文章编号：2096-1618(2019)02-0112-06

基于二进制链表的粗糙集属性约简

宋 剑，蒋 瑜，李 冬，鲍杨婉莹
(成都信息工程大学软件工程学院，四川 成都 610225)

摘要：差别矩阵是很多学者用来计算粗糙集属性约简的一种方法，该方法因其简单、直观、易于理解而得到广泛应用，但是包含在差别矩阵中的冗余元素不仅对属性约简不起作用反而增加存储空间，为消除这些冗余元素提出了一种新的存储结构：二进制链表，通过位运算将差别矩阵中所有的重复元素和父集元素删除，降低差别信息的存储空间。为验证二进制链表的有效性，提出了一种新的属性约简算法。通过UCI数据库中多组数据集对该方法进行测试，并将实验结果与其他算法进行比较，提出的算法可以更快地得到属性约简集并且能够有效地降低存储空间。

关 键 词：粗糙集；属性约简；差别矩阵；二进制链表；位运算
中图分类号：TP18 **文献标志码**：A
doi：10.16836/j.cnki.jcuit.2019.02.002

0 引言

波兰数学家Z. Pawlak在20世纪80年代提出了粗糙集理论[1]，属性约简是该理论的重要组成部分，在保持决策系统分类精度的同时删除系统中不必要的条件属性从而达到降维的目的。文献[2]指出："学者们根据不同的理论提出了许多优秀的属性约简算法"，这些方法包括：增量式属性约简算法[3-7]，正区域方法[8]，差别矩阵方法[9-11]等。

差别矩阵是一种简单而直观的属性约简方法，该方法的不足之处是矩阵中会存在大量的冗余元素，这些元素不仅对属性约简没有作用反而增加了存储空间。针对这一缺点，很多学者提出了对差别矩阵的改进算法，将差别矩阵转化成二进制差别矩阵[12-14]或者利用一些数据结构删除差别矩阵中的部分冗余元素如：树形结构[15-18]。文献[19]使用区分链表来存储差别信息，在构建链表时删除核属性的父集元素，在一定程度上降低了空间存储。但该链表仍然存在一些问题，首先链表中仍然存在重复元素，这些元素会浪费存储空间，其次如果在构造链表的过程中不存在核属性则链表存储的差别信息个数与用差别矩阵存储是相同的，文献[20]将链表结构应用在不完备的决策表中，在构建链表过程中仍然存在和文献[19]中算法的不足之处。

针对传统区分链表的不足，构建了一种新的链表结构：二进制链表。用二进制编码表示差别信息元素，在构建链表的过程中删除所有的父集元素。为验证二进制链表的有效性，基于二进制链表提出一种获取属性约简的算法，该算法在每次迭代过程从前往后在二进制链表中选择必要属性，同时删除包含必要属性的结点，并使约简算法的时间复杂度为$O(|C||U|^2)$。

1 基础概念

根据文献[1,10]给出粗糙集理论的相关知识的介绍。

定义1[1] $S=\{U,C,D,V,f\}$表示一个决策表，其中U表示论域的集合，C为条件属性集，D为决策属性集，V是值域，f表示信息函数，表1是一个决策表，$U=\{x_1,x_2,\cdots,x_{10}\}$，$C=\{a,b,c,d\}$，$V=\{0,1,2\}$，$f:U\times(C\cup D)\rightarrow V$。

表1 决策表1

U	a	b	c	d	D
x_1	1	2	0	1	1
x_2	1	2	0	1	1
x_3	2	0	0	1	0
x_4	0	0	1	2	1
x_5	2	1	0	2	1
x_6	0	0	1	2	2
x_7	2	0	0	1	0
x_8	0	1	2	2	1
x_9	2	1	0	2	2
x_{10}	2	0	0	1	0

收稿日期：2018-09-04
基金项目：四川省教育厅重点资助项目(17ZA0071)

定义 2[1]　在决策表 $S=\{U,C,D,V,f\}$ 中,A 是条件属性 C 与决策属性 D 的并集,对于任意属性子集 R,以及 U 的子集 X, X 关于 R 的下近似集用 $\underline{R}(X)$ 来表示,其定义为 $\underline{R}(X)=\cup\{R_i \mid R_i\in U/R, R_i\subseteq X\}$,$U/R=\{R_1,R_2,\cdots,R_n\}$。

