# 基于粗糙属性向量树的规则提取快速矩阵算法

文香军[1], 蔡云泽[1], 谭天乐[2], 许晓鸣[1,3]
(1. 上海交通大学自动化系, 上海 200240; 2. 上海航天局 812研究所, 上海 200233; 3. 上海理工大学, 上海 200093)

摘 要: 本文首先探讨了粗糙集中等价矩阵的基本概念及其运算性质. 借助于粗糙属性向量树(RAVT)的巧妙构造, 提出了两种能同时完成属性约简、数据清洗和规则提取的快速递推矩阵算法(RMC)和分布式并行矩阵算法(PMC). 上述算法强调规则提取的实用性和高效性, 通过一个简单实例研究验证了 PMC算法的可行性, 通过对算法复杂度的深入分析和一组对比实验验证了 RMC算法对知识发现、基于数据建模和控制的有效性.

关键词: 等价矩阵; RAVT; 规则提取; RMC; PMC
中图分类号: TP273/TP13 文献标识码: A 文章编号: 0372-2112 (2006) 01-0065-06

# Fast Matrix Computation Algorithms Based on RAVT for Rules Extraction

WEN Xiang-jun[1], CAI Yun-ze[1], TAN Tian-le[2], XU Xiao-ming[1,3]
(1. *Automation Department of Shanghai Jiaotong University, Shanghai 200240, China*; 2. *No 812 Institute of SAST, Shanghai 200233, China*; 3. *University of Shanghai for Science and Technology, Shanghai 200093, China*)

**Abstract** The concept of equivalence matrix and its operation are discussed in this paper. Based on a Rough Attribute Vector Tree (RAVT), two kinds of fast matrix computation algorithms—Recursive Matrix Computation (RMC) method and Parallel Matrix Computation (PMC) method are proposed for data cleaning and rules extraction finished synchronously in rough information system. The algorithms emphasize the practicability and efficiency of rules generation. A case study of PMC is analyzed in detail and its feasibility is shown, and a comparison experiment on computational complexity of RMC algorithm shows that it is valuable for knowledge discovery and knowledge-based modelling and control.

**Key words** equivalence matrix; RAVT; rules extraction; RMC; PMC

## 1 引言

由 Z Pawlak提出的粗糙集理论[1]是一种描述知识不确定性的方法, 它为在数据集中寻找隐藏的模式提供了一种有效的智能数据分析工具. 它能够有效地对数据进行压缩, 可以直接从数据中获取决策规则, 同时给出了一些评价数据质量的标准, 对数据分析的结果也能提供一些直观的解释, 从而使数据分析的结果容易理解[2].

作为一种从实例中发现知识的归纳学习算法, 对于决策规则的发现一直是粗糙集理论研究与实际应用中一个非常重要的研究内容, 并受到众多研究人员的关注. 到目前为止, 这类算法大致可以分为两类: 一类算法是基于对象实际含义的方法, 这类算法往往要考虑对象相应的背景知识. 如: DBlearn算法; 广义分布表(Generalization Distribution Table, GDT)法[3]; 区别矩阵法[12]; 基于属性重要性的方法; 基于信息熵的方法[4]; 遗传搜索算法; LBR 和 LEN3[5]; 二元区分表法[6,7]; 二元整数编程(Binary Integer Programming, BIP)法[8]等等. 常见的一些粗集应用系统, 如 ROSE和 Rosetta[15]中也有一些用于规则提取的工具软件. 这类算法由于在计算的过程中不能将约简过程和信息决策表的具体内容分离开来, 其通用性不强, 提取规则的效率也有限. 另一类是与数据的实际含义无关(或与计算过程相分离)的算法, 包括 C4.5(ID3), Prism, Version-space[3]和矩阵算法[9~11]等, 这类方法不用考虑样本的背景知识, 从而具有较强的通用性. 由于规则提取在粗糙集应用中的核心地位, 从而设计一种高速并行的规则提取算法具有非常重要的理论意义和实用价值.

本文首先介绍了粗糙信息系统和等价矩阵相关的一些重要概念, 然后设计了一种基于粗糙属性向量树(Rough Attribute Vector Tree, RAVT)的可用于规则提取的快速递推矩阵算法(RMC). 在递推矩阵算法的基础上, 进一步提出了一种可望用于处理海量数据规则提取的分布式并行

收稿日期: 2004-04-26; 修回日期: 2005-09-12

矩阵算法(PMC)并进行了实例研究. 通过对 RMC算法复杂性的深入分析和一组对比实验表明了该规则提取算法的实用性和高效性. 最后给出了本文的研究结论.

