一种基于扩展差别矩阵的规则获取方法*)

林晓斌　叶东毅
(福州大学数学与计算机学院　福州 350002)

摘　要　本文对 Skowron 差别矩阵[1]进行扩展，提出了一种不求决策表属性约简，直接获取决策规则的方法。此外，该方法只需根据同一决策类中扩展差别矩阵元素来提取规则，且不会造成规则冲突，适合于分布规则提取。

关键词　粗糙集，规则获取，差别矩阵

A Rule Acquisition Method Based on Extended Discernibility Matrix

LIN Xiao-Bin　YE Dong-Yi
(College of Mathematics and Computer, Fuzhou University, Fuzhou 350002)

Abstract　In this paper, by expanding Skowron's discernibility matrix, we put forward a method to extract rules from a decision table. The method processes the data in advance so that the extraction of classification rules can be efficiently done in a distributed way. Moreover, the method allows direct acquisition of rules without computing attribute reductions.

Keywords　Rough sets, Rules obtaining, Discernibility matrix

粗糙集理论自 1982 年由 Pawlak 教授[2]提出以来，在机器学习和数据挖掘等多个领域中得到了广泛的应用[3,4]。在求核、约简和规则提取方面已经有大量的研究工作[5,6]。Skowron 教授等人提出的基于差别矩阵的方法[1]以及对其进行的在不相容决策表中的改正算法[7]被大量应用在求核和约简算法中。在提取规则的过程中，将不同决策类的数据分开提取规则会导致错误，这阻碍了分布计算规则的实现，在公开文献中未发现分布式规则提取的相关论述。另外，在现有的规则提取算法中，求取决策规则的基本思想多是先求得整个论域的核 CORE(U,D)，而后求约简，再从所求得的约简中对规则逐条地求出"值核"[3]。该类算法可能存在的一个问题是 CORE(U,D)中的属性对于某一具体的规则来说可能不全是核值，甚至可能核值属性簇与 CORE 的交为空，若出现这一情况，则求 CORE 对该规则来说就是无意义的。因此研究一种无需求核或约简、直接得到规则的算法是有意义的。本文的目的是通过对 Skowron 差别矩阵进行扩展，使得计算出差别矩阵后，可以直接仅由同一决策类中矩阵元素的信息得到不冲突的规则。该算法为分布式规则提取提供了可能，本文的理论分析部分将给出其正确性的证明。

1　基本概念

一个决策表是如下形式的四元组[3]

$L=(U,C\cup D,V,F)$

其中 $U=\{x_1,x_2,\cdots,x_m\}$ 是论域，$C=\{a_1,a_2,\cdots,a_n\}$ 为条件属性集合，D 为决策属性集合，不失一般性，假设仅一个决策属性 D，其取值为 $1,2,\cdots,k$。$IND(D)$ 是 D 上的等价关系。$U/IND(D)$ 是由决策属性 D 导出的等价类，记为 $U/IND(D)=\{Y_1,Y_2,\cdots,Y_k\}$，是 U 的一个划分。

定义 1　设 U 是一个论域，P 是定义在 U 上的一个等价关系簇，$R\in P$。如果 $IND(P\backslash\{R\})=IND(P)$，则称关系 R 在 P 中是绝对不必要的；否则，称 R 在 P 中是绝对必要的。

定义 2　设 U 是一个论域，P 是定义在 U 上的一个等价关系簇，$R\in P$。如果每个关系 $R\in P$ 在 P 中都是绝对必要的，则称关系簇 P 是独立的，否则，称 P 是相互依赖的。

定义 3　设 U 是一个论域，P 是定义在 U 上的一个等价关系簇，P 中所有的绝对必要关系组成的集合称为关系簇 P 的绝对核，记作 $CORE(P)$。

定义 4　设 U 是一个论域，P 和 Q 为定义在 U 上的两个等价关系簇且 $R\subseteq P$，如果

(1) $IND(Q)=IND(P)$；

(2) Q 是独立的

则称 Q 是 P 的一个绝对约简。

定义 5　$\Lambda(a_i,c_{ij})\rightarrow D=d, d=1\cdots k$ 为决策规则集，记 $RULE=\Lambda(a_i,c_{ij})$，若 $x\in U$ 满足 $RULE$，称 $RULE$ 覆盖 x，否则称 $RULE$ 排斥 x。其中 (a_j,c_{ij}) 表明 x_i 在 a_j 属性上的值为 c_{ij}。

