一种基于约束的关联规则挖掘算法

李广原[1,2] 杨炳儒[1] 周如旗[3]
(北京科技大学计算机与通信工程学院 北京 100083)[1]
(广西师范学院计算机与信息工程学院 南宁 530023)[2] (广东第二师范学院计算机科学系 广州 510303)[3]

摘 要 基于约束的关联规则挖掘是一种重要的关联挖掘,能按照用户给出的条件来实行有针对性的挖掘。大多数此类算法仅处理具有一种约束的挖掘,因而其应用受到一定程度的限制。提出一种新的基于约束的关联规则挖掘算法MCAL,它同时处理两种类型的约束:非单调性约束和单调性约束。算法包括3个步骤:第一步,挖掘当前数据集的频繁1项集;第二,应用约束的性质和有效剪枝策略来寻找约束点,同时生成频繁项的条件数据库;最后,递归地应用前面两步寻找条件数据库中频繁项的约束点,以生成满足约束的全部频繁项集。通过实验对比,无论从运行时间还是可扩展性来说,本算法均达到较好的效果。

关键词 数据挖掘,关联规则挖掘,约束关联挖掘

中图法分类号 TP18 **文献标识码** A

Efficient Algorithm for Mining Association Rules with Constraints

LI Guang-yuan[1,2] YANG Bing-ru[1] ZHOU Ru-qi[3]
(School of Computer & Communication Engineering, University of Science & Technology Beijing, Beijing 100083, China)[1]
(School of Computer and Information Engineering, Guangxi Teachers Education University, Nanning 530023, China)[2]
(Department of Computer Science, Guangdong University of Education, Guangzhou 510303, China)[3]

Abstract Association rules mining with constraints is an important association mining method, and it can mine the rules according to the users needs. Most of algorithms deal with one constraint, but in the reality applications, usually there are two or more constraints. In this paper, a novel algorithm for mining association rules with constraint was proposed. It can deal with two constraints simultaneously, namely constraint of anti-monotone and constraint of monotone. The algorithm consists of three phases, first, frequent 1-itemsets are collected over the dataset, second, we apply some prune techniques to the constraints check and a conditional database is generated, and at the end, the final frequent itemsets which are satisfied with the constraints are generated. Experimental results show that the proposed algorithm is efficient both in run time and scalability.

Keywords Data mining, Association rules mining, Association rules mining with constraint

1 引言

关联规则挖掘是数据挖掘的一个重要任务。根据挖掘数据的性质和任务,现已提出了不少有效的算法有 Apriori 算法及基于此算法的改进算法;其它算法有划分算法、抽样算法、挖掘动态数据的增量算法和并行与分布算法、挖掘多层规则的算法、模糊关联挖掘算法等等。关联挖掘的一个重要步骤是频繁项集挖掘。当数据量很大时,所生成的项集数量以及由此生成的规则会非常大,而有些规则对用户来说可能是不相关的或不感兴趣的。为控制规则输出的数量,通常设置支持度和置信度两个参数,控制规则的输出量。但此举的局限性是明显的,主要存在3个问题[1]:第一,用户不能参与到挖掘过程中;第二,不能进行聚焦式的挖掘;第三,关联性定义严格。3个问题的存在使得挖掘缺乏灵活性、挖掘结果不能满足需要。约束关联挖掘能够允许用户通过设置相关的聚焦参数,从而使挖掘能够按照相关参数的要求来进行。根据约束的性质,约束可分为单调性约束、非单调性约束、简洁性约束、可转变的约束。大多数基于约束的挖掘算法仅处理其中的一个约束,主要原因在于存在两个或以上的约束时,在约束点上,往往其中一个约束对于在约束点处的项集及其子集总是满足的;而另一个约束对它的超集总是满足的,这样会增加处理约束的难度。本文提出一个新的基于约束的关联规则挖掘算法,算法包括3个步骤:第一步,挖掘数据集中的频繁1项集;第二,应用约束的性质和有效剪枝策略来寻找约束点,同时生成频繁项的条件数据库;最后,递归地应用前面两步来约束点条件数据库中的频繁项,以生成满足约束的全部频繁项集。通过实验对比,无论从运行时间还是可扩展性来说,本算法均达到较好的效果。

