

基于属性重要性的决策表属性约简算法

杨成福^{1,2}, 舒 兰¹

(1. 电子科技大学, 四川 成都 610054; 2. 河西学院, 甘肃 张掖 734000)

摘 要: 提出一种基于粗糙集属性重要性的属性约简算法。该算法以所有条件属性为初始约简集合, 以属性重要性为迭代准则, 通过逐步缩减来求取约简。同时给出了该算法的时间复杂度分析, 并举例验证了所提出算法的有效性和实用性。

关键词: 粗糙集; 属性重要性; 属性约简

中图分类号: TP301.6

文献标识码: A

文章编号: 1673-629X(2006)11-0062-02

An Algorithm of Attribute Reduction of Decisive Table
Based on Attribute ImportanceYANG Cheng-fu^{1,2}, SHU Lan¹

(1. University of Electronic Sci. and Techn., Chengdu 610054, China;

2. Hexi University, Zhangye 734000, China)

Abstract: Proposed an algorithm of attribute reduction based on attribute importance of rough set. With all the condition attribute as the initial reduction, this algorithm takes importance of attribute as the iterative criterion to find reduction. The time complexity of the algorithm is analyzed and an example is investigated to verify it. The results show this algorithm can find the reduction effectively.

Key words: rough set; attribute importance; attribute reduction

0 引 言

粗糙集理论是由波兰科学家 Pawlak 在 1982 年提出的一种数学理论^[1], 该理论主要用于数据分析, 尤其对不精确和不确定的数据进行分析。近年来, 它已被广泛应用到人工智能、模式识别和数据挖掘等方面。属性约简是粗糙集理论中的一个重要的课题, 一般说来, 知识库中的知识属性并不是同等重要的, 而且还存在冗余, 这不利于作出正确而简洁的决策, 属性约简要求在保持知识库的分类和决策能力不变的条件下, 删除不相关或不重要的属性。在大量的数据中, 属性是否冗余就要看属性是否相对重要, 由此, 文中提出了一个删除冗余属性的重要性算法。

1 粗糙集理论的基本概念

定义 1 论域和知识^[2]。

设 $U = \emptyset$ 是人们感兴趣的对象所组成的有限集合, 称为论域。任何子集 $X \in U$ 称为 U 的一个概念 (一般认为 \emptyset 也是一个概念)。 U 中的任何概念族称为 U 的抽象知识, 简称知识。 R 是 U 上的一个等价关系, U/R 表示 R 的所有等价类构成的集合, $[x]_R$ 表示包含 x 的 R 等价类, 所

以有: $U/R = \{X_1, X_2, \dots, X_n\}, X_i \in U, i = 1, 2, \dots, n$,
且 $X_i \cap X_j = \emptyset, \bigcup_{i=1}^n X_i = U (i \neq j, 1 \leq i, j \leq n)$ 。

定义 2 粗糙集的上一下近似^[2]。

设 $Y \subseteq U$ 是任意一个子集, R 是 U 上的等价关系, 则
 $RY = \bigcup \{X_i \mid X_i \in U/R, X_i \subseteq Y\} = \{x \in U \mid [x]_R \subseteq Y\}$
 $\bar{R}Y = \bigcup \{X_i \mid X_i \in U/R, X_i \cap Y \neq \emptyset\} = \{x \in U \mid [x]_R \cap Y \neq \emptyset\}$

分别称它们为 R 的下近似和 R 的上近似。实际上, 下近似就是所有那些被包含在 Y 里的等价类的并集; 上近似就是所有那些与 Y 有交的等价类的并集。

定义 3 等价关系相对于等价关系的正域^[3]。

设 P, Q 是 U 中的两个等价关系, Q 的 P 正域记为: $\text{POS}_P(Q)$, 即: $\text{POS}_P(Q) = \bigcup_{Y \in U/Q} PY$, 也即是: U 中的所有根据分类 U/P 的信息可以准确地划分到关系 Q 的等价类中去的对象的集合。

定义 4 近似分类质量^[4]。

设 P, Q 是 U 中的两个等价关系, 且 $U/P = \{X_1, X_2, \dots, X_n\}, U/Q = \{Y_1, Y_2, \dots, Y_m\}$, 则 P 对 Q 的近似分类

