文章编号：1000－2243(2000)05－0009－04

一个改进的粗糙集属性约简算法

叶东毅，陈昭炯

(福州大学计算机科学与技术系，福建 福州 350002)

摘要：利用单属性的逼近精度，在 Jelonek 属性约简算法的基础上，得到一个改进的属性约简算法. 实例计算结果表明，在获得同样的属性约简的情况下，该算法与 Jelonek 算法相比，计算量较少，提高了计算速度.

关键词：粗糙集；逼近精度；属性；约简

中图分类号：TP182 文献标识码：A

粗糙集理论是分析不完整、不精确的信息系统的有力工具，目前在机器学习，数据挖掘，人工神经网络等领域得到了广泛的应用[1－6]. 属性约简是粗糙集理论中的一个核心部分，可用于知识约简. 由于求所有属性约简是NP难的问题，因此到目前为止，还没有一个高效的求最佳和所有属性约简的算法. 不过，在实际应用中，往往只要求出某种次优的(相对)属性约简就可以了. 为此，人们已提出了若干个比较简单的信息表属性约简算法，Jelonek[6] 等人提出的算法是其中比较典型的一个，取得了较好的效果，但也存在一些不足，它必须计算很多不同属性子集的逼近精度才能决定如何扩展候选属性约简，因此，需要较多的计算量. 本文对此进行改进，利用单属性的逼近精度，在 Jelonek 算法的基础上，得到一个新的属性约简算法. 实例计算结果表明，在获得同样的属性约简的情况下，本算法与 Jelonek 算法相比，计算量较少，提高了计算速度.

1 改进的约简算法

给定一个信息系统[2]：

$$L = (U, Q, V_q, F_q),\ q \in Q$$

其中：U 是论域，Q 是属性集合，V_q 为属性取值的集合，F_q 是 $U \times Q \to V_q$ 的映射. 在多值决策应用中，属性集合 Q 通常分为条件属性集 C 与决策属性集 D.

设 $P \subseteq C$，$\mathfrak{L}$ 为由决策属性 D 所决定的 U 的划分 $\{Y_1, Y_2, \cdots, Y_K\}$，则对划分 $\mathfrak{L}$ 的 P－逼近精度(approximation quality)为[2]：

$$\gamma_P(\mathfrak{L}) = \sum_{i=1}^{k} \mathrm{card}(\underline{P}Y_i)/\mathrm{card}(U)$$

其中 $\underline{P}Y_i$ 为 Y_i 的下逼近(lower approximation).

设 P、$R \subseteq C$，且 $R \subseteq P$，若 $\gamma_R(\mathfrak{L}) = \gamma_P(\mathfrak{L})$，且不存在 $R' \subset R \subseteq P$，使得 $\gamma_{R'}(\mathfrak{L}) = \gamma_P(\mathfrak{L})$，则称 R 为 P 的一个相对属性约简[2]，记为 RED(P, $\mathfrak{L}$). 由此定义可以看出，约

收稿日期：1999－10－20

作者简介：叶东毅(1964－)，男，教授.

基金项目：福建省自然科学基金资助项目(A0010009)；福建省优秀留学回国人员科研资助项目

简后保持划分 $\mathfrak{L}$的逼近精度不变. 所有RED (P, $\mathfrak{L}$)的交为核 (core). 在实际计算中, 可以通过确定关键 (indispensable) 属性来得到核, 而无需求出所有的属性约简[5].

对于求相对属性约简的问题, Jelonek 等人算法[6] 的基本思想是从核 (core) 开始, 如果 $\gamma_R = \gamma_C$, 则 R 为一个属性约简; 否则, 对所有的属性 $a \in C/R$, 计算 gain $= \gamma_{R\cup\{a\}} - \gamma_R$, 若 a' 是使相应的gain达到最大的属性, 则置 $R = R \cup \{a'\}$, 继续上述过程. 在该算法中, 从核(可以是空集) 开始, 每扩展一次, 都要对所有的属性 $a \in C/R$, 计算新的逼近精度 $\gamma_{R\cup\{a\}}$, 计算量是比较大的. 如果每扩展一次, 只需对部分的属性计算逼近精度 $\gamma_{R\cup\{a\}}$, 甚至无须计算新的逼近精度, 则可望减少计算量, 提高算法速度. 本文算法就是基于这种思路对 Jelonek 算法进行改进的. 下面给出这一算法.

