

DOI:10.13232/j.cnki.jnju.2020.04.005

属性组序下基于代价敏感的约简方法

刘 鑫, 胡 军*, 张清华

(计算智能重庆市重点实验室, 重庆邮电大学, 重庆, 400065)

摘 要: 属性约简是粗糙集理论中的重要问题. 为了满足用户对属性的偏好, 人们研究了属性序下的属性约简, 然而对一些问题却很难给出完整的属性序. 针对该问题, 比较分析了属性组序下的约简子集的优劣, 并提出代价敏感下的属性组序约简的算法. 该算法通过属性组序的特点考虑用户偏好并结合属性代价以及属性重要度加权的方式选择局部属性, 可以得到更符合用户偏好的约简. 理论分析和实验结果验证了该算法的可行性和有效性, 并且能在一般情形下找到满足用户偏好的约简.

关键词: 属性约简, 用户偏好, 属性组序, 代价敏感

中图分类号: TP18

文献标识码: A

Attribute reduction based on cost sensitive under attribute group order

Liu Xin, Hu Jun*, Zhang Qinghua

(Chongqing Key Laboratory of Computational Intelligence, Chongqing University of Posts and Telecommunications,
Chongqing, 400065, China)

Abstract: Attribute reduction is an important issue in rough set theory. In order to satisfy users' preferences for attributes, people have studied attribute reduction under attribute order. In some problems, however, it is difficult to give a complete attribute order. Aiming at this problem, this paper compares and analyzes the pros and cons of reduction subsets under attribute group order, and an algorithm based on cost sensitive under attribute group order is proposed. This algorithm considers user preferences based on characteristics of the attribute group order, and selects local attributes by combining the attribute cost and attribute importance weight, which can obtain a reduction more in line with users' preferences. The theoretical analysis and experimental results verify the feasibility and effectiveness of the proposed algorithm, and we can find the reduction that satisfies the users' preferences in the general case.

Key words: attribute reduction, users' preference, attribute group order, cost sensitive

粗糙集理论(Rough Set Theory)是 Pawlak^[1]教授在 1982 年提出的一种用于处理不精确、不一致、不完整信息与知识的数学工具, 该理论已被广泛应用于机器学习、数据挖掘、模式识别、物联网、信息安全等领域. 粗糙集的核心思想是用可定义的集合来描述一个不确定概念, 从而产生上近似集和下近似集的定义. 其上近似集中的对象有可

能属于某一概念集中, 下近似集中描述的对象则可以确定归入到某一概念集中^[2]. 在经典的 Pawlak 粗糙集模型中采用等价关系, 因其要求过于严格, 所以学者们提出了各种扩展的粗糙集模型以适应不同的应用领域^[3-6].

属性约简作为粗糙集理论研究的核心内容之一, 其目标是删除属性集合中冗余的、不相关的和

基金项目: 国家自然科学基金(61876201, 61876027), 重庆市人工智能创新重大主题专项(cstc2017rgzn-zdyfX0040)

收稿日期: 2020-06-20

* 通讯联系人, E-mail: hujun@cqupt.edu.cn

不必要的属性,从而减少属性数量,但与原始属性集合相比,仍然具有对信息系统分类和决策的同等的能力^[7].近几十年,学者们在这个领域取得了大量的成果.这些研究大致可分为以下三类:基于正区域的属性约简^[8]、基于可分辨矩阵的属性约简^[9]和基于信息观的属性约简^[10-11].然而,这些研究没有考虑属性外部的信息以及属性自身所含有的语义信息.

为了得到更合理、更符合用户需求的约简,一些以用户偏好为出发点,结合属性的外部信息,考虑属性代价的算法相继被提出^[12-15],还有一些研究结合用户兴趣以及属性的语义信息,通过属性序的方式对用户偏好进行描述. Wang and Wang^[16]首次提出属性序的概念并设计了一种基于 Skowron 分辨矩阵的属性约简方法,解决了 Pawlak 约简算法完备性和给定属性序下约简的唯一性问题. Zhao and Wang^[17]利用属性序关系代表用户偏好,设计了一种包括属性顺序在前的属性约简方法以满足用户需求,同时提出三种属性序关系:全序、组序以及平衡序. Yao et al^[18-19]提出一种考虑用户对属性偏好的机器学习泛化模型,该模型结合了内部信息和外部信息,可以构建用户优选的属性集. Han and Wang^[20]给出了相邻的属性对基本定理,通过第二属性证明了属性对的决策定理. 官礼和等^[21]提出一种属性序下分层递阶的决策规则挖掘算法,能够获取较高识别率的决策规则. 韩素青和阴桂梅^[22]将约简问题转化为集合覆盖的问题,提出一种面向用户需求的属性约简算法. 胡峰和王国胤^[23]结合分治法的思想,提出一种快速属性约简算法,能够在海量数据中快速得到约简.

现有基于属性序的属性约简算法都需要给出所有属性的全序关系.然而在现实生活中,治疗某一种病有不同的治疗方式,如外科治疗、化学治疗、放射治疗和中医治疗等等,不同治疗方式还包含更具体的不同药物方法,所以不同用户对不同方式的偏好形成了属性组序的关系,而不必考虑每个属性的偏好关系.目前对于属性组序关系下的约简方法研究较少, Zhao and Wang^[17]提出一个简单的方法,利用字母序对组序中的属性排序使其变为全序关系.该方法简单但合理性不足,没有

考虑现实生活中用户的偏好问题以及获取属性所需的代价或费用等问题.为此,本文优先考虑属性组序的关系,其次在每个分组中综合考虑属性重要度和代价的影响,给出了属性组序下基于代价敏感的约简方法,并通过实例和仿真实验进行了分析和验证.

1 基础理论

粗糙集理论是属性约简的核心知识,本节主要回顾粗糙集理论和属性序的基本概念.

定义 1 决策表^[24] $DT=(U, A, V, f)$ 被称为决策表,其中 U 是非空对象集合,也称作论域, $A=C \cup D$ 为属性集合, C 和 D 分别称作条件属性集和决策属性集, $V=\bigcup_{a \in A} V_a$ 为属性取值的集合, $f: U \times A \rightarrow V$ 为信息函数,它指定 U 中每个对象 x 的属性值.

定义 2 等价关系^[24] 决策表 $DT=(U, A, V, f)$, 对于 $\forall B \subseteq A$, 其等价关系为:

$$IND(B)=\{(x, y) \in U^2 \mid b(x)=b(y), \forall b \in B\} \quad (1)$$

$[x]_B=\{y \in U \mid (x, y) \in IND(B)\}$ 表示所有与 x 满足等价关系 $IND(B)$ 的对象的集合,称为等价类.

定义 3 上下近似集^[24] 在决策表 $DT=(U, A, V, f)$ 中, 对于 $\forall X \subseteq U, \forall B \subseteq A$, 可得到 X 相对于 B 的上近似集 $\bar{B}(X)$ 、下近似集 $\underline{B}(X)$:

$$\bar{B}(X)=\{x \in U \mid [x]_B \cap X \neq \emptyset\} \quad (2)$$

$$\underline{B}(X)=\{x \in U \mid [x]_B \subseteq X\} \quad (3)$$

$\bar{B}(X)$ 表示根据 B, U 中与 X 相交不为空的对象集合, $\underline{B}(X)$ 表示根据 B, U 中所有归入 X 的对象集合.