质量记为: $r_P(Q)$, 即: $r_P(Q) = \sum_{i=1}^m |PY_i| / |U|$ 。

定义 5 属性相对重要性^[4]。

对于决策信息系统 $S = (U, A = C \cup D)$, 其中 C 为

收稿日期: 2006-02-10

作者简介: 杨成福 (1966-), 男, 甘肃武威人, 硕士研究生, 研究方向为粗糙集理论; 舒 兰, 教授, 博士生导师, 研究方向为粗糙集理论、模糊信息处理技术。

条件属性, D 为决策属性, $B \subseteq C$, $b \in B$ 在 B 中的相对重要性定义为 $\text{sig}_{B-|b|}^D(c) = r_B(D) - r_{B-|b|}(D)$ 。当 $B = C$ 时, 就为 $\text{sig}_{C-|b|}^D(c) = r_C(D) - r_{C-|b|}(D)$ 。

2 基于属性重要性的属性约简算法

对于决策信息系统 $S = (U, A = C \cup D)$, 设 $c \in C$, 若有 $\text{sig}_{C-|c|}^D(c) = 0$, 则称 c 在 C 中 D 是不重要的(可省略的)。当 C 中每个元素都不为 C 中 D 可省略时, 称 C 为 D 独立。当 $R \subseteq C$ 为 D 独立, 且 R 中所有元都是 D 不可省略时, 称 R 为 C 的一个约简。文中依据该思想给出一种基于属性重要性求取约简算法, 该算法以所有条件属性为初始约简集合, 以属性重要性是否为零为前提, 逐步缩减来求取约简。具体算法如下:

● 算法: 求取约简。

- 1) 初始化: $R = C$ // C 是条件属性集;
- 2) for each $c \in C$
- 3) 计算属性 c 的相对重要性: $\text{sig}_{R-|c|}^D(c) = r_R(D) - r_{R-|c|}(D)$;
- 4) if $\text{sig}_{R-|c|}^D(c) = 0$
- 5) $R = R - \{c\}$
- 6) Endif
- 7) Endfor
- 8) Return R // 得到约简 R

● 算法: 计算属性重要性。

- 1) $U/D = \{Y_1, Y_2, \dots, Y_n\}$
- 2) $U/R = \{X_1, X_2, \dots, X_k\}$
- 3) $U/(R - \{c\}) = \{Z_1, Z_2, \dots, Z_p\}$
- 4) For $i = 1$ to n
- 5) For $j = 1$ to k
- 6) If $(Y_i \supseteq X_j)$
- 7) $\underline{R}Y_i = \underline{R}Y_i \cup X_j$
- 8) Endif
- 9) Endfor
- 10) For $l = 1$ to p
- 11) If $(Y_i \supseteq Z_l)$
- 12) $\underline{D}Y_i = \underline{D}Y_i \cup Z_l$
- 13) Endif
- 14) Endfor
- 15) $|\underline{R}Y| = |\underline{R}Y| + |\underline{R}Y_i|$
- 16) $|\underline{D}Y| = |\underline{D}Y| + |\underline{D}Y_i|$
- 17) Endfor
- 18) 计算 c 的相对重要性:

$$\text{sig}_{R-|c|}^D(c) = \frac{1}{|U|} (|\underline{R}Y| - |\underline{D}Y|)$$

3 算法时间复杂度分析

对于计算属性重要性算法, 由文献[5]知第 1) 步的时间复杂度分别为 $O(mn)$, $O(sm)$ 和 $O((s-1)mp)$, 其

中 m 为 U 中的对象个数, s 为条件属性集合 C 中元素个数, 因 $n, k, p \leq m$, 所以第 2) ~ 第 15) 步的时间复杂度为 $O(2m^2)$ 。整个属性重要性算法时间复杂度为 $O(mn) + O(sm) + O((s-1)mp) + O(2m^2) \leq O(2m^2)$ 。对于求取属性约简算法, 由上述推理知, 第 3) 步的时间复杂度为 $O(sm^2)$ 。由于算法中的循环最多循环 s 步, 故求取属性约简算法的时间复杂度为 $O(s^2m^2)$ 。