## 2 粗糙信息系统和等价矩阵

按照粗糙集的理论[1,2], 本文将所有形如四元组 $S=(U, R, V, f)$的知识系统称为粗糙信息系统. 其中的属性集合 $R=C\cup D$. $C$ 为条件属性, 用于描述对象. $D$ 为决策属性, 用于描述分类. 论域 $U=\{x_1, x_2, \cdots, x_n\}$是一个非空有限的对象集合, 集合中的对象 $x_j$ 使用属性 $r_i\in R$ 及属性值 $v_{ij}\in V_{ri}$来描述. $V_{ri}$是属性 $r_i$ 的值域, $V=\bigcup V_{ri}$, $r_i\in R$. $f: U\times R\rightarrow V$ 是一个决策函数或决策算法.

有关规则提取的等价矩阵的定义和运算性质描述如下:

定义 1 令 $S=(U, R, V, f)$为一个粗糙信息系统, $x_i$, $x_j\in U$, 定义 $C\subseteq R$ 的 $n\times n$ 二元等价矩阵 $M_C=[a_{ij}]$. $M_C$ 中的元素 $a_{ij}$满足 $a_{ij}=\begin{cases}1, & x_iE_Cx_j\\ 0, & \text{else}\end{cases}$; $i, j=1, 2, \cdots, n$, $E_C$ 为 $U$ 上的一个等价关系.

定义 2 令 $M_1=[a_{ij}]$, $M_2=[b_{ij}]$为两个二元 $n\times n$ 等价矩阵. 矩阵 $M_1$ 与 $M_2$ 的交集 $M_1\cap M_2$ 定义如下: $M_1\cap M_2=[r_{ij}]$; 其中 $r_{ij}$为 $a_{ij}$与 $b_{ij}$的二元与, 即 $r_{ij}=a_{ij}\cap b_{ij}=\min(a_{ij}, b_{ij})$.

定义 3 令 $M_1=[a_{ij}]$, $M_2=[b_{ij}]$为两个二元 $n\times n$ 等价矩阵. 矩阵 $M_1$ 与 $M_2$ 的异或 $M_1\oplus M_2$ 定义如下: $M_1\oplus M_2=[r_{ij}]$, 其中 $r_{ij}$为 $a_{ij}$与 $b_{ij}$的二元异或, 即 $r_{ij}=a_{ij}\oplus b_{ij}=xor(a_{ij}, b_{ij})$

以上定义的等价矩阵运算具有以下性质: (1)交换率: $M_1\cap M_2=M_2\cap M_1$; (2)结合率: $(M_1\cap M_2)\cap M_3=M_1\cap(M_2\cap M_3)$; (3)幂等性: $M\cap M=M$; (4)(异或)幂零性: $M\oplus M=[0]$; (5)对称性: $M^{\mathrm{T}}=M$

引理 1[10] 令 $S=(U, R, V, f)$为一个知识系统, $C_i$, $C_j\subseteq R$, 则 $n\times n$ 二元等价矩阵 $M_{Ci}$, $M_{Cj}$ 满足: $M_{Ci}\cap M_{Cj}=M_{C\cap C_j}$.

上述理论和有关等价矩阵的运算性质形成了本文快速矩阵算法的理论基础.

## 3 粗糙属性向量树

对连续属性值进行离散化后, 我们就得到了一张包含所有规则的决策表. 决策表中的每一个对象就代表了一条基本的决策规则, 决策表就包含了论域空间上所有决策规则的集合. 但是这样的决策规则没有足够适应度从而用处不大. 为了能够从决策表中获取适应度大的规则, 就需要对决策表进行约简. 对决策表的约简包括对条件属性的约简和对规则的属性值约简两个部分. 目前, 关于属性约简求取决策规则的算法已经有很多, 其中很多算法求取最小属性约简的计算过程始终与对象的真实取值相联系, 通用性差. 文献[9, 10]采用等价矩阵方法, 通过一些彼此独立的算法, 能够迅速地找到一个原始决策表的核属性和决策表中的属性约简, 并具有很好的通用性. 在本文中, 主要考虑用基于等价矩阵的方法实现规则提取、属性约简以及数据清洗的并行处理.

粗糙集决策表中, 一个属性就对应着一个等价关系, 每个对象就对应着一个等价矩阵. 条件属性和决策属性分别对论域形成了各自的划分, 这两个划分构成了条件属性和决策属性对论域中对象的分类知识. 为了从决策表中提取所有可能的规则, 我们首先必须找到属性的所有组合形式. 不幸的是, 由于属性的组合爆炸问题, 要想直接通过等价矩阵的组合来找到一个决策表的最小约简是一个非常困难的问题. 本文中, 我们提出了一种粗糙属性向量树可以用来解决这个问题.

定义 4 令 $V^i=[v_i]_{1\times \mathrm{card}(C)}$为一个与某些条件属性组合相关的二元向量. 向量的维数为 $1\times \mathrm{card}(C)$, 其中 $\mathrm{card}(C)$是条件属性的总个数. 上标 $i=1, 2, \cdots, \mathrm{card}(C)$表示向量中元素"1"的个数(其余元素均为"0").