定义 3[1]　在决策表 $S=\{U,C,D,V,f\}$ 中,C 关于 D 的正域用 $POS_C(D)$ 来表示,计算公式为 $POS_C(D)=\cup\underline{C}(D_i)(D_i\in U/D)$,其中 $U/D=\{D_1,D_2,\cdots,D_k\}$ 表示决策属性 D 对论域 U 的划分。

定义 4[1]　在决策表 $S=\{U,C,D,V,f\}$ 中,若条件属性集 P 为一个约简集,则对于 P 中任意属性 a,都满足两个条件:$POS_p(D)=POS_C(D)$;$POS_{p-\{a\}}(D)\neq POS_C(D)$。

引理 1[1]　在决策表 $S=\{U,C,D,V,f\}$ 中,条件属性集 C 关于决策属性 D 的正域定义为 $POS_C(D)=\cup(P)$,其中 P 是 U 关于 C 的划分的一个子集并且 $|P/D|=1$。

定义 5[10]　设在决策表 $S=\{U,C,D,V,f\}$ 中,记 $U/C=\{[x_1]',[x_2]',\cdots,[x_m]'\}$,其中 $[x_m]'$ 表示与 x_m 根据 C 划分的等价类集合,S 的简化决策表(表 2)可定义为 $S'=\{U',C,D,V,f\}$,其中 $U'=\{x_1,x_2,\cdots,x_m\}$。

表 2　表 1 对应的简化决策表

U	a	b	c	d	D
x_1	1	2	0	1	1
x_3	2	0	0	1	0
x_4	0	0	1	2	1
x_5	2	1	0	2	1
x_8	0	1	2	2	1

定义 6[10]　简化决策表 $S'=\{U,C,D,V,f\}$ 的简化差别矩阵中的差别信息 m_{ij} 的定义如下:$m_{ij}=\{c \mid c\in C, f(x_i,c)\neq f(x_j,c)\}$,如果不等式 $f(x_i,D)\neq f(x_j,D)$ 成立,那么 x_i 和 x_j 均在正域中,否则 x_i 和 x_j 一个在正域中,一个在负域中。

定义 7[10]　设决策表 S 的简化差别矩阵(表 3)为 $\boldsymbol{M}$,m_{ij} 为矩阵中任意非空差别信息集,设 P 为条件属性集的 C 的子集,如果 $P\cap m_{ij}\neq\varphi$ 并且从 P 中删除任意元素该不等式均不成立,则称属性集 P 是一个约简集。

表 3　表 1 对应的简化差别矩阵

	x_1	x_3	x_4	x_5	x_8
x_1	φ	$\{ab\}$	$\{abcd\}$	$\{abd\}$	φ
x_3		φ	$\{acd\}$	$\{bd\}$	$\{abcd\}$
x_4			φ	$\{abc\}$	$\{bc\}$
x_5				φ	$\{ac\}$
x_8					φ

定义 8[10]　在决策表 S 的简化差别矩阵中,对于矩阵中任意一个差别元素集 A,都存在差别元素集 B,,满足 $B\subseteq A$,那么差别元素 A 被称为无用差别元素。

引理 2[10]　决策表 $S=\{U,C,D,V,f\}$ 的简化差别矩阵 $\boldsymbol{M}$,对于任意 C 的子集 P,如果矩阵中任意非空 m_{ij} 与 P 相交不为空,则有

$$POS_P(D)=POS_C(D)$$

2　区分链表的相关知识

文献[19-20]基于链表思想提出一种能够降低差别矩阵空间复杂度的链表结构(区分链表),差别矩阵中的每个非空且不为核心属性的元素作为一个结点插入链表中,删除链表中包含核心属性的结点,与差别矩阵相比,实现了差别信息的压缩存储。图 1 给出了基于表 1 构建的一个区分链表。

