文章编号: 1001-0920 (2011) 02-0207-06

基于 Bucket Sort 的快速属性约简算法

蒋 瑜, 刘胤田, 李 超

(成都信息工程学院 软件工程学院，成都 610225)

摘 要: 利用桶排序思想设计了一个求解 U/C 的算法, 其时间复杂度降为 $O(|C||U|)$. 由此, 给出一种无需求解正域便能判断正域是否变化的方法. 基于以上方法, 提出一种快速属性约简算法. 该算法的求解策略是在每次迭代过程中求解决策表相对核, 如果在某次迭代过程中找不到这样的核属性, 则任意排除一个条件属性. 最后通过实验分析了该算法在最坏情况下的时间复杂性, 其复杂性降为 $O(|C|^2|U/C|)$.

关键词: 粗糙集；正区域；属性约简；桶排序

中图分类号: TP18 **文献标识码:** A

Fast algorithm for computing attribute reduction based on Bucket Sort

JIANG Yu, *LIU Yin-tian*, *LI Chao*

(College of Software Engineering，Chengdu University of Information Technology，Chengdu 610225，China. Correspondent：JIANG Yu，E-mail：jiangyu@cuit.edu.cn)

Abstract: An algorithm based on Bucket Sort for computing U/C is proposed, whose complexity is cut down to $O(|C||U|)$. And a method is designed to estimate whether the change of positive region or not, which doesn't compute positive region. A fast attribute reduction algorithm based on U/C is introduced. The reduction strategy of the algorithm is to compute relative core. If in some iteration the algorithm can not find such cores, it will eliminate one condition attribute preparing for finding relative core in the next iteration. The time complexity of the algorithm in the worst case is analyzed and its temporal complexity is $O(|C|^2|U/C|)$.

Key words: rough set；positive region；attribute reduction；Bucket Sort

1 引 言

Rough 集理论是由波兰学者 Pawlak[1]提出的一种处理模糊和不确定性的数学工具, 目前已成功应用于数据挖掘、模式识别、过程控制、信息系统分析、决策支持系统、分类和故障检测等领域[2-3]. 属性约简是 Rough 集理论的核心内容之一. 所谓属性约简, 就是在保持知识库分类能力不变的前提下, 删除不相关或不重要的冗余属性.

在通常情况下, 约简算法并不需要计算出信息系统的所有约简, 仅需获得用户感兴趣或可用的约简. 依据粗糙集理论中获取约简方法的不同, 常用的计算方法有基于正区域法[4-8]和差别矩阵法[9-11].

对于差别矩阵法, 首先计算数据集的差别矩阵和差别函数; 然后通过求差别矩阵中所有项的最小析取范式来获得约简. 该算法的优点在于直观、易于理解、而且能很容易地计算出核与所有约简. 但该算法也存在着不足之处, 即在矩阵中会出现大量的重复元素 (或元素之间存在着包含关系) , 这就大大降低了属性约简算法的效率. 通常此类算法的复杂度为 $O(|C|^2|U|^2)$, 其中: $|C|$ 为属性个数, $|U|$ 为数据记录数.

上述约简方法中, 差别矩阵方法的计算过程需耗费巨大的时间和空间, 故不常采用. 其他方法中都需要频繁计算决策表的正区域, 因而计算正区域的时间复杂度直接决定了约简算法的时间复杂度.

通常采用的正区域计算方法时间复杂度为 $O(|C||U|^2)$, 文献 [7] 中基于快速排序思想给出了一种快速求解正区域算法, 其时间复杂度可降为 $O(|C||U|\log(|U|))$. 在此基础上, 文献 [12-13] 中类似地提出了一种基数排序的计算正区域算法, 其时间复

收稿日期: 2009-11-30；**修回日期:** 2010-02-17.

基金项目: 国家自然科学基金项目(60702075)；成都信息工程学院发展基金项目(KYTZ200811).

作者简介: 蒋瑜(1980−), 男, 讲师, 硕士, 从事粗糙集理论和数据挖掘的研究；刘胤田(1972−), 男, 博士副, 教授, 从事数据挖掘的研究.

