概念特化的概念格更新构造算法

杜秋香,张继福,张素兰
(太原科技大学 计算机科学与技术学院,山西 太原 030024)

摘 要:概念格是形式概念分析中的核心数据结构,概念格应用的瓶颈之一是其构造效率.针对形式背景的某个属性分解为多个新属性得到更加特化的概念,给出了一种基于概念特化的渐进式更新构造算法.该算法利用分解后的新属性及其相应的形式背景,构造出的概念格与原概念格的某个子概念格作比较,来更新构造概念格,从而减少了比较次数,提高了更新构造的效率.以天体光谱数据作为形式背景,实验验证了该算法的正确性和有效性.

关键词:概念格;渐进式构造;概念特化;更新构造
中图分类号: TP311 **文献标识码**: A **文章编号**: 1673-4785(2008)05-0443-06

An improved algorithm based on concept specialization for constructing concept lattices

DU Qiu-xiang, ZHANG Ji-fu, ZHANG Su-lan
(School of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan 030024, China)

Abstract: Concept lattices are the core data structures in formal concept analysis. The widespread application of concept analysis is limited by the difficulty of constructing a concept lattice. An incrementally updating construction algorithm based on concept specialization was developed after it was realized that the attributes in the formal context can be decomposed into several new attributes, or more specialized concepts. The algorithm, with decomposed attributes and a corresponding formal context, compares the concept lattice formed with the new attributes and one of the sub-lattices of the original concept lattice, then upgrades the concept lattice according to results from the comparisons. In this way the number of comparisons is reduced and the efficiency of constructing the concept lattice is improved. Experiment results, with celestial spectrum data as the formal context, verified the validity of the algorithm.

Keywords: concept lattice; incremental constructing; concept specialization; updating construction

"概念"的基本观点是由哲学理论中的概念发展而来,是反应事物本质属性的思维产物.形式概念理论是 20世纪 80年代初由德国教授 R. Wille提出的[1],并通过 Hasse图生动简洁地体现了概念间的泛化和特化关系,提供了一种数据分析和知识处理的有力工具,被广泛应用于知识工程、数据挖掘、信息检索和软件工程等领域,典型的应用有:Neuss和 Kent使用概念格进行 Internet上文档元信息的自动分类和分析[2]; Eklund和 Martin展示了概念层次进行 Web文档索引和导航的能力[3]; Corbett和 Burrow提出使用概念格表示建筑早期设计软件支持环境(SEED)中的状态图,使得设计中获得的知识可以重用[4]; Gordin等人还将概念格应用于类层次(class hierarchy)的设计上[5].

概念格构造效率一直是概念格应用的主要瓶颈之一.目前,概念格的构造算法主要分为批处理算法和渐进式算法两大类,典型的批处理算法有 Bordat算法、Ganter算法、Nourine算法等[6],其存在的问题是当形式背景发生变化时就要重新构造概念格.渐进式算法被认为是比较有前途的一类算法,可分为增加对象和增加属性 2类概念格的渐进式构造.增加对象的经典算法是 Godin算法[7],以及 Godin算法的改进,例如,采用树结构组织格结点进行概念格的构造[8];增加属性的典型算法是基于属性的概念格渐进式生成算法[9],以及 AddIntent算法[10].此外,对概念格的研究热点还包括:规则提取[11-12]、概念格的扩展以及与其他理论的融合[13-15]、概念格的

收稿日期: 2008-03-21.
基金项目:山西省自然科学基金资助项目 (2006011041).
通信作者:杜秋香. E-mail: janny123@163.com.

更新与维护[16-17]等方面.

概念是语义描述的基本单位,在数据库中,概念描述了各个对象的基本特征.各属性值以及概念依据抽象程度不同可构成一个层次结构,通常称为概念树或概念层次树.在概念层次树中,不同的层次表示概念不同的抽象级别.层次越高,表示概念的抽象级别越高,概念也越泛化;层次越低,表示概念的抽象级别越低,概念也越特化.概念分层为在各种抽象级别处理数据提供了方便.概念格的 Hasse图形象地体现了格中概念间的泛化—特化关系.当形式背景中若干属性合并时,概念格中的部分概念提升到一个较高层次,得到较为泛化的概念;相反,当形式背景中某个属性分解为若干个新属性时,就会得到更加特化的概念.针对形式背景中的概念特化,即属性分解问题,本文给出了一种基于概念特化的概念格渐进式更新构造算法 UCCS,从而提高了概念格更新构造的效率.

