基于二进制区分矩阵的不完备系统增量式属性约简算法

丁棉卫[1] 张腾飞[1] 马福民[2]

(南京邮电大学自动化学院 南京 210023)[1] (南京财经大学信息工程学院 南京 210023)[2]

摘 要 不完备信息系统下的增量式属性约简是动态数据挖掘技术的重要研究内容之一。求解增量式属性约简时首先需要求解容差类。当已有系统新增实例时,为了快速求解新的容差类,首先提出一种快速且稳定性较好的容差类静态求解方法,然后在此基础上提出容差类的增量式求解方法。根据增量式求得的新容差类,结合二进制区分矩阵直观及便于处理的优点,通过动态更新二进制区分矩阵方法,提出了不完备信息系统下基于二进制区分矩阵的增量式属性约简算法。通过实例及仿真实验验证了算法的有效性。

关键词 不完备信息系统,增量式,容差类,属性约简

中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.07.043

Incremental Attribute Reduction Algorithm Based on Binary Discernibility Matrix in Incomplete Information System

DING Mian-wei[1] ZHANG Teng-fei[1] MA Fu-min[2]

(College of Automation, Nanjing University of Posts and Telecommunications, Nanjing 210023, China)[1]

(College of Information Engineering, Nanjing University of Finance and Economics, Nanjing 210023, China)[2]

Abstract Incremental attribute reduction algorithm in incomplete information system is one of the important research contents in the area of data mining. For getting the attribute reduction incrementally, the tolerance class needs to be computed. For the purpose of speeding up the tolerance class calculation, an improved static algorithm with rapidity and stability is developed firstly, followed by a novel incremental algorithm, which can update the tolerance class rapidly when a new object is coming. On the basis of the obtained tolerance class and combined with the intuitive and easy of binary matrix, an incremental attribute reduction algorithm based on binary matrix in incomplete information system by updating the binary matrix was proposed. The validity of these algorithms was demonstrated by the simulation and experimental results.

Keywords Incomplete information system, Incremental, Tolerance class, Attribute reduction

1 引言

粗糙集理论是由波兰华沙理工大学 Pawlak 教授于 20 世纪 80 年代初提出的用于处理不完整、不确定知识的数学工具,已被广泛应用于数据挖掘、图像识别与处理、过程控制以及医疗诊断等领域[1-4]。传统粗糙集理论基于完备信息系统,而现实生活中的信息系统往往会因为数据丢失等原因变成不完备信息系统,所以建立在等价关系和划分之上的传统粗糙集理论的应用受到了一定的限制。通常采用两种方法来应对上述问题:1)放宽等价关系为相似关系,得到基于相似关系粗糙集;2)放宽划分为覆盖,得到覆盖粗糙集[5]。其中,覆盖粗糙集理论研究最为广泛。针对数据缺失的不完备信息系统,覆盖粗糙集理论通常会采用两种处理方法:1)对缺失值进行删除、替换等预处理操作,使其成为完备信息系统;2)对缺失值不进行处理,直接采用能处理缺失值的模型(如容差粗糙集模型等)来进行处理[6-7]。目前,研究最为广泛的是第二种方法。众多学者通过容差粗糙集中的容差关系建立覆盖来处理不完备信息系统下的属性约简问题[8-9]。

Zhang[8]介绍了条件信息量,并根据条件信息量给出了基于条件信息量的属性约简算法。文献[10]利用边界域的基数作为评价属性约简的准则,并依据该准则提出了基于边界域的启发式约简算法。上述方法主要运用于静态信息系统,而现实生活中很多信息系统总是动态变化的,当有新数据加入时,采用静态方法重新计算整个信息系统往往是很浪费时间和资源的,因此有部分学者转向研究不完备信息系统下能够处理动态数据的增量式属性约简算法。文献[11]介绍了一种增量式求核方法,即当加入新实例时,通过动态更新而非重建的方法得到更新后的区分矩阵,再对更新后的区分矩阵利用属性核判定定理求取核属性。Qian 等人[12]给出了不完备信息系统下简化矩阵的定义,并根据简化矩阵采用更新的方