杂度下降至$O(|C||U|)$. 但这两种算法的复杂度是在决策表属性值都为一位数据的前提下得到的, 由于基数排序时间复杂度为$O(K|U|)$, 其中K为属性值数据位数, 算法的复杂度应为$O(K|C||U|)$. 在本文的属性约简算法中, 提出了一种基于桶排序的正区域计算方法, 其时间复杂度可被降为$O(|C||U|)$. 基于所求划分$U/C=\{[x_1]_C,[x_2]_C,\cdots,[x_n]_C\}$, 在划分$U/C$中的每个等价类中选取一条决策规则构成新决策表$S'=(U',C,D,V',f')$, 当选取的决策规则为不相容规则时, 令该决策规则的决策属性取值为null. 由于相容决策规则集合的子集一定是相容决策规则集合, 表明在划分$U/P=\{[x_1]_P,[x_2]_P,\cdots,[x_n]_P\}(P\subseteq C)$中, 若$[x_i]_P\subseteq \mathrm{POS}_C(D)$, 则

$$U(P\bigcup\{a\})=\{([x_1]_P/\{a\})\bigcup\cdots\bigcup[x_i]_P\bigcup\cdots\bigcup([x_n]_P/\{a\})\},$$

$$U/(P\bigcup\{a\})=\{([x_1]_P/\{a\})\bigcup\cdots\bigcup([x_i]_P/\{a\})\bigcup\cdots\bigcup([x_n]_P/\{a\})\}$$

对求解$\mathrm{POS}'_{P\bigcup\{a\}}(D)(a\in C-P)$是没有影响的. 基于此, 依据决策表$S'=(U',C,D,V',f')$设计出一个不需求解正域就能判断$\mathrm{POS}'_P(D)$是否等于$\mathrm{POS}'_C(D)$的算法, 从而给出了一个不需要计算正域的快速属性约简算法. 该算法的求解策略是在每次迭代过程中求解决策表相对核, 如果在某次迭代过程中找不到这样的核属性, 则任意排除一条件属性, 为下一次迭代中找到相对核属性作准备, 从而得到决策表的约简, 其求解时间复杂度为$O(|C|^2|U/C|)$, 并通过实验和分析证明本文算法的高效性.

2 基本概念

定义 1[14] 一个决策表信息系统可定义为$S=(U,A,V,f)$. 其中: U为论域, 是对象的集合, $U=\{x_1,x_2,\cdots,x_n\}$; A为属性集, $A=\{a_1,a_2,\cdots,a_m,d\}$, A由两部分组成, $A=C\bigcup D$且$C\bigcap D=\varnothing$, C为条件属性集, D为决策属性集, 一般情况下D中只含有一个属性$D=\{d\}$; V为属性的值域, $V=\{v_{a1},v_{a2},\cdots,v_{am},v_d\}$; f为信息函数, 即$f:U\times A\to V, a\in A, x\in U$, 则有$f(x,a)\in v_a$. 表1为一决策表, 条件属性$C=\{a,b,c,d\}$, 决策属性$D=\{D\}$.

定义 2[14] 对于决策表信息系统$S=(U,A,V,f)$, 令$R\subseteq A$, 则$\mathrm{ind}(R)=\{(x_i,x_j)|f(x_i,b)=f(x_j,b),P_b\in R\}$称为$S$的不可区分关系. 显然, 不可区分关系是一个等价类, 含x的等价类记为$[x]_{\mathrm{ind}}(R)$或$[x]_R$. R在U上导出的划分记为$U/\mathrm{ind}(R)$或U/R.

定义 3[14] 在决策表信息系统$S=(U,C\bigcup D,V,f)$中, $P\subseteq C$, 其中: D的P正区域记为$\mathrm{POS}_P(D)$, 定义为

表 1 决策表

U	a	b	c	d	D
u_1	1	1	1	1	0
u_2	2	2	2	1	1
u_3	1	1	3	1	0
u_4	2	3	2	3	0
u_5	2	2	2	1	1
u_6	3	1	2	1	0
u_7	1	2	3	2	2
u_8	2	3	1	2	3
u_9	3	1	2	1	1

$$\mathrm{POS}_P(D)=\bigcup_{X\in U/D}P\_(X),$$

D的P正区域是U中所有根据$\mathrm{ind}(P)$的信息可划分到D的等价关系中的对象集合.

定义 4[14] 在决策表信息系统$S=(U,C\bigcup D,V,f)$中, $a\in C$. 如果$\mathrm{POS}_C(D)=\mathrm{POS}_{C-\{a\}}(D)$, 则称$a$为$C$中相对$D$不必要的; 否则, 称$a$为$C$中相对$D$必要的. C的所有必要属性的集合称为C相对D的核, 记为$\mathrm{Core}(C)$.

