对决策表和简化决策表的研究

王希雷，苏 静
（天津科技大学 计算机学院，天津 300222）

摘 要：简化决策表具有高效处理数据的能力，一经提出即得到广泛的重视。文中通过反例证明简化决策表在处理数据时会产生错误，改变了原决策表表达信息。通过分析得出简化决策表不具备处理不一致数据的能力，针对简化决策表存在的缺陷对其进行修正，提出新简化决策表的概念和建立算法。新简化决策表具有高效处理数据的能力，同时具备处理不一致数据的能力，并且不需要修改原有以决策表或简化决策表为基础的求核和求约简算法。

关键词：Rough集；简化决策表；约简；核
中图分类号：TP18 **文献标识码**：A **文章编号**：1673－629X(2008)04－0118－03

Research on Decision Table and Simple Decision Table

WANG Xi-lei, SU Jing
(College of Computer Sci. & Eng., Tianjin Univ. of Sci. & Techn., Tianjin 300222, China)

Abstract: Simple decision table is cared widely for it can simplify decision table. In this paper, prove simple decision table can produce errors by counterexamples and simple decision table change the information in decision table. Through analyzing know that simple decision table cannot deal with inconsistence data. Present new simple decision table to modify simple decision table. New simple table can simplify decision table and deal with inconsistence data and former algorithms based on decision table or simple decision table do not need to be modified.

Key words: rough sets; simple decision table; reduction; core

0 引 言

Rough集理论在知识发现领域取得了很大的成功。降低基于Rough集理论求核和约简算法的时间空间复杂度一直是Rough集理论的研究重点内容之一。文献[1,2]对二进制可辨矩阵的时间空间复杂度进行了大幅度的降低，有效地去掉了其中大量的无效计算和冗余信息；文献[3]证明了可辨矩阵的缺陷；文献[4]在求正区域前进行排序，大幅度降低了基于正区域计算的一系列算法的时间复杂度；文献[5]从去除决策表中冗余信息入手，降低了决策表的空间复杂度，以此为基础的算法时间空间复杂度就会大幅度下降，这也是简化决策表核心思想和优点。

但是简化决策表的定义是有缺陷的，会导致简化决策表丢失信息，并影响在此基础上求出的约简和核。

文中将分析上述问题产生原因，提出反例，并给出新简化决策表定义和建立算法。新简化决策表在保留简化决策表优点的前题下，对决策表或简化决策表为基础的算法不需改动即可继续使用。

收稿日期：2007－07－02
基金项目：天津市高等学校科技发展基金(20061011)
作者简介：王希雷(1973－)，男，黑龙江人，硕士，研究方向为Rough集理论与应用、商空间理论、粒度计算、知识发现。

1 基本概念

1.1 简化决策表及相关定义定理

定义1[5]：$T=(U,C,D,V,f)$是一个决策表，其中论域$U=(u_1,u_2,\cdots,u_n)$，条件属性集$C=(c_1,c_2,\cdots,c_m)$，决策属性$D=(d)$，$V=\bigcup_{a\in C\cup D}V_a$，$V_a$是属性$a$的值域，$f:U\times(C\cup D)\to V$是一个信息函数，它对一个对象的每一个属性赋予一个信息值，即$\forall a\in C\cup D, x\in U$，有$f(x,a)\in V_a$；每个属性子集$P\subseteq(C\cup D)$决定了一个二元不可分辨关系IND(P)：

$$IND(P)=\{(x,y)\in U\times U\mid \forall a\in P, f(x,a)=f(y,a)\}$$

关系IND(P)构成了U的一个划分，用$U/IND(P)$表示，简记为U/P，U/P中的任何元素$[x]_P=\{y\mid \forall a\in P, f(x,a)=f(y,a)\}$称为等价类。