4 应用实例

表 1 所示的决策系统 $S = (U, A = C \cup D)$, 论域 U 由 8 个对象组成, 其条件属性集 C 包括 4 个属性, 即 | 学历, 经验, 是否说法语, 证明 |。其中学历属性值的意义如下: MBA 为工商管理硕士, MCE 为土木工程学硕士, MSc 为理学硕士。决策属性为是否接受。对于决策表 1 所示的决策系统, 有 $U/C = \{|X_1|, |X_2|, |X_3|, |X_4|, |X_5|, |X_6|, |X_7|, |X_8|\}$, $U/D = \{|X_1, X_4, X_6, X_7|, |X_2, X_3, X_5, X_8|\}$, 由求取约简算法首先计算属性 a_1 的重要性得 $\text{sig}_{R-|a_1|}^D(a_1) = 0$, 此时可得属性约简为 $R = \{a_2, a_3, a_4\}$; 然后计算属性 a_2 相对于 $R = \{a_2, a_3, a_4\}$ 的重要性, 得 $\text{sig}_{R-|a_2|}^D(a_2) = 5/8$; 再计算属性 a_3 相对于 $R = \{a_2, a_3, a_4\}$ 的重要性, 得 $\text{sig}_{R-|a_3|}^D(a_3) = 0$, 此时可得属性约简为 $R = \{a_2, a_4\}$; 接着再计算属性 a_4 相对于 $R = \{a_2, a_4\}$ 的重要性, 得 $\text{sig}_{R-|a_4|}^D(a_4) = 1/4$ 。根据算法此时就可得到原决策表条件属性相对约简为 $R = \{a_2, a_4\}$ 。可以验证, $\{a_2, a_4\} = \{ \text{经验, 证明} \}$ 是决定一个对象属于哪一类的最小集。需要指出的是, 最小约简集不是唯一的。

表 1 一个两类决策系统

	学历 (a_1)	经验 (a_2)	是否说法语 (a_3)	证明 (a_4)	是否接收 (d)
X_1	MBA	中	是	优秀	是
X_2	MBA	低	是	一般	否
X_3	MCE	低	是	良好	否
X_4	MSc	高	是	一般	是
X_5	MSc	中	是	一般	否
X_6	MSc	高	是	优秀	是
X_7	MBA	高	否	良好	是
X_8	MCE	低	否	优秀	否

5 结束语

以粗糙集中属性相对重要性为基础, 由此给出求取属性约简的一种算法。该算法以所有条件属性为初始约简集合, 依次从属性集中删除不重要的属性, 最终得到一个属性约简。通过实例验证该算法是有效的, 也是合理的。

参考文献:

- [1] Pawlak Z. Rough sets[J]. Int J of Information and Computer Science, 1982, 11(5): 341-356.
- [2] 刘清. Rough 集及 Rough 推理[M]. 北京: 科学出版社,

(下转第 66 页)

串口的返回结果后,准备发送已申请关键操作标识,并与服务器端断开连接时,CPU 转向处理并发的另一客户端连接请求。因为此时系统的 Locked 属性尚未赋值为假(如果是在关键操作执行完后立即置 Locked 为假,仍然可能在上下操作之间被插入其它操作),所以另一客户端自动断开。这样服务器端断开事件中得到的是否请求过关键操作标识就是第二次的请求结果,而第一次连接中发送的标识会被丢失(服务器端读到为空)。因此没有操作将 Locked 赋值为假,串口资源无法解锁,会一直死锁下去。