定义 4 正域^[24] $DT=(U, C \cup D, V, f)$ 是一个决策表, $U/D=\{d_1, d_2, \dots, d_m\}$ 是根据 D 得到的 U 的一个划分; 对于 $\forall B \subseteq C, D$ 相对于 B 的正域是 $POS_B(D)$, 其表示为:

$$POS_B(D)=\bigcup_{i=1}^m \underline{B}(d_i) \quad (4)$$

定义 5 Pawlak 约简^[24] 在决策表 $DT=$

$(U, C \cup D, V, f)$ 中,相对于决策属性 D ,属性子集 $R \subseteq C$ 是条件属性集 C 的一个约简,当且仅当:

- (1) $POS_R(D) = POS_C(D)$;
- (2) $\forall a \in R, POS_R(D) \neq POS_{R-\{a\}}(D)$.

其中,条件(1)表明属性子集 R 和属性子集 C 拥有相同的正域,条件(2)指定 R 中任意一个属性 a 都是必要的.

定义6 属性序^[16] 在决策表 DT 中,令 $m = |C|$,在 C 上定义一个完整的序关系“ $<$ ”,将 C 中所有属性分别标上1到 m ,这样在 C 上就得到了一个关于属性的序列,记为 $O: c_1 < c_2 < \dots < c_m$.

算法1 属性序下的属性约简算法^[16]

输入:决策表 DT ,属性序 O .

输出:约简 R .

- ① 令 c_N 是一个约简属性, $R = R \cup \{c_N\}$;
- ② $E^* = \{\alpha: \alpha \in E, \alpha \cap \{c_N\} = \emptyset\}, E = E^*$;
- ③ $N' = \max\{|E/L|\}, N = N'$;
- ④ 重复以上步骤,直到 $E = \emptyset$;
- ⑤ 返回约简 R .

其中, E 是决策表 DT 的分辨矩阵元素集, c_i 为标签属性, L 是属性序在 E 上具有相同标签属性的二元关系.

2 属性组序下基于代价敏感的约简方法

在一些实际问题中,用户可能无法给出所有属性的全序,但可以对属性进行分组;组内的属性间没有序关系,但属性组间有偏序关系,称为属性组序.属性组序的描述方式较好地满足了人们对属性的偏好,以属性组序形式存在的偏好关系也非常常见,又或者部分属性间本身没有可比性.然而,现有的研究还没有针对属性组序的属性约简方法,本节针对属性组序的情形,考虑现实中的代价问题,设计了一种基于代价敏感的约简算法.

定义7 属性组序^[17] 给定决策表 $DT, C = \{a_1, a_2, \dots, a_m\}$ 是 DT 的条件属性集, $m = |C|$, $\{G_1, G_2, \dots, G_n\}$ 是 C 的一个属性划分,其包含 n 个组,并满足 $G_1 \cup G_2 \cup \dots \cup G_n = C, G_i \cap G_j = \emptyset, i, j = 1, \dots, n, i \neq j$,则属性组序记为 S :

$$S = G_n > G_{n-1} > \dots > G_2 > G_1 \quad (5)$$

该属性组序描述了用户对属性的偏好.组间 $G_n > G_{n-1}$ 表示 G_n 组中的属性优于 G_{n-1} 组中的属性,用户对同一组内的属性从语义上具有相同偏好程度.

在信息系统中,对于一个特定的问题,并不是所有的条件属性都是必要的,或者说存在冗余属性.考虑属性客观代价可以作为用户偏好的一个指标,本文结合属性组序的特点,定义了一种度量约简子集所处平均位置的指标,以此来描述该子集是否处于属性组序中更为靠前的位置,称为子集平均位置(SAP).

定义8 子集平均位置 给定属性组序 $S = G_n > G_{n-1} > \dots > G_2 > G_1$ 以及约简集合 $A = \{a_1, a_2, \dots, a_m\}$,集合 A 中每个属性所在属性组序 S 里的位置为该组的下标,记为 $p(a_m)$.则约简集合 A 的平均位置为:

$$SAP(A) = \sum_{i=1}^m p(a_i) / m \quad (6)$$

例如有 $S = G_3 > G_2 > G_1, G_3 = \{a_2\}, G_2 = \{a_1, a_3, a_5\}, G_1 = \{a_4\}$,约简集合 $A = \{a_2, a_3, a_5\}, B = \{a_1, a_4, a_5\}$.则 $SAP(A) = 2.33, SAP(B) = 1.66$,所以用户对约简 A 的偏好大于约简 B .同样容易发现,约简 A 和约简 B 除相同组中的属性外, A 集合中的属性 $a_2 \in G_3$ 是优于 B 集合中的属性 $a_4 \in G_1$.

属性组序描述的是用户的偏好,其作为描述属性的外部信息,可以直观地发现各组间的优劣关系.为了获得偏好程度更高的约简子集,本文采取一种按组删除再添加、最后对约简子集进行验证的方法,其主要分为三步:第一,结合其分组的特点,依次从偏好程度低的分组开始删除,尽可能保留偏好程度更高,分组更为靠前的属性;第二,当每次删除分组正域发生改变时,从当前分组中选择属性,直到正域与原始条件属性集的正域相同;第三,对约简子集中每个属性进行判断是否为必要属性.其流程图如图1所示.

第二步中,如何从局部组中选择合适的属性作为约简结果至关重要.如果按照属性组序依次删除分组是遵循属性语义上的偏好,则在局部分组中选择通过统计和客观评价信息,也就是属性

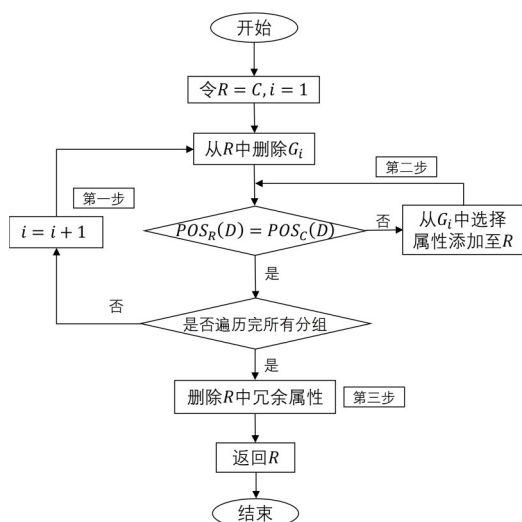


图 1 算法流程图

Fig. 1 The flowchart of our algorithm

重要度的方式来优先选择属性. 首先选择组中核属性, 其次利用加权的方式, 选择重要程度最大的属性加入约简子集中. 属性重要度作为信息表的统计信息, 代价作为现实生活中存在的客观评判标准, 都可以对各分组中的属性进行客观的评价. Zhang and Shen^[25]综合考虑用户需求, 设计了一种属性重要度和代价加权的约简算法 (Attribute Reduction algorithm with Weight, ARWW), 其代价重要度由单个属性代价占据总代价的比例决定, 而对于分组来讲, 同组内属性代价的重要度也会随着候选属性子集增加或减少而动态变化. 下面介绍属性重要度的相关定义.

