

# 基于改进的差别矩阵的快速属性约简算法

钱进,叶飞跃,徐亚平

QIAN Jin, YE Fei-yue, XU Ya-ping

江苏技术师范学院 计算机科学与工程学院,江苏 常州 213001

Institute of Computer Science and Engineering, Jiangsu Teachers University of Technology, Changzhou, Jiangsu 213001, China

E-mail: qjqjlqyf@163.com

**QIAN Jin, YE Fei-yue, XU Ya-ping. Fast attribute reduction algorithm based on improved discernibility matrices. Computer Engineering and Applications, 2008, 44(21): 102-105.**

**Abstract:** In order to solve the efficiency problem of calculating the attribute reduction based on discernibility matrices, a new algorithm based on counting sorting for computing U/C is provided, and its complexity is cut down to  $O(|C||U|)$ . Then, the shortcomings of attribution reduction algorithm are analyzed based on discernibility matrices, and the definition of the improved discernibility matrices is presented. Furthermore, core attribute computed by a fast computing core algorithm and an attribute with more frequencies can be used to generate smaller class feature matrices. So a new algorithm based on the improved class feature matrices is proposed, whose worst time complexity and space complexity are cut down to  $\max(O(|C|^2 \sum_{0 \leq i < j \leq k} |D_i||D_j|), O(|C||U|))$  and

$\max(O(|C| \sum_{0 \leq i < j \leq k} |D_i||D_j|), O(|U|))$  respectively. An example is used to illustrate the efficiency of the new algorithm. Experiments show that the new algorithm is efficient for various kinds of data sets.

**Key words:** rough set; attribute reduction; discernibility matrices; core attribute

**摘要:**为了解决基于差别矩阵属性约简的计算效率问题,首先以计数排序的思想设计了一个新的计算 U/C 的高效算法,其时间复杂度降为  $O(|C||U|)$ 。其次分析了基于差别矩阵的属性约简算法的不足,提出了改进的差别矩阵的定义,利用快速计算核属性算法生成的核属性和出现频率最多的属性来降低差别矩阵的大小,并设计了基于改进的差别矩阵的快速属性约简算法,证明了该新

算法的时间复杂度和空间复杂度分别被降为  $\max(O(|C|^2 \sum_{0 \leq i < j \leq k} |D_i||D_j|), O(|C||U|))$  和  $\max(O(|C| \sum_{0 \leq i < j \leq k} |D_i||D_j|), O(|U|))$ 。最后,实例

分析与实验结果表明了新算法具有高效性。

**关键词:**粗糙集;属性约简;差别矩阵;核属性

**DOI:** 10.3778/j.issn.1002-8331.2008.21.028 **文章编号:** 1002-8331(2008)21-0102-04 **文献标识码:** A **中图分类号:** TP18

## 1 引言

粗糙集理论<sup>[1]</sup>是由波兰科学家 Pawlak 教授提出的一种研究不精确、不确定性知识的数学工具。目前,它正被广泛地应用于机器学习、数据挖掘及模式识别等多个领域。在粗糙集理论中,属性约简是重要研究内容之一,也是知识获取的关键步骤。它反映了一个决策表的本质属性。研究人员发现,如果删除冗余属性,则大大提高系统潜在知识的清晰度。因此,研究如何快速提高属性约简算法效率<sup>[2-8]</sup>是十分重要的工作。

影响基于差别矩阵的属性约简算法效率的主要因素有两个:(1)计算 U/C 等价类;(2)差别矩阵的大小。为了降低基于差别矩阵的属性约简算法的时间复杂度和空间复杂度,经过研究,首先,以计数排序的思想设计了一个求 U/C 算法,其时间复

杂度降为  $O(|C||U|)$ ;其次,为降低差别矩阵的大小,提出了改进的差别矩阵,采用了时间复杂度为  $O(|C||U|)$  的快速求核算法来计算核属性,利用核属性和出现频率最多的属性来降低差别矩阵的大小。实验结果表明新算法不仅具有高效性而且能够处理大型决策表。