定义 5 令 $V_1^i=[v_{1i}]$, $V_2^i=[v_{2i}]$为两个 $1\times \mathrm{card}(C)$二元属性向量. $V_1^i$ 与 $V_2^i$ 的交定义如下: $V_1^i\cap V_2^i=[r_i]$, 其中 $r_i$ 为 $v_{1i}$与 $v_{2i}$的二元与, 即 $r_i=v_{1i}\cap v_{2i}=\min(v_{1i}, v_{2i})$.

定义 6 令 $V_1^i=[v_{1i}]$, $V_2^i=[v_{2i}]$为两个 $1\times \mathrm{card}(C)$二元属性向量. $V_1^i$ 与 $V_2^i$ 的异或定义如下: $V_1^i\oplus V_2^i=[r_i]$, 其中 $r_i$ 为 $v_{1i}$与 $v_{2i}$的二元异或, 即 $r_i=v_{1i}\oplus v_{2i}=xor(v_{1i}, v_{2i})$.

显然, 上述定义的属性向量满足第 2节中的矩阵运算性质(1)~(4).

定义 7 粗糙属性向量树(RAVT)是用来描述决策表中条件属性组合集合的向量树. 该树具有如下特性:

(1)树的总层数为 $\mathrm{card}(C)$;

(2)树中各层的节点由属性向量组成, 对树的第 $i$层而言, 共有 $C^i_{\mathrm{card}(c)}$个属性向量节点;

(3)对第 $i$层的节点 $j$, 其属性向量为 $V_{ij}^i$ ($j=1, 2, \cdots, C^i_{\mathrm{card}(c)}$). 其中的上标 $i$表示该向量中元素"1"的个数(其余元素为"0"). 将所有的这类向量放在同一层. (有关这类二元向量的产生算法在很多文献中都有论述, 在此恕不赘述, 本文采用的是 Steinhaus-Johnson-Trotter(SJT)算法[14]);

(4)同一层任何两个节点具有如下性质: $V_{ij}^i\oplus V_{ij}^i\neq[0]_{1\times \mathrm{card}(C)}$, $j\neq k$, $j, k=1, 2, \cdots, C^i_{\mathrm{card}(c)}$;

(5)对于相邻两层的节点, 如果满足如下关系: $V_{ij}^i\cap V_{(i-1)k}^{i-1}=V_{(i-1)k}^{i-1}$, $j=1, 2, \cdots, C^i_{\mathrm{card}(c)}$, $k=1, 2, \cdots, C^{i-1}_{\mathrm{card}(c)}$则这两个节点之间有一根树枝相连. 我们称这样的节点为邻层相关节点. 记为: $V_{(i-1)k}^{i-1}\rightarrow V_{ij}^i$.

按照粗糙集的相关理论, 上述性质(5)有助于尽快从树的上一层节点的属性组合中找到适应性较弱的规则(决

策规则的特点就是规则的前部所含的条件越少(精简),提取的规则就越有代表性,其适应性就越广). 在矩阵算法中,所有上一层相关节点的等价矩阵的交 $M^i_{C_i}=\bigcap M^{i-1}_{C-\{c\}}$, $c\in C_i$ 就形成了下一层能提取有效规则的等价矩阵的组合. 寻找所有可能的规则形式,也即是一个寻找所有可能的条件属性组合的过程. 不难算出, card(C)个条件属性的所有可能组合数目为 $\sum_{i=1}^{card(C)} C^i_{card(c)}=2^{card(C)}-1$. 由于 RAVT采用的 SJT算法能够通过 Gray码序列相邻移项运算快速生成具有上述性质 3的二元向量(即属性组合)并能保持常数摊销时间①(Constant Amortized Time, CAT)的特性,同时通过属性向量的二元逻辑运算建立规则提取的实际生成序列. 综上所述,粗糙属性向量树实际上是描述了能够用于提取规则的实际可能(而不是所有可能)的条件属性的组合关系,从而可望有效避免由于属性组合爆炸而带来的各种问题.

给定一个决策表,具有 4个条件属性 $C=C_i$, $i=1\sim4$ 一个决策属性 $D=d$, $U=\{x_1, x_2, \cdots, x_n\}$,则其属性向量树的构成如图 1所示.

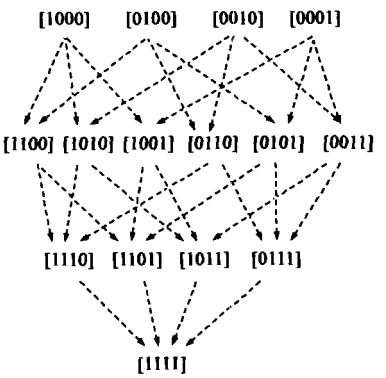


图 1　一个粗糙属性向量树实例(card(C)=4)

通过上述形象化的属性向量树,我们就可以很容易理解快速矩阵算法同时完成规则提取、属性约简和数据清洗的整个过程. 当然,如果 card(C)是一个很大的数,想绘制出一颗像图 1这样既形象又直观的树是很难的. 所幸的是,粗糙属性向量树不能绘制出来并不影响我们的快速矩阵算法的设计.

