基于规则面向属性的数据库归纳的无回溯算法*

周生炳[1] 张 钹[2] 成 栋[3]

[1](空军电讯工程学院 西安 710077)

[2](清华大学计算机科学与技术系 北京 100084)

[3](中国人民大学工商管理学院 北京 100872)

摘要 该文提出了基于规则的面向属性知识发现方法的无回溯算法.把背景知识理解为特殊的逻辑程序,并把它的子句展开为完全归结子句,然后按照用户要求,定义并确定每个属性的恰当层次.每个属性的多个值归纳为恰当层次中的值,只需一遍扫描,因此无需回溯.

关键词 知识发现,概念层次,无回溯算法.

中图法分类号 TP181,TP311

自从第一本有关知识发现(knowlegde discovery in databases,简称KDD)的文集[1]出版以来,有关KDD或数据采掘(data mining)的研究得到广泛的关注[2~4].J. Han等人提出了面向属性(attribute-oriented)的概念提升方法(简称AO方法).其后,他们围绕AO方法进行了一系列深入的工作[5~11].在文献[5]中,他们将AO方法推广为基于规则(rule-based)的AO方法(简称RBAO方法),其主要特点是根据规则对数据库进行两遍扫描以提升概念.我们提出,背景知识可看成是特殊的逻辑程序,在把每个规则展开为完全归结后,可以根据用户要求,确定每个属性概念提升的恰当层次,从而进行一遍扫描即可完成概念提升,避免了回溯.下面首先介绍RBAO方法,然后讨论它的无回溯算法.

1 基于规则的AO方法

AO方法的关键在于用概念层次表示的领域专家提供的背景知识.最常用的概念层次是概念树或概念格[6,7,12],其中概念提升是无条件的.但是,在某些情况下,用户可能希望表达有条件的概念提升,即除了依赖概念本身外,还依赖于别的概念.例如,在城乡居民经济状况调查中,600元的月收入对农民是高收入,但对城市居民则是中等收入,对某些城市的居民甚至是低收入.鉴于此,W. D. Cheung等人把AO方法推广为基于规则的AO方法,加强了AO方法的表达和归纳能力[5,13].

在基于规则的AO方法中,背景知识由一组泛化规则表示.泛化规则形如

$$C(x) \leftarrow A(x)\&B(x).$$

它的意思是,对元组x,某个属性a的概念(属性值)A可以推广到概念C(更高级的属性值),如果x满足条件B.B形如$B_1\&\ldots\&B_n$,B_i是不同于属性a的某个属性b_i的概念,或表示算术比较关系.基于规则的AO方法(简称RBAO方法)分为两步进行.我们以一个例子来说明.

第1步.从初始关系到主关系

尽可能地按属性阈值和泛化规则提升概念,合并相同的元组,得到的广义关系叫做主关系(prime-relation).

例:一个大学学生数据库由模式Student(Name,Status,Sex,Age,Birthplace,GPA)组成,学习任务是发现计

---

* 本文研究得到国家自然科学基金和中国博士后科学基金资助.作者**周生炳**,1962年生,博士后,讲师,主要研究领域为人工智能,逻辑程序,知识发现.**张钹**,1935年生,教授,博士生导师,中国科学院院士,主要研究领域为人工智能,神经网络,计算机应用.**成栋**,1968年生,博士,副教授,主要研究领域为数据仓库技术在营销分析中的应用.

本文通讯联系人:周生炳,西安 710077,空军电讯工程学院四系计算机室

本文1998-01-06收到原稿,1998-09-02收到修改稿

算机专业学生的特征规则.属性GPA的概念图如图1所示,对应的泛化规则如下,生成的主关系见表1.