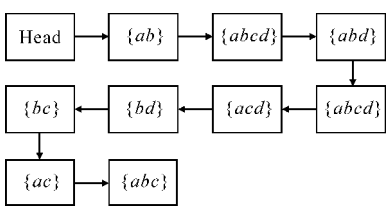

图 1　基于表 1 构建的区分链表

通过表 3 的差别矩阵和图 1 的区分链表的对比可知,该链表并没有实现差别矩阵的非空元素的压缩存储,由于差别矩阵中没有核属性所以无法根据核属性对链表进行简化。根据定义 8 可知结点 $\{a,b,c,d\}$,$\{a,c,d\}$,$\{a,b,d\}$ 都是冗余结点,这些结点不仅增加了存储空间而且也增加了算法的执行时间,因此需要设计新的链表结构来解决这个问题。

3　二进制链表的设计与实现

首先给出二进制链表的定义, 然后提出一种构建二进制链表的算法。

定义 9　二进制链表是一个有序链表,其特点是每个结点都是按照二进制编码的大小,从小到大排列的。

链表的每个结点中存储的只有用二进制编码表示的差别信息,编码的长度均为 $|C|$,差别信息中存在的元素都用“1”来表示,不存在的元素都用“0”来表示且每个编码的位置不可互换。

基于以上二进制链表的定义,二进制链表的构建算法如下所示。

算法 1 二进制链表的创建

输入:决策表

输出:二进制链表

Step1:根据定义 5 对决策表 T 进行简化,得到简化的决策表;

Step2:创建链表的头结点;

Step3:根据定义 6 计算的差别信息,将差别信息转化成二进制编码 B;

Step4:调用 AddNode(B)函数,将结点 node 插入到二进制链表中;

Step5:转到 Step3 直到计算完所有差别信息;

Step6:对二进制链表从小到大排序;

Step7:结束算法。

AddNode(B)函数实现了将编码 B 插入到二进制链表这一有序链表中,其实现步骤如下:

(1)B 中所有位的数字都为 0 则转到(5);

(2)如果链表为空则直接生成结点,并将结点插入到链表中,转到(5);

(3)遍历链表中的每个结点,取出每个结点中的二进制编码 $B1$,对 $B1$ 和 B 进行如下运算:

(i)如果 $B1\&B=B1$ 则转到(5),否则转到(ii);

(ii)如果 $B1\&B=B$ 则删除存储 $B1$ 的结点,否则转到(iii);

(iii)如果遍历结束转到(4),否则选取下一个结点的编码并转到(i);

(4)将存储 B 的结点插入到链表末尾;

(5)函数结束。

基于算法 1 和表 1,给出二进制链表的构建过程:

首先创建链表的头结点,然后开始计算第一条差别信息$\{a,b\}$,其转化成二进制编码为(1100)后插入到链表中,对于第二条差别信息$\{abcd\}$,对应的编码为(1111),因为(1111)&(1100)=(1100)所以不添加到链表中,计算第三条差别信息$\{abd\}$,编码为(1101),因为(1101)&(1100)=(1100)所以不添加到链表中。同理,重复上面构建过程直到计算完最后一条差别信息后对链表进行排序得到如图 2 所示的二进制链表。

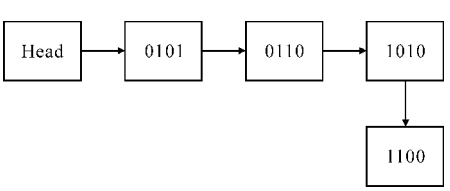

图 2 基于算法 1 和表 1 构建的二进制链表

通过图 2 和图 1 的对比可知,二进制链表删除了表 3 中差别矩阵的重复元素以及父集元素,节省了存储空间,结点中用二进制编码代替字符集合是为了便于位运算,编码中的数字是顺序不可改动的。

3.1 二进制链表的特性

由二进制链表的构建过程可以得到以下特性:

定理 1 二进制链表包含了能够获得属性约简所需要的全部差别信息。

证明 设 DS 为二进制链表中所有结点对应的差别信息的集合,由二进制链表的构建过程可知 DS 包含于 $\boldsymbol{DM}$($\boldsymbol{DM}$ 为差别矩阵),对于任意差别信息 $DI\in\boldsymbol{DM}$ 都能在 DS 集合中找到子集 DK 使得 DK 包含于 DI。由无用差别信息的定义可知 DI 对求属性约简没有用处。所以,二进制链表包含了能够获得属性约简所需要的全部差别信息。