定义 9 属性依赖度^[8] 给定决策表 DT , $\forall B \subseteq C$, D 相对于 B 的依赖度为:

$$\gamma_B(D) = \frac{|POS_B(D)|}{|U|} \quad (7)$$

其中, $|\cdot|$ 表示集合的基数, 并且 $0 \leq \gamma_B(D) \leq 1$.

定义 10 属性内部重要度^[8] 给定决策表 DT , $B \subseteq C$, $\forall a \in B$, 则属性 a 的内部属性重要度为:

$$Sig_{inner}(a, B, D) = \gamma_B(D) - \gamma_{B - \{a\}}(D) \quad (8)$$

属性 $a \in C$ 为必要属性当且仅当 $Sig_{inner}(a, B, D) > 0$.

定义 11 属性外部重要度^[8] 给定决策表 DT , $B \subseteq C$, $\forall a \in C - B$, 则属性 a 的外部属性重要度为:

$$Sig_{outer}(a, B, D) = \gamma_{B \cup \{a\}}(D) - \gamma_B(D) \quad (9)$$

属性外部重要度表明属性 a 对于集合 B 的分辨能力的提升.

定义 12 代价重要度 给定决策表 DT , 属性组序 $S = G_n > G_{n-1} > \dots > G_2 > G_1$, t_i 为各个属性的代价, T_n 为 G_n 组属性集的总代价, $\forall a \in G_m$, 则属性 a_i 的代价重要度为:

$$Sig_{cost}(a_i) = 1 - \frac{t_i}{\sum_{n > m} T_n + \sum_{n \leq m} t_k} \quad (10)$$

其中, t_k 为 G_n 组中第二步添加的属性, 此时 $n \leq m$.

代价重要度描述的是考虑获取属性所需代价的客观因素对属性产生的影响. 对用户而言, 代价越大其重要程度越低, 因为人们通常会倾向代价更低的约简子集.

结合以上属性重要度的定义, 定义加权重要度为:

$$S(a_i, B) = \lambda Sig_{cost}(a_i) + (1 - \lambda) Sig_{outer}(a_i, B, D)$$

当 $\lambda = 0$ 时, 为仅考虑属性重要度的方法, 当 $\lambda = 1$ 时, 为仅考虑代价重要度的方法. 下面给出属性组序下基于代价敏感的属性约简算法 (ARCSGO).

算法 2 属性组序下基于代价敏感的属性约简方法 (Attribute Reduction Based on Cost Sensitive under Attribute Group Order, ARCSGO)

输入: 决策表 DT , 属性组序 S , λ

输出: 约简后属性集 R

(1) 令 $R = C$;

(2) 依照组序 S , 依次删除 G_i , $1 \leq i \leq n$, 当 $POS_R(D) \neq POS_C(D)$, 执行循环;

① 对于任意属性 $a \in G_i$, 计算 $Sig_{inner}(a, C, D)$, 将 $Sig_{inner}(a', C, D) > 0$ 的属性添加至 R , 并从 G_i 中删除;

② 当 $POS_R(D) \neq POS_C(D)$ 时, 跳转至 (3), 否则退出循环;

③ 对于任意属性 $a \in G_i$, 计算 $S(a_i, R)$, 选择最大 $S(a_i, R)$ 的属性添加至 R , 并从 G_i 中删除, 直到 $POS_R(D) = POS_C(D)$;

(3) 对于任意属性 $a \in R$, 判断 $Sig_{inner}(a, R, D) > 0$ 是否大于 0, 大于 0 则保留, 否则删除;

(4) 返回约简集 R .

算法复杂度分析: 设 $|U|$ 和 $|C|$ 分别代表决策表中样本数和条件属性数, $|G|$ 表示分组数. 步骤 (2) 中依次删除分组 G_i , 将执行 $|G|$ 次, ① 中计算

内部重要度的时间复杂度为 $O(|C||U|^2)$, ②中计算正域时间复杂度为 $O(|C||U|^2)$, ③中计算外部重要度的时间复杂度为 $O(|C||U|^2)$, 即步骤(2)的时间复杂度为 $O(|G||C||U|^2)$. 步骤(3)的时间复杂度为 $O(|C||U|^2)$. 故算法的时间复杂度为 $O(|G||C||U|^2)$.

下面通过一个实例来说明算法的执行过程.

如表1所示是一个决策表, 该决策表包含五个条件属性 $\{a_1, a_2, a_3, a_4, a_5\}$ 和决策属性 $\{D\}$, 条件属性所对应的代价为 $\{10, 30, 40, 20, 10\}$, 利用算法2的约简方法, 假设用户给定属性块序 $S = G_3 > G_2 > G_1$, 其中 $G_3 = \{a_2\}$, $G_2 = \{a_1, a_3, a_5\}$, $G_1 = \{a_4\}$, $\lambda = 0.5$, 约简步骤如下:

(1) 初始化 $R = C$, 从 R 中删除 G_1 组属性, 此时 $POS_R(D) = POS_C(D)$.

(2) 从 R 中删除 G_2 组属性, 此时 $POS_R(D) \neq POS_C(D)$, 则计算 $Sig_{inner}(a_1, C, D) = 0$, $Sig_{inner}(a_3, C, D) = 0$, $Sig_{inner}(a_5, C, D) = 0$, 则该组中不含核属性. 计算 $S(a_1, G_2) = 0.65$, $S(a_3, G_2) = 0.56$, $S(a_5, G_2) = 0.58$, 则选择 a_1 属性加入至 R 中, 此时 $POS_R(D) \neq POS_C(D)$. 再计算 $S(a_3, G_2) = 0.42$, $S(a_5, G_2) = 0.73$, 选择 a_5 属性加入至 R 中, 此时 $POS_R(D) = POS_C(D)$.

(3) 从 R 删除 G_3 组属性, 此时 $POS_R(D) \neq POS_C(D)$, 则计算 $Sig_{inner}(a_2, C, D) = 0$, 所以 a_2 不是核属性, 计算 $S(a_2, G_3) = 0.34$, 所以选择 a_2 属性加入至 R 中, 最终 $POS_R(D) = POS_C(D)$. 判断 R 中属性是否为必要属性, 最终得到约简 $A_1 = \{a_1, a_2, a_5\}$, 该约简的代价为 50.

同理, 当 $\lambda = 0$ 时, 得到的约简结果为 $A_2 = \{a_2, a_3, a_5\}$, 该约简的代价为 80, 由此可以看出重要度加权的方式在局部组中也是有效的. 分别计算上述算法中 $\lambda = 0.5$ 和 $\lambda = 0$ 时的约简子集 $SAP(A_1) = SAP(A_2) = 2.33$, 表明这两个约简子集在该属性组序中的偏好程度相同.