定义 5[14] 在决策表信息系统$S=(U,C\bigcup D,V,f)$中, 若存在$x_i,x_j\in U$, 当$i\neq j$时, 有$f(x_i,C)=f(x_j,C)$且$f(x_i,D)\neq f(x_j,D)$, 则称该系统为不相容决策表信息系统, x_i和x_j称为不相容对象(冲突对象); 否则, 称为相容决策表信息系统.

3 U/C的快速计算方法及其复杂度分析

不可区分关系是Rough集理论中一个核心概念, 不可区分关系计算的复杂度直接影响基于正区域的属性约简的复杂度. 文献[7]利用快速排序的方法给出了一个计算$\mathrm{ind}(R)$的时间复杂度为$O(|R||U|\log|U|)$的算法. 文献[12-13]计算$\mathrm{ind}(R)$是基于基数排序设计的算法, 其时间复杂度为$O(|R||U|)$, 但这两种算法的复杂度是在决策表属性值都为一位数据的前提下得到的. 本文对计算$\mathrm{ind}(R)$的方法进行深入分析后, 利用桶排序的思想, 给出一个计算$\mathrm{ind}(R)$的时间复杂度为$O(|R||U|)$的算法.

算法 1 桶排序bucketSort(U^o,a), 其中: $U^o=\{u_1^o,u_2^o,\cdots,u_m^o\}\subseteq U, a\in C$, 输出$\{a\}$所对应的划分为$U^o/a$.

桶排序算法如下:

bucketSort(U^o,a)

{

1) 计算$f(u_i^o,a)(i=1,2,\cdots,m)$的最大值和最小值, 分别记为max和min;

2) 动态开辟具有$\max-\min+1$个元素的数组B. 数组B中的每个元素都是一个集合, 代表一个等价类. 且令$B[i]\Leftarrow\varnothing(i=0,1,\cdots,\max-\min)$;

3) 如果 $U^o=\varnothing$; 则 return B;

4) 在 U^o 中任选一元素, 令 $B[f(u_i^o,a)-\min] \Leftarrow B[f(u_i^o,a)-\min]\bigcup\{u_i^o\}$, 且 $U^o \Leftarrow U^o-\{u_i^o\}$;

5) Goto 3).

}

算法 2　计算 ind(R) 的所有等价类 get_ind(U, R), 其中: $U=\{u_1,u_2,\cdots,u_n\}$, $R\subseteq C$ 且 $R=\{r_1,r_2,\cdots,r_t\}$; 输出 R 所对应的划分为 U/R.

具体算法如下:

get_ind(U,R)

{

$U''\Leftarrow\{U\}$; //U'' 的每一元素都是一个集合

while($R\neq\varnothing$){

在集合 R 中任选一元素 a_i, 令 $R\Leftarrow R-\{a_i\}$;

构建一个集合 N, 令 $N\Leftarrow\varnothing$;

while($U''\neq\varnothing$){

在集合 U'' 中任选一元素 u_i'', 令 $U''\Leftarrow U''-\{u_i''\}$;

if($|u_i''|=1$) then $N\Leftarrow N\bigcup\{u_i''\}$;

else $N\Leftarrow N\bigcup$bucketSort(u_i'',a_i);

}

$U''\Leftarrow N$;

}

return U''; // U'' 是 R 所对应的划分 U/R

}

由算法 2 的求解步骤可知, 算法 2 由两重循环所构成: 外层循环 R 和内层循环 U''. 根据算法 2 和算法 1 可知, 集合 U'' 中的任意一个元素都是一个集合, 不妨设 $U''=\{u_1'',u_2'',\cdots,u_t''\}$, 且满足下面两个性质:

1) $u_i''\bigcap u_j''=\varnothing,\ i,j=1,2,\cdots,t,\ i\neq j$;

2) $u_1''\bigcup u_2''\bigcup\cdots\bigcup u_t''=U$.

内层循环 U'' 的时间复杂度主要由每趟桶排序 bucketSort(u_i'',a_i)决定, 而根据算法 1 可知, 桶排序 bucketSort(u_i'',a_i)的时间复杂度为 $O(|u_i''|)$, 从而可得内层循环 U'' 的时间复杂度为

$$\sum_{i=1}^{t}O(|u_i''|)=O(|u_1''|)+O(|u_2''|)+\cdots+O(|u_t''|)=$$

$$O(|u_1''|+|u_2''|+\cdots+|u_t''|)=O(|U|).$$

而外层循环 R 最多循环 $|R|$ 次, 因此可知算法 2 的时间复杂度为 $O(|R||U|)$.