R_1: $poor \leftarrow \{0.0-1.99\}$;

R_2: $poor \leftarrow \{2.0-2.49\} \& \{graduate\}$;

R_3: $average \leftarrow \{2.0-2.49\} \& \{undergraduate\}$;

R_4: $average \leftarrow \{2.5-2.99\}$;

R_5: $good \leftarrow \{3.0-3.49\}$;

R_6: $good \leftarrow \{3.49-3.8\} \& \{graduate\}$;

R_7: $excellent \leftarrow \{3.49-3.8\} \& \{undergraduate\}$;

R_8: $excellent \leftarrow \{3.9-4.0\}$;

R_9: $weak \leftarrow \{poor\}$;

R_{10}: $weak \leftarrow \{average\} \& \{senior, graduate\}$;

R_{11}: $strong \leftarrow \{average\} \& \{freshman, sophomore, junior\}$;

R_{12}: $strong \leftarrow \{good\}$;

R_{13}: $strong \leftarrow \{excellent\}$.

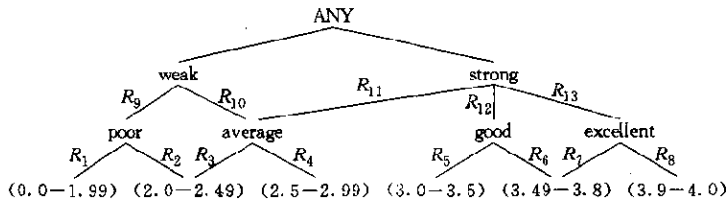

图1

表1

Status	Sex	Age	Birthplace	GPA	Vote
Undergraduate	M	16~25	Canada	Average	40
Undergraduate	M	16~25	Canada	Good	20
Undergraduate	F	16~25	Canada	Excellent	10
⋮	⋮	⋮	⋮	⋮	⋮
Graduate	M	25~30	Canada	Poor	6
Graduate	M	25~30	Canada	Good	4
Graduate	F	25~30	Canada	Excellent	4

第2步.主关系中某些属性应进一步泛化,直到关系的规模小于泛化关系阈值.在第1步泛化后,可能损失一些信息.例如,表1中第1个元组是按R_{10}提升到weak,还是按R_{11}提升到strong是无法确定的,因为身份信息(freshman,sophomore,junior,senior)在第1步泛化中丢失了.事实上,合并到这个元组的40个学生中可能有各个年级的学生.因此,第2步应用回溯算法恢复丢失的信息.

在主关系中,一个广义元组是初始关系的一个元组集合合并的结果,该元组集称为广义元级的源集,而广义元组称为覆盖元组.回溯算法的原理如下:

(1) 把主关系中的元组回溯到它们的源集.在初始关系中加入一个虚拟属性covering-tuple-id(覆盖元组标识符)来记录对应的覆盖元组.

对表1中的主关系应用这一步,结果见表2.

(2) 选择某些属性提升到更高级别,这一步在表2所示的初始关系中进行.

表 2

Name	Status	Sex	Age	Birthplace	GPA	Covering-tuple-id
—	junior	M	20	Vancouver	2.3	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮
—	sophomore	M	21	Calgary	2.3	1
—	freshman	M	18	Toronto	2.4	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮
—	junior	M	19	Ottawa	3.1	2
—	Ph.D	M	30	Waterloo	3.9	40

(3) 比较合并广义元组.对具有相同 covering-tuple-id 和被选属性值的元组进行比较,合并其中相同的元组,得到的结果称为加强主关系(enhanced-prime relation).本例的加强主关系见表 3.

表 3

Covering-tuple-id	GPA	Vote
1	Strong	10
1	Weak	30
2	Strong	20
⋮	⋮	⋮
40	Strong	4

(4) 合并元组映射回主关系并分裂主关系中的元组.加强主关系中具有相同 covering-tuple-id 的所有元组映射为主关系中对应的覆盖元组.主关系中的元组因此分裂为若干个元组,Vote 同时被调整,分裂元组中某些属性泛化为加强主关系中的对应值.这一步的结果见表 4.

表 4

Status	Sex	Age	Birthplace	GPA	Vote
Undergraduate	M	16～25	Canada	Strong	10
Undergraduate	M	16～25	Canada	Weak	30
Undergraduate	F	16～25	Canada	Strong	20
⋮	⋮	⋮	⋮	⋮	⋮
Graduate	F	25～30	Canada	Strong	4

(5) 合并分裂主关系中的广义元组.