1 基本概念

一个二维数据表 (G,M),其中 G的元素称为对象,M 的元素称为属性,若 G与 M 之间存在二元偏序关系 $G \times M = I$,在形式概念理论中称 $K = (G,M,I)$为一个形式背景,则 $(g,m) \in I$或 gIm 表示对象 g具有属性 m.设 A是对象集合 G的一个子集,B 是属性集合 M 的一个子集,若 A、B 满足:1) $f(A) = \{m \in M \mid \forall g \in A, gIm\}$ (A中对象共同属性的集合);2) $g(B) = \{g \in G \mid \forall m \in B, gIm\}$ (B中所有属性的对象的集合),则称 $C(A,B)$为概念格 $L(K)$的一个形式概念,其中,A为该概念的外延,记为$A = \mathrm{Extent}(C)$,B 为该概念的内涵,记为 $B = \mathrm{Intent}(C)$[1,6].

定义 1[1,6] 设 $C_1(A_1,B_1)$,$C_2(A_2,B_2)$为概念格 L的 2个不同概念,若 $C_1 < C_2 \Leftrightarrow A_1 \subset A_2 \Leftrightarrow B_2 \subset B_1$,并且不存在 $C_3(A_3,B_3)$有 $C_1 < C_3 < C_2$成立,则称 C_2 为 C_1 的父概念或父结点,记为 $C_2 = \mathrm{Father}(C_1)$,若 $\mathrm{Father}(C_1)$不存在,则称概念 C_1 为全概念;C_1 为 C_2 的子概念或子结点,记为 $C_1 = \mathrm{Child}(C_2)$,若 $\mathrm{Child}(C_2)$不存在,则称 C_2为零概念.

表 1 形式背景

Table 1 Formal context

	A	B	C	D
1	√	√		√
2	√		√	
3		√	√	
4	√	√		√
5	√			

概念格可以用 Hasse图的形式可视地表现出来,图 1就是表 1的形式背景对应概念格的 Hasse图形式.其中,图 1中的每个结点表示一个形式概念,结点间的连线表示概念间的父子关系.

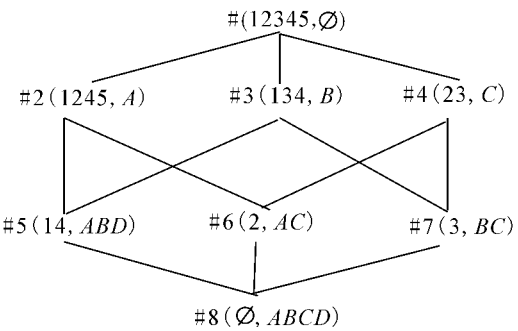


图 1 概念格的 Hasse图

Fig 1 Hasse diagram of concept lattice

2 基于概念特化的概念格更新构造

对于任意给定的形式背景,一个属性分解为若干个新属性,即形式背景中的高层概念属性特化为多个底层概念属性,使得形式背景发生变化,因此如何利用原形式背景已构造出的概念格,来快速更新构造新形式背景的概念格,对于提高概念格构造效率具有重要意义,该问题可以描述如下:

设形式背景 $K_1 = (G_1, M_1, I_1)$,由 K_1 构造的概念格为 L_1,$P \in M_1$,若属性 P分解为属性集合 $P' = \{P_1, P_2, \cdots, P_n\}$,则形式背景由 K_1 变为 $K' = (G_1, (M_1 - P) \cup \{P_1, P_2, \cdots, P_n\}, I')$,且由 K构造的概念格为 L'.如何利用原格 L_1 通过快速更新构造得到概念格 L'是本文要解决和研究的问题.利用 L_1 更新构造出的概念格设为 L,则 $L' = L$.