到稿日期:2011-02-21 返修日期:2011-03-29 本文受国家自然科学基金(60875029)资助。

李广原(1969—),男,博士生,副教授,CCF会员,主要研究方向为数据挖掘,E-mail:guangyuanli2004@163.com;杨炳儒(1943—),男,教授,博士生导师,主要研究方向为知识发现、柔性建模与集成技术;周如旗(1971—),男,副教授,主要研究方向为机器学习。

2 相关研究

文献[2]给出了第一个基于约束的关联规则挖掘算法;文献[3]利用单调性约束、非单调性约束和简洁性约束来对搜索频繁项集的空间进行剪枝;文献[4,5]给出了一个基于 FP-Growth 的算法 FIC^M 来求解这两类约束,并且介绍一种新的约束——可转变约束,该算法在剪枝前产生最大频繁项集,并且尽早检查单调性约束;文献[6]给出了算法 Dualminer,它是第一个同时处理两种类型的约束,但算法有不少限制和性能方面的问题。首先,它基于 MAFIA,而 MAFIA 产生的所有频繁模式间的支持度是没有关联的;其次,它假设整个数据集能够放在内存当中,这在实际中往往是不能做到的;最后,它采用自顶向下的寻找单调性约束的方式,使得相对于大数据集来说进行了许多无用的检查操作。文献[7]对 Dualminer 进行了并行化处理,指出在挖掘包含有 10k 到 100k 之间事务的相关性比较小的稀疏数据集时,Dualminer 要花费很多的时间进行处理,但作者并没有给出实验去进行验证。文献[8,9]在处理单调性和非单调性约束时,指出通过剔除不满足约束的事务来预处理数据,然后在缩减后的数据集上进行频繁项集挖掘。但这种方法存在缺点:首先它通过递归的方法把缩减的数据集写回磁盘,这会造成很大的 I/O 开销;另外,对初始的单调性约束的检查结果非常敏感。文献[10]给出一种新的约束,称为松散的非单调性约束,并提出通过新的数据约简策略在基于 Apriori 算法上处理这类新的约束的方法。文献[11]给出一种动态的基于约束的频繁集挖掘算法 DCF,它采用了若干优化技术来处理一个称之为分割支持图的结构,使算法获得项集支持度的界限,而且能够更好地利用约束的特性。当处理来自动态的约束时,算法根据约束增量的生成函数来精确产生满足新的约束但不满足旧约束的项集。

3 问题描述

设 $I=\{i_1,i_2,\cdots,i_n\}$ 为 n 个项,$X\subseteq I$ 称为项集,如果 X 包含有 k 个项,则称 X 为 k 项集。一个事务 t 是一个 2 元组 $\langle ID_t, I_t\rangle$,其中 ID_t 是事务 t 的标识,$I_t\subseteq I$ 是 t 包含的项,项集 X 的支持度记为 $\mathrm{Supp}(X)$,是指包含 X 的事务数占数据集中所有事务数的百分比。另设事务的支持度阈值为 ε,如果 $\mathrm{Supp}(X)\geqslant\varepsilon$,则 X 为频繁项集。

作用于项 I 的约束 C 可看成是一个谓词,表示为 $C:2^I\rightarrow\{\mathrm{true},\mathrm{false}\}$。一个项集 X 满足约束 C,当且仅当 $C(X)=\mathrm{true}$,数据集中所有满足约束 C 的项集可表示为 $S_C(I)=\{S\mid S\subseteq I\wedge C(S)=\mathrm{true}\}$。

定义 1 多维约束是指作用于多维属性上的约束。

这里的多维属性是指其中的一个属性有两个属性项。比如把一个商品看成是一个项,则该商品有若干属性,比如成本、价格等。

定义 2 给定一个约束 C,对于任一项集 X,如果 $\forall Y\subseteq X$,当 $C(X)=\mathrm{true}\Rightarrow C(Y)=\mathrm{true}$ 时,约束 C 属于非单调性约束。

在频繁项集挖掘中,频繁约束就是一个非单调约束。Apriori 算法利用了这个约束特性,即向下封闭性。一个频繁项集,其子集也是频繁的。

定义 3 给定一个约束 C,对于任一项集 X,如果 $\forall Y\supseteq X$,当 $C(X)=\mathrm{true}\Rightarrow C(Y)=\mathrm{true}$ 时,约束 C 属于单调性约束。即一个项集,当它满足一个单调性约束时,它的任一超集也满足该约束。