## 2 基本概念

限于篇幅,下面仅介绍主要用到的一些 Rough 集的概念,详细的请参考有关文献<sup>[1, 8]</sup>。

**定义 1** 设  $X \subseteq U, R \subseteq C, X$  的关于  $R$  的下近似为  $\underline{R}(X) = \cup \{x \in U : [x]_R \subseteq X\}$ 。其中:  $[x]_R$  表示  $U$  中所有与  $x$  在关于  $IND(R)$  下由等价的元素构成的集合。

**基金项目:**江苏技术师范学院青年项目资金(No.KYY07030)。

**作者简介:**钱进(1975-),男,讲师,主要研究领域为数据挖掘,粗糙集,CCF 会员;叶飞跃(1961-),男,博士,教授,主要研究领域为数据挖掘;徐亚平(1957-),男,副教授,主要研究领域为计算方法。

**收稿日期:** 2008-04-30 **修回日期:** 2008-05-30

**定义 2** 在决策表  $S=\langle U, C, D, V, f \rangle$  中, 设  $U/D=\{D_1, D_2, \dots, D_k\}$  表示由决策属性集  $D$  对论域  $U$  的划分,  $U/C=\{C_1, C_2, \dots, C_m\}$  表示由条件属性集  $C$  对论域  $U$  的划分, 其中  $C_i(i=1, 2, \dots, m)$  称为基本块, 称  $POS_C(D)=\bigcup_{D_i \in U/D} C(D_i)$  为  $C$  关于  $D$  的正区域。

**定义 3** 在决策表  $S=\langle U, C, D, V, f \rangle$  中,  $c \in C$ , 若  $POS_{C-c}(D) \neq POS_C(D)$ , 则称属性  $c$  在  $C$  中是不可缺少的;  $C$  的所有不可缺少的属性的集合称为  $C$  的核 (简称核), 记为  $Core(C)$ 。

**定义 4** 在  $S=\langle U, C, D, V, f \rangle$  中,  $S=(\bigcup_{pos} U_{pos} \cup \bigcup_{neg} U_{neg}, C, D, V, f)$  为简化的决策表。定义简化差别矩阵为:  $M=(m_{ij})$ , 矩阵的元素定义如下:  $m_{ij}=\{a|a \in C, f(x_i, a) \neq f(x_j, a), \text{当 } x_i, x_j, \text{只有一个在 } U_{pos} \text{ 中}; f(x_i, a) \neq f(x_j, a) \text{ 且 } f(x_i, D) \neq f(x_j, D), \text{当 } x_i, x_j \text{ 都在 } U_{pos} \text{ 中}\}$ ; 否则  $m_{ij}=\emptyset$ 。

### 3 快速计算 U/C 算法

为了将相容决策表和不一致决策表的属性约简过程一致起来, 将所有矛盾的对象归为一类  $\underline{C}D_0$ , 划分  $\{\underline{C}D_0, \underline{C}D_1, \dots, \underline{C}D_k\}$  则既将属于不同决策类的相容对象分开, 又将相容对象与矛盾对象分开, 这样的不一致决策表就可以看成“相容”决策表了, 而相容决策表不过是不一致决策表的“特例”,  $\underline{C}D_0=\emptyset$ 。同时, 为了区分不同决策类, 将划分  $\underline{C}D_0$  中所有不相容对象的决策属性值修改为  $\text{Max}(V_D)+1$ 。

基于差别矩阵的属性约简算法需要计算  $U/C$ 。对目前计算  $IND(C)$  的方法进行了深入研究, 利用计数排序思想对  $U$  集合进行排序, 给出了一个快速计算  $IND(C)$  的算法, 其时间复杂度为  $O(|C||U|)$ 。

算法 1: 计算  $IND(C)$

输入: 决策表  $S=\langle U, C, D, V, f \rangle, U=\{x_1, x_2, \dots, x_n\}, C=\{c_1, c_2, \dots, c_s\}$

输出:  $U/C$

1. 对每一个  $c_i(i=1, 2, \dots, s)$  统计  $f(x_j, c_i)(j=1, 2, \dots, n)$  的最大值和最小值, 分别记为  $M_i$  和  $m_i$ 。将数组  $Order[n]$  分别初始化为  $1, \dots, n$ 。  
 $TempOrder[n]$  全部初始化为 0。