定义 2 设 $K_1 = (G_1, M_1, I_1)$为形式背景,$P \in M_1$,若属性 P分解为属性集合 $\{P_1, P_2, \cdots, P_n\}$,称 P 为分解属性.

在形式背景 $K_1 = (G_1, M_1, I_1)$中,若 $g(P) = Q$,则由形式背景 $K_2 = (Q, \{P_1, P_2, \cdots, P_n\}, I_2)$构造的概念格设为 L_2,更新构造概念格 L的过程是利用 L_2 的概念与 L_1 的部分概念做比较得到的.

定义 3 设概念 $C_1(A_1,B_1) \in L_1$,$C_2(A_2,B_2) \in L_2$,若概念 C_1 与 C_2 具有相同的外延,即 $A_1 = A_2$,则称概念 C_1 与概念 C_2 互为对应概念.

定义 4 设 $L_1' = (G_1', M_1', I_1')$为任一概念格,若 $C_1'(A_1', B_1') \in L_1'$,都有 $C_1'(A_1', B_1') \in L_1$ 成立并且 L_1 的二元关系 I_1 关于概念格 L_1 也成立,则称 L_1 为 L_1 的子概念格,简称子格.

定理 1 利用概念格 L_1 更新构造概念格 L的过程中,若 L_2 的全概念 $C_2(A_2,B_2)$与 L_1 的子格 L_1'的

全概念 $C_1(A_1, B_1)$互为对应概念,则 L_2只需与概念格 L_1'做比较. 即 L_2无需与 L_1的所有概念都做比较,也就是 L_2仅与 L_1的一个子概念格做比较.

证明　设 $C'(A', B')$为概念格 L_1中除 C_1及其子孙结点以外的其他任一概念: 1)若 C为 C_1的父结点或祖先结点,则有 $A_1 \subset A$成立,因为 C_2与 C_1互为对应概念,根据定义 5有 $A_1 = A_2$,所以 $A_2 \subset A$成立,概念 C_2应作为 C的子结点,这时概念 C就会有 2个外延相同的子结点,所以 C_2不与 C_1的父概念及其祖先概念做比较; 2)若 $C'(A', B')$为除其父结点及其祖先结点外的其他结点,又分为 2种情况:如果 A与 A_1没有任何关系,则 C_2显然没有必要与 $C'(A', B')$做比较. 如果 $A' \cap A_1 \neq \varnothing$,又因为 $A_1 = A_2$,则 $C'(A', B')$与 $C_1(A_1, B_1)$必有相同的子结点,这时 L_2中的结点只与 $C'(A', B')$的子结点做比较就可以,而没有必要与 C做比较.

所以,将 L_2中的概念插入 L_1中更新构造概念格 L时, L_2只需与 L_1中以 C_1为全概念的子格 L_1'做比较.

在利用概念格 L_1、L_2更新构造概念格 L的过程中, L_2的每一概念 C_2与 L_1中具有相同父概念的结点 C_1进行比较,根据外延之间关系的不同采取不同的措施,删除概念 C_1内涵中的分解属性 P,添加新的属性,更新父子关系. 这样逐层自顶向下进行比较,直到 L_2的所有概念比较完毕. 在此过程中,根据概念外延关系的不同会产生以下几类概念:

定义 5　设概念 $C_1(A_1, B_1) \in L_1$, $C_2(A_2, B_2) \in L_2$,若 $A_1 = A_2$并且 $P \in B_1$, C_1变为 $(A_1, (B_1 - P) \cup B_2)$,则称概念 C_1为概念格 L的更新概念.

定义 6　设概念 $C_1(A_1, B_1) \in L_1$, $C_2(A_2, B_2) \in L_2$,若 $A_1 \cap A_2 \neq \varnothing$,则产生一新概念 $C(A_1 \cap A_2, (B_1 \cup B_2) - P)$,称概念 C为 L的特化概念.

定义 7　设概念 $C_1(A_1, B_1) \in L_1$, $C_2(A_2, B_2) \in L_2$,若 $A_2 \subset A_1$,则将概念 $C_2(A_2, B_2)$插入到概念格 L_1中, C_1、C_2分别变为 $(A_1, (B_1 - P))$、$(A_2, (B_1 - P) \cup B_2)$,则概念 C_1、C_2分别称为概念格 L的更新概念和新增概念.