4 MCAL 算法

4.1 算法的基本思想

MCAL 算法同时处理两类约束,即非单调性约束和单调性约束。我们以表 1 所列的数据库为例来说明算法过程,表 2 是有关数据库中项的属性描述。

表 1 事务数据库 D

TID	Items
1	A,B,D,E
2	A,B,D,C
3	A,E
4	A,C,D,E
5	A,B,C,D,E
6	B,D,E
7	B,C,D
8	C,D,E
9	D,E
10	B,D,E

表 2 事务数据库 D 中的项及其属性

ItemID	cost	price
A	40	50
B	30	35
C	55	60
D	25	42
E	20	45

在给出的示例数据库中,我们考虑两类具体的约束:$\max(S.\mathrm{cost})\leqslant\min(S.\mathrm{price})$ 和 $\mathrm{total}(S.\mathrm{price})\geqslant 100$,其中 S 是项集,S 中的每一个项有两个属性 cost 和 price。$\max(S.\mathrm{cost})$ 是指在 S 的所有项中属性 cost 最大的值,而 $\min(S.\mathrm{price})$ 是指在 S 的所有项中属性 price 最小的值。显然,第一个约束是非单调性约束,第二个约束是单调性约束。假设用 C_1 代表约束条件 $\max(S.\mathrm{cost})\leqslant\min(S.\mathrm{price})$,用 C_2 代表约束条件 $\mathrm{total}(S.\mathrm{price})\geqslant 100$,则算法的结果是输出符合约束条件 $C_1\wedge C_2$ 的数据库中的频繁项集。

定义 4[12] (1)设 pc 代表两个序列之间具有前缀关系的谓词,设有两个序列 s_1,s_2,如果 $pc(s_1,s_2)=\mathrm{true}$,则 s_1 是 s_2 的前缀,比如 $s_1=ab,s_2=abcd$,那么 $pc(s_1,s_2)=\mathrm{true}$,$s_1$ 是 s_2 的前缀。

(2)项集 β 称为 α 的最大投影事务集 $\langle tid, I_t\rangle$,当且仅当 1)$\alpha\subseteq I_t,\beta\subseteq I_t$;2)$pc(\alpha,\beta)=\mathrm{true}$;3)不存在 β 的超集 γ,使得 $\gamma\subseteq I_t,pc(\alpha,\gamma)=\mathrm{true}$。

(3)α 的条件数据库是包含 α 的最大 α 的投影事务的集合。这里,α 条件数据库是采用 FP-tree 作为它的结构,基于 FP-true 结构的 α 条件数据库记作 $T|_\alpha$。

定义 5[12] 设 α 是一个频繁项集,λ 是 α 条件数据中的频繁项集的集合,则 $\alpha\cup\lambda$ 就构成了 $T|_\alpha$ 中的潜在的最大频繁项集。

在描述算法前,先给出几个引理和定理。

引理 1[12] 设 β 是 $T|_\alpha$ 中频繁项的集合,γ 是 $T|_{\{a\}\cup\alpha}$ 中的频繁项集合的子集,C 为非单调性约束,如果在 $T|_\alpha$ 中,$C(\alpha\cup\beta)=\mathrm{true}$,则 $C(\{a\}\cup\alpha\cup\gamma)=\mathrm{true}(a\in\beta)$。

证明:由于 $\{a\}\cup\alpha\cup\gamma\subseteq\alpha\cup\beta$,而 C 为非单调性约束,由非

单调约束的性质马上可以得到结论。

引理 2[12] 设 γ 是 $T|_{\{a\}\cup\alpha}$ 中频繁项的集合中的子集，C 为非单调性约束，如果在 $T|_\alpha$ 中，$C(\{a\}\cup\alpha)=\text{false}$，$a$ 是 $T|_\alpha$ 中的一个频繁项，那么不需要生成 $T|_{\{a\}\cup\alpha}$，因为 $\{a\}\cup\alpha\cup\gamma$ 包含 $\{a\}\cup\alpha$，$C(\{a\}\cup\alpha\cup\gamma)=\text{false}$。

引理 2 的证明和引理 1 类似。

引理 3 设 β 是 $T|_\alpha$ 中频繁项的集合，γ 是 $T|_{\{a\}\cup\alpha}$ 中的频繁项集合的子集，C 为单调性约束，如果在 $T|_\alpha$ 中，$C(\alpha\cup\beta)=\text{false}$，则 $C(\{a\}\cup\alpha\cup\gamma)=\text{false}$，$a\in\beta$。

