文章编号: 1001-0920(2011)07-0984-07

基于冲突的增量式核属性更新算法

葛 浩[1a], 李龙澍[2], 杨传健[1b]

(1. 滁州学院 a. 电子信息工程系，b. 计算机系，安徽 滁州 239012；2. 安徽大学 计算机学院，合肥 230039)

摘 要: 目前的增量式核属性更新算法大多建立在可分辨矩阵的基础上, 不利于大数据集处理, 而且算法的效率也不理想. 为了克服上述不足, 在改进的基于冲突域求核算法的基础上, 提出了基于冲突的增量式核属性求解算法. 该算法不需要创建可分辨矩阵, 当有新对象加入时, 对决策表仅需进行一次遍历便可完成核属性增量更新. 算法的时间和空间复杂度分别为 $O(|C||U'|)$ 和 $O(|C|)$. 理论分析和实验结果均表明, 所提出的算法是可行且高效的.

关键词: 粗糙集；冲突域；核属性；增量计算

中图分类号: TP181 **文献标识码:** A

Incremental updating algorithm of the computation of core based on the collision

GE Hao[1a], *LI Long-shu*[2], *YANG Chuan-jian*[1b]

(1a. Department of Electronic and Information Engineering，1b. Department of Computer Science，Chuzhou University, Chuzhou 239012，China；2. School of Computer Science，Anhui University，Hefei 230039，China．Correspondent: GE Hao，E-mail：togehao@126.com)

Abstract: At present, a lot of incremental algorithms of updating the core are based on the distinguishable matrix, which are disadvantageous to the larger database, and the efficiency of algorithms is not ideal. To overcome above shortcomings, on the basis of the improved algorithm for computing the core based on the conflict region, the algorithm of incremental computing core based on the collision is proposed. The algorithm is unnecessary to create the distinguishable matrix and only needs to traverse the decision table one time, when a new object is added into the decision table. The time complexity and space complexity of the algorithm are $O(|C||U'|)$ and $O(|C|)$ respectively. Both of theoretical analysis and experimental results show that the algorithm is effective and efficient.

Key words: rough set；conflict region；core attribute；incremental computation

1 引 言

粗糙集[1]是波兰数学家 Pawlak 教授于 1982 年提出的一种处理含糊和不精确知识的数学工具, 它能有效地分析和处理不精确、不一致、不完备的信息, 从海量数据中发现隐含的知识. 属性约简是粗糙集理论研究的主要内容之一, 已引起众多学者的关注[2-6], 而属性约简算法常以核属性集为初始约简集, 因而求核是该类方法的关键. 现有的核属性求解方法有: 基于可分辨矩阵的求核方法, 基于正区域的求核方法和基于信息熵的求核方法.

Hu 等人[2]根据 Skowron 可分辨矩阵[3]提出一种基于可分辨矩阵的求核方法, 该算法的时间和空间复杂度均为 $O(|C||U|^2)$. 叶东毅等人[7]举例说明了 Hu方法中存在的问题, 对可分辨矩阵进行了改进, 并提出了新的算法, 新算法的时间和空间复杂度仍均为 $O(|C||U|^2)$. 王国胤[8]指出, 叶东毅虽然给出了修正方法, 但没有发现产生错误的根本原因是由于不相容规则的比较导致的. 杨明等人[9]提出了一种新的改进的可分辨矩阵及其求核方法, 有效地解决了 Hu 方法存在的错误, 该算法的时间和空间复杂度分别为 $\max\{O(|C||U|\log|U|), O(|C||U||\mathrm{POS}_C(D)|)\}$ 和 $O(|C||U||\mathrm{POS}_C(D)|)$; 文献 [10] 提出了基于冲突域的核属性求解算法, 该方法不需要创建可分辨矩阵,

收稿日期: 2010-03-29；**修回日期:** 2010-09-24.

基金项目: 安徽省自然科学基金项目(090412054)；安徽高校省级自然科学研究项目(KJ2010B137)；安徽省高等学校省级优秀青年人才基金项目(2010SQRL138).

作者简介: 葛浩(1976−), 男, 副教授, 硕士, 从事数据挖掘和粗糙集的研究；李龙澍(1956−), 男, 教授, 博士生导师, 从事不精确信息处理和智能软件等研究.

因而大大节省了存储空间, 算法的时间和空间复杂度分别为 $O(|C|^2|U|)$ 和 $O(|U|)$.

上述求核问题针对的是静态信息系统, 然而现实世界中信息系统的对象在不断地动态变化, 新的对象增加了, 已得到的核可能不再有效, 这就需要对核进行动态修改. 杨明[11]提出了基于可分辨矩阵的增量式核属性求解算法, 该算法的时间和空间复杂度分别为 $O(|C|(5|U_1|+3|U_2'|))$ 和 $O(|C|(|U1|^2+|U_1||U_2'|))$ (文献 [11] 在分析复杂度时没有考虑条件属性 C 的基), 但该方法需要建立在已有可分辨矩阵的基础上, 而如果处理海量信息系统, 必将占用很大的空间, 这将是制约求核算法效率的瓶颈. 为此, 本文提出一种增量式求核方法. 首先对决策表进行异类化和简化, 证明简化决策表的核属性与原始决策表核属性是等价性的, 并在此基础上对文献 [10] 提出的基于冲突域的求核算法加以改进; 然后提出基于冲突的增量式核属性求解方法, 该方法不依赖于可分辨矩阵, 仅需对信息系统进行一次遍历便可完成核属性的更新; 最后设计增量式求核算法, 算法的时间和空间复杂度分别为 $O(|C||U'|)$ 和 $O(|C|)$.

