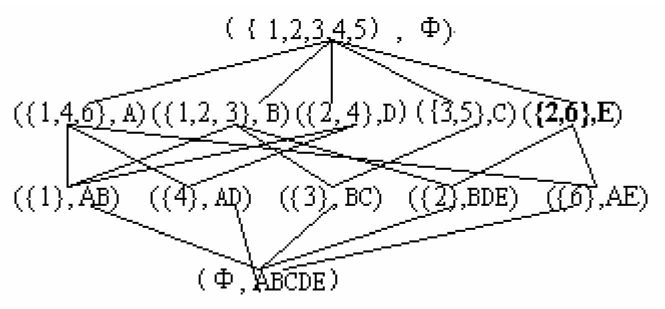

Fig. 2. Hasse Figure of Concept Lattice from a New Object

3 Pruning Based Incremental Construction of Concept Lattice

From the previous section, we know that during the incremental construction, the nodes and edges corresponding to modified nodes or newly added nodes should be modified. As a result, it is possible that unnecessary comparisons between S and B occur. Hence by eliminating or avoiding such unnecessary comparisons, the construction efficiency could be improved.

Definition 9. Let C (A, B) be a node of concept lattice L, C1 (A1, B1) and C2 (A2, B2) are two sub (child) nodes of C, C1=child(C), C2=child(C), C1 and C2 are called the sibling nodes. The sub (child) nodes C1 or C2, denoted as offs(C), offs(C) are called the offspring nodes of C.

Definition 10. Let C (A, B) be a node of concept lattice L, the number of the objects contained in A is called the support of extension of concept C, denoted as |A|. The number of the attributes contained in B is called the support of intension of concept C, denoted as |B|.

Definition 11. Let C (A, B) be a node of concept lattice L and C be a modified node with respect to a newly added object x during the incremental construction process, if C is modified as C_1 (A_1, B_1), where $A\subset A_1$ and $B_1=B$, then C_1 (A_1, B_1) is called the redundant information.

Theorem 1. Let C=(A, B) be a node of concept lattice L, x is a newly added object (its attributes set is S), if C is modified to C_1 (A_1, B_1), and B is not equal to Sup (S), then C_1 (A_1, B_1) must be redundant information.

Proof. Since B is not equal to Sup (S), there must exist a node C_2 (A_2, B_2) such that $B_2 \subset B$ and $A \subset A_2$ in the concept lattice L. Since C is a modified node, we have $B \subseteq S$, then $B_2 \subseteq S$. Node C_1 ($A \cup \{x\}$, S) could be generated from the newly added object x and C, and node C_3 ($A_2 \cup \{x\}$, S) could be generated from x and C_2. By definition 11, C_1 is redundant information.

Definition 12. In the incremental construction process of concept lattice, the operations of eliminating redundant information is called pruning.

Theorem 2. The pruning of concept lattice does not disrupt the completeness of concept lattice.

Proof. Suppose redundant information C_1 (A_1, B_1) is generated during the incremental construction process of concept lattice L, from theorem 1, C_1 ($A \cup \{x\}$, S) and C_3 ($A_2 \cup \{x\}$, S) are generated also. Hence after the pruning of concept lattice, redundant information is eliminated, and C_3 ($A_2 \cup \{x\}$, S) is generated from the newly added object x and C_2. So the pruning of concept lattice does not disrupt the completeness of concept lattice.

From Theorem 2, redundant information only adds comparing times of concept lattice intension; it does not affect the structure of concept lattice.

The comparisons of the attribute set S of the newly added object x with the intension B of a node C (A, B) of concept lattice can be listed as the following 4 cases:

1) B is a subset of S ($B \subset S$)

If $B \subset S$, it indicates that the generated nodes by adding object x are the parent nodes of C. In this case, it is unnecessary to compare with the child or offspring nodes of C;

2) S is the supermum of B

If B=Sup(S), it indicates the generated nodes by adding object x is the child nodes of C. No further operation is needed;

3) B=S

In this case, it suffices to modify node C and the extensions of all the parent nodes of C.

4) $B \cap S \neq \Phi$ and $B \not\subset S$

In this case, a new node should be added, and at the same time, this newly added node should be compared with the existing nodes.

Based on the above analysis, we can see that the process of pruning can reduce the redundant information, decrease the comparing times of concept lattice intension without affecting its structure during the construction of concept lattice. And additionally, its completeness is also preserved. Hence by combining with the Godin algorithm, the basic principle of the pruning based incremental construction of concept lattice can be outlined as:

If the attribute set of a newly added object is a subset of the intension of concept lattice nodes, it will generate redundant information from theorem 1 when compared with the sub concepts of the concept lattice nodes. In order to eliminate such redundant information, we can use the pruning of concept lattice to find the subset of the newly added object such that it does not need to compare with sub nodes and offspring nodes. Similarly if the attribute set of a newly added object is equal to the intension of concept lattice nodes, no comparison with sub nodes and offspring nodes is needed either.

The above process is an improvement of the Godin algorithm. It is capable of eliminating redundant information during the construction process, and consequently enhancing the construction efficiency. In addition, the construction process is a top-down process.