4　简化决策表

在决策表 $S=(U,C,D,V,f)$ 中, 过去的属性约简算法在计算 $\mathrm{POS}_P(D)(P\subseteq C)$ 时都是建立在整个对象集 U 上. 经分析, 在划分 $U/C=\{[x_1]_C,[x_2]_C,\cdots,[x_m]_C\}$ 中, 每个等价类 $[x_i]_C$ 存在两种可能: 1) $[x_i]_C\subseteq\mathrm{POS}_C(D)$, 2) $[x_i]_C\nsubseteq\mathrm{POS}_C(D)$. 这表明同一个等价类中的决策规则具有相同的性质, 要么都是相容规则, 要么都是不相容规则. 因此基于整个对象集 U 计算正区域 $\mathrm{POS}_P(D)(P\subseteq C)$ 并不是必要的.

定义 6　在决策表 $S=(U,C,D,V,f)$ 中, 划分 $U/C=\{[x_1]_C,[x_2]_C,\cdots,[x_m]_C\}$, $S'=(U',C,D,V',f')$ 为其简化的决策表. 其中: $U'=\{y_1,y_2,\cdots,y_m\}$, $V'=\{v_{a1},v_{a2},\cdots,v_{am},v_d'\}$, $y_i\in[x_i]_C$. 若 y_i 为不相容决策规则, 则令 $f'(y_i,D)$ 的值为 null.

算法 3　基于定义 6, getNewDesTable(S) 获得决策表 $S=(U,C,D,V,f)$ 的简化决策表 $S'=(U',C,D,V',f')$.

getNewDesTable(S)

{

构建记录集 U', 令 $U'\Leftarrow\varnothing$;

调用算法 2: get_ind(U,C) 求得划分 U/C;

对划分 U/C 中的所有等价类 $[x_i]_C$ 作如下操作:

1) 在 $[x_i]_C$ 中任一决策规则 x_j, 若 $f(x_i,D)=f(x_j,D)$, 则 $U'\Leftarrow U'\bigcup\{x_i\}$;

2) 在 $[x_i]_C$ 中存在至少一条决策规则 x_j, 若 $f(x_i,D)\neq f(x_j,D)$, 则 $U'\Leftarrow U'\bigcup\{x_i\}$, 且 $f'(x_i,D)\Leftarrow$null;

return $S'=(U',C,D,V',f)$;

}

根据算法 3 可求得表 1 所对应的新决策表, 如表 2 所示.

表 2　表 1 的简化决策表

U	a	b	c	d	D
u_1	1	1	1	1	0
u_2	2	2	2	1	1
u_3	1	1	3	1	0
u_4	2	3	2	3	0
u_6	3	1	2	1	null
u_7	1	2	3	2	2
u_8	2	3	1	2	3

定理 1　若 R 是决策表 $S'=(U',C,D,V',f')$ 的一个约简, 则 R 一定是决策表 $S=(U,C,D,V,f)$ 的一个约简.

证明:　令决策表 $S'=(U',C,D,V',f')$ 的正域为 $\mathrm{POS}'_C(D)$, 不妨设 $\mathrm{POS}'_C(D)=y_1,y_2,\cdots,y_j$, 根据定义 6 可知, 决策表 $S=(U,C,D,V,f)$ 的正域

$$\mathrm{POS}_C(D)=[y_1]_C\bigcup[y_2]_C\bigcup\cdots\bigcup[y_j]_C,$$

$$[y_i]_C\in U/C,\ i=1,2,\cdots,j.$$

由已知条件 R 是决策表 $S'=(U',C,D,V',f')$ 的一个约简可得

$$\mathrm{POS}'_R(D)=\mathrm{POS}'_C(D)=y_1,y_2,\cdots,y_j.$$

再根据定义6可知

$$\mathrm{POS}_R(D)=[y_1]_R\bigcup[y_2]_R\bigcup\cdots\bigcup[y_j]_R,$$

$$[y_i]_R\in U/R,\ i=1,2,\cdots,j.$$

由$R\subseteq C$可知$[y_i]_C\subseteq[y_i]_R$, 则有

$$|\mathrm{POS}_R(D)|\geqslant|\mathrm{POS}_C(D)|,$$

而对于决策表S而言, 有

$$|\mathrm{POS}_R(D)|\leqslant|\mathrm{POS}_C(D)|,$$

所以$|\mathrm{POS}_R(D)|$只能等于$|\mathrm{POS}_C(D)|$, 这表明R是决策表$S=(U,C,D,V,f)$的一个约简. □

5 快速属性约简算法及其复杂性分析

在基于正域的属性约简算法中, 过去的约简算法都是在不断计算$\mathrm{POS}_P(D)$是否等于$\mathrm{POS}_C(D)$. 要计算$\mathrm{POS}_P(D)$, 必须首先求得划分U/P. 经深入分析, 本文提出一个不需要计算正域就能判断正域是否变化的快速属性约简算法. 为了更方便地说明本文思想, 给出如下定理.