2. for( $i=1; i \leq s+1; i++$ )

2.1 for( $k=0; k < M_i - m_i + 1; k++$ )

$C[k]=0$ ; //  $C[k]$  存放条件属性  $c_i$  的属性值为  $m_i+k$  的对象个数

2.2 for( $j=1; j < n+1; j++$ )

$\{C[f(Order[x_j], c_i) - m_i]++$ ; // 对条件属性  $c_i$  的不同属性值进行

计数

$TempOrder[j]=Order[j]$ ; //  $TempOrder$  存放临时排序结果

2.3 for( $k=1; k < M_i - m_i + 1; k++$ )

$C[k]=C[k-1]+C[k]$ ; // 确定不同属性值的最后一个对象的位置

2.4 for( $j=n; j > 0; j--$ )

$\{k=f(TempOrder[x_j], c_i) - m_i$ ;

$Order[C[k]]=TempOrder[j]$ ;

$C[k]--$ ;

3. 设由步 2 得到的对象序列为  $x_1, x_2, \dots, x_n; t=1; B_t=\{x_1\}$ ;

for( $j=2; j < n+1; j++$ )

若任一  $c_i \in C(i=1, 2, \dots, s)$  均有  $f(x_j, c_i) \neq f(x_{j-1}, c_i)$ , 则  $B_t=B_t \cup \{x_j\}$ ;

否则  $\{t=t+1; B_t=\{x_j\}\}$ ;

4.  $U'=\emptyset$ ;

for( $i=1; i < t+1; i++$ )

若  $B_i$  中所有对象的决策属性值均相同, 则取第一个对象并入  $U'$  中;

否则修改第一个对象的决策属性值为  $\text{Max}(V_D)+1$ , 并入  $U'$ 。

(1) 分析算法 1 的时间复杂度: 经过简单分析, 算法 1 的时间复杂度为  $O(|C||U| + \sum_{i=1}^{|C|} (M_i - m_i + 1))$ 。在大多数情况下, 常有

$\max_{1 \leq i \leq |C|} (M_i - m_i + 1) < |U|$ , 故  $|C||U| + \sum_{i=1}^{|C|} (M_i - m_i + 1) \leq |C||U| + |C||U|$ , 从而

算法 1 的时间复杂度为  $O(|C||U|)$ 。算法的空间复杂度为  $O(|U|)$ 。

(2) 以决策表 1 为例说明该算法

对决策表 1 的 10 个对象  $U$ , 用算法 1 计算  $U/\{a, b, c, d, e\}$  过程如下:

由算法 1 的步 1 计算出  $M_1=1, m_1=0; M_2=1, m_2=0; M_3=1, m_3=0; M_4=1, m_4=0; M_5=1, m_5=0$ 。  $Order[1]=1, Order[2]=2, \dots, Order[10]=10$ 。

算法 1 的步 2 的第 1 次循环: 步 2.1 和步 2.2 得到  $C[0]=6, C[1]=4$ ; 步 2.3 得到  $C[0]=6, C[1]=10$ ; 步 2.4 从后向前确定各对象的位置。  $a$  属性值为 1 的有 4 个对象 3, 6, 8, 9, 所以  $Order[10]=9, Order[9]=8, Order[8]=6, Order[7]=3$ 。  $a$  属性值为 0 的有 6 个对象, 其对象的顺序如下:  $Order[6]=10, Order[5]=7, Order[4]=5, Order[3]=4, Order[2]=2, Order[1]=1$ 。最后得到临时对象顺序为  $\{1, 2, 4, 5, 7, 10, 3, 6, 8, 9\}$ 。第 2, 3, 4, 5 次循环分别依  $b, c, e, f$  属性进行排序, 过程与第 1 次循环类似。

由第 3 步得到等价类为  $\{\{x_6\}, \{x_2, x_4\}, \{x_1\}, \{x_5\}, \{x_8\}, \{x_3, x_9\}, \{x_7, x_{10}\}\}$ 。