定理1[5]：在决策表$T=(U,C,D,V,f)$中，记

$$POS_C D=\bigcup_{X\in U/C且\forall x,y\in X\Rightarrow f(x,D)=f(y,D)} X$$

该定理说明，C关于D的正区域是由在决策属性

上取值唯一的基本块的并集组成,任何决策属性值不唯一的基本块不在正区域中。称 U/C 中的等价类为基本块。

定义 2[5]:在决策表 $T=(U,C,D,V,f)$ 中,记 $U/C=\{[u'_1]_C,[u'_2]_C,\cdots,[u'_m]_C\}$,$U'=\{u'_1,u'_2,\cdots,u'_m\}$,由定理 1 可设 $POS_C(D)=[u'_{i_1}]_C\cup[u'_{i_2}]_C\cup\cdots\cup[u'_{i_t}]_C$,其中 $\forall u'_{i_s}\in U'$ 且 $|[u'_{i_s}]_C/D|=1(s=1,2,\cdots,t)$;记 $U'_{pos}=\{u'_{i_1},u'_{i_2},\cdots,u'_{i_t}\}$,$U'_{neg}=U'-U'_{pos}$。称 $T'=(U',C,D,V,f)$ 为简化决策表。

1.2 新简化决策表定义

定义 3:在决策表 $T=(U,C,D)$ 中,记 $U/C=\{[u'_1]_C,[u'_2]_C,\cdots,[u'_m]_C\}$,若 $|[u'_i]_C/D|=1$ 则提取 $u'_j\in[u'_i]_C$ 放入 U'_1,若 $|[u'_i]_C/D|\neq 1$ 则提取 $u'_i,u'_j,\cdots,u'_s\in[u'_i]_C$ 放入 U'_2 中(其中 $u'_i\neq u'_j\neq\cdots\neq u'_s$),称 $T'=(U',C,D)$ 为新简化决策表,其中 $U'=U'_1+U'_2$。

由定义 3 易知 U'_1 中的对象全为一致对象,称 U'_1 为一致对象集;U'_2 中的对象全为不一致对象,称 U'_2 为不一致对象集。

2 简化决策表的分析

举例证明简化决策表存在丢失信息的问题,并且以简化决策表为基础的约简和核也存在由此引发的错误。

表 1　决策表 T

$U\setminus A$	a	b	d
1	2	0	0
2	2	0	1
3	2	1	0
4	2	1	1
5	0	0	0
6	0	0	0

以表 1 决策表 $T=(U,C,D,V,f)$,$C=\{a,b\}$,$D=\{d\}$,为例说明简化决策表的含义:先求出条件属性集对 T 的划分 $U/C=\{\{u_1,u_2\},\{u_3,u_4\},\{u_5,u_6\}\}$,然后取每个等价类中一个对象组成 $U'=\{u_1,u_5,u_3\}$(简化决策表定义没有明确说明取等价类中的哪个对象,在文献[5]建立简化决策表算法和示例中取等价类中第一个对象;对含不一致对象的决策表无论取等价类中哪个对象,都存在丢失信息问题),$T'=(U',C,D,V,f)$ 为简化决策表(见表 2)。

按文献[5]中算法 3 的第一步(文献[5]中算法 3 的第 1 步是调用求等价类算法 1 建立简化决策表)建立 T 的简化决策表过程如下(算法的详细步骤参见文献[5]中的算法 1,3):

由算法 1 得　$\rightarrow u_1\rightarrow u_2\rightarrow u_3\rightarrow u_4\rightarrow u_5\rightarrow u_6$

第 1 趟'分配'结果:front[0] $\rightarrow u_1\rightarrow u_2\rightarrow u_3\rightarrow u_4\leftarrow$ end[0]

front[1] $\rightarrow u_5\rightarrow u_6\leftarrow$ end[1]

第 1 趟'收集'结果:$\rightarrow u_1\rightarrow u_2\rightarrow u_3\rightarrow u_4\rightarrow u_5\rightarrow u_6$

第 2 趟'分配'结果:front[0] $\rightarrow u_1\rightarrow u_2\rightarrow u_5\rightarrow u_6\leftarrow$ end[0]

front[1] $\rightarrow u_3\rightarrow u_4\leftarrow$ end[1]