证明：根据单调性约束的性质直接得到结论。

引理 4 设 γ 是 $T|_{\{a\}\cup\alpha}$ 中频繁项的集合中的子集，C 为单调性约束，如果在 $T|_\alpha$ 中，$C(\{a\}\cup\alpha)=\text{true}$，$a$ 是 $T|_\alpha$ 中的一个频繁项，那么不需要生成 $T|_{\{a\}\cup\alpha}$，因为 $\{a\}\cup\alpha\cup\gamma$ 包含 $\{a\}\cup\alpha$，$C(\{a\}\cup\alpha\cup\gamma)=\text{true}$。

证明和引理 3 类似。

引理 5 设 $X=\{x_1,x_2,\cdots,x_n\}(n\geqslant 2)$ 为一项集，C 为单调性约束，且 $C(X)=C(x_1x_2,\cdots,x_n)=\text{true}$，设 $x_1,x_2,\cdots,x_n$ 是按其与 C 有关的其属性值从大到小的顺序排列（如对于上例中的 total(S. price)≥100，按 price 值从大到小排列），如果删除前 $k(k\geqslant 1)$ 个最小属性值的项，剩余项组成的项集不满足约束 C，则对于任何 k 个项，删除它们后，剩余项集不满足约束 C。

引理 5 作为算法的一个剪枝策略。

定理 1 设 X 为一项集，C_1 和 C_2 分别为单调性约束和非单调性约束。

(1)如果 $C_1(X)=\text{false}$，那么 $\forall Y\subseteq X, C_1(Y)\wedge C_2(Y)=\text{false}$，即 X 的任何子集都不同时满足 C_1 和 C_2；(2)如果 $C_1(X)=\text{true}$，$C_2(X)=\text{true}$，则存在 $Y\subseteq X$，使得 $C_1(Y)\wedge C_2(Y)=\text{true}$；(3)如果 $C_1(X)=\text{true}$，$C_2(X)=\text{false}$，则存在 $Y\subseteq X$，使得 $C_1(Y)\wedge C_2(Y)=\text{true}$。

证明：(1)如果 $C_1(X)=\text{false}$，很显然 $\forall Y\subseteq X$。根据单调约束的性质，$C_1(Y)=\text{false}$，于是不管 $C_2(Y)$ 的值如何，$C_1(Y)\wedge C_2(Y)=\text{false}$。

(2)如果 $C_1(X)=\text{true}$，$C_2(X)=\text{true}$，对于约束 C_1，存在 $Y\subseteq X$，使得 $C_1(Y)=\text{true}$。对于约束 C_2，$\forall Y\subseteq X, C_2(Y)=\text{true}$。综上，则存在 $Y\subseteq X$，使得 $C_1(Y)\wedge C_2(Y)=\text{true}$。

(3) 如果 $C_1(X)=\text{true}$，$C_2(X)=\text{false}$，存在 $Y\subseteq X$，使得 $C_1(Y)=\text{true}$，可能存在 $Y\subseteq X$，使得 $C_2(Y)=\text{true}$，于是可能存在 $Y\subseteq X$，使得 $C_1(Y)\wedge C_2(Y)=\text{true}$。

定理 2 数据集所有的频繁项集是数据集所有频繁 1 项集组成的项集的子集。

由频繁项集的子集也是频繁的这个性质可以推出结论。

4.2 算法描述

算法 MCAL

输入：数据库 D，最小支持度 ε，非单调性约束 max(S. cost)≤min(S. price)用 C_1 表示，单调性约束 total(S. price)≥100，用 C_2 表示。数据库中的事务要先转换成 FP-tree 结构。

输出：所有同时满足 C_1 和 C_2 的频繁项集。

算法的步骤如下：

```
MCAL(α, T|α)
(1)从 T|α 的 FP-tree 头表中得到频繁项集 L 以及它们的支持度。
(2)β=L;if(C2(β∪α)=false,then exit,there are no frequent itemsets
   that satisgy C1∧C2。
(3)if C2(β∪α)=true
   应用引理 5 得出满足约束 C2 的项集中项的个数的大小 N。
(4)for each a∈β
   MCAL(α, T|α)
   if (|L|)<N // 生成的频繁项体数小于 N
   continue
   else
     for each χ∈L
     if C1(χ∪a)=true
     生成 Tχ∪a,if (|L|)>N
       输出生成的频繁项集
       endfor
   endfor
```