定义 8　设 L_1的任一概念 $C(A, B)$,若 $A \subseteq Q$不成立或 $P \notin B$,则称概念 C为概念格 L的不变概念.

根据文献[17],可以证明如下定理,即利用概念格 L_1与 L_2更新构造的概念格 L与在形式背景 K'上重新构造的概念格 L为同一概念格.

定理 2　若概念 $C(A, B)$为更新构造的概念格 L的任一概念,则概念 $C(A, B)$也为重新构造的概念格 L的一个概念,即若 $\forall C(A, B) \in L$,则 $C(A, B) \in L'$.

证明　因为 $\forall C(A, B) \in L$, 1)若概念 C为 L的不变概念,则 $C \in L_1$. 说明概念 $C(A, B)$与分解属性 P无关,所以在形式背景 K、K中,概念 $C(A, B)$均满足 $f(A) = B$, $g(B) = A$成立,所以 $C(A, B) \in L$; 2)若概念 C为特化概念,则有: $C_1 = (A \cup A_1, B_1) \in L_1$, $C_2 = (A \cup A_2, B_2) \in L_2$,并且 $A_1 \cap A_2 = \varnothing$, $(B_1 - P) \cup B_2 = B$. 那么在 K_1中有 B_1使得: $g(B_1) = A \cup A_1$, $f(A) \supseteq f(A \cup A_1) = B_1$,这时,若 $A_1 = \varnothing$,则 $f(A) = f(A \cup A_1) = B_1$,若 $A_1 \neq \varnothing$,由定义 6,只能有 $f(A) = f(A \cup A_1) = B_1$,这时概念 (A, B_1)不满足概念的完备性,因为 $A \neq g(f(A)) = A \cup A_1$,所以 $(A, B_1) \notin L_1$. 同理可证在形式背景 K_2中有 B_2使得 $g(B_2) = A \cup A_2$, $f(A) = B_2$,即 $(A, B_2) \notin L_2$. 因此,在 K中有 $(B_1 - P) \cup B_2 = B$满足 $g(B) = A$和 $f(A) = B$,即 $C(A, B) \in L$. 同理可证,当概念 C为 L的新增概念和更新概念时, $C(A, B) \in L$成立.

定理 3　若概念 $C(A, B)$为重构概念格 L的任一概念,则概念 C也为更新概念格 L的一个概念,即若 $C(A, B) \in L'$,则 $C(A, B) \in L$.

证明　因为 $C(A, B) \in L'$,假设 $B = B_1 \cup B_2$,并且 $B_1 \in K_1$和 $B_2 \in K_2$,则在概念格 L_1中有: $g(B_1 \cup P) = A \cup A_1$, $f(A \cup A_1) = B_1 \cup P$,在概念格 L_2中有: $g(B_2) = A \cup A_2$, $f(A \cup A_2) = B_2$,并且 $A_1 \cap A_2 = \varnothing$,即有: $C_1(A \cup A_1, B_1 \cup P) \in L_1$, $C_2(A \cup A_2, B_2) \in L_2$. 又因为在形式背景 K中, $f(A) = B = B_1 \cup B_2$,则在 K_1中有 $f(A) = B_1 \cup P$,又因为 $A \neq g(f(A))$,所以 $(A, B_1 \cup P) \notin L_1$. 同理可证, $(A, B_2) \notin L_2$. 根据以上生成概念的定义,概念 $C(A, B) \in L$.

定理 2与定理 3证明了利用概念格 L_2插入概念格 L_1中更新构造的概念格 L与直接在新的形式背景下生成的概念格 L的概念是相同的,在更新构造 L的过程中,格中的父子关系也随着概念的变化进行更新, L中概念和 L中概念有着相同的父子关系,所以概念格 L与概念格 L为同一概念格.

3　基于概念特化的概念格更新构造算法

基于上述分析,基于概念特化的概念格更新构造算法如下:

算法 UCCS(updating constructing based on concept specialization)

输入:概念格 L_1,对象集合 Q,分解属性 P及其分解后的属性集合 $\{P_1, P_2, \cdots, P_n\}$.