属性组序是属性完全有序和完全无序的中间

表1 决策表

Table 1 A decision table

a_1	a_2	a_3	a_4	a_5	D
2	1	1	1	1	1
1	1	1	2	1	1
1	2	0	1	2	1
1	2	1	1	1	2
0	1	2	1	1	2
2	1	1	3	2	2
2	2	1	1	2	2

状态. 当属性集完全无序时, 即分组数为 1, 此时本文提出的算法退化为 ARWW 算法,^[25] 该算法利用属性重要度和代价加权的方式在全局寻找属性, 得到不同权重时的约简子集. 当属性集处于完全有序时, 即分组数等于属性个数, 此时算法仅根据用户偏好顺序依次删除属性, 得到偏好程度较高约简集合, 因为每个分组中仅存在一个属性, 其局部选择属性的方法将失效. 在实验分析中也验证了属性集随着分组数的增加, 其参数 λ 影响力的变化.

3 实验分析

为了验证本文提出算法的有效性, 本文选取了五组 UCI 数据集进行实验, 分别用 a, b, c, d, e 代表 Lymphography 数据集、Lung-Cancer 数据集、Dermatology 数据集、Breast-Cancer-Wisconsin (BCW) 数据集以及 Connect-4 数据集的条件属性. 由于不同用户对属性有不同的偏好, 对五组数据集采用随机不均等分组的方式, 其中设置组数 $G = 3$, λ 分别设置为 0 和 0.5 进行实验, 下面仅列出 Lymphography 数据集的 10 种不同分组的方式 (如表 2 所示).

通常情况下, 属性的重要性与获取属性所需的成本呈正相关, 因此本文采用属性重要度算法对属性进行代价的设置. 例如在 Lymphography 数据集中, 根据属性重要度算法得到的约简为 $\{a_{18}, a_2, a_{13}, a_{14}, a_{15}, a_{16}\}$, 首先将不在约简集合中的属性代价设置为 10, 其次按属性的重要度由低到高依次递增 10. 即 Lymphography 数据集条件属性为 $\{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}, a_{11}, a_{12}, a_{13},$

表 2 Lymphography 的分组方式

Table 2 The grouping methods of Lymphography dataset

ID	属性组序 $S = G_3 > G_2 > G_1$
1	$a_{16}, a_{12}, a_6, a_8, a_{10} > a_3, a_{11}, a_{14}, a_{18},$ $a_1, a_2 > a_5, a_{17}, a_4, a_{13}, a_7, a_{15}, a_9$
2	$a_2, a_{11}, a_{12}, a_3, a_{16}, a_9, a_6 > a_{13}, a_{17},$ $a_{10}, a_7, a_{14}, a_5 > a_8, a_1, a_4, a_{15}, a_{18}$
3	$a_{15}, a_{14}, a_1, a_{10}, a_{12}, a_{11} > a_5, a_{18}, a_9,$ $a_{17}, a_2 > a_8, a_3, a_4, a_{13}, a_7, a_{16}, a_6$
4	$a_9, a_3, a_{12}, a_{18}, a_1, a_8 > a_5, a_{15}, a_2,$ $a_{14}, a_4, a_{13} > a_6, a_{10}, a_7, a_{17}, a_{11}, a_{16}$
5	$a_9, a_{13}, a_{18}, a_5, a_{11}, a_7 > a_{15}, a_{12}, a_{14},$ $a_2, a_6, a_{17} > a_8, a_3, a_{16}, a_{10}, a_4, a_1$
6	$a_{17}, a_{16}, a_{10}, a_{12}, a_{13} > a_7, a_2, a_5, a_3,$ $a_4, a_{18} > a_{11}, a_{14}, a_9, a_1, a_8, a_{15}, a_6$
7	$a_6, a_2, a_{17}, a_3, a_8 > a_{11}, a_{16}, a_{15}, a_7,$ $a_{10}, a_{18}, a_4 > a_5, a_{14}, a_1, a_{13}, a_9, a_{12}$
8	$a_{18}, a_3, a_{11}, a_{10}, a_{15} > a_{12}, a_9, a_2, a_{17},$ $a_7, a_5, a_6 > a_1, a_{13}, a_8, a_{16}, a_{14}, a_4$
9	$a_3, a_8, a_4, a_{11}, a_{10}, a_{18}, a_9 > a_{15}, a_{17},$ $a_{12}, a_7 > a_{16}, a_{14}, a_2, a_6, a_5, a_{13}, a_1$
10	$a_{15}, a_{16}, a_3, a_1, a_4, a_5 > a_{13}, a_{17}, a_9,$ $a_{12}, a_8 > a_6, a_{12}, a_2, a_7, a_{18}, a_{10}, a_{14}$

$a_{14}, a_{15}, a_{16}, a_{17}, a_{18}$ }, 设置代价为 $\{10, 60, 10, 10, 10, 10, 10, 10, 10, 10, 10, 50, 40, 30, 20, 10, 70\}$.

表 3 至表 7 是 ARCSGO 算法在属性重要度和代价加权 ($\lambda=0.5$) 和仅考虑属性重要度 ($\lambda=0$) 时在五个数据集上的不同约简结果. 首先该算法针对不同的分组方式可以得到不同的约简结果, 其次在分组数 $G=3$ 时, 实验数据集所得的约简结果都有较高的 SAP 值, 这是因为算法在删除分组时优先保留了较高偏好的分组. 从代价角度来看, 局部加权的方法同样在属性组序的关系下成立, $\lambda=0.5$ 时和 $\lambda=0$ 时得到的约简结果相比, 代价相等或更低. 这里讨论一种最坏的情况, 即当所有重要度较大的属性恰好为一个约简结果处于最高偏好的分组当中时, 通过该算法优先按照属性组序的方式删除属性, 最终得到的约简结果就为属性重要度算法的约简结果, 其代价也就最大. 所以该算法以属性组序作为用户首要的偏好关系, 其次从各分组中添加属性, 可以得到较高 SAP 的约简结合, 其代价与用户分组方式有关.

表 3 Lymphography 数据集在 10 种不同分组方式下的约简结果

Table 3 Reduction results of Lymphography dataset under ten different grouping methods

