

基于逻辑距离判定的最简规则提取算法

何 群

(南京工业大学 信息科学与工程学院, 江苏 南京 210009)

摘 要: 针对目前以属性约简为设计目标的算法增大了最终规则的冗余度, 提出以属性值(逻辑值)为基本单位, 从整个决策表的不可分辨关系出发, 先由逻辑距离确定每个属性值的分辨强度, 再以其强度为依据采用择强、覆盖算法对表进行多值逻辑化简, 通过实例证明所得规则最简。

关键词: 不可分辨关系; 多值逻辑; 逻辑最小项; 决策规则

中图分类号: TP181

文献标识码: A

文章编号: 1000-7180(2010)04-0038-05

An Algorithm of Minimizing Rules Extraction Based on Distinguishing by Logic Distance

HE Qun

(College of Information Science and Engineering, Nanjing University of Technology, Nanjing 210001, China)

Abstract: In allusion to increase redundance of the rules for some algorithm of attribute reduction with belong to attribute is basic unit. Put forward taking belonging to attribute value (logical value) as the basic unit, in the indefinite relation of the whole decision, the different degree of each logic value for the logic distance among the smallest items of different class is decided, taking its strength as the basis minimize the table of multiplied logic by choosing strong and overlay, and by the examples proved that the rules extracted is minimized.

Key words: indefinite relation; multiplied logic; logical minterm; decision rules

1 引言

从决策表中提取规则就是机器学习、模式识别和人工智能的关键。规则提取和应用是决策系统的两个阶段, 提取阶段可在相对充裕的时间和空间环境下完成, 是一次性的准备工作; 应用阶段则在严格时间和空间环境下完成, 是一个反复、实时、准确的判定过程, 所以在提取阶段使规则达到最简是决策系统的核心任务。粗糙集理论被认为是约简规则的数学工具, 当前基于该理论所提出的算法^[1-2]其基本步骤是先求核属性, 由核属性确定最佳独立集, 再由最佳独立集确定核属性值(规则), 然而这些算法仍不能完全解决规则提取的最简化问题, 主要表现在以下几个方面:

(1) 将寻找核属性和最佳独立集作设计目标, 但如设 $\{A, C\}$, $\{A\} = A_1, A_2, \dots, A_n$. $\{A, C\}$ 中可分为 2 个或 2 个以上集 $\{A_r\}$, $\{A_s\}$, $\dots$ 且每个集都可完整决定 C 时:

$$f(A_r) = C \quad (A_r \{A\}, r < n)$$

$$\text{且 } f(A_s) = C \quad (A_s \{A - A_r\}, r + s < n) \dots\dots$$

因为对 C 有多个决定的属性集, 所以产生 $\text{CORE}(A, C) = \bigcap \text{RED}(A, C) = \{\Phi\}$ 的情况, 按粗糙集所定义该表无核属性, 即用求核属性算法不能准确地确定关键属性。

(2) 由于在建立最佳独立集过程中约去了按粗糙集所定义的“不必要属性”, 而求得的规则对整个决策表并非最简。即有些“不必要属性”中的部分属性值对规则的最简化是非常重要的。

收稿日期: 2009-03-16; 修回日期: 2009-05-16

基金项目: 江苏省建筑业科研计划项目(JGJH2008-17)

(3) 用可信度因子这种模糊的概念确定规则支持程度,这样的规则显然是不实用的.

问题(1)、(2)的主要原因是所选基本单位是属性,一个重要属性(比如核属性)中部分属性值对分辩必不可少,但并不等于该属性中的全部属性值对分辩都重要,反之亦然.因此以整个属性为单位进行约简往往会约去部分重要属性值而使所提取规则非最简;问题(3)的主要原因是试图用低代价的局部择优算法实现全局最优化问题.为此,文中以粗糙集不可分辩关系和多值逻辑化简为理论基础,用逻辑观点研究逻辑值之间的划分及合并,直接求逻辑最简式(提取规则),提出了一种逻辑化简提取规则的完备算法,该算法将逻辑值(属性值)作为基本量化单位,先求出每个逻辑值的行分辩强度和列组合强度,以此为依据,采用行和列综合择优、覆盖递进的化简算法,使最终所得规则准确且全局性最简.

2 原理和算法

2.1 多值逻辑化简原理

决策表的实质是多值逻辑的真值表^[3].对于这种真值表可以采用类似二值逻辑化简方法^[4]进行化简.下面给出相关定义和定理.