定理 2 二进制链表中每个结点的编码第一个不为 0 的坐标为 i,设 R 为从条件属性集 C 中选取第 i 个属性组成的集合,则 $POS_R(D)=POS_C(D)$。

证明 由定理 1 可知二进制链表包含获得属性约简的全部差别信息,设 DS 为二进制链表中所有结点对应的差别信息的集合,对于 DS 中任意差别信息元素 DI,R 与 DI 相交不为空,从而证明了 $POS_R(D)=POS_C(D)$ 成立。

基于定理 2,只要删除 R 中所有不必要的条件属性,就可以求得决策表的一个属性约简。由图 1 得 $R=\{c,b,a\}$,因为 $POS_{R-\{a\}}(D)=POS_C(D)$ 并且对于新的属性集合 $S=\{b,c\}$,删除任意一个属性所得的正域与原来的正域都不相等,所以 a 为不必要属性应当删除。$\{b,c\}$ 为决策表 1 的一个属性约简。

性质 1 由定理 2 得到的属性集合 R 中,第一个条件属性一定是必要的条件属性。

3.2 二进制链表的复杂度分析

对于一个决策表假设其经过算法 1 中第一步简化后有$|U|$个对象,条件属性仍然为$|C|$个, 则差别矩阵中最多具有$|U|^2$ 个非空差别信息。假设差别矩阵中不存在相互包含关系的差别信息的个数为 N(一般情况 N 远小于$|U|^2$),由二进制链表的构建过程可知,每个结点包含$|C|$个二进制编码,链表中最多包含 N 个结点。在最坏情况下即 N 与$|U|^2$ 相等,二进制链表的空间复杂度为 $O(|C||U|^2)$。

另外,在构建二进制链表的过程中,由于算法最多迭代$|U|^2$ 次,且每次最多插入一个结点,删除 N_i 个结点,二进制链表的时间复杂度为$|U|^2+(N_1+N_2+\cdots+N_{|U|^2})$,因为二进制链表中最多包含$|U|^2$ 个结点,所以

$N_1+N_2+\cdots+N_{|U|^2}$的值最多为$|U|^2$,且每次要与$|C|$个二进制编码比较从而可得二进制链表的时间复杂度为$O(|C||U|^2)$。

4　基于二进制链表的属性约简

为验证文中提出的二进制链表的有效性, 基于二进制链表提出一个属性约简算法。该算法基于定理 2 和性质 1, 在每次迭代过程中从前往后在二进制链表中选择必要属性, 同时删除包含必要属性的结点。算法具体描述如下:

算法 2　在二进制链表上的属性约简的算法

输入:二进制链表

输出:属性约简集 R

Step1:初始化一个空集 R;

Step2:如果链表长度大于 0,转到 Step3,否则算法结束输出约简集 R;

Step3:选择头结点的下一个结点中的编码 B,得到 B 中第一个不为 0 的坐标 i;

Step4:初始化一个长度为$|C|$除第 i 位为 1 其余均为 0 的二进制编码 $B1$;

Step5:取出每个结点的编码 B,若 $B\&B1=B1$ 则删除这个结点,否则取下一个结点的编码直至链表末尾;

Step6:将第 i 个条件属性放入集合 R,转到 Step2。

基于图 2 和算法 2 的求解过程如下:条件属性集 $=\{a,b,c,d\}$,初始化约简集 $R=\varphi$,头结点的下一个结点的编码为(0101),其中第二位为 1,初始化一个二进制编码 $B1=(0100)$,因为$(0101)\&B1=B1$ 所以删除(0101)的结点,$(0110)\&B1=B1$,所以删除(0110)的结点,条件属性集中第二个属性为 b 所以 $R=R\cup\{b\}$。此时,下一个结点的编码为(1010),其中第一位为 1,初始化一个二进制编码 $B1=(1000)$,因为$(1010)\&B1=B1$ 所以删除(1010)的结点,$(1100)\&B1=B1$,所以删除(1100)的结点,条件属性集中第一个属性为 a 所以 $R=R\cup\{a\}$,此时链表长度为 0 算法结束输出约简集 $R=\{a,b\}$。