输出:概念格 L.

1)在形式背景 $K_2 = (Q, \{P_1, P_2, \cdots, P_n\}, I_2)$上生成概念格 L_2

2) For every node $C_2(A_2, B_2)$ in L_2

3) If Father$(C_2) = \varnothing$

4)调用函数 Find(C_2, C_1')//根据外延寻找 C_2 在 L_1 中的对应概念 $C_1'(A_1', B_1')$

5) $C_1'(A_1', B_1')$更新为 $C_1'(A_1', (B_1' - P) \cup B_2)$

6) End if

7) If father$(C_2) \neq \varnothing$,

8)调用函数 Find()//在 L_1'中寻找 father(C_2) 的对应概念 $C'(A', B')$

9) 执行过程 Compare $(C_2,$ Child $(C'))$ // Child$(C') = C_1(A_1, B_1)$

10) next node

11) End for

12) If Intent$(C_1') = \varnothing$//只判断 L_1'的全概念

13)删除概念 C_1',更新父子关系

14) End if

Find(C_2, C_1') //将 L_1 的概念按外延排序,设 L_1 的任一概念为 $C_1'(A_1', B_1')$

1) For every node C_1' in lay$(|A_2|)$//在 L_1 中,外延个数为 $|A_2|$层次上的每个概念

2) If $A_1 = = A_2$

3) Return $C_1'(A_1', B_1')$

4) End if

5) End for

Compare $(C_2,$ Child $(C'))$//设 Child $(C') = C_1(A_1, B_1)$

1) For every Child(C')

2) If $A_1 = = A_2$//若外延相同,只更新 C_1 的内涵

3) $C_1(A_1, B_1)$更新为 $C_1(A_1, (B_1 - P) \cup B_2)$

4) End if

5) Else if $A_1 \subseteq A_2$//将 C_2 加入 L_1 中

6)增加边 $C_2 -> C_1$

7) $C_2(A_2, B_2)$更新为 $(A_2, B_2 \cup (B' - P))$

8) $C_1(A_1, B_1)$更新为 $(A_1, (B_1 - P) \cup B_2)$

9) End if

10) Else if $A_1 \cap A_2 \neq \varnothing$//产生一特化概念 C''

11)生成特化概念 $C''(A_1 \cap A_2, (B_1 - P) \cup B_2)$

12)增加边 $C_1(A_1, B_1) -> C''(A_1 \cap A_2, (B_1 - P) \cup B_2)$

13)将概念 C_2 加入 L_1 中

14)增加边 $C_2 -> C''$,$C' -> C_2$

15) End if

16) Else //$A_1 \neq A_2$,上述情况除外,若 C_2 和 C' 的每个子结点都不相同,将 C_2 添加到 L_1 中

17) $C_2(A_2, B_2)$更新为 $(A_2, B_2 \cup (B' - P))$

18)增加边 $C' -> C_2(A_2, B_2 \cup (B' - P))$

19) End if

20) End for

在此算法中,时间开销有 2部分组成:L_2 的建格时间和将 L_2 插入到概念格 L_1 中的时间. 由于形式背景 K_2 相对于 K 较小,所以 L_2 的建格时间相对于 L 较小. 设 L_2 的结点数为 N_2,则将 L_2 插入 L_1 中的复杂度为 $O(N_2 * O(\text{Find}()) * O(\text{Compare}()))$. 其中,设 L_1 中外延数为 $|A_2|$的结点个数为 N_1,则 $O(\text{Find}()) = N_1 * |A_2|^2$,设 L_1 中和 C_2 具有相同父结点(父结点的外延相同)的结点数为 N_3,外延的最大个数为 N_4,则 $O(\text{Compare}()) = N_3 * |A_2| * N_4$,设 $N = N_1 * N_2 * N_3 * N_4$,则 $O(N_2 * O(\text{Find}()) * O(\text{Compare}())) = N * |A_2|^3$,算法的复杂度为多项式级. 而经典的 Godin 算法时间复杂度为 $O(2^{2k}||G||)$(k为属性的个数),复杂度是呈指数增长的,显然本文算法较优. 由前面的分析可以看出,分解得到的新属性个数越小,随着数据量的不断增大,本文算法的优越性也越明显.