定义 6　若 $\exists x_i,x_j\in U,[x_l]_c=[x_j]_c\wedge[x_i]_d<>[x_j]_d$，则称 x_i,x_j 冲突，此时决策表为不相容表。

定义 7　正区域

$POS(U)=\{x_i\mid\forall x_j\in U, x_i$ 与 x_j 不冲突$\}$

定义 8　差别矩阵

$$M_{ij}=\begin{cases}0 & \text{if } d(i)=d(j)\\ \{\wedge(a_k,c_{ik})\mid c_{ik}<>c_{jk}, k=1\cdots n\} & d(i)<>d(j)\end{cases}$$

矩阵的第 i 行中所有非 0 元素记录了对象 x_i 与不同类的区别特征，同时记录了 C_{ik} 的值。显然若 $M_{ij}=\Phi$，则表明 x_i,x_j 冲突。

2　算法描述

记 $S_i=\{M_j\mid d_j=i, x_i\in POS_C(U,D)\}$，即对应于差别矩阵中无空元素的同一决策类的行集合。以下描述由 S 获得对应该类的规则的方法：

获得决策规则的步骤描述

第一部分，获得基本规则集 $RULE_i$

1) $i=1$

*)福建省自然科学基金资助项目(A0610014)和福建省高新科技重点资助项目(2005H028)。

2) set $RULE_i=\Phi, r=\Phi$,
3) for all M_{ik} in row i, $r=\wedge M_{ik}$,
if $r<>\Phi$ then
for each (a_k,c_{ik}) in r,
$RULE_i=RULE_i\cup(a_k,c_{ik})$
else
begin //若 r 为空,
for each $|M_{ik}|=1$ do
if $r=\Phi$ then set $r=M_{ik}$ else set $r=r\wedge M_{ik}$, //记为 $r=r+M_{ik}$
for each M_{jk} in row i who contains M_{ik}, set $M_{jk}=\Phi$
end for
while not all $M_{jk}=\Phi$ do
get the most frequent (a_k,c_{ik}) in row i. set $r=r+M_{ik}$
for each M_{jk} in row i who contains (a_k,c_{ik}), set $M_{jk}=\Phi$
end of while
end
4)$RULE_i=RULE_i\cup r$, $i=i+1$, while $i<=m$ do goto step 2。

该部分程序结束后,得到 $RULE_i$, $(i=1,\cdots,m)$, $RULE_i$ 为能够区分 x_i 与不同决策类对象的合取式集合。记 $RULE(S)=\{RULE_i\}$, $i=1,\cdots,m$。

定义两条合取规则 $r1,r2$ 矛盾为:

$\exists(a_k,c_{ik})\in r_1, \exists(a_k',c_{ik}')\in r_2$,

$a_k=a_k'$ 但 $c_{ik}\neq c_{ik}'$

第二部分,化简基本规则集,得到 RL

1) 取某个 S 对应的 $RULE(S)$,记 $|RULE(S)|=k$,记 $ITEM(i)=\{i\}$,表示该规则对应的决策类
2) For $i=1$ to k
For $j=i+1$ to k
set NR=Φ,flag=0
for each $r1\in RULE_i, r2\in RULE_j$
{
if $r1\subseteq r2$ then $NR=NR\cup r2$
flag=1
else if $r2\subseteq r1$
then $NR=NR\cup r1$,flag=1
else if $r1$ and $r2$ do not conflict then
if all x_j in $ITEM(j)$ fit $r1$ then $NR=NR\cup r1$,flag=1
if all x_i in $ITEM(i)$ filt $r2$ then $NR=NR\cup r2$,flag=1
}
if flag=1, then set
$RULE_i=RULE_i\cup NR, RULE_j=\varphi$
$ITEM_i=ITEM_i\cup ITEM_j$
//End For j
//End For i
3) 取 $RL[d]=\cup RULE_i$,RULE $[d]$为能够表示决策类 d 的规则集。对于所有的决策类,重复以上步骤,求出所有的 $RL[d]$。
4) 取各 $RL[d]$中的最短规则进行组合,算法结束