2　决策表简化及核的等价性

决策表

$$S = (U, A, V, f).$$

其中: U 为论域, 是对象的有限集; A 为属性集, $A = C \bigcup D$ 且 $C \bigcap D = \phi$, C 为条件属性集, D 为决策属性集; $V = \bigcup\limits_{a \in A} V_a$, V_a 是属性 a 的值域; $f : U \times A \to V$ 为信息函数, $a \in A$, $x \in U$, 有 $f(x, a) \in V_a$. 不失一般性, 也为了便于操作, 假设 D 仅有一个决策属性值, 其取值范围是 $1, 2, \cdots, k$ 的整数.

决策表 S 中, 若 $\exists x_i, x_j \in U (i \neq j)$, 有

$$f(x_i, C) = f(x_j, C) \bigwedge f(x_i, D) \neq f(x_j, D),$$

则称 S 为不相容决策表, x_i 和 x_j 称为不相容对象 (或称冲突对象); 否则, 称 S 为相容决策表.

关于粗糙集的其他一些概念请参见文献 [1].

定理 1 [12]　给定决策表 $S = (U, C \bigcup D, V, f)$, $\forall a \in C$ 为核的充分必要条件是

$$\mathrm{POS}_{C-\{a\}}(D) \neq \mathrm{POS}_C(D).$$

性质 1　在决策表 $S = (U, C \bigcup D, V, f)$ 中, 设 $P \subseteq C$, 则 $\mathrm{POS}_P(D) = \{x_i | \forall x_j \in [x_i]_P (i \neq j)$, 有 $f(x_i, P) = f(x_j, P)$ 且 $f(x_i, D) = f(x_j, D)\}$.

证明　令 $U/D = \{Y_1, Y_2, \cdots, Y_m\}$, 则

$$\mathrm{POS}_P(D) = P\_Y_1 \bigcup P\_Y_2 \bigcup \cdots \bigcup P\_Y_m.$$

设 $\exists x_i \in U, x_i \in \mathrm{POS}_P(D)$, 则 $\exists Y_k \in U/D$, $1 \leqslant k \leqslant m$, 有 $x_i \in P\_Y_k$. 由下近似集定义 $P\_Y_k = \{x \in U | [x]_P \subseteq Y_k\}$, 可得 $[x_i]_p \subseteq Y_k$, 则 $\forall x_j \in [x_i]_p$ 有 $f(x_i, P) = f(x_j, P)$ 且 $f(x_i, D) = f(x_j, D)$. □

定义 1　在决策表 $S = (U, C \bigcup D, V, f)$ 中, 新的决策表 $S_1 = (U, C \bigcup D, V', f')$, 其满足

$$f'(x_i, C) = f(x_i, C);$$

$$f'(x_i, D) = \begin{cases} V_\xi, \ \exists x_j (i \neq j), \\ \quad f(x_i, C) = f(x_j, C) \bigwedge f(x_i, D) \neq f(x_j, D); \\ f(x_i, D), \ \text{otherwise}. \end{cases}$$

其中 V_ξ 表示不同于 S 中任何决策值, 即使有动态数据加入, 也不会出现与之相同的决策值.

由定义 1 可知, 决策表 S_1 是将原决策表 S 中不相容对象异类化, 使原始决策表 S 转化为异类化的决策表 S_1. 可见 S_1 为相容决策表.

定义 2　在决策表 $S_1 = (U, C \bigcup D, V', f')$ 中, 设

$$U/C = \{[x_1]_C, [x_2]_C, \cdots, [x_l]_C, [x_{l+1}]_C, \cdots, [x_t]_C\}.$$

其中: 对于 $\forall x_i, x_j \in [x_r]_C (i \neq j, 1 \leqslant r \leqslant l)$, 有 $f(x_i, D) f(x_j, D)$; 在 $[x_s]_C (l < s \leqslant t)$ 中, $\exists x_i, x_j \in [x_s]_C (i \neq j)$, 有 $f(x_i, D) \neq f(x_j, D)$; 记 $U_1' = \{x_1, x_2, \cdots, x_l\}$, $U_2' = \{x_{l+1}, \cdots, x_t\}$, $U' = U_1' + U_2' = \{x_1, x_2, \cdots, x_t\}$, 则称 $S_1' = (U', C \bigcup D, V', f')$ 为 S 的简化决策表.

由定义 2 可知, S_1' 是 S_1 消除了重复对象后的决策表, 也是相容的.

性质 2　对于决策表 $S_1' = (U', C \bigcup D, V', f')$, 有 $\mathrm{POS}_C'(D) = U'$.

证明　因为 S_1' 为相容决策表, 所以由性质 1 可得 $\mathrm{POS}_C'(D) = U'$. □

定理 2　设决策表 S 的核为 $\mathrm{Core}(C)$, 决策表 S_1' 的核为 $\mathrm{Core}'(C)$, 则有 $\mathrm{Core}(C) = \mathrm{Core}'(C)$.

证明　1) 首先证明 $\mathrm{Core}(C) \subseteq \mathrm{Core}'(C)$. 只要证明对于 $\forall a \in \mathrm{Core}(C)$, 有 $a \in \mathrm{Core}'(C)$ 即可.

因为 $a \in \mathrm{Core}(C)$, 故有

$$\mathrm{POS}_{C-\{a\}}(D) \neq \mathrm{POS}_C(D),$$

设 $\exists x \in U$, $x \in \mathrm{POS}_C(D)$, 但 $x \notin \mathrm{POS}_{C-\{a\}}(D)$, 则 $\exists y \in U (x \neq y)$, 有

$$f(x, C - \{a\}) = f(y, C - \{a\}) \bigwedge f(x, D) \neq f(y, D),$$

由定义 2, 可设 $x \in [x']_C$, $y \in [y']_C$. 因为 $x \in \mathrm{POS}_C(D)$, 故 $x' \in \mathrm{POS}_C(D)$, 也有 $x' \in \mathrm{POS}_C'(D)$. 对于 y', 存在以下 2 种情况:

① $y' \in U_1'$, 有 $f(x, D) = f'(x, D)$, $f(y, D) = f'(y, D)$. 因为 $f(x, D) \neq f(y, D)$, 故 $f'(x', D) \neq f'(y', D)$.