下面用例子加以说明。

以表 1 作为例子，假设非单调性约束 C_1：max(S. cost)≤min(S. price)，单调性约束 C_2：total(S. price)≥100，最小支持度为 20%。

(1)求出数据库频繁 1 项集（按支持度由大到小排列）：D,E,B,A,C。

(2)$C_2(DEBAC)=\text{false}$，得出 N 值至少大于等于 2。

(3)$C_1(D)=\text{true}$，$C_1(E)=\text{true}$，$C_1(B)=\text{true}$，$C_1(A)=\text{true}$，$C_1(C)=\text{true}$。

下面仅以 A 为例。注意，D 的支持度最大，没有必要生成 D 的条件数据库，因为它的支持度最大，已经没有前缀项了。

(4)求出 A 的条件数据库，其投影事务在原数据库中分别为（项按支持度由大到小排列）：$\{DEB,DB,E,DE,DEB\}$。

(5)求出频繁 1 项集：D,E,B。

由于 $DEBA$ 为 4 项，大于 N 的值，因此后面考虑继续生成各项的条件数据库。

(6)$C_1(DEBA)=\text{false}$；$C_1(DA)=\text{true}$；$C_1(EA)=\text{true}$；$C_1(BA)=\text{false}$。

(7) 根据引理 2、引理 3，不用生成 BA 的条件数据库。DA 已没有前缀，但 $C_2(DA)=\text{false}$，仅需生成 EA 的条件数据库，得到投影事务集为 $\{D,D,D,D\}$，D 为条件数据库中的频繁 1 项集，$C_1(DEA)=\text{true}$，$C_2(DEA)=\text{true}$。

(8)由于 DEA 的子集除了 DEA 是 3 项外，其余为小于等于 2，经检查，均不满足 C_2，最后输出满足 $C_1\wedge C_2$ 的 A 的条件数据库中的频繁项集为 DEA。

程序的正确性可由定理 2 推证得到。

5 实验分析

为了测试算法的性能，我们选择算法 FP-growth+ 作为比较的对象。FP-growth+ 采用 FP-growth 算法来找出所有频繁项集，然后从中筛选出满足约束的频繁项集。测试程序运行在 Windows XP 系统上，CPU 为 PIV2.4，内存为 2G，编程语言为 C++，数据集采用文献[13]的方法生成测试数据集。对每个项赋予两个属性，即 cost 和 price，属性值采用随机赋值。所生成的数据集的参数如表 3 所列。

表 3 中的参数 δ，其含义是指数据集中不满足约束的频繁项集占全部频繁项集的百分比，$\delta=0$ 代表每一个频繁项集都满足约束，$\delta=100$ 代表没有一个频繁项集满足约束。

表 3　测试用的相关参数

参数符号	含义	设置大小
N	项的数目	300
M	项的属性个数	2
|T|	事务的个数	20K-100KB
V	项集的平均长度	20
F	最大频繁项集的平均长度	15
ε	最小支持度	0.001～0.01
δ	选择约束率	0～100%

实验结果如图 1—图 3 所示。其中图 1 是测试没有满足约束的频繁项集所占比例对运行时间产生的影响，图 2 是测试事务大小对可扩展性的影响，图 3 是不同的支持度对运行时间的影响。

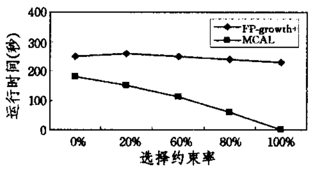

图 1　不同选择约束率和运行时间对应图

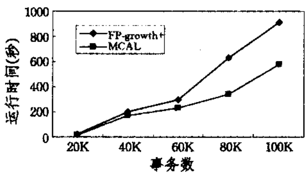

图 2　不同事务数和运行时间的对应图

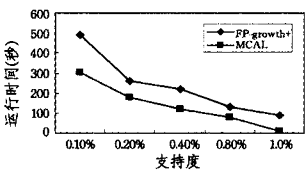

图 3　不同支持度和运行时间的对应图

从实验结果可以看出，MCAL 算法是有效的，且扩展性能较好，它充分利用非单调性约束和单调性约束的性质并与有效的剪枝技术相结合，大大减少了不必要的扫描，使其在运行时间和可扩展性方面均优于 FP-growth+算法。