将差别矩阵的正区域按决策类分开,由上述算法的第二部分得到规则集,可以实现规则获取的分布处理。其正确性将在第 4 部分给出。

对于非正区域,由于 $M_{ij}=\phi$ 记录了所有的$[x]_c$ 等价关系,因此显然可以在 $O(kn)$时间内得到所有的等价类($k=|U\backslash POS(C,D)|$)。用上述方法可得到描述该等价类的规则。记 $D(i,j)$为非正区域中与 x_i 等价,且决策规则为(d,j)的对象集合,则对于 $D(i,j)$可以得到规则:

$$RULE_i\rightarrow(d,j)\quad \beta=\frac{|D(i,j)|}{|[x_i]_C|}。$$

3 理论分析

S K M Wong 和 W Ziarko 已经证明了找出一个决策表的最小约简是 NP-hard 问题,导致 NP-hard 问题的主要原因是属性的组合爆炸问题。决策表的规则获取是在求属性约简的基础上进行值约简,因此也是 NP-hard 算法。上述算法本身也是一个指数时间复杂性的算法,但得到扩展差别矩阵的过程的时间复杂性是 $O(d*n^2)$,其中 $d=|C|, n=|U|$。因此,整个算法的 cup 时间主要耗费在从扩展矩阵中获取规则。所以若能通过分布式的方式提取规则,将有效提高算法效率。由上述算法描述可知,在获取规则过程中,对于正区域的各决策类,仅由本类的扩展矩阵中的元素即可得最终的规则,无需再参照其他类中的元素。因此,扩展矩阵构造完成后,各类的规则求取过程互不冲突,在分布式环境下可以分别计算。

结论 1 若$[x_i]_C\neq[x_j]_C$,且 $d[x_i]\neq d[x_j]$,则 $RULE_i$ 排斥 x_j,$RULE_j$ 排斥 x_i。

证明:由$[x_i]_C\neq[x_j]_C$,且 $d[x_i]\neq d[x_j]$得 $M_{ij}\neq\phi$, $\forall(a_k,c_{ik})\in M_{ij}$,都有$(a_k,c_{ik})$排斥 x_j,又 $RULE_i\cap M_{ij}\neq\varphi$,因此$\exists(a_k,c_{ik})\in RULE_i$ 且(a_k,c_{ik})排斥 x_j,可得 $RULE_i$ 排斥 x_j。

同理可得 $RULE_j$ 排斥 x_i。

结论 2 若 $x_i\in POS(C,D)$,则 $\forall x_j, d[x_j]\neq d[x_i]$, $RULE_i$ 排斥 x_j。

证明:由 $x_i\in POS(C,D)$,且 $d[x_j]\neq d[x_i]$,可得$[x_i]_C\neq[x_j]_C$,由结论 1 得 $RULE_i$ 排斥 x_j。

结论 3 若 $x_i\notin POS(C,D)$, $\forall x_j\notin[x_i]_C$,则 $RULE_i$ 排斥 x_j

证明:由 $x_i\notin POS(C,D)$得$|[x_i]|\geqslant 2$,则 $\exists x_k\in[x_i]_C$, $[x_k]_C=[x_i]_C$ 且 $d[x_k]\neq d[x_i]$,因此对于 $\forall x_j\notin[x_i]$, $\exists x_m\in[x_i]$,使得$[x_m]_C\neq[x_j]_C$, $d[x_m]\neq d[x_j]$,因此 $RULE_m$ 排斥 x_j,

又$[x_m]_C=[x_i]_C$,因此 $RULE_i$ 排斥 x_j。

结论 2 和结论 3 保证了分布求规则的正确性。

4 算例

某决策表如表 1 所示。

表 1 决策表

U	C_1	C_2	C_3	d
1	0	0	1	1
1	0	0	0	1
0	0	0	0	0
1	1	0	1	0
1	1	0	2	2
2	1	0	2	2
2	2	2	2	2

由第一部分计算过程可得

1. $(C_1,1)(C_2,0)\cup(C_2,0)(C_4,1)$ →1
2. $(C_1,1)(C_2,0)\cup(C_1,1)(C_4,0)$ →1
3. $(C_1,0)$ →0
4. $(C_2,1)(C_4,1)$ →0
5. $(C_4,2)$ →2
6. $(C_1,2)\cup(C_4,2)$ →2
7. $(C_1,2)\cup(C_2,2)\cup(C_3,2)\cup(C_4,2)$ →2