设信息系统中条件属性集 C 中有 m 个属性: C_1, C_2, ……, C_m, 其值域为有限离散集合, 决策属性为 D, 其等价类构成 U 的划分 $\mathfrak{L}$为$\{Y_1, Y_2, \cdots, Y_k\}$. 算法的基本过程如下:

1) 计算下逼近$\underline{C}Y_i$, $i = 1, \cdots, k$, 由此可得 $\gamma_C(\mathfrak{L})$; 分别对每个条件属性 C_j, $j = 1, \cdots, m$, 计算 $\gamma_{C_j}(\mathfrak{L})$; 令 $P =$ 核 core(其计算思想参见 [5]), $D = C$.

2) 如果 P 满足 $\gamma_P(\mathfrak{L}) = \gamma_C(\mathfrak{L})$, 则停止, P 为一个 RED (C, $\mathfrak{L}$); 否则, $\gamma_P(\mathfrak{L}) < \gamma_C(\mathfrak{L})$, 转步骤 3.

3) 计算 $\gamma_{C_g}(\mathfrak{L}) = \max(\gamma_{C_i}(\mathfrak{L}): C_i \in D)$, 如果 $\gamma_{C_g}(\mathfrak{L})$ 是唯一的最大值, 则置 $P = P \cup \{C_g\}$, $D = D \setminus \{C_g\}$, 转回步骤 2 继续执行; 否则, 记 $Q = \{C_i \in D: \gamma_{C_i}(\mathfrak{L}) = \gamma_{C_g}(\mathfrak{L})\}$, 计算 $\gamma_{P\cup\{C_q\}}(\mathfrak{L}) = \max(\gamma_{P\cup\{C_i\}}(\mathfrak{L}): C_i \in Q)$(如果有多个最大值, 则取属性取值个数最少的那个属性), 置 $P = P \cup \{C_q\}$, $D = D \setminus \{C_q\}$, 转回步骤 2 继续执行.

在上述算法的步骤 3 中, 有两种情形: ① 如果 $\gamma_{C_g}(\mathfrak{L})$ 是唯一的最大值, 则无须计算新的逼近精度即可扩展 P, 因此, 计算量少. 这点与 Jelonek 算法是不同的; ② 在另一种情况下, 当 $Q = D$ 时, 与 Jelonek 算法是等价的, 当 $Q \subset D$ 时, 比 Jelonek 算法节省计算量.

2 算例

例 1 表 1 所示的是一组汽车数据[3], 分析汽车行驶里程价格与其他 9 个影响因素之间的关系. 将表 1 看作一个信息系统, 则论域 U 就是 21 种汽车, 决策属性集合 D 为{里程}, 条件属性集合 C 为{车型, 汽缸, 涡轮机, 燃料, 位移, 压缩率, 功率, 挂档, 重量}, 由 D 决定的划分 $\mathfrak{L}= \{Y_1, Y_2, Y_3\} = \{\{1, 2, 3, 5, 9, 10, 13, 16, 17, 18, 21\}, \{4, 8, 11, 12, 14, 15, 19, 20\}, \{6, 7\}\}$ 根据本文算法, 先扩展到核 $P =$ {重量, 燃料}, 这与Jelonek 算法相同; 当扩展到 $P =$ {重量, 燃料, 车型} 时, 不要再计算新的逼近精度(对应步骤 3①), 而Jelonek 算法需要对 7 个不同的 3 − 属性集({重量, 燃料, 车型, }, {重量, 燃料, 汽缸}, {重量, 燃料, 涡轮机}, {重量, 燃料, 位移}, {重量, 燃料, 压缩率}, {重量, 燃料, 功率}, {重量, 燃料, 挂档. }) 计算它们的逼近精度, 因此, 计算量明显增加. 两个算法再扩展到最小属性约简 $C^0 =$ {重量, 车型, 燃料, 位移} 的过程是相同的. 因此, 就总的计算量而言, 本文算法比较少.

例 2 表 2 所示的是一组信用卡申请的数据[2], 决策属性集 D 为{d}, 条件属性集 C 为 $\{C_1, C_2, C_3, C_4\}$, 划分 $\mathfrak{L}= \{Y_1, Y_2\} = \{\{1, 4, 6, 7\}, \{2, 3, 5, 8\}\}$. 根据本文算法, 容易求得最小属性约简 $P = \{C_1, C_2\}$. 同例 1 一样, 与 Jelonek 算法相比, 本算法计算量较少.