4 举例分析

表 1为任意形式背景,图 1是其构造出的概念格,将形式背景中的 B 属性分解为属性 B_1、B_2、B_3,B_1、B_2、B_3 形成的形式背景及其构造出的概念格分别为表 2和图 2所示. 采用本文的算法,更新构造新步骤如下.

表 2 形式背景

Table 2 Formal context

	B_1	B_2	B_3
1	√	√	√
3	√	√	√
4		√	

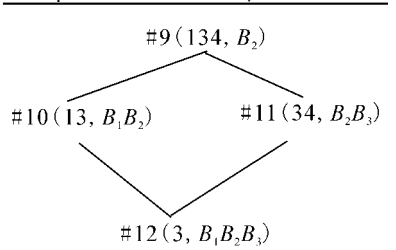


图 2 概念格的 Hasse图

Fig 2 Hasse diagram of concept lattice

首先在图 1中找到图 2中概念 #9$(134, B_2)$的对应概念 #3$(134, B)$,更新概念 #3$(134, B)$为 #3$(134, B_2)$,由定义 7,概念 #3为更新概念. 将概念 #9的每个子概念与概念 #3的子概念做比较,首先,将 #9概念的子结点 #10$(13, B_1B_2)$与概念 #5$(14, ABD)$做比较,概念 #10与概念 #5的外延有交集 1,又因为

#5的子概念中没有外延为 1的子概念,由定义 8产生一特化概念 #13 $(1, AB_1B_2D)$,再将概念 #10 $(13, B_1B_2)$与概念 #7 $(3, BC)$做比较,因为概念 #7的外延 $\{3\}$为概念 #10外延 $\{1, 3\}$的子集,由定义 9,概念 #10 $(13, B_1B_2)$添加到概念格 L_1 中,将概念 #7、#5作为它的子结点,并将 #7更新为 $(3, B_1B_2C)$. 再将 #9的第 2个子结点 #11 $(34, B_2B_3)$与 #3 $(134, B_2)$的每个子结点 #5、#7作比较,产生特化概念 #14 $(4, AB_2B_3D)$, #7 $(3, B_1B_2C)$又被更新为 $(3, B_1B_2\ B_3C)$. 最后将概念 #12 $(3, B_1B_2\ B_3)$与 L_1 中的下一层上的结点做比较, #8更新为 $(\varnothing, AB_1B_2B_3CD)$. 综上所述,在此更新过程中的特化概念有 #13 $(1, AB_1B_2D)$、#14 $(4, AB_2B_3D)$,新增概念有 #11 $(34, B_2B_3)$、#10 $(13, B_1B_2)$,更新概念有 #3 $(134, B_2)$、#5 $(14, A\ B_2D)$、#7 $(3, B_1B_2B_3C)$、#8 $(\varnothing, AB_1B_2B_3CD)$,而概念 #1 $(12345, \varnothing)$、#2 $(1245, A)$、#4 $(23, C)$则为不变概念. 可以很容易看出此算法与在表 3的形式背景重新构造的概念格是相同的,进而证明了此算法的正确性.