② $y' \in U_2'$, 有 $f'(y', D) = V_\xi$, 而 $f'(x', D) \neq V_\xi$,

故 $f'(x',D)\neq f'(y',D)$. 因为 $f(x,C-\{a\})=f(y,C-\{a\})$, 由定义 1 和定义 2 知

$$f'(x',C-\{a\})=f'(y',C-\{a\}).$$

由 ① 和 ② 均得 $f'(x',D)\neq f'(y',D)$, 故

$$f'(x',C-\{a\})=f'(y',C-\{a\})\bigwedge f'(x',D)\neq f'(y',D),$$

则 $x'\notin \mathrm{POS}'_{C-\{a\}}(D)$, 因此可得

$$\mathrm{POS}'_{C-\{a\}}(D)\neq \mathrm{POS}'_{C}(D),$$

故 $a\in \mathrm{Core}'(C)$. $\mathrm{Core}(C)\subseteq \mathrm{Core}'(C)$ 得证.

2) 证明 $\mathrm{Core}'(C)\subseteq \mathrm{Core}(C)$. 只要证明对于 $\forall a\in \mathrm{Core}'(C)$, 有 $a\in \mathrm{Core}(C)$ 即可.

因为 $a\in \mathrm{Core}'(C)$, 故有

$$\mathrm{POS}'_{C-\{a\}}(D)\neq \mathrm{POS}'_{C}(D),$$

设 $\exists x\in U'$, $x\in \mathrm{POS}'_C(D)$, 但 $x\notin \mathrm{POS}'_{C-\{a\}}(D)$, 则 $\exists y\in U'(x\neq y)$, 有

$$f'(x,C-\{a\})=f'(y,C-\{a\})\bigwedge f'(x,D)\neq f'(y,D).$$

x,y 存在以下 2 种情况:

① $x,y\in U'_1$, 则 $x,y\in \mathrm{POS}_C(D)$. 由定义 1 和定义 2, 有 $f(x,C-\{a\})=f(y,C-\{a\})\bigwedge f(x,D)\neq f(y,D)$, 得 $x\notin \mathrm{POS}_{C-\{a\}}(D)$.

② $x\in U'_1$, $y\in U'_2$, 则 $x\in \mathrm{POS}_C(D)$. 由于 $y\in U'_2$, 由定义 1 和定义 2 知, $\exists y'\in U(y\neq y')$, 有 $f(y,C)=f(y',C)\bigwedge f(y,D)\neq f(y',D)$. 故 $f(x,D)\neq f(y,D)$和 $f(x,D)\neq f(y',D)$ 至少有一个成立. 不妨设 $f(x,D)\neq f(y,D)$, 则有 $f(x,C-\{a\})=f(y,C-\{a\})\bigwedge f(x,D)\neq f(y,D)$, 得 $x\notin \mathrm{POS}_{C-\{a\}}(D)$.

由 ① 和 ② 可得 $\mathrm{POS}_{C-\{a\}}(D)\neq \mathrm{POS}_C(D)$, 故 $a\in \mathrm{Core}(C)$. $\mathrm{Core}'(C)\subseteq \mathrm{Core}(C)$ 得证.

综合 1) 和 2), 有 $\mathrm{Core}(C)=\mathrm{Core}'(C)$. □

3 改进的求核算法

定义 3 [10] 决策表 $S=(U,C\bigcup D,V,f)$ 中, $P\subseteq C$, D 的 P 冲突域记为 $\mathrm{ConSet}(P)$, 定义为

$$\mathrm{ConSet}(P)=U/P-\mathrm{POS}_P(D)=\{Z_1,Z_2,\cdots,Z_k\}.$$

由定义 3 可知, $\mathrm{ConSet}(P)$ 是对 S 按条件属性集 P 划分后, 冲突对象类的集合. 用 $|\mathrm{ConSet}(P)|$ 表示 $\mathrm{ConSet}(P)$ 中冲突对象的数目.

性质 3 决策表 $S=(U,C\bigcup D,V,f)$ 中, 对于 $\forall x_i\in Z_s(1\leqslant s\leqslant k)$, $\exists x_j(i\neq j)\in Z_s$, 有

$$f(x_i,P)=f(x_j,P)\bigwedge f(x_i,D)\neq f(x_j,D).$$

由定义 3 可以得证, 此略.

定义 4 [10] 决策表 $S=(U,C\bigcup D,V,f)$ 中, 核属性集 $G\mathrm{Core}(C)$ 满足以下条件:

$$G\mathrm{Core}(C)=\begin{cases}\{a|a\in C\},\ |\mathrm{ConSet}(C-\{a\})|>|\mathrm{ConSet}(C)|;\\ \phi,\ \text{otherwise.}\end{cases}$$

定理 3 [10] 对于决策表 $S=(U,C\bigcup D,V,f)$, 有 $\mathrm{Core}(C)=G\mathrm{Core}(C)$.

定义 5 决策表 $S'_1=(U',C\bigcup D,V',f')$ 中, 核属性集 $\mathrm{Core}'(C)$ 满足以下条件:

$$\mathrm{Core}'(C)=\begin{cases}\{a|a\in C\},\ |\mathrm{ConSet}(C-\{a\})|>0;\\ \phi,\ \text{otherwise.}\end{cases}$$

定理 4 对于决策表 $S=(U,C\bigcup D,V,f)$, 有 $\mathrm{Core}(C)=\mathrm{Core}'(C)$.