由第 4 步得到  $U'$  为  $\{x_6, x_2, x_1, x_5, x_8, x_3, x_7\}$ , 其中  $x_2, x_3, x_7$  为矛盾对象, 将这样的不相容对象归为一类, 修改其决策属性值为 2。

表 1 决策表

$U$	$a$	$b$	$c$	$e$	$f$	$D$
$x_1$	0	0	0	0	1	0
$x_2$	0	1	1	1	0	1
$x_3$	1	1	0	1	1	1
$x_4$	0	1	1	1	0	0
$x_5$	0	0	1	0	1	0
$x_6$	1	1	0	1	0	1
$x_7$	0	1	1	1	1	1
$x_8$	1	1	1	0	1	1
$x_9$	1	1	0	1	1	0
$x_{10}$	0	1	1	1	1	0

## 4 改进的差别矩阵及其快速属性约简算法

### 4.1 改进的差别矩阵定义

文献[7, 8]都对差别矩阵进行了改进, 但给出的差别矩阵里面仍然存在许多空集元素, 这显然浪费了大量存储空间。为了构造改进的差别矩阵, 首先将简化决策表中的对象按其决策属性进行排序。如果有多个决策属性, 则可将多个决策属性看成一个决策属性, 多个决策属性组合值为该属性的取值范围。不失一般性, 假设不一致决策表  $S$  仅有一个决策属性  $d$ , 其决策属性值映射为  $1, 2, \dots, k$ 。按算法 1 处理后, 共有  $k+1$  类对象, 其决策属性值为  $k+1, 1, 2, \dots, k$ 。这样将整个决策表分为  $k+1$  个子决策表  $D_0, D_1, \dots, D_k$ , 每个子决策表中包含同一决策属性值的对象, 其对象个数分别为  $|D_0|, |D_1|, \dots, |D_k|$  (如果  $S$  为相容决策表,  $|D_0|$  为 0)。

**定义 5** (类间差别矩阵)  $D_i$  中对象与  $D_j$  中对象关于属性  $C$  的类间差别矩阵  $M_{ij}^C$  的每个元素定义为:  $C_{ij}=\delta_1, \delta_2, \dots, \delta_s$ 。其中:

$$\delta_l = \begin{cases} 0 & f(x_i, c_l) = f(x_j, c_l) \\ 1 & f(x_i, c_l) \neq f(x_j, c_l) \end{cases}, \text{其中 } l=1, 2, \dots, s, x_i \in D_i, x_j \in D_j$$

从定义5可以看出,关于条件属性C的类间差别矩阵删除了差别矩阵中大量无用的空集元素,对于多数大容量的决策表,类间差别矩阵占用了较小的存储空间,且容易分析各属性的重要性。

**性质1** 对决策表S,设  $C=(c_1, c_2, \dots, c_s)$ ,下列等式成立:

$$M_{ij}^C = M_{ij}^{c_1} (+) M_{ij}^{c_2} (+) \dots (+) M_{ij}^{c_s}$$

证明:由定义5直接证得。

**定义6** 关于属性C的  $D_i$ 中对象的差别矩阵  $M_i^C$ 定义为:

$$M_i^C = \cup M_{ij}^C, \text{其中 } j=i+1, \dots, k。$$

**定义7** (改进的差别矩阵)对给定的决策表S,定义改进的差别矩阵  $M^C$ 为:  $M^C = \bigcup_{i=0}^{k-1} M_i^C$ 。

根据定义7,  $M^C$ 则包含  $\sum_{i=0}^{k-1} \sum_{j=i+1}^k |D_i||D_j|$ 个元素。当等价类的个数都为1,则空间复杂度最坏为  $O(n^2)$ 。通常情况,  $M^C$ 比文献[7,8]中  $M^C$ 少  $\sum_{0 \leq i < j \leq k} |D_i||D_j|$ 个元素。

### 4.2 快速属性约简算法

为了降低差别矩阵中元素个数,文献[8]采用任选一个属性来降低差别矩阵的大小。而核属性是属性约简中不可缺少的属性,那么,能否利用核属性来生成较小的差别矩阵成为一个重要的研究内容。