Theorem 3. In the construction process of concept lattice by the pruning based incremental construction algorithm, if no redundant information is generated, the pruning based algorithm is degenerated to the Godin algorithm.

Proof. If no redundant information is generated, it means there is no super-node relationship between the newly added object with the other nodes, i.e., comparisons with the offspring of node C do not occur, hence the construction process by the pruning based algorithm is identical to that by the Godin algorithm, and the two algorithms also possess the same time complexity.

Theorem 4. The construction efficiency of the pruning based algorithm is better than that of the Godin algorithm.

Proof. If redundant information is generated in the incremental construction process, according to theorem 1 and theorem 2, the pruning process reduces the number of the comparisons between the intensions, hence its computational efficiency is better than that of the Godin algorithm. On the other hand, if no redundant information is generated in the incremental construction process, according to theorem 3, the two construction algorithms have the same computational complexity. Combining these two cases, we can see that the construction efficiency of the pruning based algorithm is no worse than that of the Godin algorithm in all possible cases.

4 Pruning Based Incremental Construction Algorithm of Concept Lattice Entries

A pseudo-code of our pruning based incremental construction algorithm of concept lattice (PCL in short) could be described as follow:

PCL (The pruning based concept lattice) algorithm:

```
Input:The original concept lattice and a new added object,
the nodes of the original concept lattice are sorted in the
deceasing order of their support of intensions.
output: A new concept lattice
/* inte: the intension of a concept lattice node; exte: the
extension of a concept lattice node; father: the super node
of a concept lattice node; child : the sub node of concept
lattice node; now:is the current label number of the concept
lattice node */
```

```
(1) Input a new object
(2) Search the concept lattice nodes from top to down, and
      compare the attribute set of the new object with the
      intension of the current concept lattice.
(3) Set the new object to a new node new c, its intension
      is new, its extension is code, its super node is
      newfather, its sub node is newchild
(4) for I=now to 1
(5)   determine the relationship between new and inte
(6)      call procedure "judge" (new,inte) ;
(7) next I
(8) end PCL;
  judge(new,inte)
(1)  if  new⊆inte then
(2)        fetch newfahter
(3)     if I∉newfather then
(4)          add a new edge I->new
(5)          code=exte∪code
(6)          father=father∪code
(7)      newchild=newchild∪code
(8)        end if
(9)        exit for
(10)  elseif inte⊆new then
(11)       fetch newchild
(12)    if I∉newchild then
(13)         add a new edge new->I
(14)         newfather=I∪newfather
(15)         exte=exte∪code
(16)         child=child∪code
(17)       end if
(18) elseif inte=new then
(19)       fetch I and the extension exte of all its
     supernodes
(20)       exte=exte∪code
(21)       delete node new
(22)       exit for
(23)  elseif  inte∩new≠Φ then
(24)       add a new concept lattice node newjoin, the
     intersection intension is join
(25)       for j=1 to now
(26)            if  j≠now then
(27)              if  join=inte  then
(28)                 update j
(29)                 delete node newjoin
```

```
(30)             end if
(31)           end if
(32)      next for
(33)   if not exist equal then
(34)       njexte=code∪exte
(35)       njinte=join
(36)       njchild=code∪I
(37)       determine the superconcept relationship
     between newjoin and other concept lattice nodes
(38)       if ∃newjoin's super node, then
(39)         repeat the above modifying operations of the
     super node
(40)       end if
(41)     end if
(42)   end if;
(43)  end judge;
```

In the process of the traditional incremental construction, when a new object is added to the formal context, it must be compared with all the nodes of the original concept lattice. As a result, the algorithm computational complexity increases exponentially at the worst case. More specifically, time complexity of the general concept lattice is O ($2^K|U|$).

For our PCL, when a newly added object is compared with the current concept lattice nodes, if it exists the relationship of the super nodes as before, it becomes unnecessary to continue the comparison process. As a result, it can reduce the comparisons, and the time complexity of our PCL algorithm PCL is no larger than O ($2^K|U|$).

5 Experiment Analysis

Now a large telescope called LAMOST (Large Sky Area Multi-Object Fiber Spectroscopic Telescope) is under construction in the National Observatory in Beijing, China. After its scheduled completion in 2006, it is expected to collect more than 40,000 spectrums in a single observation night Such voluminous data demand automatic spectrum processing and data mining [12]. In this section, we will give some results of our experiments on the concept lattice construction from the observed celestial spectrums.

The experiment setup is: PentiumIII-1.0G CPU, 256M memory, Windows 2000 operating system and ORACLE 9i DBMS. Both the PCL algorithm and the Godin algorithm are coded in Visual Basic 6.0. The star spectra are treated by the following steps, then used as the formal context:

1) For each star spectrum, choose 200 wave-length from 3510 A to 8330 A at a step of 20 A as its attributes set.

2) At the above each chosen wave length, the corresponding flux, peak width and shape information of the spectrum are quantified respectively into one of the 13 different intervals in total.