第 2 趟'收集'结果:$\rightarrow u_1\rightarrow u_2\rightarrow u_5\rightarrow u_6\rightarrow u_3\rightarrow u_4$

得到 $U/\{a,b\}$ 为:$\{u_1,u_2\}$,$\{u_5,u_6\}$,$\{u_3,u_4\}$,

得到 $U'=\{u_1,u_5,u_3\}$,$T'=(U',C,D,V,f)$ 为简化决策表。

表 2　T 的简化决策表 T'

$U\setminus A$	a	b	d
1	2	0	0
5	0	0	0
3	2	1	0

虽然 T' 比 T 的空间少了很多(说明简化决策表能减小决策表空间),但是很明显 T' 表达信息与 T 表达信息不同。通过核和约简定义易得 T 的核和约简均为 $\{a\}$,简化后 T' 的核和约简均为 ϕ(T' 决策属性值全是 0,与条件属性无关),可以看出 T' 改变了 T 表达的信息,改变了 T 的核和约简。

3 新简化决策表建立算法

简化决策表定义实质上是针对决策表中存在大量条件属性相同的对象(简化决策表观点认为它们是冗余信息,如:T 中的 u_5,u_6;u_1,u_2;u_3,u_4),通过仅保留相同条件属性对象中的一个对象来简化决策表的空间。但是条件属性值相同的对象由于决策属性值不同(不一致对象)表达信息不完全相同,简化决策表定义错误的根源在于认为不一致对象表达信息完全相同(如:认为 T 中的 u_1,u_2 完全相同,u_3,u_4 完全相同),即丢失了部分不一致对象的信息,所以产生了第 2 节中指出的错误。因此对简化决策表的修正,只要把丢失的信息放入简化决策表中即可。定义 3 为文中给出的新简化决策表定义,其中'若 $|[u'_i]_C/D|\neq 1$ 则提取 $u'_i,u'_j,\cdots,u'_s\in[u'_i]_C$ 放入 U'_2 中'就是把不一致对象信息无损失地放入新简化决策表中(例如:把 T 中 u_1,u_2 都放入简化决策表中,即认为 u_1,u_2 不同),实现了对简化决策表的修正,新简化决策表把一致对象和

不一致对象分别放入 U'_1, U'_2 中，为基于新简化决策表的算法或理论提供了尽可能多的信息，方便了基于新简化决策表的算法或理论的进一步发展。

设存在决策表 $T = (U, C, D)$，$u = \{u_1, u_2, \cdots, u_n\}$，$C = \{c_1, c_2, \cdots, c_m\}$，$D = \{d\}$。

新简化决策表建立算法：NSDT 算法

输入：决策表 T

输出：新简化决策表 T_N

step1. 统计每个属性 c_i 值的个数 N_{c_i}，用数字 1,2,… 替换各属性值，记下替换规则。

step2. 用静态链表存储对象 $u_1, u_2, \cdots, u_n$，令表头指针指向 u_1。

step3. for $(i = 1; i < m + 1; i++)$。

step3.1 第 i 趟分配：建立 N_{c_i} 个空队列，令 front_k 和 $\text{end}_k (k = 0,1,2,\cdots,N_{c_i})$ 分别为第 k 个队列头指针和尾指针，将链表中对象 $u \in U$ 按链表中次序分配到第 k 个队列中。

step3.2 第 i 趟收集：表头指针指向第 1 个非空队列的头指针，修改每一个非空队列的尾指针，令其指向下一个非空队列的尾指针，令其指向下一个非空队列的对头对象，将所有队列重新组成一个链表。

step4. 设由 step3 的得到链表中的对象序列为 $u'_1, u'_2, \cdots, u'_n, i = 1$

do

while $u_i = u_{i+1}$ do $\{i = i + 1\}$;

if $i = n$ then $\{u_i \to U'_1$, break$\}$;

if $u(c)'_i \neq u(c)'_{i+1}$ then $\{u_i \to U'_1, i = i + 1\}$;