若要判断  $c_i \in C(i=1, 2, \dots, s)$ 是否是核属性,就要计算  $|POS_C(D)|$ 与  $|POS_{C-\{c_i\}}(D)|$ 之间的差值。用计数排序算法计算  $POS_{C-\{c_i\}}(D)$ 、 $POS_C(D)$ 的时间复杂度均为  $O(|C||U|)$ 。但计算所有  $c_i \in C$ 的时间复杂度为  $O(|C|^2|U|)$ ,这并不是所期望的。在计算  $|POS_{C-\{c_i\}}(D)|$ 时,要对U中的对象依  $c_1, c_2, \dots, c_{i-1}, c_{i+1}, \dots, c_s$ 进行排序。计算  $|POS_{C-\{c_{i+1}\}}(D)|$ 时,要对U中的对象依  $c_1, c_2, \dots, c_i, c_{i+2}, \dots, c_s$ 进行排序。在这两次排序中可以发现,有  $s-2$ 次排序操作是重复的,只有一次操作与之不同,所以可以依  $c_1, c_2, \dots, c_i, c_{i+2}, \dots, c_s$ 进行排序时,可以从上次排列的结果出发,只要对  $c_i$ 进行一次排序就可以得到本次的排序结果,从而对所有  $c_i \in C$ 进行排序的时间复杂度仅为  $O(|C||U|)$ 。

用  $Order(U, R)$ 表示U中的对象按  $R(R \subseteq C)$ 的条件属性依次排序后的结果。根据计数排序算法思想和文献[9],同样得到定理1和性质2。

**定理1**  $Order(U, C-\{c_{i+1}\})$ 等价于  $Order(U, C-\{c_i\}) \cdot Order(U, \{c_i\})$ ,其中  $c_i \in C$ 。

**性质2**  $Order(U, C-\{c_i\})$ 等价于  $Order(U, C)$ 。

算法2:快速计算属性约简

输入:决策表S

输出:属性约简 Reduct

**步骤1** 计算  $U/C$  等价类,得到  $Order(U, C)$ ,并且计算  $|POS_C(D)|$ ;

**步骤2** 根据性质2,计算  $|POS_{C-\{c_i\}}(D)|$ ,如果  $|POS_{C-\{c_i\}}(D)| < |POS_C(D)|$ ,则  $Core(C) = Core(C) \cup \{c_i\}$ ;

**步骤3** 根据定理1,依次计算  $Order(U, C-\{c_i\})(i=2, \dots,$

$s)$ ,并且分别计算  $|POS_{C-\{c_i\}}(D)|$ ,如果  $|POS_{C-\{c_i\}}(D)| < |POS_C(D)|$ ,则  $Core(C) = Core(C) \cup \{c_i\}$ ;

**步骤4**  $Reduct = Reduct \cup Core(C)$ ,利用核属性计算出出现频率最多的属性  $c'$ ,  $Reduct = Reduct \cup \{c'\}$ ,根据  $Reduct$  生成  $M^{C-Reduct}$ ;

**步骤5** 计算  $M^{C-Reduct}$  中属性出现频率,选择出现频率最多的属性  $c$ ,  $Reduct = Reduct \cup \{c\}$ ;

**步骤6**  $DelElement(M^{C-Reduct}, c)$ ;

**步骤7** 如果  $M^{C-Reduct}$  为空,则转步骤8;否则转步骤5;

**步骤8** 输出属性约简 Reduct。

(1)算法正确性分析

算法2的第1、2、3步生成核属性,第4步生成  $M^{C-Reduct}$ ,即在  $M^C$ 中删除所有包含属性 Reduct 的元素后剩余的差别矩阵,但  $M^{C-Reduct}$  比  $M^C$ 小很多。第5步统计出现频率最多的属性  $c$ ,第6步是在差别矩阵中删除包含属性  $c$  的元素。故该算法是与基于差别矩阵的属性约简算法是等价的,从而算法是正确的。