ID	ARCSGO($\lambda=0.5$)			ARCSGO($\lambda=0$)		
	Reduct	COST	SAP	Reduct	COST	SAP
1	$a_1, a_3, a_{11}, a_{14}, a_2, a_{10}, a_{12}, a_8, a_6$	170	2.44	$a_{14}, a_{18}, a_3, a_1, a_{10}, a_{12}, a_{16}, a_{18}$	180	2.5
2	$a_{14}, a_{13}, a_6, a_{12}, a_{11}, a_3, a_9, a_{16}, a_2$	220	2.77	$a_{14}, a_{13}, a_2, a_{12}, a_{11}, a_6, a_3, a_{16}, a_9$	220	2.77
3	$a_8, a_5, a_2, a_{14}, a_{15}, a_{10}, a_{12}, a_1$	180	2.5	$a_8, a_5, a_2, a_{14}, a_1, a_{12}, a_{10}, a_{15}$	180	2.5
4	$a_6, a_{14}, a_{15}, a_{13}, a_{12}, a_1, a_{18}$	220	2.28	$a_6, a_{13}, a_2, a_{15}, a_{14}, a_{18}, a_{12}$	270	2.14
5	$a_{12}, a_{14}, a_6, a_{15}, a_2, a_{11}, a_5, a_{13}$	220	2.37	$a_{14}, a_{15}, a_{12}, a_2, a_{18}, a_{13}, a_{11}$	270	2.42
6	$a_8, a_6, a_2, a_{18}, a_{12}, a_{10}, a_{16}, a_{13}$	240	2.25	$a_{14}, a_2, a_{18}, a_{13}, a_{10}, a_{17}, a_{16}$	260	2.42
7	$a_5, a_{10}, a_{15}, a_{16}, a_{18}, a_6, a_8, a_{17}, a_2$	230	2.33	$a_5, a_{18}, a_{15}, a_{10}, a_{16}, a_2, a_8, a_6, a_{17}$	230	2.33
8	$a_8, a_{12}, a_{17}, a_6, a_7, a_5, a_2, a_{10}, a_{15}, a_{18}$	230	2.2	$a_{13}, a_2, a_6, a_{17}, a_{18}, a_{10}, a_{15}, a_{11}$	250	2.37
9	$a_6, a_1, a_{12}, a_{17}, a_7, a_{15}, a_{10}, a_8, a_3, a_{11}, a_{18}$	190	2.27	$a_{14}, a_{15}, a_{12}, a_{18}, a_3, a_8, a_{10}, a_{11}$	190	2.5
10	$a_{10}, a_6, a_{12}, a_{17}, a_8, a_{13}, a_1, a_5, a_{16}, a_{15}$	170	2.22	$a_{14}, a_6, a_{13}, a_{17}, a_1, a_5, a_{15}, a_{16}$	180	2.25

图 2 至图 6 分别表示在五个数据集中 10 种不同分组方式下, ARCSGO 算法、ARWW 算法以及属性重要度算法 (Attribute Reduction algorithm with Importance, ARWI)^[26] 的 SAP 值. 不难发现 ARCSGO 算法比 ARWW 算法和 ARWI 算法有更高的 SAP 值, 表明本文提出的算法能够找到当前分组情形下更为靠前, 也就是偏好程度更高的

属性集. 其原因在于该算法考虑了用户的属性组序偏好关系, 优先保留了具有较高偏好的属性, 其中参数 λ 作为局部分组中选取属性时的变量, 对 SAP 的影响较小, 可以发现 $\lambda=0$ 和 $\lambda=0.5$ 的 SAP 值相差不大. 当 ARWW 算法或 ARWI 算法的约简集合位于偏好程度最高的分组中时, 此时 ARCSGO 算法与其余两种算法求得的约简集合

表4 Lung-Cancer数据集在10种不同分组方式下的约简结果

Table 4 Reduction results of Lung-Cancer dataset under ten different grouping methods

ID	ARCSGO($\lambda=0.5$)			ARCSGO($\lambda=0$)		
	<i>Reduct</i>	<i>COST</i>	<i>SAP</i>	<i>Reduct</i>	<i>COST</i>	<i>SAP</i>
1	$b_{33}, b_{35}, b_{32}, b_{23}, b_7, b_{36}, b_{14}$	80	3	$b_{14}, b_{12}, b_{23}, b_{36}, b_{33}, b_7$	100	3
2	$b_{41}, b_{53}, b_{19}, b_3, b_{12}$	100	3	$b_{40}, b_6, b_{12}, b_3, b_{34}$	190	3
3	$b_{37}, b_{35}, b_{29}, b_7, b_{20}, b_4, b_{53}$	70	3	$b_{37}, b_{35}, b_{29}, b_7, b_{20}, b_4, b_{53}$	70	3
4	$b_{33}, b_{10}, b_8, b_{28}, b_{29}, b_{32}$	60	3	$b_{28}, b_8, b_{10}, b_{25}, b_{32}, b_5, b_{38}$	70	3
5	$b_{23}, b_{26}, b_{14}, b_{54}, b_{13}$	60	3	$b_{43}, b_6, b_{54}, b_4, b_{26}$	90	3
6	$b_2, b_7, b_{20}, b_{34}, b_{35}$	50	3	$b_{43}, b_2, b_{53}, b_3, b_{13}, b_7$	80	3
7	$b_{37}, b_{35}, b_{13}, b_3, b_{33}, b_2, b_{49}$	90	3	$b_{40}, b_{35}, b_2, b_{49}, b_3, b_{24}$	130	3
8	$b_{37}, b_{51}, b_{16}, b_{34}, b_7, b_{33}, b_{38}$	70	3	$b_{37}, b_{51}, b_6, b_{34}, b_7$	90	3
9	$b_{37}, b_{16}, b_{29}, b_3, b_{53}, b_{24}, b_{18}$	90	3	$b_{40}, b_6, b_{20}, b_2, b_{24}, b_{29}$	150	3
10	$b_{41}, b_{56}, b_{13}, b_{34}, b_3$	70	3	$b_{41}, b_3, b_{14}, b_6, b_{19}$	120	3

表5 Dermatology数据集在10种不同分组方式下的约简结果

Table 5 Reduction results of Dermatology dataset under ten different grouping methods

ID	ARCSGO($\lambda=0.5$)			ARCSGO($\lambda=0$)		
	<i>Reduct</i>	<i>COST</i>	<i>SAP</i>	<i>Reduct</i>	<i>COST</i>	<i>SAP</i>
1	$c_{21}, c_5, c_{28}, c_{32}, c_{17}, c_{18}, c_{20}, c_{31}, c_8, c_1, c_4$	150	2.9	$c_{22}, c_8, c_5, c_{31}, c_{28}, c_2, c_4, c_{17}, c_{32}, c_1, c_{18}$	210	2.9
2	$c_3, c_5, c_9, c_{14}, c_{20}, c_{29}, c_{17}, c_{16}, c_{28}$	110	2.11	$c_4, c_{16}, c_9, c_8, c_{19}, c_{32}, c_{28}, c_{18}, c_{13}, c_{10}$	150	2.7
3	$c_5, c_{29}, c_{15}, c_{10}, c_{16}, c_{32}, c_7, c_{18}, c_2, c_1, c_4$	170	2.9	$c_5, c_{22}, c_{29}, c_{15}, c_4, c_7, c_{32}, c_{16}, c_1, c_{18}$	220	2.9
4	$c_4, c_{19}, c_{17}, c_2, c_6, c_{24}, c_{10}, c_{34}$	150	2.87	$c_4, c_{34}, c_{19}, c_2, c_6, c_{22}, c_{17}, c_{24}$	210	2.87
5	$c_{32}, c_{21}, c_9, c_{14}, c_{26}, c_5, c_3, c_{18}, c_8, c_4$	130	2.8	$c_{34}, c_{22}, c_4, c_3, c_{14}, c_5, c_6$	200	2.85
6	$c_{26}, c_{28}, c_{14}, c_{19}, c_{18}, c_1, c_{33}, c_{34}$	130	3	$c_{25}, c_{34}, c_{19}, c_{28}, c_1, c_{18}, c_{14}, c_{26}$	130	3
7	$c_{34}, c_5, c_{19}, c_3, c_{32}, c_{18}, c_{10}, c_7$	120	2.62	$c_{34}, c_5, c_4, c_{32}, c_3, c_{10}$	130	2.5
8	$c_2, c_{17}, c_{21}, c_{18}, c_{16}, c_{33}, c_5, c_3, c_{28}, c_9$	120	2.5	$c_{34}, c_{22}, c_4, c_3, c_{28}, c_{33}, c_5$	200	2.71
9	$c_{19}, c_{17}, c_{28}, c_{34}, c_{18}, c_{33}, c_4$	140	2.57	$c_{16}, c_{19}, c_{34}, c_4, c_{18}, c_{10}, c_{29}$	160	2.71
10	$c_5, c_{21}, c_{17}, c_{25}, c_{19}, c_9, c_3, c_{18}, c_7, c_{16}$	120	2.7	$c_{34}, c_{28}, c_3, c_{16}, c_{19}, c_{18}, c_{11}, c_7$	140	2.75