$l = i$;

if $u(c)'_i = u(c)'_{i+1}$

then$\{\{$do $i = i + 1$ until $u(c)'_i \neq u(c)'_{i+1}\}, \{u'_l, u'_{l+1}, \cdots, u'_i\} \to U'_2, i = i + 1\}$;

until $i > n$

step5. 设 $c_{m+1} = d$，对 U'_2 中对象仿照 step2 建立静态链表，执行 step3.1，step3.2，进行第 $m + 1$ 趟分配和收集。

step6. 逐个比较 step5 得到对象序列中相邻对象，去掉重复对象得到化简后的 U'_2。

step7. 按 step1 中保留替换规则，恢复 U'_1、U'_2 中数据。

step8. 得到 $T' = (U', C, D)$ 为新简化决策表，其中 $U' = U'_1 + U'_2$。

注：$u(c)_i$ 表示 u_i 的条件属性值集，$u(c)'_i \neq u(c)'_{i+1}$ 即 2 个对象的条件属性值不完全相同。

step1、step4、step5、step6 时间复杂度 $O(|C||U|)$，step2、step3、step5 时间复杂度 $O(|U|)$，所以 NSDT 算法的时间复杂度为 $O(|C||U|)$。

以决策表 T 为例说明新简化决策表定义。定义 3 计算得 T 的新俭化决策表样本集 $U' = \{u_5, u_1, u_2, u_3, u_4\}$，其中 $U'_1 = \{u_5\}$，$U'_2 = \{u_1, u_2, u_3, u_4\}$，有效地去除了决策表中的冗余对象 u_6，保留了 u_2、u_4。计算新简化决策表 T_N（见表 3）的核和约简均为 $\{a\}$，与决策表 T 的核和约简相同，说明新简化决策表 T_N 没有改变决策表 T 的信息。

表 3 T 的新简化决策表 T_N

$U \setminus A$	a	b	d
5(或 6)	0	0	0
1	2	0	0
2	2	0	1
3	2	1	0
4	2	1	1

以 T 为例说明 NSDT 算法。由 step1 得 N_a、N_b、N_d 分别为 2、2、2，对应替换规则见表 4。

表 4 替换规则

属性	a	a	b	b	d	d
T 中属性值	0	2	0	1	0	1
替换值	1	2	1	2	1	2

由 step2 得到 head $\to u_1 \to u_2 \to u_3 \to u_4 \to u_5 \to u_6$。

由 step3 得：

第 1 趟‘分配’结果：front[0] $\to u_1 \to u_2 \to u_3 \to u_4 \leftarrow$ end[0]

front[1] $\to u_5 \to u_6 \leftarrow$ end[1]

第 1 趟‘收集’结果：$\to u_1 \to u_2 \to u_3 \to u_4 \to u_5 \to u_6$

第 2 趟‘分配’结果：front[0] $\to u_1 \to u_2 \to u_5 \to u_6 \leftarrow$ end[0]

front[1] $\to u_3 \to u_4 \leftarrow$ end[1]

第 2 趟‘收集’结果：$\to u_1 \to u_2 \to u_5 \to u_6 \to u_3 \to u_4$

由 step4 得 $U'_1 = \{u_6\}$，$U'_2 = \{u_1, u_2, u_3, u_4\}$；由 step5 得 $U'_2 = \{u_1, u_3, u_2, u_4\}$；由于 U'_2 中没有重复对象，所以 step6 没有改变 U'_2 的结果；由 step7 得 $u_1 = (2,0,0)$，$u_2 = (2,0,1)$，$u_3 = (2,1,0)$，$u_4 = (2,1,1)$，$u_6 = (0,0,0)$；step8 得 $T = \{u_6, u_1, u_3, u_2, u_4\}$，与用定义得到结果是一致的（决策表中对象顺序可以任意交换；$u5$、$u6$ 是完全一样的对象）。所以用定义和 NSDT 算法得到的新简化决策表相同。