结束语　本文给出一个多约束关联挖掘算法。通过实验对比，无论从运行时间还是可扩展性来说，MCAL 算法均获得了较好的性能。约束关联挖掘是一种重要的关联挖掘，以有效的剪枝策略，结合约束的性质来挖掘是主要的方法。基于动态、多关系、分布式数据的多维、多约束挖掘是今后研究的一个主要方向。

参 考 文 献

[1] Ng R T, Lakshmannan L V S, Han Jia-wei. Exploratory mining and pruning optimizations of constrained associations rules[C]// Proceedings ACM SIGMOD International Conference on Management of Data. Seattle, Washington, USA, June 1998

[2] Srikant R, Vu Q, Agrawal R. Mining association rules with item constraints[C]// Proceedings of ACM International Conference on Knowledge Discovery and Data Mining. 1997:67-73

[3] Lakshmanan L, Ng R, Han J, et al. Optimization of constrained frequent set queries with 2-variable constraints[C]// ACM SIGMOD Conference on Management of Data. 1999:157-168

[4] Pie J, Han J. Can we push more constraints into frequent pattern mining? [C]// ACM SIGKDD Conference. 2000:350-354

[5] Pie J, Han J, Lakshmanan L. Mining frequent itemsets with convertible constraints[C]// IEEE ICDE Conference. 2001:433-442

[6] Bucila C, Gehrke J, Kifer D, et al. Dualminer: Adual-pruning algorithm for itemsets with constraints[C]// EightACM SIGKDD Internationa Conf. on Knowledge Discovery and Data Mining. Edmonton, Alberta, August 2002:42-51

[7] Ting R M, Bailey J, Ramamohanarao K. Paradualminer: An efficient parallel implementation of the dualminer algorithm[C]// Eight Pacific-Asia Conference (PAKDD 2004). Sydney, Australia, May 2004:96-105

[8] Bonchi F, Giannotti F, Mazzanti A, et al. Examiner: Optimized level-wise frequent pattern mining with monotone constraints [C]// IEEE ICDM. Melbourne, Florida, November 2004

[9] Bonchi F, Lucchese C. On closed constrained frequent pattern mining[C] // IEEE International Conference on Data Mining Brighton. UK, November 2004

[10] Bonchi F, Lucchese C, Trasarti R. Pushing tougher constraints in frequent pattern mining[C]// 9th Pacific-Asia Conference on Knowledge Discovery and Data Mining. Hanoi, Vietnam, May 2005

[11] Laks V, Lakshmanan S, Ng R T. Efficient dynamic mining of constrained frequent sets[J]. ACM Transactions on Database Systems, 2003, 28(4)

[12] Anthony J, Lee T, Lin Wan-chuen, et al. Mining association rules with multi-dimensional constraints[J]. The Journal of Systems and Software, 2006(79):79-92

[13] Agrawal R, Srikant R. Fast algorithms for mining association rules[C] // Proceedings of International Conference on Very Large Data Bases. 1994:487-489

(上接第 227 页)

[14] Wang Guo-yin, et al. Theoretical study on attribute reduction of rough set theory: comparison of algebra and information views [C]// Proceedings of the Third IEEE International Conference on Cognitive Informatics. Canada: IEEE Computer Society, 2004:148-155

[15] Fleuret F. Fast Binary Feature Selection with Conditional Mutual Information[J]. Journal of Machine Learning Research, 2004, 5: 1531-1555

[16] 胡清华，于达仁，谢宗霞. 基于邻域粒化和粗糙逼近的数值属性约简[J]. 软件学报, 2008, 19(3): 640-649

[17] 杨明. 一种基于一致性准则的属性约简算法[J]. 计算机学报, 2010, 33(2): 231-239

[18] Hu Qing-hua, Yu D, Xie Zong-xia, et al. EROS: Ensemble rough subspaces[J]. Pattern Recognition, 2007: 3728-3739

[19] Kryzkiewicz M. Comparative study of alternative types of knowledge reduction in inconsistent systems [J]. International Journal of Intelligent Systems, 2001, 16: 105-120

[20] 张文修，米据生，吴伟志. 不协调目标信息系统的知识约简[J]. 计算机学报, 2003, 26(1): 12-18

[21] Kurgan L A, Cios K J. CAIM Discretization Algorithm[J]. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(2): 145-153

[22] Pawlak Z. Rough Sets: Theoretical Aspects of Reasoning About Data[M]. Dordrecht: Kluwer Academic Publishers, 1991