定义1 设多值逻辑函数 $C(A_n)$, $\{A_n\} = A_1, A_2, \dots, A_n$; $\{A_i\} = a_{1i}, a_{2i}, \dots, a_{mi}$, $\{C\} = c_1, c_2, \dots, c_m$. $A_i (i = 1, 2, \dots, n)$ 为逻辑变量, $a_{ji} \in \{N\}$ 为逻辑值; $c_i \in \{N\}$ 为逻辑函数值.逻辑变量每组取值对应一个逻辑函数值:

$$a_{i1} \wedge a_{i2} \wedge \dots \wedge a_{ik} \wedge \dots \wedge a_{in} = c_i$$

称为最小项表达式,记为 $m_i = c_i$;而称 m_i 为最小项.

多值逻辑函数 $C(A_n)$ 所构成的数据结构是 $n + 1$ 列、 m 行的真值表, $\{A_i\}$ 是逻辑变量, C 是逻辑函数,表的一行是一个 $m_i = c_j$.

定义2 设两个最小项 m_i 和 m_j ,将两者对应逻辑值分别进行比较运算(用符号 $\oplus$ 表示):若两值相同时结果等于0,否则两值不同时结果等于1.再将这些结果值累加,所得的累加和称为这两个最小项的逻辑距离,记为 Dm_{ij} .

$$Dm_{ij} = \sum_{(k=1,2,\dots,n)} (a_{ik} \oplus a_{jk} \mid a_{ik} \in m_i, a_{jk} \in m_j)$$

逻辑距离 Dm_{ij} 是粗集不可分辩关系用多值逻辑阈运算的数学表式,说明两个最小项 m_i 和 m_j 可分辩的程度.

定理1 设 $m_i = c_i, m_j = c_j (c_i \neq c_j)$,若 $Dm_{ij} = 1$,并设其中 $a_{ik} \oplus a_{jk} = 1$,则 a_{ik} 和 a_{jk} 是不可约简值,所对应的 A_k 是不可约简变量.

证明 设最小项表达式:

$$m_i = a_{i1} \wedge a_{i2} \wedge \dots \wedge a_{ik} \wedge \dots \wedge a_{in} = c_i$$

$$m_j = a_{j1} \wedge a_{j2} \wedge \dots \wedge a_{jk} \wedge \dots \wedge a_{jn} = c_j$$

$$\text{设 } a_{iv} = a_{jv} (v = 1, 2, \dots, k-1, k+1, \dots, n); \\ a_{ik} \neq a_{jk}; c_i \neq c_j.$$

因为只有 $a_{ik} \oplus a_{jk} = 1$,若约简变量 A_k ,

$$m'_i = a_{i1} \wedge a_{i2} \wedge \dots \wedge a_{ik-1} \wedge a_{ik+1} \wedge \dots \wedge a_{in} = c_i$$

$$m'_j = a_{j1} \wedge a_{j2} \wedge \dots \wedge a_{jk-1} \wedge a_{jk+1} \wedge \dots \wedge a_{jn} = c_j$$

则有 $m'_i = m'_j = c_i \vee c_j$,即产生了相同的最小项可得不同的逻辑函数值的逻辑矛盾.所以 A_k 是不可约简变量, a_{ik} 和 a_{jk} 是不可约简值,得证.

推理1 设 $m_i = c_i, m_j = c_j (c_i \neq c_j)$,若 $Dm_{ij} = N (1 \leq N \leq n)$,设 $a_{ik} \oplus a_{jk} = 1 (k = 1, 2, \dots, N)$,若 a_{ik}, a_{jk} 所对应的 $\{A_k\} (k = 1, 2, \dots, N)$ 中任选一个 A_q 保留,则其他 $\{A_p\} (p = 1, 2, \dots, q-1, q+1, \dots, N-1)$ 是可约简的.

2.2 算法

用 Dm_{ij} 为每个逻辑值确定一对强度(行分辩强度和列组合强度),行分辩强度为从行的方向确定每个 $m_i = c_i$ 的最简式的择优条件,而列组合强度为从列的方向确定多个 $m_i = c_i$ 组合的择优条件;以最简表达式性质为准则,以列组合强度和行分辩强度的综合择优条件为标准,求出多个 $m_i = c_i$ 的最简式.

定义3 设有两个最小项表达式 $m_i = c_s, m_j \neq c_s$.由式

$$d_j(a_{ik}) = (a_{ik} \oplus a_{jk})^{(n-v)}. (k = 1, 2, \dots, n)$$

称为逻辑值 a_{ik} 的单项行分辩强度.