用求核定义求 T_N 的核和约简易得核为 $\{a\}$ 约简

（下转第 181 页）

映射文档在应用程序启动时编译,它可为Hibernate提供关于持久对象的相应类、它们的结构、它们应该映射到哪个数据库表格、哪个字段以及如何映射的信息。

3.4 进行DAO的定义

通过使用接口的编程方法,可以在方法的调用者和方法的具体实现之间直接建立一个屏障,即他们不存在任何的关联,这样,当方法的调用者发生改变时,不影响方法的具体实现;反之,当方法的具体实现发生改变时,不影响方法的调用。

本功能模块中的DAO为:

```
public interface S_PostalPreOrderDao extends Dao {
public Serializable save(S_PostalPreOrder s_PostalPreOrderObject)
throws DaoException;
public S_PostalPreOrder load(S_PostalPreOrderPK pk) throws
DaoException;
public void saveOrUpdate (S_PostalPreOrder s_PostalPre-
OrderObject)throws DaoException;
.....
}
```

3.5 业务逻辑层和表示层的配置

在本系统的各个功能模块中,逻辑层和表示层因为使用了Hibernate实现数据持久层而轻松了许多,对数据库的操作变成了对某个对象的操作,对于对单个对象的增加、删除、修改、更新、查找操作,基本不涉及SQL语句,而对于多个对象之间的级联操作,也可以方便地解决。

如下代码所示,简洁方便地完成了插入数据的操作,并且不用关心底层数据库的结构。

```
S_PostalPreOrder preorder=new S_PostalPreOrder ();
  preorder.setServiceNo ("9999");
  preorder.setCustomerName ("wanglin");
  S_PostalPreOrderDao preOrderDao=DaoFactory.getDaoFactory
().getS_PostalPreOrderDao();
  preOrderDao.save(preorder)
```

基于Hibernate的邮政11185系统的实现,由于使用了对象-关系映射技术,缩短了程序开发的周期,并且程序的可靠性得到了提高。

4 结束语

对象-关系映射技术是采用面向对象的方法技术来设计应用系统,通过将对象和数据库中的表之间建立的映射关系,来关注数据的实际存储,即使后台数据库发生变化,也不会导致对应用程序的修改。

文中通过研究持久层的各种解决方案,对邮政11185系统提出了基于Hibernate的持久层解决方案。在介绍Hibernate技术基础上,结合邮政11185系统的例子,具体说明了如何运用Hibernate实现系统持久层的设计。

参考文献:

[1] 夏　昕,曹晓钢,唐　勇.深入浅出Hibernate[M].北京:电子工业出版社,2005.

[2] Adatia R,Arni F,Gabhart K.EJB编程指南[M].北京:电子工业出版社,2002.

[3] Tyagi S,McCammon K.JDO核心技术[M].北京:清华大学出版社,2005.

[4] 陈天河.Hibernate项目开发宝典[M].北京:电子工业出版社,2006.

[5] King G, Andersen M R, Bernard E, et al. Hibernate Core for Java[EB/OL].2007-04-02.http://www.hibernate.org/344.html.

(上接第120页)

为{a},与T的核和约简相同,可见新简化决策表定义和NSDT算法没有改变决策表表达的信息,所以也没有改变决策表的核和约简。

4 结　语

文献[5]提出的简化决策表,从减小决策表的空间入手,有效地降低了求决策表核和约简算法的空间、时间复杂度。但是由于在化简过程中对不一致问题考虑不完全,造成了丢失信息并且导致简化决策表的核和约简与决策表不同。文中分析了产生错误的原因,并且在保留简化决策表优点的前提下,修正了错误,给出了新简化决策表定义及建立算法。使用新简化决策表定义或NSDT算法,不用对原来的基于决策表或基于简化决策表的算法进行改动即可在新简化决策表下使用并得到正确结果。

参考文献:

[1] 王希雷,王　磊,马　涛,等.二进制区分的简化方法[J].微机发展,2003,13(6):101-103.

[2] 王希雷,王　磊,马　涛,等.一种高效属性约简算法[J].微机发展,2002,12(增刊):12-15.

[3] 王希雷,王　磊.粗集中区分矩阵对不一致问题处理的研究[J].微机发展,2003,13(2):119-120.

[4] 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.

[5] 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.