2 RBAO 方法的无回溯算法

无条件概念树确定了属性值从一般到特殊的依赖关系,而泛化规则还确定了属性之间的依赖关系.例如,上例GPA 依赖于属性 Status.RBAO 方法的两阶段算法没有考虑这一点,因此回溯是必需的.下面,我们用类似于逻辑程序的某些技术对泛化规则进行变换,按依赖关系对属性排序,按此顺序对属性进行泛化,即可避免回溯.

泛化规则重写为下述形式:

$$(a(x)=C)\leftarrow(a(x)=A)\&B_1(x)\&\ldots\&B_n(x)\,|\,B_1(x)\&\ldots\&B_n(x). \tag{1}$$

其中 $a(x)=C$ 称为规则(1)的头,$(a(x)=A)\&B_1(x)\&\ldots\&B_n(x)$ 或 $B_1(x)\&\ldots\&B_n(x)$ 称为规则(1)的体;$B_i(x)$ 形如 $b_i(x)=B_i$ 或 $b_i(x)*B_i$,$*$ 是比较运算符,a,b_i 是属性名.这个规则的意思是:对元组 x,属性 a 的值 A 提升到更高级值 C,如果属性 b_i 的值为 B_i 或 b_i 的值与 B_i 满足算术关系 $*$(小于,大于等).

属性 a 依赖于 $B_1\&\ldots\&B_n$ 中出现的属性 b_i(如果 $a\neq b_i$),记为 $a<b_i$,并具有传递性:$a<b,b<c$,则 $a<c$.这里,我们只考虑不同属性之间的依赖关系,不考虑不同层次属性值之间的依赖关系.如果对任何属性 a,不存在属性序列 $b_1,\ldots,b_n$,使 $a<b_1,b_1<b_2,\ldots,b_{n-1}<b_n,b_n<a$,则称泛化规则是不循环的.

对规则

$$(a(x)=C)\leftarrow(a(x)=A)\&B_1\&\ldots\&B_n, \tag{2}$$

如果存在下述形式的规则：

$$(a(x)=A)\leftarrow(a(x)=D)\&E_1\&\ldots E_m \tag{3}$$

或

$$(b_i(x)=E_i)\leftarrow(b_i(x)=F_i)\&H_1\&\ldots\&H_k, \tag{4}$$

那么，规则

$$(a(x)=C)\leftarrow(a(x)=D)\&E_1\&\ldots\&E_m\&B_1\&\ldots\&B_n \tag{5}$$

或

$$(a(x)=C)\leftarrow(a(x)=A)\&B_1\&\ldots\&B_{i-1}\&(b_i(x)=F_i)\&H_1\&\ldots\&H_k\&B_{i+1}\&\ldots\&B_n \tag{6}$$

称为规则(2)的一个归结(resolution). 如果规则(2)不存在归结，则称这个规则是一个完全归结.

在属性 a 的规则集合中，概念 C 的定义是以 $a(x)=C$ 为头的所有规则的集合.

对于关系模式 $R(a_1,\ldots,a_n)$ 中的所有属性，如果属性 a 不依赖其他属性(称为独立属性)，那么，不论是否存在概念层次，对它的处理都与 AO 方法相同. 因此，不妨设 $a_1<a_2<\ldots<a_n$(称为依赖链).

无回溯算法的关键是以关于 a_i 的规则的完全归结来代替初始规则，并按 $a_1,\ldots,a_n$ 的顺序进行概念提升. 为了控制泛化过程的终止，需要计算各个属性的概念图中每一层的节点个数.

设 $L(a)$ 表示属性 a 的所有规则的集合，令

$node_a(0)=\{A\mid a(x)=A$ 仅在 $L(a)$ 的规则体中出现$\}$,

$node_a(i+1)=\{C\mid(a(x)=C)\leftarrow(a(x)=A)\&B_1\&\ldots\&B_n\in L(a)$并且 $A\in node_a(i)\}$,

$\max_a=\{C\mid a(x)=C$ 在规则头中出现，但不在体中出现$\}$.