其中

$$a_{ik} \in m_i, a_{jk} \in m_j; v = Dm_{ij}; n = |A_n|;$$

“.”是可变长IP地址格式的级别符号,计算时可转换为相应的基数,如设“.”为10.

定义4 设一个最小项 $m_i = c_i$ 和真值表中其他满足 $m_j \neq c_i (j = 1, 2, \dots, q)$ 的最小项, m_i 对每个 m_j 分别求出每个的 $v_j = Dm_{ij}$ 和 $d_j(a_{ik}) (j = 1, 2, \dots, q)$.

$$D(a_{ik}) = \bigcup_{(j=1,2,\dots,q)} (d_j(a_{ik})) (k = 1, 2, \dots, n)$$

称 $D(a_{ik})$ 为逻辑值 a_{ik} 的行分辨强度;而

$$D(c_i) = \bigvee D(a_{ik}) (k = 1, 2, \dots, n)$$

称 $D(c_i)$ 为逻辑函数值 c_i 的行分辨强度;

其中, $\bigcup$ 是按级别分别进行二进制增位合并. 如设 a, b, c, d 是二进制数值, 则有 $a \cdot b \bigcup c \cdot d = ac \cdot bd$.

求行分辨强度的算法如下(算法 1):

```
for(; (m_i = c_i);) {
  for(; (m_j ≠ c_j);) {
    dm_ij = ∑ (a_ik ⊕ a_jk) (k = 1, 2, ..., n);
    da_ik = (a_ik ⊕ a_jk) (·)^(n-dm_ij) (k = 1, 2, ..., n);
    Da_ik ∪ = da_ik (k = 1, 2, ..., n);
    Dc_i ∨ = Da_ik (k = 1, 2, ..., n);
  }
}
```

根据定义 2, $D(a_{ik})$ 值越大表示 a_{ik} 对 c_i 的分辨率能力越强. 对于真值表 $\{A_n, C\}$ 的每个逻辑值和逻辑函数值 $a_{ij}, c_i (i = 1 \sim n, j = 1 \sim m)$ 都有各自的 $D(a_{ij}), D(c_i) \in \{D_{n+1}\}$ 与之对应. 根据定理 1 和推理 1, 当在 i 行中任取 $s (1 \leq s \leq n)$ 个行分辨强度 $D(a_{is})$, 若

$$\bigvee D(a_{is}) = D(c_i)$$

则

$$\bigwedge a_{is} = c_i$$

由 $D(a_{is})$ 和 $D(c_i)$ 可以准确地计算出构成逻辑简式的 a_{is} 和 c_i , 一般可有一组 a_{is} 对 c_i 的化简式, 这就解决了每个 $m_i = c_i$ 化简的基本问题, 经对 $D(a_{ik})$ 选大、覆盖的处理可得每个 $m_i = c_i$ 的最简式 r_i , 算法如下(算法 2):

```
for(i = 1; i ≤ m; i++) {
  for(; {D(a_ij)} ≠ {Φ}; (j = 1, 2, ..., n)) {
    Da_ik = max({D(a_ij)}); // 选大
    D(a_ij) ⊕ = Da_ik; // 覆盖
    r_i+ = a_ik;
  }
}
```

上述是单行 $m_i = c_i$ 求最简式的方法. 然而, 当 $\{C\}$ 中有多个 c_i 时, 就有多个最小项表达式与之对应, 这就要进行多个最小项表达式合并化简. 列组合强度值是对于同一逻辑函数值 c_i 的各最小项相同的强度, 在多个最小项表达式合并化简时它是进行逻辑值优选的依据, 下面给出其定义.

定义 5 设 $m_i = c_s$ 和 $m_j = c_s (j = 1, 2, \dots, q; i \neq j)$. m_i 对 m_j 分别求出各自的 $v_j = Dm_{ij}$ 和 $d_j(a_{ik})$.

$$S(a_{ik}) = \bigcup_{(j=1, 2, \dots, q)} (d_j(a_{ik})) (k = 1, 2, \dots,$$

$n)$

称 $S(a_{ik})$ 为逻辑值 a_{ik} 的列组合强度值; 而

$$S(c_i) = \bigvee S(a_{ik}) (i = 1, 2, \dots, m)$$

称 $S(c_i)$ 为逻辑函数值 c_i 的列组合强度值.