证明 由于是 S'_1 相容决策表, 由性质 2 和定义 3 可知 $\mathrm{ConSet}(C)=\phi$. 再根据定义 4 和定理 3 可得证. □

定理 4 表明, 若删除某个属性 a 后出现了冲突, 则 a 为核属性; 否则, a 不是核属性. 该方法避免了采用可分辨矩阵所需的大量空间和时间开销.

为了便于进行增量式求核, 对每个核属性 a 添加一个计数器 count, 用于标记 $\mathrm{ConSet}(C-\{a\})$ 中冲突对的数目. 在增量式运算中, 如果 count 一旦为 0, 则该核属性将从核属性集中删除.

改进的核属性集定义如下:

性质 4 决策表 $S'_1=(U',C\bigcup D,V',f')$, 对于 $\forall a\in C$, $a\in \mathrm{Core}'(C)$, 设有 $\mathrm{ConSet}(C-\{a\})=\{Z_1,Z_2,\cdots,Z_t\}$, 核属性集表示为 $N\mathrm{Core}'(C)=\{(a,\mathrm{count})\}$, 其中 $\mathrm{count}\neq 0$ 且满足

$$\mathrm{count}=\sum_{i=1}^{t}\sum_{j=1}^{|z_i|-1}j,$$

$|Z_i|$ 表示冲突类 Z_i 中对象的个数.

在定理 4 和性质 4 基础上, 下面给出改进的基于冲突域求核算法.

算法 1 改进的基于冲突域求核算法.

输入: 决策表 $S'_1=(U',C\bigcup D,V',f')$;

输出: 决策表的核 $\mathrm{Core}(C)$.

Step 1: $N\mathrm{Core}'(C)=\mathrm{Core}(C)=\phi$, $\mathrm{ConSet}(C)=\phi$.

Step 2: for $i=1$ to $|C|$ Do

Step 2.1: {对属性 $C-\{a_i\}$ 采用分布计数的基数排序方法[10], 排序决策表 S, 得 s 和 $E=\{E_1,E_2,\cdots,E_s\}$;

Step 2.2: $\mathrm{ConSet}(C-\{a_i\})=\phi,\mathrm{count}=0$;

Step 2.3: for $j=1$ to s Do

if (E_j 中存在冲突) then { $\mathrm{ConSet}(C-\{a_i\})=\mathrm{ConSet}(C-\{a_i\})\bigcup E_j$; $\mathrm{count}=\mathrm{count}+|E_j|*$

$(|E_j|-1)/2;\}$

Step 2.4: if$(|\mathrm{ConSet}(C-\{a_i\})|>0)$ then $N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+\{(a_i,\mathrm{count})\}$;

$\}$ // end_Step 2.

Step 3: $\mathrm{Core}(C)=N\mathrm{Core}'(C)$.

Step 4: 输出 $\mathrm{Core}(C)$.

算法 1 中: Step 2.1 的时间复杂度为 $O(|C||U'|)$, Step 2.3 的时间复杂度为 $O(|C||U'|)$; Step 2 循环体总的时间复杂度为 $O(|C||U'|)+O(|C||U'|)=O(|C||U'|)$, 而 Step 2 循环次数为 $|C|$, 则 Step 2 的时间复杂度为 $O(|C|^2|U'|)$. 因此算法 1 的时间复杂度为 $O(|C|^2|U'|)$, 空间复杂度为 $O(|U'|)$.

4　增量式求核算法

4.1　增量式求核理论

为了便于增量式求核, 这里按决策属性值对决策表 S_1' 进行划分, 可以分成 $|U_1'/D|+1$ 个部分 (其中 $|U_1'/D|$ 表示 U_1'/D 中等价类的数目), 即 $\{Y_1,Y_2,\cdots,Y_{|U_1'/D|},Y_\xi\}$, 共 $|U_1'/D|+1$ 个类.

当动态添加对象 x 后, 将 x 与 S_1' 中的对象进行比较, 存在 5 种情况:

1) x 与 U_1' 中的某个对象相同;

2) x 与 U_2' 中的对象冲突;

3) x 为一个已有类, 与 U' 中对象不存在冲突;

4) x 为一个新类, 与 U' 中对象不存在冲突;

5) x 与 U_1' 中的某个对象冲突.

定理 5　对于情况 1) 和 2), 核属性集保持不变.

证明　x 为情况 1) 或 2) 时, 根据定义 1 和定义 2, 对添加 x 后的决策表 S_1' 相容化和简化, 设获得决策为表 S_2', 则 S_2' 与 S_1' 是一样的. 因此, 情况 1) 和 2) 核属性集保持不变. □

定理 6　对于情况 3), 核属性集 $N\mathrm{Core}'(C)$ 满足 $\exists y\in U',\exists a\in C, f'(x,a)\neq f'(y,a)\bigwedge f'(x,C-\{a\})=f'(y,C-\{a\})\bigwedge f'(x,D)\neq f'(y,D)$, 则 $N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+(a,1)$.

证明　情况 3)中, x 属于已有的类 (设为 Y_k), 且 x 与原 U' 中的对象没有冲突, 因此在这种情况下, 新核只可能在 x 与 $U'-Y_k$ 中的对象比较时产生. 根据条件 $\exists y\in U', f'(x,C-\{a\})=f'(y,C-\{a\})\bigwedge f'(x,D)\neq f'(y,D)$, 有 $y\notin \mathrm{POS}'_{C-\{a\}}(D)$; 由于 S_1' 是相容决策表, 则 $y\in \mathrm{POS}'_C(D)$, 有 $\mathrm{POS}'_{C-\{a\}}(D)\neq \mathrm{POS}'_C(D)$, 故 $a\in N\mathrm{Core}'(C)$. 因此

$$N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+(a,1).$$ □

定理 7　对于情况 4), 核属性集 $N\mathrm{Core}'(C)$ 满足 $\exists y\in U',\exists a\in C, f'(x,a)\neq f'(y,a)\bigwedge f'(x,C-\{a\})=f'(y,C-\{a\})$, 则 $N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+(a,1)$.