表 1　汽车数据库

序号	车型	汽缸	涡轮机	燃料	位移	压缩率	功率	挂档	重量	里程
1	小型	6	Y	EFI	中等	高	高	自动	中等	中等
2	小型	6	N	EFI	中等	中等	高	手动	中等	中等
3	小型	6	N	EFI	中等	高	高	手动	中等	中等
4	小型	4	Y	EFI	中等	高	高	手动	轻	高
5	小型	6	N	EFI	中等	中等	中等	手动	中等	中等
6	小型	6	N	2— BBL	中等	中等	中等	自动	重	低
7	小型	6	N	EFI	中等	中等	高	手动	重	低
8	微型	4	N	2— BBL	小	高	低	手动	轻	高
9	小型	4	N	2— BBL	小	高	低	手动	中等	中等
10	小型	4	N	2— BBL	小	高	中等	自动	中等	中等
11	微型	4	N	EFI	小	高	低	手动	中等	高
12	微型	4	N	EFI	中等	中等	中等	手动	中等	高
13	小型	4	N	2— BBL	中等	中等	中等	手动	中等	中等
14	微型	4	Y	EFI	小	高	高	手动	中等	高
15	微型	4	N	2— BBL	小	中等	低	手动	中等	高
16	小型	4	Y	EFI	中等	中等	高	手动	中等	中等
17	小型	6	N	EFI	中等	中等	高	自动	中等	中等
18	小型	4	N	EFI	中等	中等	高	自动	中等	中等
19	微型	4	N	EFI	小	高	中等	手动	中等	高
20	小型	4	N	EFI	小	高	中等	手动	中等	高
21	小型	4	N	2— BBL	小	高	中等	手动	中等	中等

表 2　信用卡申请的数据

申请者	C_1(帐户)	C_2(余额)	C_3(职业)	C_4(月消费)	d 决策
1	银行	中等	有	低	同意
2	银行	低	有	高	拒绝
3	无	低	有	中等	拒绝
4	其他金融机构	高	有	高	同意
5	其他金融机构	中等	有	高	拒绝
6	其他金融机构	高	有	低	同意
7	银行	高	无	中等	同意
8	无	低	无	低	拒绝

应用上述算法对文献[4] —[6] 中其他的一些典型例子进行计算，与 Jelonek 算法相比，计算量明显减少，取得良好的效果.

3　结语

本文对 Jelonek 算法进行改进，提出一个改进的属性约简算法. 实际计算结果表明，该算法容易实现，计算量相对较少，而且在很多情况下，能求得一个最小属性约简.

参考文献:

[1] Pawlak Z. Rough set: theoretical aspects of reasoning about data[M]. Dordrecht: Kluwer Academic Publishers, 1991.

[2] Pawlak Z, Slowinski R. Rough set approach to multiattribute decision analysis, invited review [J]. European Journal of Operational Research, 1994, 72: 443—459.

[3] 王 珏. Rough Sets 约简与数据浓缩[J]. 高技术通讯, 1997, 11: 40—45.

[4] Mrozek A. Rough sets and dependency analysis among attributes in computer implementations of expert' sinference models[J]. Int J Man—Machine Studies, 1989, 30: 457—471.

[5] Pawlak Z, Wong S K M, Ziarko W. Rough sets: probabilistic versus deterministic approach[J]. Int J Man—Machine Studies, 1988, 29: 81—95.

[6] Jelonek J, Krawiec K, Slowinski R. Rough set reduction of attributes and their domains for neural networks [J]. Computational Intelligence, 1995, 11(2): 339—347.

An improved reduction algorithm of attribute in rough set

YE Dong—yi, CHEN Zhao—jiong

(Department of Computer Science and Technology, Fuzhou University, Fuzhou, Fujian 350002, China)

Abstract: Based on Jelonek' s algorithm and approximation quality for a single attribute, an improved reduction algorithm of attributes is given in this paper. Numerical experiments show that this algorithm, obtaining the same reduction as Jelonek' s algorithm , requires less computational effort than the latter, thus improving the efficiency of attribute reduction.

Keywords: rough set; approximation quality; attribute; reduction