表6 BCW数据集在10种不同分组方式下的约简结果

Table 6 Reduction results of BCW dataset under ten different grouping methods

ID	ARCSGO($\lambda=0.5$)			ARCSGO($\lambda=0$)		
	<i>Reduct</i>	<i>COST</i>	<i>SAP</i>	<i>Reduct</i>	<i>COST</i>	<i>SAP</i>
1	d_6, d_8, d_1, d_5	110	2.25	d_6, d_5, d_8, d_7, d_9	110	2.6
2	d_6, d_3, d_8, d_4	110	2.5	d_6, d_3, d_1, d_8	120	2.5
3	d_6, d_8, d_4, d_3	110	2	d_6, d_3, d_4, d_7, d_9	120	2.4
4	d_6, d_7, d_2, d_9, d_5	110	2.2	d_6, d_7, d_2, d_5, d_4	110	2.2
5	d_6, d_2, d_7, d_4, d_5	110	2.2	d_6, d_3, d_7, d_5	130	2
6	d_6, d_7, d_9, d_8, d_5	110	2.2	d_6, d_3, d_4, d_8	110	2.25
7	d_6, d_7, d_9, d_8, d_5	110	2.2	d_6, d_1, d_5, d_8	110	2.25
8	d_6, d_7, d_1, d_4	90	2.5	d_6, d_7, d_1, d_4	90	2.5
9	d_6, d_2, d_7, d_1	90	2.25	d_6, d_5, d_1, d_8	110	2.25
10	d_6, d_2, d_7, d_1	90	2	d_6, d_3, d_1, d_8	120	2.25

具有相同的SAP值.一般情形下,ARCSGO算法能得到更符合用户偏好的约简子集.

图7至图11表示的是ARCSGO算法在 $\lambda=0, \lambda=0.5$ 以及 $\lambda=1$ 取值下,随着分组数的增加,代价变化的曲线图.这里选取每个数据集中 $ID=1$ 的例子不断细化分组,在细化的过程中保留上一级分组的偏好关系.可以发现,随着分组数的增加,不同 λ 值下的代价最终都会趋于稳定,其原因在于随着分组数增加,每组中的属性个数在减少,用户对属性的偏好越明确,并且本文提出的算法是优先考虑属性组序关系, λ 仅作为局部选择属性的参数,随着分组数的增加,其影响力也逐渐变小.

表 7 Connect-4 数据集在 10 种不同分组方式下的约简结果

Table 7 Reduction results of Connect-4 dataset under ten different grouping methods

ID	ARCSGO($\lambda=0.5$)			ARCSGO($\lambda=0$)		
	Reduct	COST	SAP	Reduct	COST	SAP
1	$e_{26}, e_{32}, e_{14}, e_7, e_8, e_{21}, e_{25}, e_1,$ $e_2, e_{31}, e_{22}, e_{28}, e_{37}$	670	1.76	$e_{14}, e_{21}, e_{15}, e_{26}, e_1, e_{31}, e_{25}, e_2,$ $e_{38}, e_{22}, e_{37}, e_{13}, e_{24}, e_{16}$	760	2.07
2	$e_{14}, e_1, e_{17}, e_7, e_{21}, e_{15}, e_{26}, e_{22}, e_{16},$ $e_3, e_{31}, e_8, e_{38}, e_{28}, e_{25}, e_{32}, e_2$	720	1.94	$e_{14}, e_{21}, e_1, e_{37}, e_{15}, e_{22}, e_{31}, e_8,$ $e_{26}, e_3, e_{25}, e_{32}, e_{13}, e_2$	850	1.92
3	$e_{16}, e_2, e_{31}, e_{22}, e_{21}, e_{38}, e_{32}, e_7,$ $e_8, e_{37}, e_1, e_{25}, e_{15}$	700	2	$e_{31}, e_2, e_{26}, e_{22}, e_{21}, e_{37}, e_8, e_7,$ $e_{32}, e_{38}, e_1, e_{15}, e_{25}$	700	2
4	$e_{22}, e_1, e_{17}, e_8, e_{15}, e_{25}, e_{38}, e_{16}, e_{28},$ $e_{14}, e_7, e_{26}, e_{31}, e_3, e_{32}, e_2, e_{21}$	720	2.05	$e_1, e_{37}, e_{13}, e_{15}, e_{25}, e_{14}, e_7, e_{16},$ $e_{21}, e_{31}, e_{23}, e_2, e_{32}$	940	2.07
5	$e_{38}, e_{17}, e_7, e_{21}, e_1, e_{25}, e_{16}, e_{32}, e_2,$ $e_8, e_{22}, e_{26}, e_3, e_{28}, e_{14}, e_{31}, e_{15}$	720	2.17	$e_{37}, e_{21}, e_1, e_8, e_{25}, e_2, e_{13}, e_{32},$ $e_{22}, e_{15}, e_{31}, e_{26}, e_3, e_{14}$	850	2.28
6	$e_{39}, e_{32}, e_1, e_7, e_{25}, e_{31}, e_3, e_{16}, e_{27},$ $e_{17}, e_2, e_8, e_{22}, e_{14}, e_{15}, e_{37}, e_{13}$	870	2.05	$e_1, e_{21}, e_{26}, e_{38}, e_{31}, e_{25}, e_2, e_{23},$ $e_8, e_{14}, e_{15}, e_{37}, e_{22}, e_{13}$	960	2.07
7	$e_{22}, e_{38}, e_{32}, e_{25}, e_{31}, e_1, e_{21}, e_7,$ $e_8, e_{16}, e_2, e_{37}, e_{15}$	700	2.07	$e_{22}, e_{38}, e_{14}, e_{21}, e_{31}, e_1, e_{25}, e_{26},$ $e_7, e_{37}, e_{23}, e_{15}, e_2, e_{16}$	810	2.14
8	$e_{25}, e_{28}, e_{32}, e_{14}, e_8, e_{22}, e_{38}, e_3, e_2,$ $e_{17}, e_{15}, e_{31}, e_1, e_{26}, e_{16}, e_7, e_{21}$	720	2.05	$e_{37}, e_{14}, e_{25}, e_{23}, e_{15}, e_2, e_{22}, e_{38},$ $e_{31}, e_{21}, e_7, e_1, e_{16}, e_{26}$	810	2.21
9	$e_{22}, e_{16}, e_1, e_{26}, e_{38}, e_{24}, e_{14}, e_{15},$ $e_{37}, e_{25}, e_{31}, e_9, e_2, e_7, e_{21}$	700	2.2	$e_1, e_{22}, e_{16}, e_{37}, e_{15}, e_{14}, e_{26}, e_{38},$ $e_{21}, e_7, e_{31}, e_{23}, e_2, e_{25}$	810	2.21
10	$e_{38}, e_1, e_{32}, e_2, e_{17}, e_{14}, e_8, e_{22}, e_{25},$ $e_{31}, e_{15}, e_{26}, e_{16}, e_3, e_{28}, e_7, e_{21}$	720	1.94	$e_{14}, e_{37}, e_1, e_2, e_{38}, e_{31}, e_{25}, e_{15}, e_{13},$ $e_{21}, e_7, e_{26}, e_{16}, e_9, e_{28}, e_{33}$	840	2.12