到稿日期:2016-05-24 返修日期:2016-09-29 本文受国家自然科学基金项目(61105082,61403184),江苏省'青蓝工程'基金(QL2016),南京邮电大学'1311人才计划'基金(NY2013),南京邮电大学科研项目基金(NY215149)资助。

丁棉卫(1991—),男,硕士生,主要研究方向为粗糙集理论;张腾飞(1980—),男,博士,教授,硕士生导师,主要研究方向为智能信息处理、智能控制等,E-mail:tfzhang@126.com(通信作者);马福民(1979—),女,博士,副教授,硕士生导师,主要研究方向为智能信息处理、智能生产系统等。

法增量式求取属性约简，其不足在于简化区分矩阵的存储空间仍然较大。文献[9]介绍了一种在不完备信息系统下基于正域的增量式属性约简算法，但是该方法一次只能处理增加单个实例的情况，在增加一组实例时的执行效果不佳。由于在容差关系下求解属性约简时需要求解容差类，因此高效的容差类求解方法有助于提高属性约简算法的运行效率，故有不少学者对高效的容差类求解方法进行了研究。

相比于传统计算容差类方法的高时间复杂度，文献[8]提出了通过求解$\sum_{b\in B}[(b,v)]$来快速求解容差类的方法，其不足在于当有新实例加入时，需要重新计算每一个(b,v)。文献[9]提出了一种利用$S_{\{1\}}(x)\subset S_{\{1,2\}}(x)$容差特性来建立高效计算容差类的方法，该方法的优点在于通过逐级计算避免了重复比较某些属性值而造成的时间浪费，不足在于当缺失值较多时运行速度不佳。文献[15]采用基数排序和标记技术的设计思想，使计算容差类的时间复杂度从传统的$O(|C|^2|U|)$降为$O(m|C||U|)$，但该方法不易理解。以上求解容差类的方法主要运用于静态求解容差类，当有新实例加入数据集时需要重复运行静态容差类计算方法，从而消耗大量时间，因此采用这些方法来处理增量式数据集时会影响求解容差类的效率。

针对加入新实例时需要重新计算所有容差类的问题，本文首先提出了一种快速且稳定性较好的容差类静态求解方法，然后在此基础上提出了容差类的增量式求解方法。根据增量式求得的新容差类，结合二进制区分矩阵直观及便于处理的优点，通过动态更新二进制区分矩阵方法，提出了不完备信息系统下基于二进制区分矩阵的增量式属性约简算法，其有效地缩减了二进制区分矩阵的存储空间并能够快速处理增加一组实例集的情况。理论分析、计算实例以及仿真实验均验证了算法的有效性。

2　相关概念

定义 1[16]　在给定的信息系统 $S=(U,A,V,f)$ 中，$U=\{x_1,x_2,\cdots,x_n\}$是非空的有限实例集合，也称作论域，$A=C\cup D$，$C\cap D=\varnothing$，$C=\{c_1,c_2,\cdots,c_m\}$是条件属性集，D 是决策属性集，对于$\forall a\in C\cup D$，映射 $f:U\to V_a$ 是信息函数，其中 V_a 是 a 的值域。对每个 $B\subseteq R$，定义信息系统的不可分辨二元关系为：

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

定义 2[10]　对于给定的信息系统 S，如果在条件属性 C 的值域中有缺失的数值，则称这样的信息系统为不完备信息系统，表示为 $IS=(U,A,V,f)$，其中缺失的属性值通常用“$*$”来表示。

定义 3[14]　在给定的不完备信息系统 IS 中，对$\forall P\subseteq A$，定义 U 上的一个容差关系 $TR(P)$ 为：

$$TR(P)=\{(x,y)\in U^2\mid \forall a\in P,(f(x,a)=f(y,a))\vee(f(x,a)=*)\vee(f(y,a)=*)\}$$