(2)算法的复杂度分析

经过简单分析,算法2的最坏时间复杂度为  $\max(O(|C|^2$

$$\sum_{0 \leq i < j \leq k} |D_i||D_j|))。因为 \cup_{1 \leq i \leq k} D_i = U'_{pos}, D_0 = U'_{neg}, |U'| = |U/C| = |U'_{pos}| + |U'_{neg}|, \sum_{0 \leq i < j \leq k} |D_i||D_j| = U'_{pos}(U'_{pos} + U'_{neg}) - \sum_{1 \leq i \leq k} |D_i|^2 < U'_{pos}(U'_{pos} + U'_{neg}),$$

故算法2的时间复杂度比文献[8]中基于简化差别矩阵的时间复杂度  $O(|C|^2(|U'_{pos}| |U/C|))$ 要好。算法2的空间复杂度为  $O(|C|$

$$\sum_{0 \leq i < j \leq k} |D_i||D_j|)$$

,也比基于简化差别矩阵的时间复杂度  $O(|C|(|U'_{pos}| |U/C|))$ 要好。本文给出的属性约简算法总的最坏时间复杂度是

$$\max(O(|C|^2 \sum_{0 \leq i < j \leq k} |D_i||D_j|), O(|C||U|))$$

$$\max(O(|C| \sum_{0 \leq i < j \leq k} |D_i||D_j|), O(|U|))。$$

### 5 实例分析与实验结果

用决策表1为例说明新算法。利用算法1对不一致决策表进行处理,得到简化决策表  $U = \{x6, x1, x5, x8, x2, x3, x7\}$ 。利用算法2对U进行计算,利用核属性f和出现频率最多的属性e生成的较小差别矩阵仅为3。而文献[7]中却包含28个元素  $(4 \times (4+3) = 28, U'_{pos} = 4, U'_{neg} = 3)$ ,文献[8]采用任选一属性,如选a,需包含6个元素,多余的元素主要是同一的相容对象生成的空集元素。通过计算,得到  $Reduct = \{f, e, a\}$ 。

为了进一步考察所提出的算法,选用UCI机器学习数据库中的部分决策表,在PC(Core™ 2 Duo CPU 2.0 G, 1.99 GB, Windows XP)上进行实验,编写了差别矩阵算法(简称算法A)、基于文献[8]中随机选择一个属性的差别矩阵算法(简称算法B)和本文算法(算法2)进行知识约简,算法A、B都采用算法1计算划分,算法B采用随机选择一个属性(表2中执行时间为分别选择5张表的第2个、第11个、第9个、第18个、第13个属性,并使用.NET平台下C#语言实现,其运行结果如表2所示。为了比较差别矩阵大小,引用文献[10]中蒸发率的概念。令原始差别矩阵大小为  $M_A$ ,采用新算法后生成的差别矩阵大小为  $M_B$ ,则差别矩阵蒸发率  $= (1 - M_B/M_A) * 100\%$ 。其中用  $N_{B/A}$ 表示



表2 约简结果、执行时间和差别矩阵蒸发率比较

数据集	实例数	条件个数	算法 A		算法 B			算法 2		
			约简结果	执行时间/s	约简结果	执行时间/s	$N_{red}/\%$	约简结果	执行时间/s	$N_{red}/\%$
表 1	10	4	3	0.031 250	4	0.015 625	62.5	3	0.015 625	81.3
Iris	150	4	4	0.015 625	4	0.015 625	64.2	4	0.015 625	96.2
Tic-tac-toe	958	9	8	0.625 000	8	0.265 625	70.0	8	0.015 625	88.4
Chess end-game	3 196	36	29	26.437 500	29	32.781 250	43.9	29	11.468 750	70.0
mushroom	8 124	22	4	95.031 250	5	33.390 625	65.9	4	6.328 125	99.4

算法 B 中差别矩阵对算法 A 中差别矩阵的蒸发率,  $N_{red}$  表示算法 2 中差别矩阵对算法 A 中差别矩阵的蒸发率。从表 2 可以看出, 算法 2 约简后条件属性数比算法 A 和算法 B 约简后条件属性数更少, 运行时间更短。同时, 差别矩阵包含的元素更少。实验结果与前面分析相符合, 验证了算法 2 的正确性和高效性。