## 4　规则提取的快速矩阵算法的设计

### 4.1　递推矩阵算法的设计

首先,我们按照定义 1生成每个属性的等价矩阵. 以 3节的决策表为例,我们将得到 4个 $n\times n$的条件属性等价矩阵 $M_{C_i}$ ($i=1\sim4$)和一个决策属性矩阵 $M_D$.

其次,将 4个条件属性的等价矩阵分别与属性向量树第一层的某个属性向量绑定. 例如将[1000]/[0100]/[0010]和[0001]分别与 $M_{C_i}$ ($i=1\sim4$)绑定,则根据属性向量树,我们可以立即推出其他各层的等价矩阵形式. 对第二层,[1100]就对应着 $M_{21}=M_{C_1}\cap M_{C_2}$,[1010]对应着 $M_{22}=M_{C_1}\cap M_{C_3}$等;对第三层的[1110]就对应着 $M_{31}=M_{21}\cap M_{22}\cap M_{24}=M_{C_1}\cap M_{C_2}\cap M_{C_3}$(等价矩阵的结合率)等. 通过对树中各层的等价矩阵分别与决策等价矩阵(如有多个决策属性则为各个决策属性联合等价矩阵)进行相交和异或运算,就可逐层将规则提取出来.

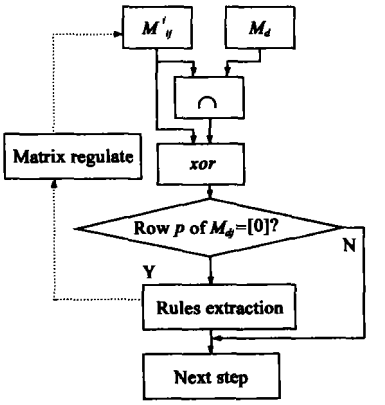


图 2　基于矩阵逻辑运算的规则提取过程

以树中的第 $i$ 层为例,规则提取过程的逻辑运算关系如图 2所示.

下面给出了递推矩阵算法第 $i$层规则提取的程序实现.

递推矩阵算法程序(第 $i$层):

1. For $j=1$ to $C^i_{card(C)}$
2. 如果存在 $V^{i-1}_{(i-1)k}\rightarrow V^i_{ij}$? ($k=1, 2, \cdots, C^{i-1}_{card(c)}$),则 $M^i_{ij}=M^i_{ij}\cap M^{i-1}_{(i-1)k}$
3. 否则,下一步
4. 计算决策等价矩阵和第 $j$个条件矩阵的交和异或: $M_{Rj}=M^i_{ij}\cap M_d$ and $M_{dj}=M^i_{ij}\oplus M_{Rj}$
5. For $p=1$ to $n$
6. 如果 $M_{dj}$第 $p$行 $\neq[0]_{1\times n}$, next $p$,否则下一步
7. For $q=p$ to $n$
8. 是否存在元素 $a_{pq}\in M^i_{ij}$为"1"? 如不是, next $p$
9. 否则置 $M^i_{ij}$的 $q$行元素为零, next $q$
10. 基于对象 $p$ 产生规则: $\wedge([C_i]\cap V^i_{ij}, v_c)\Rightarrow\wedge(d, d_p)$, next $p$
11. Next $j$

其中,第 10步 $[C_i]$是所有条件属性组成的行向量 $[C_1, C_2, \cdots, C_{card(c)}]$, $C_i\in C$,该步算法将提取决策规则的条件属性组合和决策属性以及相应的实际对象联系起来.

在规则提取的过程中,第一层条件属性的等价矩阵按定义 1从决策表中直接生成,所以本算法的第 2, 3步将不起作用. 第 $i$层的各个节点对应的等价矩阵是根据属性向量树的性质 5由邻层相关节点的等价矩阵同时递推生成的. 经过第 $i$层的规则提取之后,等价矩阵 $M^i_{ij}$变成为新的等价矩阵 $M^i_{ij}{}'$并进入下一层的运算中. 这样,经过总共 Card(C)层的循环运算后,我们就能将隐藏在决策表中的所有规则提取出来.