基于前面的分析,链表中最多有$|U|^2$ 个结点,由算法 2 的求解过程可知,该算法最多迭代$|C|$次,假设每次最多删除 N_i 个结点,则在$|C|$次迭代过程中该算法删除的结点最多为 $N_1+N_2+\cdots+N_{|C|}=|U|^2$,且需要与每个结点的$|C|$位二进制编码比较,所以该算法的时间复杂度为 $O(|C||U|^2)$。

5　实验结果及分析

选用 6 个 UCI 数据集在 Intel(R) Core 2.20 GHz (6 GB内存, Microsoft Windows 7 操作系统)上用 JAVA 语言进行实验来验证二进制链表这一结构的有效性,同时给出了创建区分链表和二进制链表的时空复杂度对比结果,如表 4 所示。

表 4　二进制链表与区分链表的实验结果的比较

数据集名称	论域数目 U	原属性个数 C	链表的结点数		创建链表所用的时间/s	
			二进制链表	区分链表	二进制链表	区分链表
Lenses	24	5	0	0	0	0
zoo	101	16	12	105	0.006	0.011
balance-scale	625	4	0	0	0.045	0.071
Tic-tac-toe	958	9	36	207832	0.127	0.156
car	1728	6	0	0	0.231	0.429
chess	3196	36	2	18	2.566	4.104

由表4 可知,二进制链表中的结点数远小于$|U|^2$,二进制链表的空间复杂度不会大于区分链表的空间复杂度并且从结点数量上看二进制链表的结点数量可以与区分链表的数量相等例如:基于 Lenses, balance-scale 和 car 的数据集二进制链表和区分链表的结点数均为 0 是因为数据集中没有相对必要的属性,所有结点都可以根据核属性删除,除了这 3 个数据集之外,区分链表的结点都比二进制链表的结点多,最少的是基于 zoo 的数据集,其结点数达到了 8 倍多,最多的是基于 Tic-tac-toe 数据集由于该数据集没有核属性,区分链表无法根据核属性对链表进行简化。

再从时间效率上看,二进制链表的创建时间比区分链表少并且随着数据集论域数目的增加,两种链表的创建时间差距越来越大。

此外,结合算法 1 和算法 2,从 UCI 数据集中选取 6 个数据集进行实验,分别给出通过二进制链表,区分链表以及二进制差别矩阵进行约简的运行时间对比,结果如表 5 所示。

表5 基于二进制链表和其他约简算法的运行时间对比

数据集名称	论域数目 U	原属性个数 C	约简后属性个数 R	算法运行时间/s		
				区分链表	文中算法	二进制差别矩阵
zoo	101	16	5	0.288	0.284	0.294
BreadCancer	367	9	7	0.350	0.346	0.352
balance-scale	625	4	4	0.383	0.364	0.386
car	1728	6	6	0.780	0.678	0.85
chess	3196	36	28	6.221	5.657	6.986
letter-recognition	20000	16	12	330.220	190.285	241.664

由表5可知,基于二进制链表的算法比另外两种算法的运行时间要少,同时也可以看出,数据集中记录数目的增加会导致文中算法与另外两种算法的运行时间差距增大。实验结果表明本文算法与基于区分链表以及文献[14]二进制差别矩阵算法相比,具有更小的时间复杂度。

6 结束语

属性约简是研究粗糙集理论的一个重要研究部分,提出一种新的链表结构:二进制链表来消除矩阵中的冗余元素,最后提出一种属性约简算法。下一步的工作是将该算法应用到分类算法中,利用约简后得到的约简集构建分类器,提高分类器泛化能力。

致谢:感谢成都信息工程大学中青年学术带头人科研基金项目(J201609)对本文的支持

参考文献:

[1] Pawlak Z. Rough Sets[J]. International Journal of Computer and In-formation Sciences,1982,11:341-356.