容差关系 $TR(P)$ 具有自反性和对称性。

定义 4[14]　在给定的不完备信息系统 $IS=(U,A,V,f)$ 中，容差类 $S_P(x)$ 表示根据关系 P 与实例 x 不能相互区分的最大实例集合，其定义为：

$$S_P(x)=\{y\in U\mid (x,y)\in TR(P)\}$$

定义 5[14]　在给定的不完备信息系统 $IS=(U,A,V,f)$ 中，给定任意属性子集 $P\subseteq A$ 和实例子集 $X\subseteq U$，X 关于容差关系 $TR(P)$ 的下近似集和上近似集定义为：

$$\underline{apr}_PX=\{x\in U\mid S_P(x)\subseteq X\}$$

$$\overline{apr}_PX=\{x\in U\mid S_P(x)\cap X\neq\varnothing\}$$

定义 6[14]　在给定的不完备信息系统 $IS=(U,A,V,f)$ 中，给定任意属性子集 $P\subseteq A$ 和实例子集 $X\subseteq U$，X 的 P 边界定义为 $BN_P(X)=\overline{apr}_PX-\underline{apr}_PX$；$X$ 的 P 正域定义为 $POS_P(X)=\underline{apr}_PX$；实例子集 X 的 P 负域定义为 $NEG_P(X)=U-\overline{apr}_PX$。当 X 为 U 时，简记 X 的正域为 U_{pos}，负域为 U_{neg}。

为了便于计算正域，给出文献[14]中改进型正域的定义。

定义 7[14]　在给定的不完备决策信息系统 $IS=(U,C\cup D,V,f)$ 中，对$\forall P\subseteq C$，D 为决策属性，定义正域为：

$$POS_P(D)=\{x\in U\mid S_P(x)/IND(D)\mid=1\}$$

定义 8[17]　在不完备信息系统 $IS=(U,C\cup D,V,f)$ 中，$U=U_{pos}\cup U_{neg}$，定义差别矩阵 $M=(m(i,j))$，其中元素 $m(i,j)$ 的表达式定义为：

$$m(i,j)=\begin{cases}c_k,\ c_k\in C, f(x_i,c_k)\neq *\wedge f(x_j,c_k)\neq *\wedge f(x_i,c_k)\neq f(x_j,c_k), f(x_i,D)\neq f(x_j,D)\text{且 }x_i,x_j\text{ 在 }U_{pos}\text{中；}\\ f(x_i,c_k)\neq *\wedge f(x_j,c_k)\neq *\wedge f(x_i,D)\neq f(x_j,D)\\ \text{且 }x_i,x_j\text{ 一个在 }U_{pos}\text{中，而另一个在 }U_{neg}\text{中}\\ \varnothing,\ \text{其它}\end{cases}$$

其中，$k=1,2,\cdots,m$，U_{pos} 为 U 的 C 正域，U_{neg} 为 U 的 C 负域。

3　改进的容差类求解方法

3.1　改进的容差类静态求解算法

根据定义 3 对容差类的定义可知，当两个实例进行容差关系判断时，需要满足两者对应属性值相同或者两者对应属性值中含有缺失值时才符合容差关系。因此，当实例 x_i 和实例 x_j 判断单个属性 a 是否满足容差关系时，其判断流程如图 1 所示。

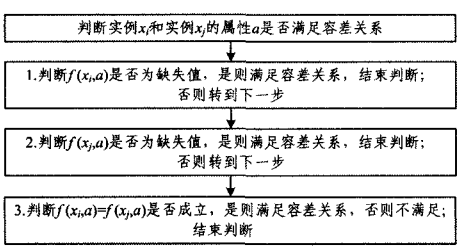

图 1　实例对单个属性容差关系进行判断的流程

从图 1 中可以看出，单个属性在最坏情况下需要判断 3 次是否满足容差关系条件。当 $f(x_i,a)$ 和 $f(x_j,a)$ 中某一个为缺失值时，则认定满足容差关系。因此，当实例中某些属性值为缺失值时，则可不用判断该属性而直接认定其满足容差关系。为了减少对单个属性的比较次数，对决策表中实例的条件属性集做预处理，对实例 x_i 中为缺失值的属性进行统计，将为缺失值的属性保存在 $missingAtt(x_i)$ 中。此时若实例 x_i 和 x_j 进行是否满足容差关系的判断，则因为已知 $missingAtt(x_i)$ 和 $missingAtt(x_j)$，所以对剩余属性中的单个属性进行判断时图 1 中的步骤 1 和步骤 2 可以省略，只需判断步骤 3 即可。此时将容差关系修改为定义 9。