证明　情况 4) 中, x 属于一个新类, 故 $f'(x,D)\neq f'(y,D)$, 且 x 与原 U' 中的对象没有冲突, 因此在这种情况下, 新核只可能在 x 与 U' 中的对象比较时产生. 根据条件 $\exists y\in U', f'(x,C-\{a\})=f'(y,C-\{a\})\bigwedge f'(x,D)\neq f'(y,D)$, 有 $y\notin \mathrm{POS}'_{C-\{a\}}(D)$; 由于 S_1' 是相容决策表, 则 $y\in \mathrm{POS}'_C(D)$, 有 $\mathrm{POS}'_{C-\{a\}}(D)\neq \mathrm{POS}'_C(D)$, 故 $a\in N\mathrm{Core}'(C)$. 因此

$$N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+(a,1).$$ □

定理 8　对于情况 5), 设 x 属于类 k, $1\leqslant k\leqslant |U_1/D|$, 即 Y_k, 核属性集 $N\mathrm{Core}'(C)$ 满足以下 2 个条件:

1) 如果 $\exists y\in Y_k,\exists a\in C, f'(x,a)\neq f'(y,a)\bigwedge f'(x,C-\{a\})=f'(y,C-\{a\})$, 则

$$N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+(a,1);$$

2) 如果 $\exists y\in Y_\xi,\exists a\in C, f'(x,a)\neq f'(y,a)\bigwedge f'(x,C-\{a\})=f'(y,C-\{a\})$, 则

$$N\mathrm{Core}'(C)=N\mathrm{Core}'(C)-(a,1).$$

证明　对于情况 5), 设 $\exists z\in U_1'$, z 与 x 有冲突, 根据定义 1 和定义 2, x 在添加到决策表 S_1' 后, z 与 x 将归为异类且仅保留一个. 不妨设保留的是 x, 这时存在 2 种情况:

1) x 加入异类后, 可能会与 Y_k 中的对象产生核, 条件 1) 即是针对这种情况的. 如果 $\exists y\in Y_k$, $f'(x,D)\neq f'(y,D)\bigwedge f'(x,C-\{a\})=f'(y,C-\{a\})$, 有 $y\notin \mathrm{POS}'_{C-\{a\}}(D)$; 因 S_1' 是相容决策表, 有 $y\in \mathrm{POS}'_C(D)$, 故 $a\in N\mathrm{Core}'(C)$. 因此, $N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+(a,1)$.

2) x 加入异类后, 需将 x 与原来 Y_ξ 产生的核属性剔除, 那么首先需要将这样的核属性找出来, 条件 2) 就是针对这种情况的. 如果 $\exists y\in Y_\xi$, $f'(x,C-\{a\})=f'(y,C-\{a\})\bigwedge f'(x,D)\neq f'(y,D)$, 有

$$y\notin \mathrm{POS}'_{C-\{a\}}(D);$$

因为 $x\in U_1'$, 有 $x\in \mathrm{POS}'_C(D)$, 则

$$\mathrm{POS}'_{C-\{a\}}(D)\neq \mathrm{POS}'_C(D),$$

故 a 属性为这样的核属性, 需要将其从 $N\mathrm{Core}'(C)$ 中剔除, 即 $N\mathrm{Core}'(C)=N\mathrm{Core}'(C)-(a,1)$.

综合 1) 和 2), 定理 8 得证. □

4.2　增量式求核算法

根据上述分析, 为了实现增量求核, 须先扫描整个决策表 S_1', 判断 x 属于 5 种情况中的哪一种, 再根据不同情况进行相应处理, 这样至少需要对 S_1' 进行 2 次操作, 从而影响了核更新效率. 为此, 本文给出仅对 S_1' 进行一次扫描便完成核增量更新的方法.

由于 S_1' 按决策属性值分成 $|U_1'/D|+1$ 个类, 算

法的思路是: 根据增量对象的所属类, 对于已有的决策表中不同决策类给予合理的处理顺序, 并针对不同类执行不同运算, 当一次遍历结束后便完成了核属性集增量更新. 具体实现如下.

算法 2 增量式求核算法.

输入: 决策表 $S_1'=(U',C\bigcup D,V',f')$, S_1' 对应的核属性集 $\mathrm{Core}(C)=\{(a,\mathrm{count})\}$, 新增对象 x;

输出: 新的核属性集 $\mathrm{Core}(C)$.

Step 1: $N\mathrm{Core}'(C)=\mathrm{Core}(C)$, flag$=tag=0$;

Step 2: if $(f'(x,D)==V_k)$

Step 2.1: {for $i=1$ to $|U_1'-Y_k|$ Do

if (x_i 与 x 冲突) then{将该 x_i 标识为 y, flag$=1$, $N\mathrm{Core}'(C)=\mathrm{Core}(C)$, break;}

else $\forall a_j\in C$, if (Condition)then$\{N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+(a_j,1),\mathrm{tag}=3;\}$

Step 2.2: for $i=1$ to $|Y_k|$ Do

{ if (x_i 与 x 相同) then $\{N\mathrm{Core}'(C)=\mathrm{Core}(C),\ \mathrm{tag}=1$, goto Step 5; }

if (flag) $\forall a_j\in C$, if (Condition) then $\{N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+(a_j,1),\ \mathrm{tag}=5;\}$