① 即总的计算时间(total amount of computation)/对象数目(objects)=常数(constant)

在粗糙集里, 选择不同的条件属性集 $C_i$ 的组合和决策属性集 $D$, 就可以找到相应的规则, 形如 $\bigwedge_{i=1}^{card(C)} C_i = v_{ci} \Rightarrow \bigwedge_{i=1}^{card(D)} D_i = v_{di}$ $C_i \subset C$. 这些规则需要满足协调性要求. 在等价矩阵中, 决策规则的协调性就体现在决策表中条件属性表示的等价关系 $E_C$ 和决策属性表示的等价关系 $E_D$ 存在 $E_C \subseteq E_D$. 在上述递推矩阵算法中, 条件属性等价矩阵的第 $i$行表示了对象 $x_i$ 在条件属性上与其他对象的等价关系, 决策属性等价矩阵的第 $i$行表示了对象 $x_i$ 在决策属性上与其他对象的等价关系. 条件属性等价矩阵第 $i$行与决策属性等价矩阵的第 $i$行相交的结果与条件属性等价矩阵的第 $i$行一致(异或运算为零行向量), 则表示条件属性等价矩阵第 $i$行中"1"所对应的那些对象当中, 存在 $E_C \subseteq E_D$, 即满足决策规则协调性的要求.

当决策表中存在冗余对象或不相容对象时, 对于冗余对象, 本算法经过第 7, 8, 9步可以避免在下一层产生冗余规则. 对于不相容的对象时, 本算法通过条件属性组合的等价矩阵与决策属性的等价矩阵相与, 使不相容数据不会产生任何规则. 上述过程实质上是一个对 RAVT同一层节点同时进行属性约简和隐含的数据清洗处理的并行规则提取过程. 由于粗糙属性向量树与实际的决策表内容相对无关(只与 card$(C)$有关), 可以独立构造, 同时在程序的设计中, 以二元等价矩阵为单元进行的"块"运算, 这将极大地提高算法的运算效率.

### 4.2 并行矩阵算法设计及实例

从 4.1节递推矩阵算法的设计中, 我们发现: 在属性向量树生成以后, 各层中每个节点的等价矩阵就可以由第一层的单个属性的等价矩阵和对应节点的属性向量唯一确定. 由各个节点的等价矩阵进行规则提取生成的规则矩阵也可以由节点的属性向量唯一确定. 因此, 如果我们对上述递推矩阵算法程序稍加改变, 就可以一次性并行地提取树中各层所有节点的规则, 然后再对提取出来的规则矩阵按属性向量树反向进行冗余处理, 则可以将我们的快速递推算法推广到分布式的并行计算环境中.

并行矩阵算法的程序设计如下:

1. 按照属性向量树生成所有节点的等价矩阵.

2. 按递推矩阵算法 (去掉第 2步寻找上层关联节点的过程) 同时并行提取各层的规则. 对第 $i$层的节点 $j$而言, 我们可以获得相应的规则矩阵 $R_{ij}^{i}$, 其中 $i = 1, 2, \cdots,$ card$(C)$; $j = 1, 2, \cdots, C_{\mathrm{card}(C)}^{i}$; 上标 $i$表示该规则矩阵中组成规则的条件属性个数.

3. For $i$= card$(C)$ to 1(减 1)

4. 如果存在节点 $R_{(i-1)k}^{i-1} \rightarrow R_{ij}^{i}$ (规则矩阵所在节点为关联节点)? 如没有, next $i$

5. 否则清除 $R_{ij}^{i}$中冗余的规则. (按照粗糙集的理论, 上一层规则因其具有较少的属性而比下一层的相应的规则具有更强的适应性, 应该予以保留).

6. 是否 $i$= 1? 不是, $i$= $i$- 1转第 3步

7. 否则, 退出

基于属性向量树设计的并行矩阵算法具有分布式的特点, 各个节点的规则提取过程可以同时独立完成, 这样将有望处理海量数据的规则提取. 由于缺乏分布式并行计算环境进行仿真, 同时为便于理解规则并行提取的过程, 在此我们给出了一个分布式并行矩阵算法实现的简单实例:

表 1 决策表

| Object | $a$ | $b$ | $c$ | $D$ |
|---|---|---|---|---|
| $X_1$ | 2 | 2 | 2 | 2 |
| $X_2$ | 2 | 2 | 3 | 3 |
| $X_3$ | 2 | 3 | 2 | 4 |
| $X_4$ | 2 | 3 | 3 | 3 |
| $X_5$ | 3 | 2 | 2 | 4 |
| $X_6$ | 3 | 2 | 3 | 2 |

表 1是一个来自粗糙信息系统的决策表.

首先, 我们按定义 7生成 card$(C)$= 3的属性向量树, 同时并行生成各层的条件属性等价矩阵和决策属性等价矩阵.