定义 9 给定不完备信息系统 $IS=(U,C\cup D,V,f)$，对于 $\forall P\subseteq C$，定义 U 上的一个容差关系 $TR(P)$ 为：

$$TR(P)=\{(x_i,x_j)\in U^2 \mid \forall a\in(P-missingAtt(x_i,x_j)),f(x_i,a)=f(x_j,a)\}$$

其中，$P-missingAtt(x_i,x_j)$ 表示从属性集 P 中去掉实例 x_i，x_j 中为缺失值的属性。

当两个实例进行判断时，只需要考虑 $P-missingAtt(x_i,x_j)$ 剩余的属性值是否相同，而无需再考虑其是否为缺失值。在判断 $P-missingAtt(x_i,x_j)$ 剩余的属性值是否相同时，当出现某个属性值不相同时，则停止剩余属性判断，认定两个实例不满足容差关系。为了避免重复计算，设 $U=\{x_1,x_2,x_3,x_4\}$，当求得 $S_C(x_1)$ 时，根据 $S_C(x_1)$ 中含有的实例，假设含有 x_2，则根据容差关系的对称性将 x_1 添加到 $S_C(x_2)$，这样在计算 $S_C(x_2)$ 时，只需判断 x_2 与 x_3 和 x_4 而无需再与 x_1 进行判断。以此类推，计算 $S_C(x_3)$ 时只需判断 x_3 与 x_4 而无需判断 x_3 与 x_1 和 x_2。通过以上方法的改进，使得判断的次数从传统最坏的 $3|C||U|^2$ 次降低到最坏情况下的 $\frac{|C||U|^2}{2}$ 次。

结合采用的求取容差类方法，改进的静态容差类求解算法如算法 1 所示。

算法 1 改进的静态容差类求解算法

输入：IS=(U,C∪D,V,f)，missingAtt(x_i)，i=1,2,…,n

输出：容差类集 ToleranceSet

Step1 建立 $S_C(x_i)=\{x_i\}$，其中 i=1,2,…,|U|，并将其添加到 Tolerance-Set 中；

Step2 将实例集中实例的编号设为 i，并将第一个实例编号 1 赋值给 i；

Step3 判断实例编号 i 是否大于|U|−1，大于则转到 Step6，不大于则从实例集 U 中取出实例 x_i，取出和其需要进行比较的第一个实例编号并设为 j，其中 j=i+1；

Step4 判断 j 是否大于|U|，若大于则将 i 自增 1 并转到 Step3，若不大于则取出 x_j，并对 x_i 和 x_j 进行属性比较，依次比较 remainedAtt=C−missingAtt(x_i)中的属性，当出现属性值不等时，停止剩余属性比较，令 j=j+1，并转到 Step4；

Step5 此时 x_i 与 x_j 满足容差关系，在 $S_C(x_i)$ 中添加 x_j，在 $S_C(x_j)$ 中添加 x_i，令 j=j+1，并转到 Step4；

Step6 输出 ToleranceSet。

3.2 容差类的增量式求解算法

算法 1 主要运用于静态求取容差类，但现实生活中很多信息系统往往是动态变化的，若在有新实例加入时重新计算所有容差类，必然会浪费时间。为了避免新实例加入时重复计算已有容差类，提出一种增量式求解容差类方法。当新实例 x 加入时，先求解 $S_C(x)$，设 $S_C(x)=\{x_1,x_3,x\}$，因为 $S_C(x)$ 中除自身外还包含实例 x_1 和 x_3，根据容差类的对称性，将实例 x 分别添加到 $S_C(x_1)$ 和 $S_C(x_3)$ 中，从而完成已有容差类的更新。通过这样的更新方式，避免了对已求解的容差类重复进行求解。