表 3　形式背景

Table 3　Formal context

	A	B_1	B_2	B_3	C	D
1	√	√	√			√
2	√				√	
3		√	√	√	√	
4	√		√	√		√
5	√					

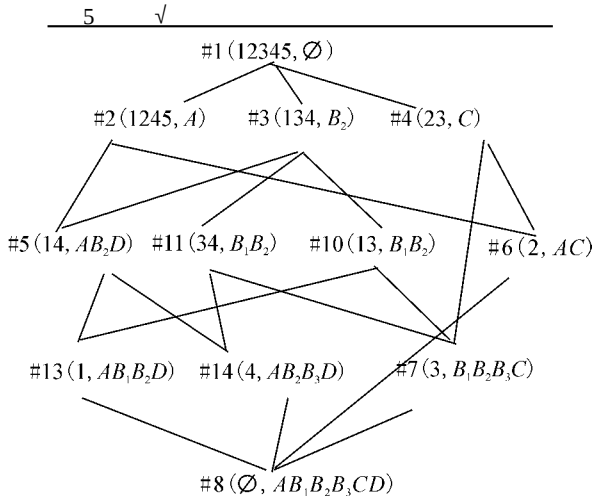


图 3　概念格的 Hasse图

Fig 3　Hasse diagram of concept lattice

5　实验分析

在 Windows XP操作系统,数据库为 Oracle 9i,采用 Visual C++ 6.0实现了 Godin算法和 UCCS算法. 实验数据采用国家天文台提供的恒星光谱数据,将这些数据经过以下处理构成本实验中的形式背景: 1)选定间隔为 20的 150个波长 S3810, S3830, …, S6810作为属性集; 2)依据每一波长处的流量、峰宽和形状,将其离散化为 13种数值之一,并作为该波长处取值. 实验中将波长为 S3850处的属性分解为 S6830、S6850、S6870、S6890、S6910等 5种波长,实验结果与 Godin算法比较的结果如表 4所示:

表 4　不同对象的 Godin算法与 UCCS算法实验比较

Table 4　Experimental comparison between Godin algorithm and UCCS algorithm of different objects

对象数	Godin/s	UCCS/s	生成结点数
2 000	523	19	12 221
3 000	1 250	44	19 155
4 000	2 239	81	25 077
6 000	4 579	122	30 314
8 315	4 649	275	42 679

从实验结果可以看出,UCCS算法更新构造概念格的结点数和 Godin算法生成的结点数是相同的,从而验证了本文算法的正确性. 采用基于链表结构的概念格渐进式构造算法大大提高了寻找相应概念的效率[18]. 当分解后的新属性的个数远小于形式背景中属性的个数时,UCCS算法更新构造概念格的效率远高于利用 Godin算法重新构造概念格的效率. 若新属性和原形式背景中的属性相当,退化为两概念格的合并问题,由文献 [17]知,其效率也明显高于利用传统算法重新构造概念格的效率.

6　结束语

本文利用概念格概念间的泛化与例化关系,给出了一种基于概念特化的概念格更新构造算法 UCCS,当形式背景中某个属性分解,新的概念格产生较之原格更加特化的概念,利用较高层的概念格,只对原格的某个子格进行更新处理,得到了概念较为特化的概念格. 最后,通过与 Godin算法重新构造概念格作比较,实验验证了算法的正确性和有效性.

参考文献:

[1]WILLE R. Restructuring lattice theory: an approach based on hierarchies of concepts[M]. Dordrecht: Reidel, 1982: 415-470.

[2]CHRISTIAN N, KENT R E. Conceptual analysis of resource meta-information[J]. Computer Networks and ISDN System, 1995, 27(6): 973-984.

[3]EKLUND P W, MARTIN P. WWW indexation and document navigation using conceptual structures[C]// 2nd IEEE Conference on Intelligent Information Processing Systems(IC IPS98). [S l]. IEEE Press, 1998: 217-221.

[4] CORBETT D, BURROW A L. Knowledge reuse in SEED exploiting conceptual graphs[C]//International Conference on Conceptual Graphs(ICCG96). Sydney, 1996: 56-60.

[5] GODIN R, HAFEDH M H, MINEAU G W, et al. Design of class hierarchies based on concept(Galois) lattices[J]. Theory and Application of Object Systems, 1998, 4(2): 117-134.

[6] 胡可云,陆玉昌,石纯一. 概念格及其应用进展[J]. 清华大学学报(自然科学版), 2000, 40(9): 77-81.
HU Keyun, LU Yuchang, SHI Chunyi. Advances in concept lattice and its application[J]. J Tsinghua Univ(Sci & Tech), 2000, 40(9): 77-81.

[7] GODIN R. Incremental concept formation algorithm based on Galois (concept) lattices[J]. Computational Intelligence, 1995, 11(2): 246-267.

[8] 谢志鹏,刘宗田. 概念格的快速渐进式构造算法[J]. 计算机学报, 2002, 25(5): 490-496.
XIE ZhiPeng, LIU ZongTian. A fast incremental algorithm for building concept lattice[J]. Chinese J Computers, 2002, 25(5): 490-496.