} // end_Step 2.2

Step 2.3: for $i=1$ to $|Y_\xi|$ Do

{ if (x_i 与 x 冲突) then $\{N\mathrm{Core}'(C)=\mathrm{Core}(C),\ \mathrm{tag}=2$, goto Step 5; }

if (flag) $\forall a_j\in C$, if (Condition) then $\{N\mathrm{Core}'(C)=N\mathrm{Core}'(C)-(a_j,1),\ \mathrm{tag}=5;\}$

else $\forall a_j\in C$, if (Condition) then $\{N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+(a_j,1),\ \mathrm{tag}=3;\}$

} // end_Step 2.3

} // end_Step 2

Step 3: else

Step 3.1: { for $k=1$ to $|U_1'/D|$

{ $T\mathrm{Core}'(C)=\phi$;//增量核临时存储空间

for $i=1$ to $|Y_k|$ Do {

if (x_i 与 x 冲突) then {将该 x_i 标识为 y, flag $=2$, $N\mathrm{Core}'(C)=\mathrm{Core}(C)$;}

$\forall a_j\in C$, if(Condition) then $\{T\mathrm{Core}(C)=T\mathrm{Core}(C)+(a_j,1),\ \mathrm{tag}=4;\}$

} // end_for_i

$N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+T\mathrm{Core}(C)$;

if (flag) { tag $=5$,break;}

}// end_Step 3.1

Step 3.2: for $i=1$ to $|Y_\xi|$ Do

{ if (x_i 与 x 冲突) then { $N\mathrm{Core}'(C)=\mathrm{Core}(C),\ \mathrm{tag}=2$, goto Step 5; }

if (flag) $\forall a_j\in C$, if (Condition) then $\{N\mathrm{Core}'(C)=N\mathrm{Core}'(C)-(a_j,1),\ \mathrm{tag}=5;\}$

else $\forall a_j\in C$, if (Condition) then $\{N\mathrm{Core}'(C)=N\mathrm{Core}'(C)+(a_j,1),\ \mathrm{tag}=4;\}$

} // end_Step 3.2

}// end_Step 3

Step 4: if (tag $==5$) { $Y_k=Y_k-y, Y_\xi=Y_\xi+y$;}

else if (tag$==4$) {构造新类$Y_{\mathrm{new}}=\phi, Y_{\mathrm{new}}=Y_{\mathrm{new}}+x, U'=U'+Y_{\mathrm{new}}$;}

else if (tag $==3$)$Y_k=Y_k+x$;

Step 5: $\mathrm{Core}(C)=N\mathrm{Core}'(C)$.

Step 6: Ouput $\mathrm{Core}(C)$.

其中: Condition 为

$$f'(x,a_j)\neq f'(x_i,a_j)\bigwedge f'(x,C-\{a_j\})=f'(x_i,C-\{a_j\});$$

flag 为冲突标识, tag 为 x 所属情况标识.

算法 2 中: 因为决策表 S_1' 的决策属性值是 $1,2,\cdots,|U_1'/D|$, V_ξ 的整数, 所以判断 x 所属决策类的时间复杂度仅为 $O(1)$; 最坏情况下, 执行 Step 2 和 Step 3 任何一个分支的时间复杂度都为 $O(|C||U'|)$. 故算法 1 的时间复杂度为 $O(|C||U'|)$, 远低于文献 [11] 的 $O(|C|(5|U_1|+3|U_2'|))$. 算法 2 的空间开销仅为核临时存储空间 $T\mathrm{Core}(C)$ 和核属性集 $N\mathrm{Core}'(C)$ 的开销, 因此空间复杂度为 $O(|C|)$, 而文献 [11] 的方法需构建可分辨矩阵, 其空间复杂度为 $O(|C|(|U_1|^2+|U_1||U_2'|))$.

5 实例分析和实验比较

5.1 实例分析

实例 1 表 1 为一决策表 S, $C=\{a,b,c,d\}$ 为条件属性, D 为决策属性, 有 8 个样本对象.

表 1 决策表 S

U	a	b	c	d	D
x_1	1	1	0	1	2
x_2	1	0	1	0	1
x_3	0	1	0	0	1
x_4	1	1	1	0	2
x_5	1	0	1	0	2
x_6	0	1	1	1	1
x_7	0	1	0	0	1
x_8	1	0	1	0	3

分析表 1, 有

$$\mathrm{POS}_C(D)=\{\{x_1\},\{x_3,x_7\},\{x_4\},\{x_6\}\},$$

$$\mathrm{ConSet}(C)=\{\{x_2,x_5,x_8\}\},$$

其中 x_2, x_5 和 x_8 为不相容对象, 表 1 为不相容的决策表. 根据定义 1 和定义 2, 对 S 异类和简化, 并按决策

属性值进行分类, 可生成决策表 S_1' (见表 2). 其中

$$U' = \{x_1, x_4, x_3, x_6, x_2\},$$

$$U_1' = \{x_1, x_4, x_3, x_6\},\ U_2' = \{x_2\},$$

$$Y_1 = \{x_3, x_6\},\ Y_2 = \{x_1, x_4\},\ Y_\xi = \{x_2\}.$$

显然, S_1' 为相容决策表.

表 2　决策表 S_1'

	Y_i	U'	a	b	c	d	D
U_1'	Y_1	x_3	0	0	0	0	1
		x_6	0	1	1	1	1
	Y_2	x_1	1	1	0	1	2
		x_4	1	1	1	0	2
U_2'	Y_ξ	$x2$	1	0	1	0	V_ξ

根据算法 1, 对每个条件属性进行分析, 得

$$\mathrm{ConSet}(C - \{a\}) =$$

$$\mathrm{ConSet}(C - \{c\}) = \mathrm{ConSet}(C - \{d\}) = \phi,$$

$$\mathrm{ConSet}(C - \{b\}) = \{\{x_2, x_4\}\}.$$

因为 $|\mathrm{ConSet}(C - \{b\})| = 2 > 0$, 故

$$\mathrm{Core}(C) = N\mathrm{Core}'(C) = \{(b, 1)\}.$$

为了说明算法 2 的方法, 下面通过 4 个实例分别分析情况 3)~5).