为了更好地说明更新容差类方法，算法 2 给出了增量式求解容差类算法的描述。

算法 2 增量式求解容差类算法

输入：1)IS=(U,C∪D,V,f)，missingAtt(x_i)，i=1,2,…,|U|，已有容差类集 ToleranceSet；2)新加入实例 x

输出：更新后的容差类集 UpdatedTolSet，缺失属性集 missingAtt(x_i)，U

Step1 得到新实例 x 为缺失值的属性 missingAtt(x)，建立 x 的容差类 $S_C(x)$，并将 x 添加到 $S_C(x)$ 中；

Step2 将实例集中实例的编号设为 i，并将第一个实例编号 1 赋值给 i；

Step3 判断实例编号 i 是否大于|U|，大于则转到 Step5，不大于则取出 x_i，并对实例 x_i 和实例 x 进行属性比较，依次比较 remainedAtt=C−missing{x_i,x}中的剩余属性，当出现属性值不等时，停止剩余属性比较，将 i 自增 1，并转到 Step3；

Step4 此时实例 x_i 与实例 x 满足容差关系，则在 $S_C(x)$ 中加入 x_i，在 $S_C(x_i)$ 中加入实例 x，将 i 自增 1，并转到 Step3；

Step5 U=U∪{x}，missingAtt{x_i}=missingAtt{x_i}∪missingAtt{x}，UpdatedTolSet=ToleranceSet∪$S_C(x)$；

Step6 输出 UpdatedTolSet，U，missingAtt{x_i}

3.3 实验结果及分析

为了验证算法的有效性，在内存为 2GB，CPU 为 AMD 2.2 GHz，操作系统为 Windows 7 的 PC 上，采用 MATLAB 2010A 实现本文中的算法 1 和算法 2、文献[8]中的求解容差类方法以及文献[9]中的求解容差类方法。文献[8]中算法用算法 A 表示，文献[9]中算法用算法 B 表示。选用表 1 中的 UCI 机器学习数据库数据源进行算法测试，*total_n* 表示实例数总数，$|C|$ 表示条件属性个数，m 表示属性值缺失元素占总属性元素的百分比，t 表示算法执行时间。对选用的 4 个数据库采用算法 1、算法 A 以及算法 B 进行静态容差类计算，算法的运行时间如表 2 所列。

表 1 UCI 数据库

数据库	*total_n*	$\|C\|$	*m*/%
anneal	798	38	64.9
water-treatment	527	38	2.8
soybean-large	307	35	6.6
hepatitis	155	19	5.7

表 2 求解静态容差类算法的时间 *t*/s

数据库	算法 1	算法 A	算法 B
anneal	39.1	18.1	243.4
water-treatment	10.6	220.5	2.1
soybean-large	3.2	2.4	3.0
hepatitis	1.1	3.6	0.2

从表 2 中可以看出，在 anneal 数据集中算法 B 的运行时间明显长于算法 1 以及算法 A 的运行时间，其中算法 A 的运行时间最短，其主要原因在于 anneal 数据中有大量的缺失值，而算法 B 在缺失值较多时运行效果较差，算法 A 的在面对缺失值偏多时则具有较好的效果。在表 2 的 water-treatment 数据集中，算法 A 的运行时间明显长于算法 1 和算法 B 的运行时间。导致算法 A 的运行速度较差的主要原因是 water-treatment 中具有较少的缺失值，而算法 A 在较少缺失值时的运行效果欠佳。由属性缺失值个数适中的 soybean-large 及 hepatitis 数据集可以看出，3 种算法的运行速度基本持平。通过以上分析可以看出，本文提出的算法 1 在面对数据库中缺失值偏多或者偏少时都不会出现极端情况，在应对各种数据库时处理速度更为均衡。同时从表 2 中的 anneal 及 water-treatment 数据集中可以明显看出，当数据集中的实例个数越多时，这种平稳性表现得就越为明显。