第一层:

$$M_a = \begin{bmatrix} 1&1&1&1&0&1 \\ 1&1&1&1&0&1 \\ 1&1&1&1&0&1 \\ 1&1&1&1&0&1 \\ 0&0&0&0&1&0 \\ 1&1&1&1&0&1 \end{bmatrix} \quad M_b = \begin{bmatrix} 1&1&0&0&1&1 \\ 1&1&0&0&1&1 \\ 0&0&1&1&0&0 \\ 0&0&1&1&0&0 \\ 1&1&0&0&1&1 \\ 1&1&0&0&1&1 \end{bmatrix}$$

$$M_c = \begin{bmatrix} 1&0&1&0&1&1 \\ 0&1&0&1&0&0 \\ 1&0&1&0&1&1 \\ 0&1&0&1&0&0 \\ 1&0&1&0&1&1 \\ 1&0&1&0&1&1 \end{bmatrix} \quad M_D = \begin{bmatrix} 1&0&0&0&0&1 \\ 0&1&0&1&0&0 \\ 0&0&1&0&1&0 \\ 0&1&0&1&0&0 \\ 0&0&1&0&1&0 \\ 1&0&0&0&0&1 \end{bmatrix}$$

第二层:

$$M_{ab} = \begin{bmatrix} 1&1&0&0&0&0 \\ 1&1&0&0&0&0 \\ 0&0&1&1&0&0 \\ 0&0&1&1&0&0 \\ 0&0&0&0&1&1 \\ 0&0&0&0&1&1 \end{bmatrix} \quad M_{ac} = \begin{bmatrix} 1&0&1&0&0&0 \\ 0&1&0&1&0&0 \\ 1&0&1&0&0&0 \\ 0&1&0&1&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&1 \end{bmatrix}$$

$$M_{bc} = \begin{bmatrix} 1&0&0&0&1&0 \\ 0&1&0&0&0&1 \\ 0&0&1&0&0&0 \\ 0&0&0&1&0&0 \\ 1&0&0&0&1&0 \\ 0&1&0&0&0&1 \end{bmatrix}$$

第三层: $M_{abc} = \begin{bmatrix} 1&0&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&0&1&0&0&0 \\ 0&0&0&1&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&1 \end{bmatrix}$

其次, 同时从树中各层提取规则.

第一层:

$$M_{aD}\oplus M_a=\begin{bmatrix}0&1&1&1&0&0\\1&0&1&0&0&0\\1&1&0&1&0&0\\1&0&1&0&0&0\\0&0&0&0&0&1\\0&0&0&0&1&0\end{bmatrix},$$

$$M_{bD}\oplus M_b=\begin{bmatrix}0&1&0&0&1&0\\1&0&0&0&1&1\\0&0&0&1&0&0\\0&0&1&0&0&0\\1&1&0&0&0&1\\0&1&0&0&1&0\end{bmatrix},$$

$$M_{cD}\oplus M_D=\begin{bmatrix}0&0&1&0&1&0\\0&0&0&0&0&1\\1&0&0&0&0&0\\0&0&0&0&0&1\\1&0&0&0&0&0\\0&1&0&1&0&0\end{bmatrix}.$$

第一层无规则生成: $R^1_{11}=[\,]$, $R^1_{12}=[\,]$, $R^1_{13}=[\,]$.

第二层:

$$M_{abD}\oplus M_{ab}=\begin{bmatrix}0&1&0&0&0&0\\1&0&0&0&0&0\\0&0&0&1&0&0\\0&0&1&0&0&0\\0&0&0&0&0&1\\0&0&0&0&1&0\end{bmatrix},$$

$$M_{acD}\oplus M_{ac}=\begin{bmatrix}0&0&1&0&0&0\\0&0&0&0&0&0\\1&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\end{bmatrix}$$

$R^2_{21}=[\,]$, $R^2_{22}$提取了三条规则:

$x_2$: $\wedge(C, a)\wedge(C, c)\Rightarrow\wedge(D, d_2)$,

$x_5$: $\wedge(C, a)\wedge(C, a)\Rightarrow\wedge(D, d_5)$

$x_6$: $\wedge(C, a)\wedge(C, a)\Rightarrow\wedge(D, d_6)$.

还有一条冗余规则

$x_4$: $\wedge(C, a)\wedge(C, c)\Rightarrow\wedge(D, d_4)$

由于 4.1算法的 7、8、9步在提取 $X_2$规则时等价矩阵 $M_{ac}$的第 4行元素被置零从而被自动清除.

$R^2_{23}$提取了两条规则:

$x_3$: $\wedge(C, b)\wedge(C, c)\Rightarrow\wedge(D, d_3)$

$x_4$: $\wedge(C, b)\wedge(C, c)\Rightarrow\wedge(D, d_4)$

第三层:

$$M_{abcD}\oplus M_{abc}=\begin{bmatrix}0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\end{bmatrix}$$

粗看似乎我们可以提取六条规则. 实际上, 在 PMC算法的 4、5步, 我们是将最后两层的节点同时考虑:

$x_1$: $\wedge(C, a)\wedge(C, b)\wedge(C, c)\Rightarrow\wedge(D, d_1)$, 不存在 $R^2_{2k}\rightarrow R^3_{3l}$ $(k=1, 2, 3)$. 因此该规则被提取出来.

$x_2$: $\wedge(C, a)\wedge(C, b)\wedge(C, c)\Rightarrow\wedge(D, d_2)$, 因与邻层相关节点 $R^2_{22}$规则 $x_2$ 构成冗余而被清除. $x_3$, $x_4$, $x_5$, $x_6$ 规则因同样原因被清除.