1) 对于 S_1', 若新增对象 x 为(1,1,1,1,1).

判断 x 为一个已存在的类 Y_1, 执行 Step 2 的 if 子语句部分. 先与 $U_1' - Y_1$ 中对象 $\{x_1, x_4\}$ 比较: 与 x_1 比较, 有 $f'(x, c) \neq f'(x_1, c) \bigwedge f'(x, C - \{c\}) = f'(x_1, C - \{c\})$, 所以 $N\mathrm{Core}'(C) = N\mathrm{Core}'(C) + (c, 1) = \{(b, 1), (c, 1)\}$; 与 x_4 比较, 有核 d 产生, 则 $N\mathrm{Core}'(C) = \{(b, 1), (c, 1), (d, 1)\}$. 再与 Y_1 中对象 $\{x_3, x_6\}$ 比较, $N\mathrm{Core}'(C)$ 无变化. 最后与 Y_ξ 中 x_2 比较, 无核产生. 最终

$$\mathrm{Core}(C) = \{(b, 1), (c, 1), (d, 1)\}.$$

2) 对于 S_1', 若新增对象 x 为 (1,1,1,1,4).

判断 x 为一个新类, 执行 Step 2 的 else 子语句部分. 先与 Y_1 中对象 $\{x_3, x_6\}$比较: 与 x_3比较, 无核属性产生; 与 x_6 比较, 有核 a 产生, 则 $N\mathrm{Core}'(C) = \{(a, 1), (b, 1)\}$. 再与 Y_2 中对象 $\{x_1, x_4\}$ 比较: 与 x_1 比较, 有核 c 产生, 则 $N\mathrm{Core}'(C) = \{(a, 1), (b, 1), (c, 1)\}$; 再与 x_4 比较, 有核 d 产生, 则 $N\mathrm{Core}'(C) = \{(a, 1), (b, 1), (c, 1), (d, 1)\}$. 最后与 Y_ξ 中 x_2 比较, 无核属性产生. 最终

$$\mathrm{Core}(C) = \{(a, 1), (b, 1), (c, 1), (d, 1)\}.$$

3) 对于 S_1', 若新增对象 x 为 (1,1,1,0,1).

判断 x 为一个已存在的类 Y_1, 执行 Step 2 的 if 子语句部分. 先与 $U_1' - Y_1$ 中对象 $\{x_1, x_4\}$ 比较, 与 x_1 比较, 无核产生; 与 x_4 比较, 互相冲突, 标记 flag = 1, 撤销之前的 $N\mathrm{Core}'(C)$ 更新, 还原为初始值 $\mathrm{Core}(C)$, 并提前结束与 $U_1' - Y_1$ 中对象的比较. 再与 Y_1 中对象 $\{x_3, x_6\}$ 比较, 均无核产生. 最后与 Y_ξ 中 x_2 比较, 有核属性 $(b, 1)$ 存在, 因为 flag = 1, 所以该核需被剔除, 即 $N\mathrm{Core}'(C) = N\mathrm{Core}'(C) - (b, 1)$, $N\mathrm{Core}'(C) = \phi$. 最终

$$\mathrm{Core}(C) = \phi.$$

4) 对于 S_1', 若新增对象 x 为(1,1,1,0,4).

判断 x 为一个新类, 执行 Step 2 的 else 子语句部分. 先与类 Y_1 中对象 $\{x_3, x_6\}$ 比较, 均无核产生. 再与 Y_2 中对象 $\{x_1, x_4\}$比较: 与 x_1比较, 无核产生; 与 x_4比较, 互相冲突, 标记 flag = 2, 撤销之前的 $N\mathrm{Core}'(C)$ 更新, 还原为初始值 $\mathrm{Core}(C)$, 并记录当前类 Y_2 的 $T\mathrm{Core}'(C)$, 因为 $T\mathrm{Core}'(C)$ 为 ϕ, 故 $N\mathrm{Core}'(C)$ 不变, 提前结束与 U_1' 中元素的比较. 最后与 Y_ξ 中 x_2 比较, 有核属性 $(b, 1)$ 存在, 因为 flag = 2, 故该核需被剔除, 即 $N\mathrm{Core}'(C) = N\mathrm{Core}'(C) - (b, 1)$, $N\mathrm{Core}'(C) = \phi$. 最终

$$\mathrm{Core}(C) = \phi.$$

5.2　实验比较

为了验证本文算法 1 和算法 2 以及文献 [11] 算法 2 的性能, 在 P4 2.8 G, RAM 512 M, VS.NET 2005 平台的 VC++ 环境下编程, 采用 UCI 数据库中的数据集为测试数据, 设计 2 组实验方案 (文献 [11] 的算法 2 以及本文算法 1 和算法 2, 分别记为算法 A, 算法 B 和算法 C; |CN| 表示最终核属性个数).

1) 选取 UCI 数据库中 6 个数据集为测试数据, 每个数据集随机抽取 90%的记录构成基准决策表, 余下 10%作为新添加的数据对象. 采用算法 A, 算法 B 和算法 C 实验, 结果如表 3 所示.