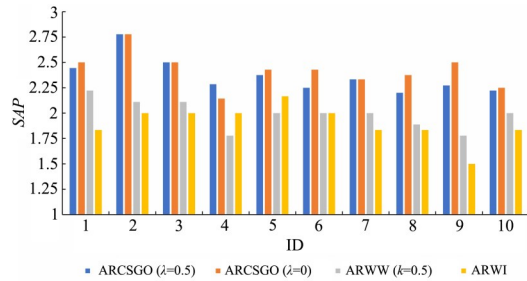


图 2 Lymphography 数据集的 SAP 值

Fig. 2 SAP value of Lymphography dataset

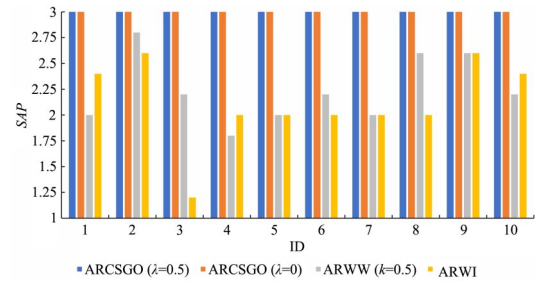


图 3 Lung-Cancer 数据集的 SAP 值

Fig. 3 SAP value of Lung-Cancer dataset

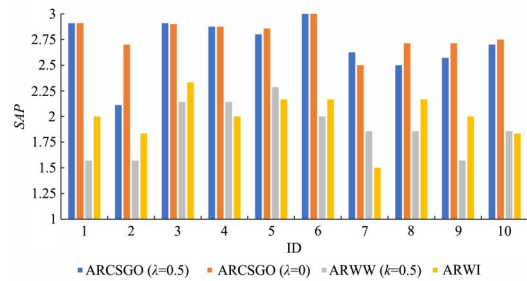


图 4 Dermatology 数据集的 SAP 值

Fig. 4 SAP value of Dermatology dataset

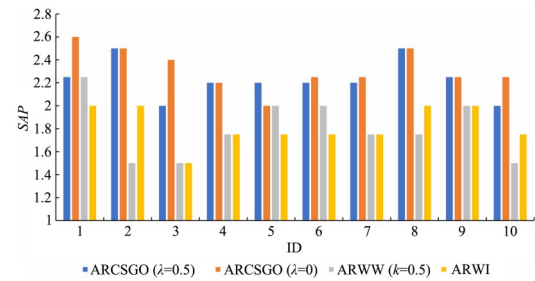


图 5 BCW 数据集的 SAP 值

Fig. 5 SAP value of BCW dataset

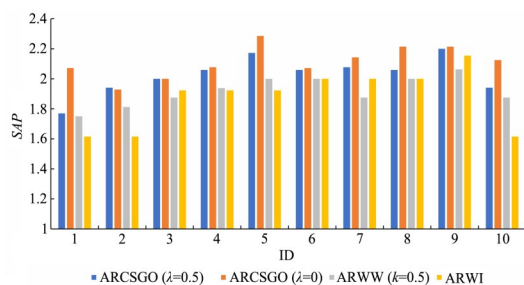


图6 Connect-4数据集的SAP值

Fig. 6 SAP value of Connect-4 dataset

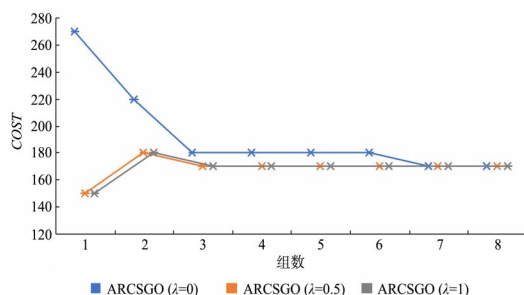


图7 Lymphography数据集在不同组数下的代价

Fig. 7 Cost under different groups of Lymphography dataset

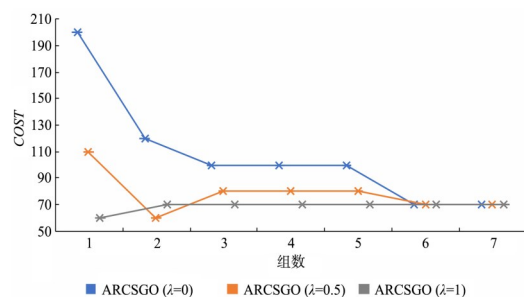


图8 Lung-Cancer数据集在不同组数下的代价

Fig. 8 Cost under different groups of Lung - Cancer dataset

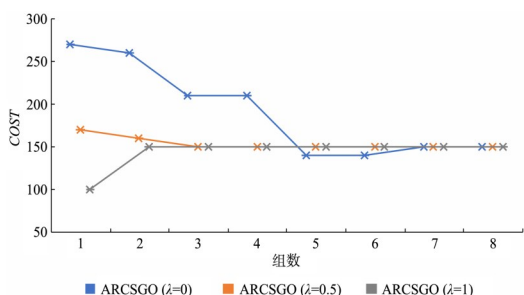


图9 Dermatology数据集在不同组数下的代价

Fig. 9 Cost under different groups of Dermatology dataset

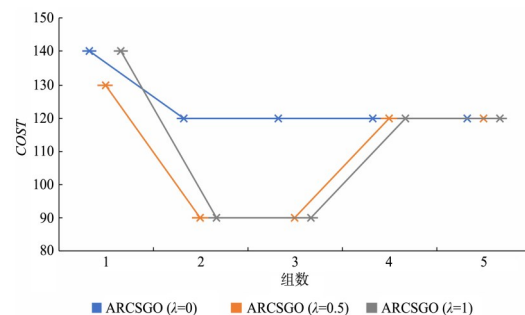


图10 BCW数据集在不同组数下的代价

Fig. 10 Cost under different groups of BCW dataset

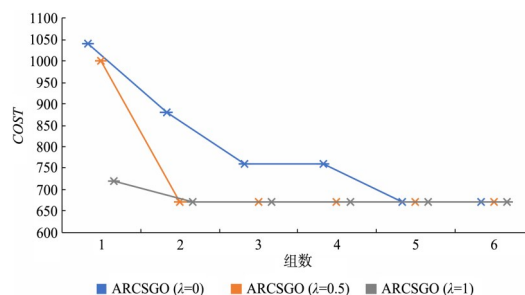


图11 Connect-4数据集在不同组数下的代价

Fig. 11 Cost under different groups of Connect-4 dataset

4 结 论

考虑用户偏好及现实生活中的代价敏感问题,本文提出一种属性组序下基于代价敏感的约简算法.该算法结合分组的思想,可对整组属性进行删除,然后再对组内属性进行添加.通过局部加权的方式添加属性和利用属性重要度选择属性相比,能够获取代价更低的约简集合.本文定义平均子集位置来描述子集在该分组中所处位置的高低,代表用户对子集的偏好程度,所设计的算法能获取较高的SAP值.实例分析以及在UCI数据集上的实验证明了该算法的可行性和有效性.本文主要针对符号数据采用等价关系进行研究,对于实数型数据或者混合型数据,可以将等价关系拓展到领域关系或者模糊关系.其次,用户对属性的分组方式是千变万化的,用户对属性目标越明确,加权方式越会随着分组数的增加而导致影响力的降低.何确定合适的组数以得到代价更低和SAP值更高的约简,是下一步研究的重点.

参考文献

- [1] Pawlak Z. Rough sets. *International Journal of Computer and Information Sciences*, 1982, 11(5): 341—356.
- [2] 王国胤,姚一豫,于洪. 粗糙集理论与应用研究综述. *计算机学报*, 2009, 32(7): 1229—1246. (Wang G Y, Yao Y Y, Yu H. A survey on rough set theory and applications. *Chinese Journal of Computers*, 2009, 32(7): 1229—1246.)
- [3] 于洪,王国胤,姚一豫. 决策粗糙集理论研究现状与展望. *计算机学报*, 2015, 38(8): 1628—1639. (Yu H, Wang G Y, Yao Y Y. Current research and future perspectives on decision; theoretic rough sets. *Chinese Journal of Computers*, 2015, 38(8): 1628—1639.)
- [4] 陈昊,杨俊安,庄镇泉. 变精度粗糙集的属性核和最小属性约简算法. *计算机学报*, 2012, 35(5): 1011—1017. (Chen H, Yang J A, Zhuang Z Q. The core of attributes and minimal attributes reduction in variable precision rough set. *Chinese Journal of Computers*, 2012, 35(5): 1011—1017.)
- [5] Yao Y Y, Lin T Y. Generalization of rough sets using modal logics. *Intelligent Automation & Soft Computing*, 1996, 2(2): 103—119.
- [6] Yao Y Y. Decision - theoretic rough set models// *Proceedings of 2nd International Conference on Rough Sets and Knowledge Technology*. Springer Berlin Heidelberg, 2007.
- [7] 杨传健,葛浩,汪志圣. 基于粗糙集的属性约简方法研究综述. *计算机应用研究*, 2012, 29(1): 16—20. (Yang C J, Ge H, Wang Z S. Overview of attribute reduction based on rough set. *Application Research of Computers*, 2012, 29(1): 16—20.)
- [8] Qian Y H, Liang J Y, Pedrycz W, et al. Positive approximation: an accelerator for attribute reduction in rough set theory. *Artificial Intelligence*, 2010, 174(9—10): 597—618.