由第二部分计算过程可得

1、2 中$(C_1,1)(C_2,0)$重复,所以

$RULE_1=\Phi, RULE_2=(C_1,1)(C_2,0)$

$ITEM_2=\{1,2\}$

3、4 不符合合并条件,即执行完后 flag=0,因此 RULE 和 ITEM 无变化

5、6 得到$(C_4,2)\rightarrow 2$ 而后 6、7 得到$(C_4,2)\rightarrow 2$

因此得到规则集

$(C_1,1)(C_2,0)\rightarrow 1$

$(C_1,0)\rightarrow 0$

$(C_2,1)(C_4,1)\rightarrow 0$

$(C_4,2)\rightarrow 2$

结束语 本文提出了一种求最简规则的方法。该方法的一个优点是将差别矩阵扩展后，可以实现分布式的规则获取。该方法的另一个优点是能够求得多个最简规则，当最终结果的 $RL[d]$ 中存在多个长度相同的合取式时，可求得多个可选的最简规则。

参 考 文 献

1 Skowron A, Rauszer C. The discernibility matrices and functions in information systems [A]. In: Slowinski I. Intelligent decision support- handbook of applications and advances of the Rough Sets theory [C]. Dordrecht: Kluwer Academic Publisher, 1991. 331～362
2 Pawlak Z. Rough Set approach to multi- attribute decision analysis [J]. European Journal of Operational Research, 1994, 72: 443～459
3 王国胤. Rough 集理论与知识获取[M]. 西安：西安交通大学出版社，2001
4 韩祯祥，张琦，文福拴. 粗糙集理论及其应用综述. 控制理论与应用[J]. 1999, 16(2): 153～165
5 Hassanien A E, et al. Rough Set Approach for Generation of Classification Rules of Breast Cancer Data. Institute of Mathematics and Informatics, 2004, 15(1): 23～38
6 Pawlak Z. Rough sets and intelligent data analysis [J]. Information Science, 2002(147): 1～12
7 叶东毅，陈昭炯. 不相容决策表属性约简计算的一个可辨识矩阵方法[J]. 福州大学学报(自然科学版)，2005, 33(1)
8 常黎云，王国胤，吴渝. 一种基于 Rough Set 理论的属性约简及规则提取方法. 软件学报，1999, 10(11)

(上接第 72 页)

其在时间为 0 的平面内的投影点坐标为(sig(Allen),0,0)，满足算法的要求。这个查询空间在时间为 0 的平面的投影中不包含满足(sig(Allen),0,0)的节点。这样为该查询空间记录这条直线的 pcd，为 1。对于另一个立方体，穿过的直线在时间为 0 的平面内的投影点坐标为(sig(Allen),91,0)，满足算法要求，记录其 pcd，同样为 1。两个空间的 pcd 的交集就是{1}。因此满足要求的路径内容为 pcd 为 1 的 Allen/91。由于要求的为 employee 对应的值，因此结果为 Allen。

5 实验

我们的实验环境的 CPU 为 Pentium4 3.00GHz，1GB 内存，120G IDE 硬盘，操作系统为 Microsoft XP。使用 Microsoft Visual C++ 6.0 进行实验。实验采用的数据集是包含更多员工数据的 XML 文件，大小 30MB。结点数目为 1022976。采用文[4]的索引 TempIndex 作为比较。同时比较使用 DOM 解析的时间，这是一种对无索引情况下查询的比较。针对 XML 文件建立的 UB-tree 索引大小为 1.5k，TempIndex 索引中结点数目为 2732。

对于查询：COMPANY/employee[@from≥97 and @to≤98 and /score[@from≥92 and @to≤93]]。其性能比较的柱状图如下。