反向检查树中第二层是否存在冗余规则.

最后, 我们获得的决策规则如表 2所示.

表 2　提取规则结果

| Object | Condition | Decision |
|---|---|---|
| $X_1$ | $(a, 2)\wedge(b, 2)\wedge(c, 2)$ | $(D, 2)$ |
| $X_2$ | $(a, 2)\wedge(c, 3)$ | $(D, 3)$ |
| $X_3$ | $(b, 3)\wedge(c, 2)$ | $(D, 4)$ |
| $X_4$ | $(b, 3)\wedge(c, 3)$ | $(D, 3)$ |
| $X_5$ | $(a, 3)\wedge(c, 2)$ | $(D, 4)$ |
| $X_6$ | $(a, 3)\wedge(c, 3)$ | $(D, 2)$ |

## 5　RMC算法的复杂性和对比实验

### 5.1　算法的复杂性

对于具有 $n$个对象的决策表, 通过本算法, 生成一颗属性向量树的条件属性等价矩阵的数量为 $\sum_{i=1}^{\mathrm{card}(C)} C^i_{\mathrm{card}(C)}=2^{\mathrm{Card}(C)}-1$, 从 $C^{i-1}_{\mathrm{Card}(C)}$个属性向量中寻找 $i$个属性向量的计算量为 $C^{i-1}_{\mathrm{Card}(C)}\times n$, 寻找所有需要的属性向量的计算量为 $n\times(2^{\mathrm{Card}(C)}-2)$. 为了提取规则, 每个条件属性等价矩阵要和决策属性等价矩阵进行一次逻辑与和异或运算, 计算复杂度均为 $O(n^2)$. 本算法最多可能得到的规则形式数量为 $2^{\mathrm{Card}(C)}-1$. 实际上, RMC算法结合二元逻辑运算, 充分利用等价矩阵的 (异或) 幂零性和对称性的性质, 因此算法能够减少近一半的矩阵计算量. 综上所述, 本算法的实际计算复杂度约为 $O(n^2\times 2^{\mathrm{Card}(C)-1})$.

### 5.2　对比实验

在同类的矩阵算法中, 文献 [9]中提出的矩阵算法 (以下简称 MC算法) 是一种基于等价矩阵进行规则提取和数据清洗的面向工程的实用性算法, 已经较成功地应用于故障诊断、过程建模和粗糙控制中[13], 和知名的基于粗糙集的数据分析软件 Rosetta 1.4.41相比, 具有较高计算效率和通用性. 下图给出了 RMC算法和 MC算法在同一测试数据集下对不同对象数目 (50~1200) 和不同的条件属性数 (3~10) 时运算效率的对比实验.

实验结果表明,随着对象数目的增多, RMC算法和 MC算法均表现出多项式的增长规律;随着属性数目的增大, RMC算法和 MC算法均表现出指数增长的规律.这与上述计算复杂度的理论分析结论是一致的.借助于 RAVT的巧妙构造和等价矩阵运算性质的运用,相比而言,当对象或属性较多时, RMC算法比 MC算法具有更高的规则提取运算效率.

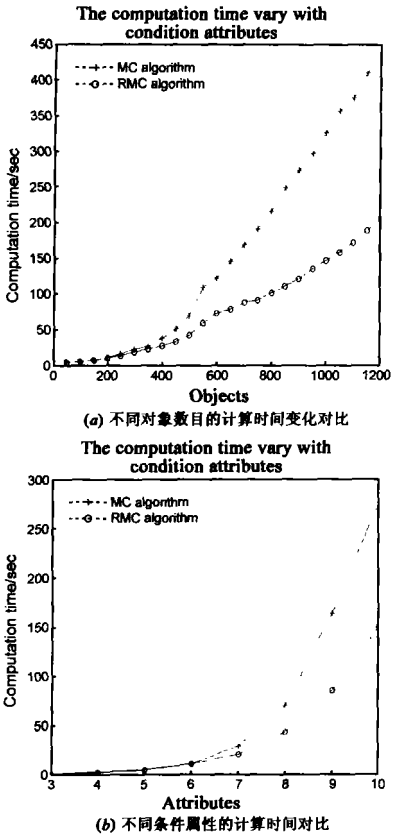


图 3 对比实验结果 (1.7GHz, Matlab 6.5)

## 6 结论

综上所述,基于粗糙属性向量树并运用等价矩阵的知识,我们可以设计出从粗糙信息系统中获取决策规则的高效快速矩阵算法.由于上述矩阵算法在规则提取的过程与对象的真实含义相分离,能同时完成将规则提取、属性约简和数据清洗的过程,从而具有较强的通用性和鲁棒性.通过一个简单实例的研究表明了分布式 PMC算法是可行的;通过对算法复杂性的深入分析和一组对比实验表明,与同类的矩阵算法相比,本文提出的 RMC算法具有更高的计算效率.上述算法为基于粗糙集的大规模数据分析、在线学习动态规则提取提供了一个基础和新的思路,将促进等价矩阵方法在大规模智能数据处理和非机理知识建模以及基于规则的粗糙控制中的应用.