[9] 李 云,刘宗田,陈 崚,等. 基于属性的概念格渐进式生成算法[J]. 小型微型计算机系统, 2004, 25(10): 1768-1771.
LI Yun, LIU Zongtian, CHEN Ling, et al. Attribute-based incremental formation algorithm of concept lattice[J]. Mini-Micro Systems, 2004, 25(10): 1768-1771.

[10] Van MERWE D, OBIEDKOV S, KOURIE D. AddIntent: a new incremental algorithm for constructing concept lattices[C]//Proc of the Second International Conference on Formal Concept Analysis. Sydney, 2004: 372-385.

[11] WANG Zhihai, HU Keyun, HU Xuegang. General and incremental algorithms of rule extraction based on concept lattice[J]. Chinese J Computers, 1999, 22(1): 66-70.

[12] 胡可云,陆玉昌,石纯一. 基于概念格的分类和关联规则的集成挖掘方法[J]. 软件学报, 2000, 11(11): 1478-1483.
HU Keyun, LU Yuchang, SHI Chunyi. An integrated mining approach for classification and association rule based on concept lattice[J]. Journal of Software, 2000, 11(11): 1478-1483.

[13] 刘宗田,强 宇,周 文,等. 一种模糊概念格模型及其渐进式构造算法[J]. 计算机学报, 2007, 30(2): 184-188.
LIU Zongtian, QIANG Yu, ZHOU Wen, et al. A fuzzy concept lattice model and its incremental construction algorithm[J]. Chinese J Computers, 2007, 30(2): 184-188.

[14] 张继福,张素兰,胡立华. 约束概念格及其构造方法[J]. 智能系统学报, 2006, 1(2): 31-38.
ZHANG Jifu, ZHANG Sulan, HU Lihua. Constrained concept lattice and its construction method[J]. CAAI Transactions on Intelligent Systems, 2006, 1(2): 31-38.

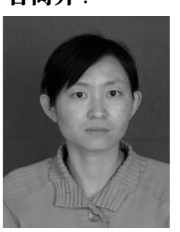
[15] 战立强,刘大昕. 基于概念格的频繁闭项集增量挖掘算法研究[J]. 哈尔滨工程大学学报, 2007, 28(2): 194-198.
ZHANG Liqiang, LIU Daxin. Study on FCI mining algorithm based on concept lattice[J]. Journal of Harbin Engineering University, 2007, 28(2): 194-198.

[16] 屠 莉,陈 崚,李 云. 一种基于属性的概念格生成及维护算法[J]. 计算机应用, 2004, 24(10): 116-118.
TU Li, CHENG Ling, LI Yun. Attribute-based algorithm for constructing and maintenance of concept lattice[J]. Computer Applications, 2004, 24(10): 116-118.

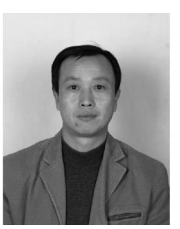
[17] 李 云,刘宗田,陈 崚,等. 多概念格的横向合并算法[J]. 电子学报, 2004, 32(11): 1849-1854.
LI Yun, LIU Zongtian, CHEN Ling, et al. Horizontal union algorithm of multiple concept lattices[J]. Acta Electronica Sinica, 2004, 32(11): 1849-1854.

[18] 蒋义勇,张继福,张素兰. 基于链表结构的概念格渐进式构造[J]. 计算机工程与应用, 2007, 43(11): 178-180.
JIANG Yiyong, ZHANG Jifu, ZHANG Sulan. Incremental construction of concept lattice based on linked list structure[J]. Computer Engineering and Applications, 2007, 43(11): 178-180.

作者简介:



杜秋香,女, 1982年生,硕士研究生,主要研究方向为概念格与数据挖掘.



张继福,男, 1963年生,教授,博士,主要研究方向为数据挖掘、模式识别与智能信息系统. 已发表学术论文 60余篇,其中被 SCI、EI收录 20余篇.

张素兰,女, 1971年生,副教授,主要研究方向为概念格与数据挖掘. 已发表学术论文 20余篇,其中被 SCI、EI收录 10余篇.