- [9] Lazo-Cortés M S, Martínez-Trinidad J F, Carrasco-Ochoa J A, et al. A new algorithm for computing reducts based on the binary discernibility matrix. *Intelligent Data Analysis*, 2016, 20(2): 317—337.
- [10] Gao C, Lai Z H, Zhou J, et al. Maximum decision entropy - based attribute reduction in decision - theoretic rough set model. *Knowledge - Based Systems*, 2018, 143: 179—191.
- [11] Wang Y B, Chen X J, Dong K. Attribute reduction via local conditional entropy. *International Journal of Machine Learning and Cybernetics*, 2019, 10(12): 3619—3634.
- [12] Min F, He H P, Qian Y H, et al. Test-cost-sensitive attribute reduction. *Information Sciences*, 2011, 181(22): 4928—4942.
- [13] Fang Y, Min F. Cost-sensitive approximate attribute reduction with three - way decisions. *International Journal of Approximate Reasoning*, 2019, 104: 148—165.
- [14] Ma X A, Zhao X R. Cost-sensitive three-way class-specific attribute reduction. *International Journal of Approximate Reasoning*, 2019, 105: 153—174.
- [15] Jia X Y, Liao W H, Tang Z M, et al. Minimum cost attribute reduction in decision - theoretic rough set models. *Information Sciences*, 2013, 219: 151—167.
- [16] Wang J, Wang J. Reduction algorithms based on discernibility matrix; the ordered attributes method. *Journal of Computer Science and Technology*, 2001, 16(6): 489—504.
- [17] Zhao K, Wang J. A reduction algorithm meeting users' requirements. *Journal of Computer Science and Technology*, 2002, 17(5): 578—593.
- [18] Yao Y Y, Zhao Y, Wang J, et al. A model of machine learning based on user preference of attributes// *International Conference on Rough Sets and Current Trends in Computing*. Springer Berlin Heidelberg, 2006: 587—596.
- [19] Yao Y Y, Zhao Y, Wang J, et al. A model of user-oriented reduct construction for machine learning. *Transactions on Rough Sets VIII*. Springer Berlin Heidelberg, 2008: 332—351.
- [20] Han S Q, Wang J. Reduct and attribute order. *Journal of Computer Science and Technology*, 2004, 19(4): 429—449.
- [21] 官礼和,王国胤,胡峰. 一种基于属性序的决策规则挖掘算法. *控制与决策*, 2012, 27(2): 313—316. (Guan L H, Wang G Y, Hu F. A decision rules mining algorithm based on attribute order. *Control and Decision*, 2012, 27(2): 313—316.)
- [22] 韩素青,阴桂梅. 一种面向用户需求的属性约简算法. *模式识别与人工智能*, 2014, 27(3): 281—288.

- (Han S Q, Yin G M. An user-oriented attribute reduct construction algorithm. Pattern Recognition and Artificial Intelligence, 2014, 27(3): 281—288.)
- [23] 胡峰, 王国胤. 属性序下的快速约简算法. 计算机学报, 2007, 30(8): 1429—1435. (Hu F, Wang G Y. Quick reduction algorithm based on attribute order. Chinese Journal of Computers, 2007, 30(8): 1429—1435.)
- [24] 王国胤. Rough集理论与知识获取. 西安: 西安交通大学出版社, 2001: 23—26, 133—136.
- [25] Zhang Q H, Shen W. Research on attribute reduction algorithm with weights. Journal of Intelligent & Fuzzy Systems, 2014, 27(2): 1011—1019.
- [26] Hu K Y, Lu Y C, Shi C Y. Advances in rough set theory and its applicatinons. Journal of Tsinghua University (Science and Technology), 2001, 41(1): 64—68.

(责任编辑 杨可盛)