Table2 are the results of the concept lattice construction, where 150 wave-lengths are used as the attribute set, 500, 1000, 1500, 2000, 2404 B-type star spectra are used as the data objects. Table 3 are the construction results of concept lattice, where the size of the attributes set is 100, 125, 150, 175, 200, and 1500 B-type star spectra are used as the objects.

Table 2. Experimental Comparison of Various Object SetsBetween the Godin and PCL Algorithms

The number of objects	Godin algorithm (seconds)	PCL algorithm (seconds)	The number of nodes
500	108	75	1035
1000	409	348	1462
1500	830	724	1652
2000	1511	1368	2051
2404	2639	2452	2997

Table 3. Experimental Comparison of Various Attribute Sets Between the Godin and PCL Algorithms

The number of attributes	Godin algorithm (seconds)	PCL algorithm (seconds)	The number of nodes
100	123	96	227
125	195	147	373
150	830	724	1652
175	1577	1450	2393
200	2533	2278	3259

From Table2 and Table3, it can be concluded that:

(1) The Godin algorithm and the PCL algorithm can construct the same concept lattice. In other words, the number of the constructed nodes, the corresponding intension, extension, and father-son relationships are the same for both the two algorithms. This verifies the theoretical correctness of the PCL algorithm;

(2) The PCL algorithm is more efficient than the Godin algorithm. This indicates that in the incremental construction process, redundant information is indeed generated for newly added objects, and by pruning, such redundant information is indeed removed.

(3) Dependent on formal context, the efficiency improvement of the PCL algorithm over the Godin algorithm varies. But on average, a 15% improvement is obtained. In addition, since redundant information could occur only for modified nodes, the improvement of the PCL algorithm is only related to the modified nodes, not the total nodes. For example, in Table2, when the number of objects is 2404, the efficiency improvement is not as significant as the other cases. In Table3, the improvement for the case of 175 attributes is not as significant as that for the case of

125 attributes. Their underlying main reason is that although the newly added nodes are large in these cases, the modified nodes are relatively small, as a result, the chance of generating redundant information is relatively small, hence small improvements of their computational efficiency.

6 Conclusions

An improved algorithm to the Godin algorithm, a benchmark of the incremental construction of concept lattice in the literature, is proposed in this work. The key novelty of our proposed algorithm is that during the construction process of concept lattice, a pruning process is activated to detect and eliminate possible generated redundant information, by which the number of the comparisons of concept lattice intensions is largely reduced when a new node is added, and the construction efficiency is consequently increased. In addition, our pruning based incremental construction algorithm is tested using the star spectra from the LAMOST project as the formal context. The preliminary experimental results show that our pruning based algorithm could have a 15% improvement on average on the construction efficiency over the Godin Algorithm. Finally, as the performance of our pruning based algorithm depends crucially on the formal context, and our currently used attributes of spectra are rather simple, our future work will focus on how to select more appropriate attributes to further boost the construction efficiency in the star spectra mining for the LAMOST project.

References

1. Wille R, Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts. In: Rival I ed. Ordered sets, M. Dordrecht:Reidel, (1982) 415–470
2. Wille R, Knowledge Acquisition by Methods of Formal Concept Analysis. In: Diday E ed. Data Analysis, Learning Symbolic and Numeric Knowledge, C. New York: Nova science publisher, (1989) 365–380
3. Belen Diaz-Agudo, Pddro A. Gonzalez-Calero. Formal Concept Analysis As a Support Technique for CBR, In: Knowledge-based systems, Vol 14. (2001) 163–171
4. Godin R, Missaoue R. An Incremental Concept Formation Approach for Learning From Database. Theoretical Computer Science, Vol. 133. (1994) 387–419
5. Godin R, Missaoue R, Alaui H. Incremental Concept Formation Algorithms Based on Galois (Concept) lattice. Computational Intelligence, Vol 1(2). (1995) 246-267
6. Nourine L. Raynaud O. A Fast Algorithm for Building Lattices. In: Workshop on Computational Graph Theory and Combinatories, C. Victoria, Canada, May(1999)1-12
7. J. Han, M. Kambr. Data Mining Concepts and Techniques. In: Morgan Kaufmann Publishers, M. (2000)
8. Hu Ke -Yun, Lu Yu-Chang, Shi Chun-Yi. Advances in Concept Lattice and Its Application. Tsinghua Univ (Sci & Tech), Vol 40(9). (2000) 77-81
9. Xie Zhi-Peng, Liu Zong-Tian. A Fast Incremental Algorithm for Building Concept Lattice. Chinese Journal of Computers, Vol 25(5). (2002) 490-496

10. Wang Zhi-Hai, Hu Ke-Yun, Hu Xue-Gang et al. General And Incremental Algorithms of Rule Extraction Based On Concept Lattice. Chinese Journal of Computers, Vol 22(1). (1999) 66-70
11. Hu Ke-Yun, Lu Yu-Chang, Shi Chun-Yi. An Integrated Mining Approach for Classification and Association Rule Based on Concept Lattice. Chinese Journal of Software, Vol 11(11). (2000)1478-1484
12. QIN Dong-Mei. Studies on Automated Spectral Recognition of Celestial Objects. Ph.D.Thesis. Institute of Automation, Chinese Academy of Sciences,(2003)