## 6 结论

本文研究了基于差别矩阵属性约简算法计算效率问题, 设计了计数排序算法, 提高了计算 U/C 等价类效率, 其时间复杂度降为  $O(|C||U|)$ 。针对差别矩阵中存在空集元素, 提出了改进的差别矩阵定义, 并采用快速求核算法计算出核属性和出现频率最多的属性来进一步降低差别矩阵大小。实验结果表明该算法是一个高效的算法。作者下一步将研究无核的决策表如何提高属性约简效率。

## 参考文献:

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

- [2] 刘少辉, 盛秋骥, 吴斌, 等. Rough 集高效算法研究[J]. 计算机学报, 2003, 26(5): 524-529.
- [3] 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.
- [4] Hu X H, Cercone N. Learning in relational databases: a rough set approach[J]. International Journal of Computational Intelligence, 1995, 11(2): 323-338.
- [5] Wang Jue, Wang Ju. Reduction algorithms based on discernibility matrix: the ordered attributes method[J]. Journal of Computer Science and Technology, 2001, 16(6): 489-504.
- [6] 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.
- [7] 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.
- [8] 高学东, 丁军. 基于简化差别矩阵的属性约简算法[J]. 系统工程理论与实践, 2006(6): 101-107.
- [9] 周江卫, 冯博琴, 刘洋. 一种新的快速求核算法[J]. 西安交通大学学报, 2007, 41(6): 688-691.
- [10] 王珏, 王任, 苗夺谦, 等. 基于 Rough Set 理论的“数据浓缩”[J]. 计算机学报, 1998, 21(5): 393-400.

(上接 48 页)

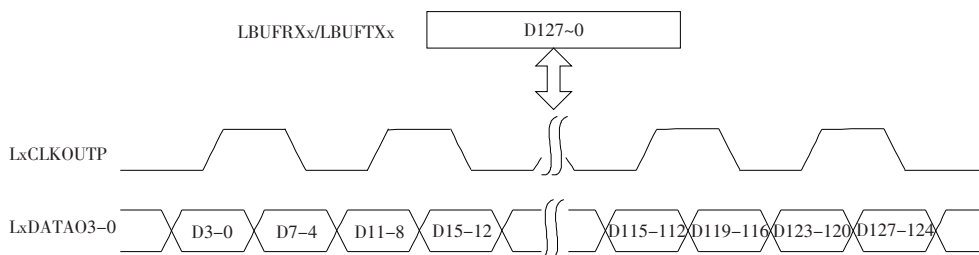


图8 4位模式链路通信协议

用来配合接收模块和发送模块作为传输时的数据缓冲区, 并且实现与系统中其他接口或者 FPGA 中其他模块的接口的数据传输功能。

## 4 结论

本文设计了一种以 FPGA 芯片为核心处理器的多通道高速 CMOS 图像采集系统, 实现了对全分辨率下帧频高达 450 帧/s 的 1 280×1 024 像素阵列的高速图像采集、数据传输和存储。LVDS 技术和乒乓操作方式在数据传输和存储过程中的应用大大提高了整个系统的可靠性和集成度, 降低了噪声对于整个系统数据传输过程的影响。同时, 通过 FPGA 芯片实现的整个系统的时序控制、数据转换和存储功能, 具有较高的系统可

重构性。系统已经应用于某高速视觉测量系统中, 实际应用证明, 具有良好的数据采集和传输能力, 能够实现对空间高速运动目标的实时图像采集、传输和存储功能。

## 参考文献:

- [1] 尚玉全, 曾云, 滕涛, 等. CMOS 图像传感器及其研究[J]. 半导体技术, 2004, 29(5): 19-24.
- [2] 周如辉. 实时视频处理系统的乒乓缓存控制器设计[J]. 单片机与嵌入式系统应用, 2006(7): 25-27.
- [3] 王智, 罗新民. 基于乒乓操作的异步 FIFO 设计及 VHDL 实现[J]. 电子工程师, 2005, 31(6): 13-16.
- [4] 甘旭军, 李平康, 杜秀霞. 多 DSP 并列高速信号处理器的设计与实现[J]. 仪器仪表学报, 2002, 23(3): 574-576.