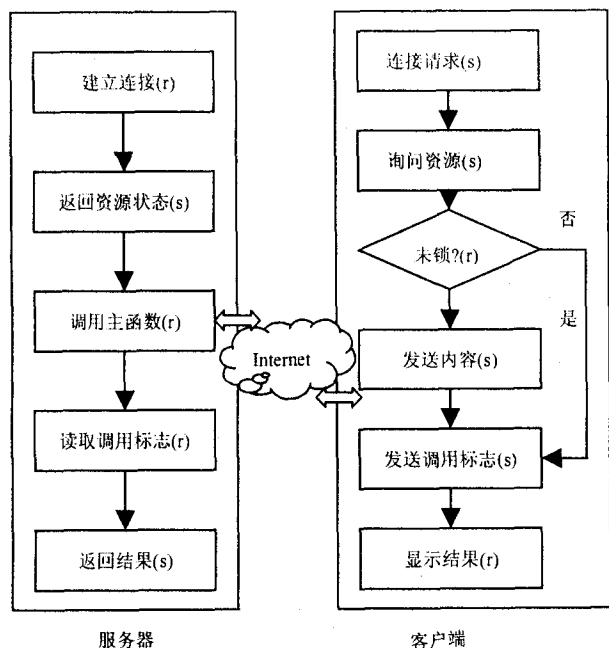


图3 握手协议二次开发方案程序执行顺序图

1.2.4 快速反应方案

摒弃二次开发 TCP 握手过程。采用对客户端每一次申请串口资源尝试一次连接,以希望能对每次尝试连接快速给出反应^[5]。

当客户端与服务器建立连接后,首先检查服务器端监听的线程数。如果线程数等于 1,则与服务器端相连的只有一个客户端,可以将串口资源分配给它。当线程数大于 1 时,直接调用断开操作。如图 4 所示。

在实际测试过程中,同样可能出现在客户端调用断开操作后,服务器端触发断开事件前,被插入其它客户端或该客户端本身的另一请求。若请求来自其它客户端,则对系统可靠性不产生影响,因为此方案中未使用 Locked 变量。若请求来自该客户端本身,则可能出现已断开连接,又要求二次断开的情况,这样系统会捕捉到访问 0 内存地

址异常。解决方法是在客户端以某种方式保证在上一次请求未得到返回结果前,不触发下一次请求(例如可以将请求的 Button 设置为不可点击)。

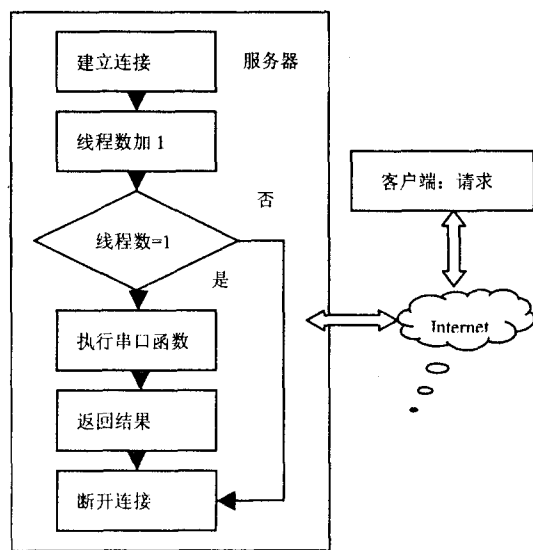


图4 快速反应方案的程序执行顺序图

2 结束语

前两种方案适用于客户端具有多线程处理能力的情况。这种情况下,系统处于多线程并发执行状态,虽然总的执行速度和后面方案的串行执行一样,但对客户端做到了真正的即时响应。但由于交互过程比较复杂,因此系统稳定性较差,而后两种方案对于客户端要求比较低,整个系统相当于完全处于串行执行状态,因此逻辑实现比较简单,系统也相对稳定。

参考文献:

- [1] 李 维. Delphi 5. x 多层分布式应用[M]. 北京:机械工业出版社,2000.
- [2] 张尧学,史美林. 计算机操作系统教程(第2版)[M]. 北京:清华大学出版社,2000.
- [3] 鲍 敏,吴 昊. Delphi 网络高级编程[M]. 北京:人民邮电出版社,2001.
- [4] 微软公司. 微软管理解决方案:作业调度[EB/OL]. 2002. <http://www.microsoft.com/china/technet/itsolutions/tcchguide/msm/smf/smfjobsc.mspx>.
- [5] 宗大华,宗 涛. 操作系统[M]. 北京:人民邮电出版社,2002.

(上接第 63 页)

2001.

- [3] 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.
- [4] 王国胤. Rough 集理论与知识获取[M]. 西安:西安交通大

学出版社,2001.

- [5] Sever H. Knowledge structuring for database mining and text retrieval using past optimal queries[D]. LA, USA: The university of Southwestern Louisiana, 1995.