设 $Th(a)$ 表示属性 a 的阈值，属性 a 的恰当层次 S_a 定义如下：

· 如果 $|node_a(0)|\leqslant Th(a)$，则 $S_a=node_a(0)$；

· 如果 $|node_a(k)\cup(\bigcup_{i=0}^{k}node_a(i))\cap\max_a|>Th(a)$，并且 $|node_a(k+1)\cup(\bigcup_{i=0}^{k}node_a(i))\cap\max_a|\leqslant Th(a)$，则 $S_a=node_a(k+1)\cup(\bigcup_{i=0}^{k}node_a(i))\cap\max_a$.

因为有些概念在概念图中比较低的层次上，但它不能再提升，因此，恰当层次并不与概念图中某一层的节点完全相同，还得考虑这种低层概念.

在做了这些准备工作之后，无回溯算法就很简单了.

设 $a_1<a_2<\ldots<a_n$，对每个属性 a_i 指派一个阈值 $Th(a_i)$. 从 a_1 开始，对 S_a 中每个概念 C，求 $a_1=C$ 的定义中每个规则的完全归结，按照这些完全归结把属性 a_1 的值提升为 C. 再对 $a_2,\ldots,a_n$ 重复上述过程，即把所有属性均泛化到恰当的层次. 然后可以去除算术比较条件中属性值较多而不存在概念层次的属性.

把无回溯算法应用于前例.

在关系 Student 的所有属性中，$GPA<status$，其他属性都是独立的. 设 GPA 和 status 的阈值均为 2，则 $S_{GPA}=\{weak,strong\}$，$S_{status}=\{undergraduate,graduate\}$，关于 weak，strong 的完全归结为：

(1) $(GPA=weak)\leftarrow(GPA\in\{2.5-2.99\})\&(status=senior)$；

(2) $(GPA=weak)\leftarrow(GPA\in\{2.0-2.49\})\&(status=senior)$；

(3) $(GPA=weak)\leftarrow(GPA\in\{2.5-2.99\})\&(status=graduate)$；

(4) $(GPA=weak)\leftarrow(GPA\in\{2.0-2.49\})\&(status=graduate)$；

(5) $(GPA=weak)\leftarrow(GPA\in\{0.0-1.99\})$；

(6) $(GPA=strong)\leftarrow(GPA\in\{3.0-3.49\})$；

(7) $(GPA=strong)\leftarrow(GPA\in\{2.0-2.99\})\&(status=freshman)$；

(8) $(GPA=strong)\leftarrow(GPA\in\{2.0-2.99\})\&(status=sophomore)$；

(9) $(GPA=strong)\leftarrow(GPA\in\{2.0-2.99\})\&(status=junior)$；

(10) $(GPA=strong)\leftarrow(GPA\in\{3.5-3.79\})\&(status=graduate)$;

(11) $(GPA=strong)\leftarrow(GPA\in\{3.8-4.0\})$;

(12) $(GPA=strong)\leftarrow(GPA\in\{3.5-3.79\})\&(status=freshman)$;

(13) $(GPA=strong)\leftarrow(GPA\in\{3.5-3.79\})\&(status=sophomore)$;

(14) $(GPA=strong)\leftarrow(GPA\in\{3.5-3.79\})\&(status=jounior)$;

(15) $(GPA=strong)\leftarrow(GPA\in\{3.5-3.79\})\&(status=senior)$.

首先提升属性 GPA，结果见表 5(属性 Age，Birthplace 的泛化与 AO 方法一致，这里假定已经完成).

表 5

Status	Sex	Age	Birthplace	GPA
Junior	M	16～25	Canada	strong
Sophomore	F	16～25	Canada	strong
⋮	⋮	⋮	⋮	⋮
Senior	M	16～25	Canada	weak
⋮	⋮	⋮	⋮	⋮

再泛化属性 Status，结果见表 6.

表 6

Status	Sex	Age	Birthplace	GAP
Undergraduate	M	16～25	Canada	strong
Undergraduate	F	16～25	Canada	strong
⋮	⋮	⋮	⋮	⋮
Graduate	M	16～25	Canada	weak
⋮	⋮	⋮	⋮	⋮

在泛化过程中，相同元组随时合并，记录在属性 Vote 中.