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

[3] Yunge Jing,Tianrui Li,Hamido Fujita,et al. An incremental attribute reduction method for dynamic data mining[J]. Information Sciences,2018,465:202-218.

[4] Yunge Jing, Tianrui Li, Junfu Huang, et al. A Group Incremental Reduction Algorithm with Varying Data Values[J]. International Journal of Intelligent Systems,2017,32(9):900-925.

[5] Yunge Jing,Tianrui Li,Hamido Fujita,et al. An incremental attribute reduction approach based on knowledge granularity with a multi-granulation view [J]. Information Sciences,2017,411:23-38.

[6] Yunge Jing,Tianrui Li,Junfu Huang,et al. An incremental attribute reduction approach based on knowledge granularity under the attribute generalization [J]. International Journal of Approximate Reasoning,2016,76:80-95.

[7] Hongmei Chen,Tianrui Li,Yong Cai,et al. Parallel attribute reduction in dominance-based neighborhood rough set [J]. Information Sciences,2016,373:351-368.

[8] Wen S D,Bao Q H. A fast heuristic attribute reduction approach to ordered decision systems[J]. European Journal of Operational Research,2018,264:440-452.

[9] Skowron A,Rauszer C. The discernibility matrices and functions in information systems[C]. Intelligent Decision Support,Handbook of Applications and Advances of the Rough Sets Theory. Dordrecht,1991:331-362.

[10] 徐章艳,杨炳儒,宋威. 基于简化差别矩阵的完备属性约简算法[J]. 计算机工程与应用,2006,26(3):167-169.

[11] 周建华,徐章艳,章晨光. 改进的差别矩阵的快速属性约简算法[J]. 小型微型计算机系统,2014,35(4):831-834.

[12] Felix R, Ushio T. Rough sets-based machine learning using a binary discernibility matrix[C]. Proceeding of 2nd International Conference on Intelligent Processing and Manufacturing of Materials,Ha-waii,1999:299-305.

[13] 蒙祖强,史忠植. 一种新的基于简化二进制可辨矩阵的相对约简算法[J]. 控制与决策,2008,23(9):976-978.

[14] 王亚琦,范年柏. 改进的基于简化二进制分辨矩阵的属性约简方法[J]. 计算机科学,2015,42(6):210-215.

[15] Yang M,Yang P. A novel condensing tree structure for rough set feature selection[J]. Neurocom-

puting,2008,71(4):1092-1100.
[16] 蒋瑜. 基于差别信息树的 rough set 属性约简算法[J]. 控制与决策,2015,30(8):1531-1536.
[17] 张文阳,蒋瑜. 基于键树的粗糙集属性约简算法[J]. 成都信息工程大学学报,2017,32(6):618-622.
[18] 唐坤剑,容强. 基于加权浓缩树的粗糙集属性约简算法[J]. 计算机工程与应用,2018,54(2):76-81.
[19] 梅红岩,刘井莲,刘海霞. 基于区分链表的属性约简改进算法[J]. 计算机与信息技术,2008,(Z1):55-56.
[20] 陈炼,吴灵芝. 基于链表的不完备决策表属性约简算法[J]. 科学技术与工程,2015,15(3):250.

Attribute Reduction with Rough Set based on Binary Linked List

SONG Jian, JIANG Yu, LI Dong, BAOYANG Wanying
(College of Software Engineering, Chengdu University of Information Technology, Chengdu 610225, China)

Abstract: The difference matrix is a method used by many scholars to calculate the attribute reduction of rough sets. This method is widely used because it is simple, intuitive and easy to understand, but the redundant elements contained in the difference matrix are not only take no effect on attribute reduction, but also have the problem of high cost of spatial storage. In order to eliminate these redundant elements, a new storage structure is proposed: a binary linked list. All repetitive elements and parent set elements in the discernibility matrix are deleted by bit operation to reduce the storage space of discernibility information. In order to verify the validity of the binary linked list, a new attribute reduction algorithm is proposed. The method is tested by multiple sets of data sets in the UCI database, and the experimental results are compared with other algorithms. The proposed algorithm can get the attribute reduction set faster and can effectively reduce the storage space.

Keywords: rough set; attribute reduction; discernibility matrix; binary linked list; bit operations