为了验证增量式求解容差类算法 2 的运行效率，继续采用表 1 中的 UCI 数据集进行测试。对测试数据库取 n 个实例作为已有实例，然后将剩余实例作为增加的数据加入，依次对算法 2、算法 A、算法 B 进行容差类计算，运行结果如表 3

所列，其中 n 表示已有实例数，$incre\_n$ 表示增加实例数。

表 3　增量式求解容差类算法的时间 t/s

数据库	n	$incre\_n$	算法 2	算法 A	算法 B
anneal	600	198	39.1	2863.2	32467.1
water-treatment	400	127	10.9	22595.2	181.4
soybean-large	200	107	3.3	196.6	171.4
hepatitis	50	105	1.1	186.9	14.9

由表 3 可以看出，当加入新实例时，采用增量式求解容差类的算法 2 的运行速度明显快于算法 A 和算法 B。这主要是因为算法 2 在算法 1 的基础上求解了新加入实例的容差类，然后根据求得的新实例的容差类反向更新已有容差类，而非重新建立所有容差类，从而实现增量式求解容差类，提高了算法的运行效率。从算法 A 和算法 B 的运行时间可以看出，在增量式属性约简中，求解容差类花费的时间较长时会严重影响增量式求解属性约简的时间。因此，在增量式求解属性约简中，采用增量式求解容差类算法，可以有效提高增量式求解属性约简的速度。

4　不完备信息系统下基于二进制区分矩阵的增量式属性约简

4.1　基于二进制区分矩阵的不完备系统增量式属性约简算法

利用二进制数便于计算和存储的优点，将区分矩阵（见定义 8）修改为二进制区分矩阵（见定义 10）。

定义 10　在给定的不完备信息系统 IS 中，根据 U^* 中的实例对，定义二进制区分矩阵 $BM^*=[exampair\ m_{ij}^*]$，其中 $exampair$ 为 U^* 中的实例对，m_{ij}^* 为矩阵的集合元素，$m_{ij}^*=[r_1,r_2,\cdots,r_n]$，$m_{ij}^*$ 中的单个元素 r_k 定义为：

$$r_k=\begin{cases}0, & \{f(x_i,c)=*\vee f(x_j,c)=*\vee f(x_i,c)=f(x_j,c)\wedge((f(x_i,d)\neq f(x_j,d)\wedge x_i,x_j\in U_1)\vee(x_i\in U_1\wedge x_j\in U_2))\}\\ 1, & \{f(x_i,c)\neq *\wedge f(x_j,c)\neq *\wedge f(x_i,c)\neq f(x_j,c)\wedge((f(x_i,d)\neq f(x_j,d)\wedge x_i,x_j\in U_i)\vee(x_i\in U_1\wedge x_j\in U_2))\}\end{cases}$$

其中，$U_1=POS_C(D)$，$U_2=U-U_1$，$U^*=\{(x_i,x_j)\in U\times U|(x_i,x_j\in U_1)\vee(x_i\in U_1\wedge x_j\in U_2)\}$。

根据定义 10 建立二进制区分矩阵，为了缩减二进制矩阵的存储空间，引入压缩二进制矩阵的定义。

定义 11　在给定的不完备信息系统 IS 中，压缩二进制矩阵定义为：

$$CBM=[exampair\ totalOne_{ij}\ ms_{ij}]$$

其中，$exampair$ 为 U^* 中的实例对；$totalOne_{ij}$ 为二进制矩阵各行中 1 的个数，表示为 $\sum_{l=1}^{n}r_l$；ms_{ij} 为组合的二进制数，表示为 $\sum_{l=1}^{n}2^{n-l}r_l$。

采用压缩矩阵方法，使得二进制矩阵的存储空间从传统的 $|C|+1$ 列减为 3 列，有效缩减了二进制矩阵的存储空间。

为了求取不完备信息系统下的增量式属性约简，本文根据增加的一组实例 X，采用动态更新 CBM 的方法，先得到更新后的 CBM，然后利用静态方法求取核属性和属性约简，从而避免了加入新实例时重新建立整个 CBM 而带来的时间浪费。算法的运行过程如图 2 所示。