列组合强度是同类逻辑值间以最小项为单位相同的程度, 其算法与算法 1 类同, 根据定义 2, $S(a_{ik})$ 值越大表示在 $C = c$ 的条件下, k 列中 $a (a = a_{ik})$ 的个数越多; 而 $S(c_i)$ 越小表示它与 $S(c_j) (j \neq i, c_i = c_j = c)$ 的组合的条件越少. 对于真值表 $\{A_n, C\}$ 的每个逻辑值和逻辑函数值 $a_{ij}, c_i (i = 1 \sim n, j = 1 \sim m)$ 都有一个 $S(a_{ij}), S(c_i) \in \{S_{n+1}\}$ 与之对应.

逻辑化简是对所有 $m_k = c_i (k \geq 1)$ 化简为 q 个与式(即乘积项)相或而组成的最简表达式, 最简表达式有以下性质^[4-5]:

性质 1 在 q 个与式中所有的逻辑值之和个数最少;

性质 2 每个与式中逻辑值的个数最少;

性质 3 与式的个数 q 最少.

性质 1 是必需条件, 而性质 2、3 是在性质 1 约束下的最优状态. 依据上述定义的行分辨、列组合强度是逻辑值化简强度的量化值, 下面给出解决多个最小项表达式化简的准则和相应的方法:

准则 1 必选逻辑值多的 m_i 必需先化简且在化简中最大限度的包括其他必选逻辑值多的 m_j 一并化简(性质 2).

方法 1: 选 $\max(\{D(c_i)\})$ 的 m_i 先化简, 且尽量包括其他行分辨强度大的 m_j 一并化简.

准则 2 可选性最少的 m_i 优先化简, 这样, 可使后续的化简可选余地更大, 避免增加与式的个数 q (性质 3).

方法 2: 在 $\{S(c_i)\}$ 中, 选列组合强度最小的 m_i 先化简.

综合准则 1、2 的要求, 将 $\{D(c_i)\} - \{S(c_i)\}$ 大小顺序作为行选优条件; 在选定行中, 以 $\{da_{ik}\} - \{sa_{ik}\}$ 大小顺序作为列选优条件, 采用选大、覆盖的算法(算法 2)实现多行化简. 算法如下:

输入: 原始样表 $\{A_n, C\}$; 输出: 规则表 $\{A_r, C\}$

(1) $\{A_n, C\} \rightarrow \{D(A_n), D(C)\}$;

(2) for(; $\{A_n\} \neq \{\emptyset\}$);

(3) $\{A_n, C\} \rightarrow \{S(A_n), S(C)\}$;

(4) $(\{a_{ik}\}, c_i, \{da_{ik}\}, \{sa_{ik}\}) = \max(D_s(C) - S_s(C))$

$(s = 1, 2, \dots, m);$

(5) for($\{da_{ik}\} \neq \{\Phi\}$);

$da_{ip} = \max(\{da_{ik}\} - \{sa_{ik}\})$;

$\{da_{is}\} += da_{ip}; \{da_{ik}\} \oplus = da_{ip} (k = 1, 2, \dots, n)$;

(6) $(\wedge a_{is} = c_i) \rightarrow \{A_r\}$;

(7) $\{A_n\} -= \{a_{is}\}$;

说明:

(1) 按算法 1 生成 $\{A_n, C\}$ 的行辩强度 $\{D(A_n), D(C)\}$.

(2) 以 $\{A_n\} \neq \{\phi\}$ 为条件循环, 每次得一个最简式.

(3) 按算法 1 生成 $\{A_n, C\}$ 的列组强度 $\{S(A_n), S(C)\}$.

(4) 得最优行 i 的相关数据 $(\{a_{ik}\}, c_i, \{da_{ik}\}, \{sa_{ik}\})$.

(5) 在 $\{da_{ik}\}$ 和 $\{sa_{ik}\}$ 中选出最简集 $\{a_{is}\}$, $\{a_{is}\}$ 是多行 m_i 的最简式. 每次在当前 $\{da_{ik}\}$ 和 $\{sa_{ik}\}$ 选中行辩、列组合强度差值最大者 da_{ip} (满足准则 1 和 2 条件), 将 da_{ip} 加入 $\{da_{is}\}$ 并用 da_{ip} 对 $\{da_{ik}\}$ 覆盖, 再次循环直到 $\{da_{ik}\} = \{\Phi\}$.

(6) 将最简式 $(\wedge a_{is} = c_i)$ 存入 $\{A_r\}$;

(7) 从 $\{A_n\}$ 中删除已化简的行集 $\{a_{is}\}$.