算法的复杂性分解为归纳部分和演绎部分的代价. 归纳部分是有效的，见文献[5]. 由于相对于数据库的规模，规则集合的规模要小得多，而且大多数情况下是简单的条件规则，因此，这部分复杂性可以忽略不计.

H. Huang 和 W. C. Fu 提出了 AO 方法和 RBAO 方法的另一种有效实现[13]. 但他们的算法要求概念图是平衡的，即每条从极小节点到极大节点的路径的长度相等. 另外，该方法不能处理算术比较条件. 我们的无回溯算法统一处理算术比较条件，因而在这两方面优于他们的算法.

参考文献

1 Piatetsky-Shapiro G, Frawley W J. Knowledge Discovery in Databases. Menlo Park, CA: AAAI/MIT Press, 1991

2 Chen M, Han J, Yu P S. Data mining: an overview from database perspective. IEEE Transactions on Knowledge and Data Engineering, 1996,8(6):866～883

3 Silberschatz A, Stonebraker M, Ullman J D. Database research: achievements and opportunities into the 21st century. SIGMOD Record, 1996,25(1):52～63

4 Ziarko W. Rough Sets, Fuzzy Sets and Knowledge Discovery. Berlin: Springer-Verlag, 1994

5 Cheung D W, Fu A W C, Han J. Knowledge discovery in databases: a rule based attribute oriented approach. In: Zbigniew R ed. Methodologies for Intelligent systems: 8th International Symposium. Berlin: Springer-Verlag, 1994. 164～173

6 Han J, Cai Y, Cercone N. Data driven discovery of quantitative rules in relational databases. IEEE Transactions on Knowledge and Data Engineering, 1993,5(1):29～40

7 Han J. Towards efficient induction mechanisms in database systems. Theoretical Computer Science, 1994,133(1):161～185

8 Han J, Fu Y. Dynamic generation and refinement of concept hierarchies for knowledge discovery in databases. In: Fayyad U M, Uthurusamy R eds. Proceedings of the KDD'94: the AAAI'94 Workshop on Knowledge Discovery in

Databases. AAAI Technical Report, WS-94-03. Menlo Park, CA: AAAI Press, 1994. 157～168

9 Han J, Fu Y. Exploration of the power of attribute-oriented induction in data mining. In: Fayyad U M *et al* eds. Advances in Knowledge Discovery and Data Mining. Menlo Park, CA: AAAI/MIT Press, 1996. 399～421

10 Han J, Huang Y, Cercone N *et al*. Intelligent query answering by knowledge discovery techniques. IEEE Transactions on Knowledge and Data Engineering, 1996,8(3):373～390

11 Hu X, Cercone N. Learning in relational databases: a rough set approach. Computational Intelligence, 1995,11(2):323～338

12 Godin R, Missaoui R. An incremental concept formation approach for learning from databases. Theoretical Computer Science, 1994,133(2):387～419

13 Huang Y, Fu W C. Efficient algorithms for attribute-oriented induction. In: Fayyad U M, Uthurusamy R eds. Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining. Menlo Park, CA: AAAI Press, 1995. 168～173

A Nonbacktracking Algorithm for the Rule Based Attribute-oriented Database Induction Approach

ZHOU Sheng-bing[1] ZHANG Bo[2] CHENG Dong[3]

[1](*Air Force Telecommunication Engineering Institute Xi'an* 710077)

[2](*Department of Computer Science and Technology Tsinghua University Beijing* 100084)

[3](*Business School Renmin University of China Beijing* 100872)

Abstract In this paper, a nonbacktracking algorithm is presented for the rule based attribute-oriented database induction approach. The background knowledge is considered as a limited logic program, and every clause of it is expanded as a complete resolution clause. Then, the concept of exact level for an attribute in the concept hierarchies is defined and specified according to its threshold required by the users. The values of each attribute are inducted as ones in its exact level by only one pass without backtracking.

Key words Knowledge discovery in databases, concept hierarchies, nonbacktracking algorithm.