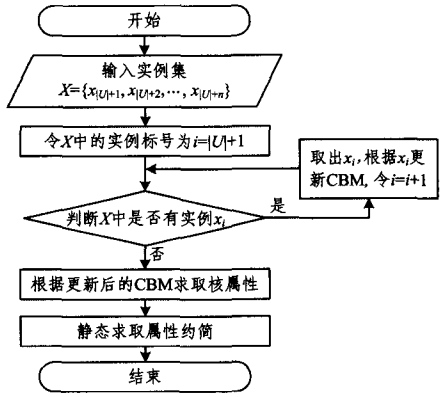

图 2　算法的运行过程

由于本文所采用的在不完备信息系统下基于二进制区分矩阵的增量式属性约简算法是以核为出发点的，因此需要先对建立的 CBM 增量式求取核属性。增量式求取核属性的关键是动态更新压缩二进制区分矩阵。

由于本文根据 U_1 和 U_2 建立压缩二进制区分矩阵，因此当有新实例 x 加入时，需通过判定 U_1 和 U_2 的更新情况来更新压缩二进制区分矩阵。U_1 和 U_2 的更新情况与新实例的容差类 $S_C(x)$ 密切相关。当 $|S_C(x)|=1$ 时，因为 $S_C(x)$ 只有 x 一个实例，所以其一定能加入到正域，因此在 U_1 中增加 x；当 $|S_C(x)|\neq 1$ 时，根据 $S_C(x)$ 中包含的实例相应更新 U_1 和 U_2。

结合上述描述，将压缩二进制区分矩阵的更新方法总结如下：

(1) $|S_C(x)|=1$

此时 $x\in U_1$，将新加入的实例 x 与 U_1 和 U_2 分别建立增加的压缩二进制矩阵，并将其加入到 CBM 中，$U_1=U_1\cup\{x\}$。

(2) $|S_C(x)|\neq 1$

1)依次判断 $S_C(x)$ 中除去 x 的剩余实例 x_i 是属于 U_1 还是属于 U_2。

①若 x_i 属于 U_1，则根据更新后的 $S_C(x_i)$ 判断此时 x_i 是否还属于 U_1。

(a)如果属于 U_1，则不更新压缩二进制区分矩阵。

(b)如果不属于 U_1，则在压缩二进制矩阵中将 x_i 与 U_1，U_2 建立的压缩二进制区分矩阵删除，此时 $U_1=U_1-\{x_i\}$，$U_2=U_2\cup x_i$，将 x_i 与 U_1 重新建立压缩二进制区分矩阵，并将其加入到 CBM 中。

②若属于 U_2，则此时的 CBM 不变。

2)根据 $S_C(x)$ 判断 x 属于 U_1 还是属于 U_2。

①$x\in U_1$，此时将 x 与 U_1，U_2 中的实例按要求依次建立增加的压缩二进制区分矩阵，并将其加入到 CBM 中，$U_1=U_1\cup\{x\}$。

②$x\in U_2$，此时将 x 与 U_1 中的实例按要求依次建立增加的压缩二进制区分矩阵，$U_2=U_2\cup\{x\}$。

更新后压缩二进制区分矩阵，在压缩二进制区分矩阵中将文献[15]中给出的核属性判定方法修改为：在压缩二进制矩阵中找到 $totalOne_{ij}$ 为 1 的行，则该行中组合二进制数 ms_{ij} 中为 1 的位所对应的属性即为核属性。

在得到核属性后，对剩余的属性利用属性频率函数挑选属性频率最大的属性加入到约简集。在压缩二进制矩阵中，

属性频率函数如定义 12 所示。

定义 12 在给定的压缩二进制区分矩阵中，属性频率函数定义为：

$$F(c_k)=\sum_{i=1}^{N}\{bit(ms_{ij},2^{|C|-k})\cdot totalOne_{ij}^{-1}\}$$

其中，N 为 CBM 中建立的实例对总数；$|C|$ 为条件属性的个数；$bit(ms_{ij},2^{N-k})$ 为 ms_{ij} 与 2^{N-k} 相与，结果为 0 时 $bit(ms_{ij},2^{N-k})$ 为 0，不为 0 时 $bit(ms_{ij},2^{N-k})$ 为 1。