表 3　3 种求核算法的时间比较

数据集	$\|U\|$	$\|C\|$	$\|U'\|$	$\|\mathrm{CN}\|$	算法执行时间 / ms		
					算法 A	算法 B	算法 C
Monks	432	6	432	3	1.50	2.03	0.32
Monkey	556	17	432	2	7.45	19.57	3.16
Car	1 728	6	972	6	4.28	6.25	3.09
Mushroom	8 124	23	8 124	0	465.89	1 136.6	331.68
Nursery	12 960	8	12 960	8	94.67	158.12	31.04
Poker	25 010	10	25 008	5	116.13	482.16	36.65

分析表 3, 算法 A 和算法 C 性能优于算法 B, 这是因为算法 A 和算法 C 采用了增量式求核方法. 另外, 算法 C 性能优于算法 A, 这是因为算法 A 需先遍历一次决策表确定新增加对象的情况, 然后修改可分辨矩阵, 接下来才能执行更新核属性的操作; 而算法 C 不需要创建可分辨矩阵, 并且对决策表仅进行一次遍历便可完成一次增量求核.

2) 选择 UCI 中的 Car, Mushroom 和 Nursery 共 3 个数据集, 以原始决策表中 40%的数据作为基准决

策表, 每次按原始决策表数据量10%的比例增加, 采用非增量式求核算法(算法B)和增量式核属性更新算法(算法C)进行比较实验, 结果如表4所示.

表 4 算法B和算法C增量时间比较 ms

数据集	Car		Mushroom		Nursery	
增量比/%	算法B	算法C	算法B	算法C	算法B	算法C
40	2.01	0	401.57	0	92.33	0
40 + 10	2.52	0.92	495.23	32.65	185.04	10.39
40 + 20	2.99	1.31	607.12	165.67	207.45	43.96
40 + 30	3.51	1.86	726.68	379.43	241.07	83.54
40 + 40	4.11	2.75	815.35	626.02	285.89	145.76
40 + 50	4.77	4.15	922.02	917.14	311.53	220.16
40 + 60	6.35	6.32	1 136.6	1 268.37	358.11	337.52

分析表4可见: 当增量数据为原始数据集的10%～40%时, 增量算法效率的优越性明显; 当增量数据达到原始数据集的50%时, 增量算法的优势有所下降; 当增量数据达到原始数据集的60%时, 增量算法在处理Car和Nursery数据集的时间开销与非增量算法的时间开销相当, 对Mushroom数据集处理时的效率还不如非增量算法, 这是因为Mushroom中条件属性C的数目相对比较大造成的($|C| = 23$). 该组测试结果表明, 增量式求核方法并不是处理任何数量的增量数据都高效, 对于本文算法2而言, 如果增量数据大于基准数据量时, 则不太适合采用增量式方法求核, 采用一般求解方法可能还会快一些. 但在实际问题中, 一次性增加很多数据的情况比较少见.

6 结 论

本文对基于冲突域的求核方法进行了改进, 提出了一种增量式核属性求解算法. 该算法不需要创建可分辨矩阵, 而是根据决策值将决策表分成不相交的决策类, 当有新的对象增加后, 针对不同的决策类执行相应操作, 仅需对决策表进行一次遍历便可完成核属性增量更新, 算法的时间和空间复杂度分别为$O(|C||U'|)$和$O(|C|)$. 理论分析和实验结果均表明, 本文提出的算法是可行、高效的.

参考文献(References)

[1] Pawlak Z. Rough sets[J]. Int J of Computer and Information Science, 1982, 11(5): 341-356.

[2] Hu X H, Cercone N. Learning in relational databases: A rough set approach[J]. Computational Intelligence, 1995, 11(2): 323-337.

[3] Skowron A, Rauszer C. The discernibility matrices and functions in information systems[C]. Intelligent Decision Support-handbook of Applications and Advances of the Rough Sets theory. Dordrecht: Kluwer Academic Publisher, 1991: 331-362.

[4] 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.

[5] 刘少辉, 盛秋戬, 吴斌, 等. Rough集高效算法的研究[J]. 计算机学报, 2003, 26(5): 524-529.
(Liu S H, Sheng Q J, Wu B, et al. Research on efficient algorithms for Rough set methods[J]. Chinese J of Computers, 2003, 26(5): 524-529.)

[6] Wang Jue, Wang Ju. Reduction algorithms based on discernibility matrix: The ordered attributes method[J]. J of Computer Science and Technology, 2001, 16(6): 489-504.

[7] 叶东毅, 陈昭炯. 一个新的差别矩阵及其求核方法[J]. 电子学报, 2002, 30(7): 1086-1088.
(Ye D Y, Chen Z J. A new discernibility matrix and the computation of a core[J]. Acta Electronica Sinica, 2002, 30(7): 1086-1088.)

[8] 王国胤. 决策表核属性的计算方法[J]. 计算机学报, 2003, 26(5): 611-615.
(Wang G Y. Calculation nethods for core attributes of decision table[J]. Chinese J of Computers, 2003, 26(5): 611-615.)

[9] 杨明, 孙志挥. 改进的差别矩阵及其求核方法[J]. 复旦学报: 自然科学版, 2004, 43(5): 865-868.
(Yang M, Sun Z H. Improvement of discernibility matrix and the computation of a core[J]. J of Fudan University: Natural Sciencd , 2004, 43(5): 865-868.)

[10] 葛浩, 李龙澍, 杨传健. 一种核属性快速求解算法[J]. 控制与决策, 2009, 24(5): 738-742.
(Ge H, Li L S, Yang C J. A quick algorithm for computing the core attribute[J]. Control and Decision, 2009, 24(5): 738-742.)

[11] 杨明. 一种基于改进差别矩阵的核增量式更新算法[J]. 计算机学报, 2006, 29(3): 407-413.
(Yang M. An incremental updating algorithm of the computation of a core based on the improved discernibility matrix[J]. Chinese J of Computers, 2006, 29(3): 407-413.)

[12] Wang G Y, Zhao J, An J J, et al. A comparative study of algebra viewpoint and information viewpoint in attribute reduction[J]. Fundamenta Informaticae, 2005, 68(6): 289-301.