定理 2 在决策表$S=(U,C,D,V,f)$中, 假设

$$U/P=\{[x_1]_P,[x_2]_P,\cdots,[x_n]_P\}(P\subseteq C);$$

$$U/D=\{[y_1]_D,[y_2]_D,\cdots,[y_m]_D\}.$$

对于$[x_i]_P\in U/P$, 若存在$[y_j]_D\in U/D$, 使得$[x_i]_P\subseteq[y_j]_D$, 则有$[x_i]_P\subseteq\mathrm{POS}_C(D)$.

证明 根据已知条件$[x_i]_P\subseteq[y_j]_D$可知, 任意两个决策规则$x_1,x_2\in[x_i]_P$, 有$f(x_1,C)=f(x_2,C)$且$f(x_1,D)=f(x_2,D)$成立, 即条件属性P下所对应的值相同且决策属性值也相同. 因此x_1,x_2为相容决策规则, $x_1,x_2\in\mathrm{POS}_C(D)$, 即$[x_i]_P\subseteq\mathrm{POS}_C(D)$. □

定理 3 设$U_2\subseteq U_1$, 如果$U_1\subseteq\mathrm{POS}_C(D)$, 则$U_2\subseteq\mathrm{POS}_C(D)$.

由"$\subseteq$"运算的传递性可证明定理3, 此略.

定理 4 在决策表$S=(U,C,D,V,f)$中, 假设

$$U/P=\{[x_1]_P,[x_2]_P,\cdots,[x_i]_P,\cdots,[x_n]_P\},P\subseteq C;$$

$$U/D=\{[y_1]_D,[y_2]_D,\cdots,[y_m]_D\}.$$

对于$[x_i]_P\in U/P$, 若存在$[y_j]_D\in U/D$, 使得$[x_i]_P\subseteq[y_j]D$, 则有

$$U/(P\bigcup\{a\})=\{([x_1]_P/\{a\})\bigcup([x_2]_P/\{a\})\bigcup\cdots\bigcup([x_i]_P/\{a\})\bigcup\cdots\bigcup([x_n]_P/\{a\})\},$$

$$U/(P\bigcup\{a\})=\{([x_1]_P/\{a\})\bigcup([x_2]_P/\{a\})\bigcup\cdots\bigcup[x_i]_P\bigcup\cdots\bigcup([x_n]_P/\{a\})\}.$$

所求得的$\mathrm{POS}_{P\bigcup\{a\}}(D)$是相等的.

证明 根据已知条件, 在决策表$S=(U,C,D,V,f)$中, 假设

$$U/P=\{[x_1]_P,[x_2]_P,\cdots,[x_i]_P,\cdots,[x_n]_P\},P\subseteq C;$$

$$U/D=\{[y_1]_D,[y_2]_D,\cdots,[y_m]_D\}.$$

对于$[x_i]_P\in U/P$, 若存在$[y_j]_D\in U/D$, 使得$[x_i]_P\subseteq[y_j]_D$, 则根据定理2可知$[x_i]_P\subseteq\mathrm{POS}_C(D)$. 换而言之, 等价类$[x_i]_P$是一个相容等价类, 其所有规则都是相容规则, 若再对等价类$[x_i]_P$细分, 则只能把$[x_i]_P$划分为更小的相容等价类集合, 这样不会影响等价类$[x_i]_P$中规则的相容性变化, 从而不影响$\mathrm{POS}_{P\bigcup\{a\}}(D)$的求解结果. □

定理 5 在决策表$S=(U,C,D,V,f)$中, 假设

$$U/P=\{[x_1]_P,[x_2]_P,\cdots,[x_i]_P,\cdots,[x_n]_P\},P\subseteq C.$$

对于$[x_i]_P\in U/P$, 若任意$x_j\in[x_i]_P,x_j\notin\mathrm{POS}_C(D)$, 则从划分$U/P$中去掉等价类$[x_i]_P$不影响$\mathrm{POS}_C(D)$的求解结果.