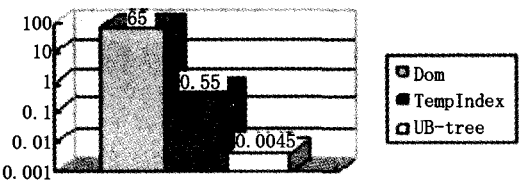

图 4 时态查询的性能比较

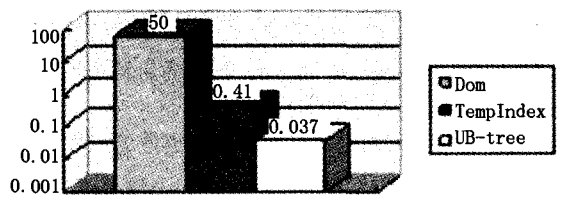

图 5 非时态查询的性能比较

从图 4 中可以看出，这三种方法的处理性能有着较大的差别。其中 DOM 思想，只能基于整个文档的遍历，这样当处理数据量大的 XML 的时候，其性能往往较差，因为存在大量无谓的数据遍历和磁盘访问的问题。TempIndex[4] 要直接操作一定量文档节点。而使用 UB-tree 处理这样的查询，只需根据查询表达式产生查询空间。由于 UB-tree 高度较小，在 UB-tree 的结构中可以很快地定位到查询空间对应的 Z-区域，从而得到对应的文档内容的表示，因此有着比较好的性能。

对于非时间谓词的查询 COMPANY/Employee[/score>91]]。图 5 给出了其性能的比较。DOM 方法依然是遍历整个文档。而 TempIndex 的查询消耗时间有所降低，因为其借助 1-index 的思想，针对简化了的 XML 文档图进行操作，然而对于谓词还是要进行局部的文档遍历。而 UB-tree 依然有着较好的性能，并且消耗时间较上一个查询有所减少。原因在于 UB-tree 没有进行大量的文档遍历，同时由于没有时间谓词，查询空间减小，只针对于时间轴为 0 的空间，查询处理的复杂度降低，所以消耗的时间更少。

总结 本文提出了在时态 XML 文档下的一种新的索引思想，使用 UB-tree 进行索引。基本思想是，将时态 XML 文档转换成多维空间的节点或者是直线。利用 UB-tree 对其进行索引，并给出了针对时态查询表达式的查询算法。由于 UB-tree 有着比较小的高度，因此能够较快地定位到要求的节点。经过理论和实验数据的分析，可以看出这种方式比之前文献介绍的 TempIndex 能够在一定程度上减少不必要的路径遍历和磁盘访问次数，因此有更好的性能。下一步的工作，在 XML 文档到 n 维空间转换过程中，用到了多条直线。改进转换算法，简化这种结构，可以有效地提高我们的查询效率。

参 考 文 献

1 Vaisman A A, Mendelzon A O, Molinari E, Tome P. Temporal XML: Model, Language and Implementation
2 Salzberg B, Tsotras V J. Comparison of Access Methods for Time-Evolving Data. ACM Computing Surveys, 1999, 31(2)
3 Dyreson C E. Observing transaction-time semantics with TTX-Path. In WISE, 2006. 193～202
4 Mendelzon A O, Rizzolo F, Vaisman A. Indexing Temporal XML Documents. In: Proceedings of the 30th VLDB Conference, Toronto, Canada, 2004
5 孔令波，唐世渭，杨冬青，王腾蛟，高军. XML 数据索引技术. 软件学报，2005, 16(12)
6 Bayer R. The Universal B-Tree for multidimensional indexing: General Concepts. In: Proc. of World-Wide Computing and its Applicazions 97 (WWCA 97). Tsukuba, Japan, 1997
7 Kaushik R, Bohannon P, Naughton J F, Korth H. Covering indexes for branching path queries. In ACM SIGMOD, Wisconsin, Madison, 2002. 133～144
8 Goldman R, Widom J J. Dataguides: Enabling query formulation and optimization in semistructured databases. In VLDB, Athens, Greece, 1997. 436～445
9 Milo T, Suciu D. Index structures for path expressions. In: Beeri C, Buneman P, eds. Proc. of the 1999 Int'l Conf. on Database Theory (ICDT). LNCS 1540, Jerusalem: Springer-Verlag, 1999. 277～295
10 Cooper B, Sample N, Franklin M J, Hjaltason G R, Shadmon M. A fast index for semistructured data. In VLDB, Rome, Italy, 2001. 341～350
11 Li Q, Moon B. Indexing and querying XML data for regular path expressions. In VLDB, Rome, Italy, 2001. 361～370
12 Markl V. Mistral: Processing Relational Queries using a Multidimensional Access Technique, http://mistral.in.tum.de/results/publications/Mar99.pdf, 1999