参考文献:

[1] Z Pawlak. Rough sets [J]. International Journal of Computer and Information Science, 1982, 11: 341-356

[2] Z Pawlak. Rough sets and intelligent data analysis[J]. Information Sciences, 2002, 147(1-4): 1-12

[3] Zhong Ning, Dong Juzhen, Ohsuga Setsuo. Rule discovery by soft induction techniques[J]. Neurocomputing, 2001, 36(1-4): 171-204.

[4] Miao Duoqian, Wang Jue. Information-based algorithm for reduction of knowledge[A]. 1997 IEEE Inter Conf on Intelligent Processing Systems[C]. Bejing, China, 1997. 1155-1158

[5] Lan Shu, Mo Zhiwen, Hu Dan. Methods of learning rules based on rough set LBR and LEM3[A]. IFSA World Congress and 20th NAFIPS International Conference[C]. Vancouver, Canada, 2001. 753-756.

[6] R Felix, T Ushio. Rule induction from inconsistent and incomplete data using rough sets[A]. 1999 IEEE Inter Conf on Systems, Man, and Cybernetics[C]. Tokyo, Japan, 1999. 154-158

[7] R Felix, T Ushio. Rough sets-based machine learning using a binary discernibility matrix[A]. Proceedings of the Second International Conference on Intelligent Processing and Manufacturing of Materials IPMM'99[C]. Hawaii, USA: 1999. 299-305

[8] A A Bakar, M N Sulaiman, et al. Finding minimal reduct with binary integer programming in data mining[A]. TENCON 2000[C]. Kuala Lumpur, Malaysia, 2000. 2: 141-146

[9] Tan Tianle, Song Zhihuan, Li Ping. Matrix computation for data cleaning and rule extraction in knowledge System [A]. Proceedings of 2002 Inter Conf on Machine Learning and Cybernetics[C]. Beijing, China, 2002. 116-120

[10] JW Guan, D A Bell, Z Guan. Matrix computation for information systems[J]. Information Sciences, 2001, 131(1): 129-156

[11] JW Guan, D A Bell. Rough computational methods for information systems[J]. Artificial Intelligence, 1998, 105: 77-103.

[12] Wang Jue, Wang Ju. Reduction algorithms based on discernibility matrices: the ordered attributes method[J]. Journal of Compute Science and Technology, 2001, (16): 489-504

作者简介:

文香军 男, 1974年生于广西,上海交通大学在读博士生,主要研究领域为信息融合、机器学习和智能控制.

E-mail: wenxiangjun@sjtu.edu.cn.

许晓鸣 男, 1957年生于福建,上海交通大学教授,博士生导师,主要研究领域为复杂工业对象和过程的智能控制.

(下转至 64页)

[12] 周宏潮, 朱炬波, 王正明. SAR图像增强的前向-后向扩散方程方法[J]. 电子学报, 2004, 32(12): 2070-2073.
ZHOU Hong-chao, ZHU Ju-bo, WANG Zheng-ming. Forward and backward diffusion processes for SAR image enhancement[J]. Acta Electronica Sinica, 2004, 32(12): 2070-2073. (in Chinese)

[13] 高鑫, 刘来福, 黄海洋. 基于PDE和几何曲率流驱动扩散的图像分析与处理[J]. 数学进展, 2003, 32(3): 285-294.
GAO Xin, LIU Lai-fu, HUANG Hai-yang. A survey-image analysis and processing using PDE and geometry curvature-driven diffusion[J]. Advances in Mathematics, 2003, 32(3): 285-294. (in Chinese)

作者简介:

谢美华 女, 1976年 8月出生于湖南省宁乡县. 现为国防科学技术大学理学院数学系讲师、博士, 在国内外发表学术论文 20余篇, 主要研究方向为图像处理与实验数据处理.
E-mail: xmhdjh@163.com.

王正明 男, 1962年 2月出生于湖南省长沙市, 现为国防科学技术大学理学院院长、博士生导师, 在国内外发表学术论文 80余篇, 主要研究方向为图像处理中的数学方法、装备试验分析与评估.

(上接第 70页)

[13] Tan Tianle, et al. Rough set-based modeling and controller design in an internal model control system[A]. 2003 IEEE Inter Conf on Systems, Man and Cybernetics[C]. Hangzhou, China, 2003. 3559-3563.

[14] Frank Ruskey. Combinatorial Generation (working version)[M]. Victoria, Canada: University of Victoria, 2001.

[15] A Øhrn. Discernibility and rough sets in medicine: tools and applications[D]. PhD thesis. Norwegian University of Science and Technology, 1999.