证明 根据已知条件$[x_i]_P\in U/P$, 若任意$x_j\in[x_i]_P,x_j\notin\mathrm{POS}_C(D)$, 则等价类$[x_i]_P$中所包含的规则都是非相容规则, 并且在再对等价类$[x_i]_P$细分过程中, 只能把$[x_i]_P$划分为更小的等价类集合, 不会影响等价类$[x_i]_P$外其他规则的相容性变化, 从而不影响正域的变化. □

根据定理4和定理5可知, 利用属性集$P(P\subseteq C)$对对象集U进行划分时, 若在划分求解过程中, 存在$B\subseteq P$, 使得划分U/B包含定理4和定理5所描述的等价类, 则抛去划分U/B中这样的等价类, 不会影响其他规则的相容性变化.

定理 6 在对划分$U/P(P\subseteq C)$的求解过程中, 若在划分的求解过程中, 从划分中不断抛去定理4和定理5所描述的等价类, 若最后所得划分U/P为空, 则$\mathrm{POS}_P(D)=\mathrm{POS}_C(D)$.

证明 设对象集U_1和U_2分别是在求解划分$U/P(P\subseteq C)$过程中抛去定理4和定理5所描述等价类集合的并集. 由定理4和定理5可知, 对象U_1中所有规则为相容规则, 即$U_1\subseteq\mathrm{POS}_C(D)$, 对象$U_2$中所有规则为非相容规则, 即在$U_2$中不存在一规则属于$\mathrm{POS}_C(D)$, 则$U_1\bigcap U_2=\varnothing$. 又因为最后所得划分$U/P$为空, 则$U_1\bigcup U_2=U$, 即$U_1$是所有相容规则的集合, 从而$U_1=\mathrm{POS}_C(D)$, 即$\mathrm{POS}_P(D)=\mathrm{POS}_C(D)$. □

性质 1 在决策表$S=(U,C,D,V,f)$中, 假设

$$U/P=\{[x_1]_P,[x_2]_P,\cdots,[x_i]_P,\cdots,[x_n]_P\},P\subseteq C;$$

$$e|[x_i]_P|=1.$$

则抛去等价类$[x_i]_P$并不影响正域的变化.

根据定理4~定理6可证明性质1, 此略.

基于定理6, 依据决策表$S'=(U',C,D,V',f')$设计出一种不需求解正域就能判断$\mathrm{POS}'_P(D)$是否等于$\mathrm{POS}'_C(D)$的算法.

算法 4　在 $S'=(U',C,D,V',f')$ 中, 利用 isPOS(U',A) 判断属性集 A 是否破坏了 $\mathrm{POS}'_C(D)$, 其中 $A\subseteq C$. 返回 true 表示 $\mathrm{POS}'_A(D)=\mathrm{POS}'_C(D)$.

具体算法如下:

Bool isPOS(U',A)

{

$U''\Leftarrow\{U'\}$; // U'' 的每一元素都是一个集合

while($A\neq\varnothing$){

在集合 A 中顺序选择一元素 a_i, 令 $A\Leftarrow A-\{a_i\}$;

构建一个集合 N, 令 $N\Leftarrow\varnothing$;

while($U''\neq\varnothing$){

在集合 U'' 中任选一元素 u''_i,

令 $U''\Leftarrow U''-\{u''_i\}$;

若存在 $x_m,x_n\in u''_i$,

$f'(x_m,D)\neq f'(x_n,D)$;

则 $N\Leftarrow N\bigcup\mathrm{bucketSort}(u''_i,a_i)$;

}

if($N=\varnothing$) return true; else $U''\Leftarrow N$;

}

根据定理 4 和定理 5 抛弃 U'' 中的等价类;

if($U''=\varnothing$) return true; else return false;

}

由算法 4 可知, 在 isPOS(U',A) 判断过程中, 不断丢弃不影响 $\mathrm{POS}'_C(D)$ 的等价类, 这些等价类分别是: 1) 只有一个元素的等价类; 2) 相容等价类; 3) 决策属性 D 下取值都为 null 的不相容等价类.

图 1 给出了基于表 2 和 isPOS(U',A) 的求解过程, 划线部分是在求解过程中不断抛弃的等价类, 最后可得 U'' 为空, 算法返回 true, 即 $\mathrm{POS}_A(D)=\mathrm{POS}_C(D)$.