3 实例与分析

表 1 是一个无核属性决策表. 按算法 1 求得该表中每个逻辑值 $\{a_{ik}\}$ 及逻辑函数值 $\{c_i\}$ 的行辩强度值, 见表 2. 以 $m_i = c_i \in C$ 将表 1 作划分, 每个划分集按算法 1 求得该集中每个逻辑值 $\{a_{ik}\}$ 及逻辑函数值 $\{c_i\}$ 的列组合强度值, 见表 3.

表 1 原始真值表

U	A_1	A_2	A_3	A_4	A_5	C
1	2	<u>1</u>	<u>2</u>	3	1	<u>2</u>
2	4	1	2	4	1	2
3	<u>3</u>	2	2	3	<u>1</u>	<u>3</u>
4	3	3	1	1	1	3
5	3	4	3	1	1	3
6	2	1	<u>4</u>	<u>1</u>	1	<u>3</u>
7	1	5	4	1	2	3
8	3	<u>1</u>	<u>1</u>	<u>1</u>	2	<u>4</u>
9	4	1	1	1	3	4
10	<u>1</u>	1	4	3	<u>3</u>	5
11	1	1	1	4	3	5

表 2 行辩强度值表

U	DA_1	DA_2	DA_3	DA_4	DA_5	DC
1	1.1.31	1.0.24	2.1.31	2.0.31	0.1.07	3.1.31
2	11.15	1.05	14.15	13.15	6.10	15.15
3	1.1.1	1.1.3	0.0.3	0.1.2	0.0.3	1.1.3
4	0.1.7	1.1.7	0.0.3	0.0.7	1.1.4	1.1.7
5	0.7	1.7	1.7	0.7	1.4	1.7
6	0.15.1	0.00.0	1.14.1	1.03.1	0.13.1	1.15.1
7	2.2	3.3	2.3	1.1	1.3	3.3
8	0.11.07	1.05.08	0.07.15	0.08.15	1.14.15	1.15.15
9	1.11.7	0.02.3	0.13.7	1.12.4	0.00.7	1.15.7
10	14.7	1.3	12.7	7.5	11.7	15.7
11	1.3.14	0.0.05	0.2.11	1.1.15	0.3.15	1.3.15

表 3 列组合强度值表

U	SA_1	SA_2	SA_3	SA_4	SA_5	SC
1	1.0.0	0	0	1.0.0	0	1.0.0
2	1.0.0	0	0	1.0.0	0	1.0.0
3	0.1	3.1	3.1	3.1	0	3.1
4	0.2.1	1.3.1	1.3.1	0.1.0	0.0.1	1.3.1
5	0.2.1	1.3.1	1.3.1	0.1.0	0.0.1	1.3.1
6	7.1	7.1	3.1	0.1	4.0	7.1
7	1.3	1.3	0.3	0	1.3	1.3
8	1.0.0	0	0	0	1.0.0	1.0.0
9	1.0.0	0	0	0	1.0.0	1.0.0
10	0	0	1.0.0	1.0.0	0	1.0.0
11	0	0	1.0.0	1.0.0	0	1.0.0

对 $C = 2$, 行选优: 按 $D(C) - S(C)$ 的大小顺序是 U_1, U_2 , 选中 U_1 .

列选优: 由 $DA_{3_1} - SA_{3_1} = 2.1.31$ 最大, 选 A_3 (第 3 列), 用 DA_{3_1} 对 $DA_{J_1} (J = 1, 2, \dots, 5)$ 覆盖 (见算法 2), 此后 $DA_{2_1} - SA_{2_1} = 1.0.3$ 最大, 用 DA_{2_1} 再对 DA_{J_1} 覆盖一次后 $DA_{J_1} = 0 (J = 1, 2, \dots, 5)$, 结束选择, 即 $(A_{2_1} = 1) \wedge (A_{3_1} = 2)$ 是 U_1 和 U_2 行的化简式, 将该式记入规则表, 见表 1 中 U_1 行中的下划线部分.

对 $C = 3$, 第 1 次行选优: 按 $(D(C) - S(C))$ 降序的结果是 U_6, U_3, U_7, U_4, U_5 , 先选中 U_6 .