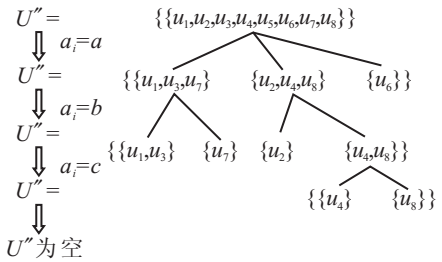

图 1　isPOS(U',A) 的求解过程

算法 5　快速属性约简算法 (FARA) 如下:

输入: 决策表 $S'=(U',C,D,V',f')$;

输出: 决策表 S 的一个约简 R;

Step 1: 构建空约简集合 R;

Step 2: 构建集合 P, 令 $P\Leftarrow C$;

Step 3: 如果 P 为空, 则结束算法;

Step 4: 从条件属性集 P 中任取一个属性 a, 且 $P\Leftarrow P-\{a\}$;

Step 5: 如果 isPOS$(U',R\bigcup P)$ 为假, 则 $R\Leftarrow R\bigcup\{a\}$;

Step 6: goto 3);

Step 7: 算法结束, 输出 R.

由算法 5 的求解步骤可知, FARA 由一层循环构成, 且该循环的循环次数最多不超过 $|C|$. 而该循环的时间复杂度主要由每趟判断正域是否变化的 isPOS$(U',R\bigcup P)$ 决定. 由算法 4 的求解步骤可知, 算法 4 由两重循环构成: 外层循环 A 和内层循环 U''. 由算法 4 可知, 集合 U'' 中的任意一元素都是一个集合, 不妨设 $U''=\{u''_1,u''_2,\cdots,u''_t\}$, 且满足下面两个性质:

1) $u''_i\bigcap u''_j=\varnothing,\ i,j=1,2,\cdots,t,\ i\neq j$;

2) $u''_1\bigcup u''_2\bigcup\cdots\bigcup u''_t\subseteq U'$.

内层循环 U'' 的时间复杂度主要由每趟桶排序 bucketSort(u''_j,a_i) 来决定, 而根据算法 1 可知, 桶排序 bucketSort(u''_j,a_i) 的时间复杂度为 $O(|u''_j|)$, 从而可得内层循环 U'' 的时间复杂度为

$$\sum_{i=1}^{t}O(|u''_i|)=O(|u''_1|)+O(|u''_2|)+\cdots+O(|u''_t|)=$$

$$O(|u''_1|+|u''_2|+\cdots+|u''_t|)\leqslant O(|U|).$$

而外层循环 A 最多循环 $|A|$ 次. 由此可知, 算法 4 的时间复杂度为 $O(|A||U'|)$. 从而算法 5 的时间复杂度为 $O(|C||A||U'|)$. 根据算法 3 可得 $|U'|=|U/C|$, 再由 $A\subseteq C$, 可得算法 5 在最坏情况下的复杂度为 $O(|C|^2|U/C|)$.

6　实　验

选用 UCI 机器学习数据库中的几个数据库在 Pentium processor 1.73GHz(768MB , Windows XP) 上进行实验, ALG1 表示基于 qsort (VC 的一个库函数) 快速排序的分类算法, 用 ALG2 表示赵军[6]的 EABKF 算法, ALG3 表示刘少秋[7]的算法, ALG4 表示葛浩[13]的算法, ALG5 表示徐章艳[12]所提出的算法. 实验结果如表 3 和图 2 所示.

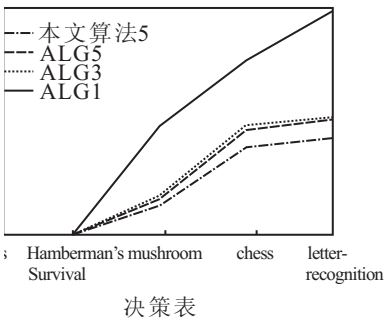

图 2　4 种约简算法运行时间对比

表 3 几种求解U/C算法时间复杂度对比

决策表	条件属性数	记录数	U/C中等价类数	算法执行时间/s			
				AGL1	AGL2	AGL4	本文算法 2
lenses	4	24	24	0.000	0.000	0.000	0.000
Haberman's Survival	3	306	283	0.015	0.015	0.015	0.000
chess	36	3196	2807	0.790	0.528	0.525	0.468
mushroom	22	8 124	5 440	1.518	0.989	0.929	0.821
letter-recognition	15	20 000	18 593	1.871	1.060	1.031	0.906

7 结 论

在基于正区域的属性约简算法中, 计算划分U/C和判断$\mathrm{POS}_P(D)$是否等于$\mathrm{POS}_C(D)$是致关重要的, 其求解速度直接影响到属性约简算法的效率. 本文采用桶排序的思想设计了一种求U/C的算法, 其复杂度被降为$O(|C||U|)$. 基于所求划分U/C, 以快速缩小搜索空间为目的, 本文设计了一种不需求解正域就能判断正域是否变化的方法. 该方法在求解过程中不断抛弃不影响正域变化的对象, 这样搜索空间U会以更快的速度减小, 从而提高算法的效率, 该算法的时间复杂度为$O(|C|^2|U/C|))$.

参考文献(References)

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

[2] Slowinski R. Intelligent decision support-handbook of applications and advances of the rough sets theory[M]. London: Kluwer Academic Publishers, 1992.

[3] 唐彬, 李龙澍, 李伟, 等. 一类对Jenolek属性约简算法的新的改进方法[J]. 系统仿真学报, 2005, 17(5): 1087-1091.
(Tang B, Li L S, Li W, et al. New improvement to Jenolek's attribute reduction algorithm[J]. J of System Simulation, 2005, 17(5): 1087-1091.)

[4] Hu X H, Nick C. Learning in relational databases: A rough set approach[J]. Int J of Computational Intelligence, 1995, 11(2): 323-338.

[5] 叶东毅. 一个改进的Jelonek的属性约简算法[J]. 电子学报, 2000, 28(12): 81-82.
(Ye D Y. An improvement to Jelonek's attribution reduction algorithm[J]. Acta Electronica Sinica, 2000, 28(12): 81-82)

[6] 赵军, 王国胤, 吴中福, 等. 一种高效的属性核计算方法[J]. 小型微型计算机系统, 2003, 24(11): 1950-1953.
(Zhao J, Wang G Y, Wu Z F, et al. An efficient approach to computer feature core[J]. Mini-Micro Systems, 2003, 24(11): 1590-1593.)

[7] 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 algorithm for rough set method[J]. Chinese J of Computers, 2003, 26(5): 524-529.)

[8] 杜金莲, 迟忠先, 翟巍. 基于属性重要性的逐步约简算法[J].小型微型计算机系统, 2003, 24(6): 976-978.
(Du J L, Chi Z X, Zhai W. An improved algorithm for reduction of knowledge based on significance of attribution[J]. Mini-Micro System, 2003, 24(6): 976-978.)

[9] Skowron A, Rauszer C. The discernibility matrices and functions in information system[C]. Intelligent Decision Support Handbook of Applications and Advances of the Rough Sets Theory. Dordrecht: Kluwer Academic Publishers, 1992: 331-362.

[10] 叶东毅, 陈昭炯. 一个新的二进制可辨识矩阵及其核的计算[J].小型微型计算机系统, 2004, 25(6): 965-967.
(Ye D Y, Chen Z J. New binary discernibility matrix and the computation of a core[J]. Mini-Micro Systems, 2004, 25(6): 965-967.)

[11] 刘文军, 谷云东, 冯艳宾, 等. 基于可辨别矩阵和逻辑运算的属性约简算法[J]. 模式识别与人工智能, 2004, 17(1): 119-123.
(Liu W J, Gu Y D, Feng Y B, et al. An improved attribute reduction algorithm of decision table[J]. Pattern Recognition and Artificial Intelligence, 2004, 17(1): 119-123.)

[12] 徐章艳, 刘作鹏, 杨炳儒, 等. 一个复杂度为$\max(O(|C||U|), O(|C|^2|U/C|))$的快速属性约简算法[J]. 计算机学报, 2006, 29(3): 391-399.
(Xu Z Y, Liu Z P, Yang B R, et al. A quick attribute reduction algorithm with complexity of $\max(O(|C||U|), O(|C|^2|U/C|))$[J]. Chinese J of Computers, 2006, 29(3): 391-399.)

[13] 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. Quick algorithm for computing core attribute[J]. Control and Decision, 2009, 24(5): 738-742.)

[14] 张文修, 吴伟志, 梁吉业, 等. 粗糙集理论与方法[M]. 北京: 科学出版社, 2001.
(Zhang W X, Wu W Z, Liang J Y, et al. Theory and method of rough set[M]. Beijing: Science Press, 2001.)