第 1 次列选优: 由 $DA_{3_6} - SA_{3_6}$ 最大, 选 A_3 (第 3 列), 用 DA_{3_6} 对 $DA_{J_6} (J = 1, 2, \dots, 5)$ 覆盖 (见算法 2), 此后 $DA_{4_6} - SA_{4_6}$ 最大, 用 DA_{4_6} 再对 DA_{J_6} 覆盖一次后 $DA_{J_6} = 0 (J = 1, 2, \dots, 5)$, 结束选择, 即 $(A_{3_6} = 4) \wedge (A_{4_6} = 1)$ 是 $U_6 \sim U_7$ 行的化简式. 将该式记入规则表, 见表 1 的 U_6 行中的下划线部分.

对 $C = 3$, 第2次行、列选优是在 $U_3 \sim U_5$ 中完成的, 过程与上述 $C = 2$ 的情况类似, 得表 1 的 U_3 下划线部分, 这样, 对 $C = 3$ 的 $U_3 \sim U_7$ 行共用 2 个最简式实现了化简.

类似 $C = 2$ 的情况, 对 $C = 4$ 和 5, 各生成 1 条规则, 见表 1 的 U_8 和 U_{10} 的下划线部分. 对原始真值表 1 用 11 个逻辑值化简为 5 个最简逻辑式(规则), 表 1 中全部下划线部分. 即

- (1) $A2.1 \wedge A3.2 = C.2$;
- (2) $A1.3 \wedge A5.1 = C.3$;
- (3) $A3.4 \wedge A4.1 = C.3$;
- (4) $A2.1 \wedge A3.1 \wedge A4.1 = C.4$;
- (5) $A1.1 \wedge A5.3 = C.5$.

若按属性约简算法, 表 1 可得的最佳独立集是 $\{A_2, A_3, A_4, C\}$, 由于约简了 $A_1 = 3, 1$ 和 $A_5 = 1, 3$ 这两组强化简的逻辑值, 使所得规则为 8 个, 而所需用 14 逻辑值. 即

- (1) $A2.1 \wedge A3.2 = C.2$;
- (2) $A2.2 = C.3$;
- (3) $A2.3 = C.3$;
- (4) $A2.4 = C.3$;
- (5) $A3.4 \wedge A4.1 = C.3$;
- (6) $A2.1 \wedge A3.1 \wedge A4.1 = C.4$;
- (7) $A3.4 \wedge A4.3 = C.5$;
- (8) $A3.1 \wedge A4.4 = C.5$.

相比文中算法, 属性约简算法的规则多增了 3 个、逻辑值多增了 3 个. 对于较大的决策表, 这种增加更加明显.

4 结束语

文中从决策表的全局出发, 以属性值为基本量化单位完备地实现规则的最简提取, 给出以下结论:

- (1) 指出当前以属性为基本单位的属性约简算

法往往会造成最终规则的冗余, 即粗集所定义的“不必要属性”并非其每个属性值对约简都不重要、核属性并非其每个属性值对约简都重要^[6].

(2) 提出多值逻辑距离的定义, 应用该定义量化每个逻辑值的可分辨强度(行分辨强度)和相似强度(列组合强度), 使化简精准无误; 并用可变长 IP 地址格式表示量化值, 有效地压缩数据空间.

(3) 提出以逻辑值为基本量化单位, 以行分辨强度与列组合强度之差作为择优依据, 判别条件简洁、准确, 具体实现多值逻辑化简的理想准则. 使所得规则最简.

(4) 对有核、无核决策表统一处理.

(5) 选用可分辨关系极弱的无核决策表(见表 1)作为实例进行分析, 经与属性约简算法比较本算法所得的规则最简, 从而证明了算法的正确性.

参考文献:

- [1] 刘洋. 一种改进的基于差别矩阵的属性约简算法[J]. 微电子学与计算机, 2007, 24(5): 133-135.
- [2] 陈淑珍. 基于粗集的几种属性约简算法分析[J]. 武汉工业学院学报, 2005(9): 118-120.
- [3] 沈继忠. 基于多值方法的模糊逻辑函数最小化[J]. 计算机学报, 2004, 27(4): 555-560.
- [4] 康华光. 电子技术基础数字部分[M]. 5 版. 北京: 高等教育出版社, 2006: 29-55.
- [5] 邱建林. 多输入多输出单边逻辑函数优化系统的设计研究[J]. 南京邮电大学学报: 自然科学版, 2006, 26(5): 65-69.
- [6] 赵辉. 基于不完备信息系统的粗模糊属性值的知识获取[J]. 微电子学与计算机, 2008, 25(8): 132-134.

作者简介:

何 群 男, (1955-), 高级工程师, 硕士生导师. 研究方向为数据挖掘、粗糙集.