根据属性频率函数，借鉴文献[16]中提出的二进制区分矩阵属性约简算法思路，可以进一步给出压缩二进制区分矩阵下的静态属性约简算法：首先将核加入到初始约简集中，从压缩二进制区分矩阵中删除核属性所在列元素非 0 的行；然后依据属性频率函数，选择分辨能力大的属性 c_k 加入到约简集中，再删除压缩二进制矩阵中 c_k 所在列元素非 0 的行。重复以上操作，直到压缩二进制区分矩阵 CBM 为空。为了保证约简的完备性，对所获得的约简集中非核属性进行反向消除，删除可以删除的属性。

结合所采用的方法，不完备信息系统下基于二进制区分矩阵的增量式属性约简过程的描述如算法 3 所示。

算法 3 不完备信息系统下基于二进制区分矩阵的增量式属性约简算法

输入：1)IS=(U,C∪D,V,f),missingAtt{x_i},i=1,2,…,n,CBM,U_1,U_2,ToleranceSet;2)新加入的实例集 X={$x_{|U|+1}$,$x_{|U|+2}$,…,$x_{|U|+n}$}

输出：约简集 Reduction

Step1 设实例编号 i=|U|+1。

Step2 判断 X 中是否有实例 x_i，若没有则转到 Step6，若有则取出 x_i 并转到 Step3。

Step3 对实例 x_i 运行算法 2，增量式求解加入新实例后的容差类 UpdatedTolSet，并判断 UpdatedTolSet 中 $|S_C(x_i)|$ 是否等于 1，若等于则转到 Step4，若不等于则转到 Step5。

Step4 此时 $x_i\in U_1$，将新加入的实例 x_i 与 U_1 和 U_2 中实例分别建立增加的压缩二进制矩阵，在新建的压缩二进制区分矩阵中去除 ms_{ij} 为 0 的行，并将剩余的矩阵加入到 CBM 中。$U_1=U_1\cup\{x_i\}$。令 i=i+1，并转到 Step2。

Step5 此时 $S_C(x)\neq1$，根据 $S_C(x)$ 包含的实例更新 U_1，U_2 和 CBM。

Step5.1 将 $S_C(x_i)$ 中除去 x_i 的剩余实例设为 remainedobj，假设实例在 remainedobj 中的编号为 y_j，并初始化 j=1(1≤j≤$|S_C(x_i)|-1$)。

Step5.2 判断 remainedobj 中是否有实例 x_{yj}，若没有则转到Step5.5；若有则从 remainedobj 中取出实例 x_{yj}，并判断 x_{yj} 是属于 U_1 还是属于 U_2，若属于 U_1 则转到 Step5.3，若属于 U_2 则 j=j+1 并转到 Step5.2。

Step5.3 根据更新后的 $S_C(x_{yj})$，判断此时 x_{yj} 是否还属于 U_1，若属于则 j=j+1 并转到 Step5.2，若不属于则转到 Step5.4。

Step5.4 在 CBM 中将 x_{yj} 与 U_1，U_2 中实例建立的压缩二进制区分矩阵删除，此时 $U_1=U_1-\{x_{yj}\}$，$U_2=U_2\cup x_{yj}$，将 x_{yj} 与 U_1 中实例重新建立压缩二进制区分矩阵，在新建的压缩二进制区分矩阵中去除 ms_{ij} 为 0 的行并将剩余的矩阵加入到 CBM 中。令 j=j+1 并转到 Step5.2。

Step5.5 根据 $S_C(x_i)$ 判断 x_i 是属于 U_1 还是属于 U_2，若属于 U_1 则转到 Step5.6，若属于 U_2 则转到 Step5.7。

Step5.6 将 x_i 与 U_1，U_2 中的实例按要求依次建立增加的压缩二进制区分矩阵，在新建的压缩二进制区分矩阵中去除 ms_{ij} 为 0 的行并将剩余的矩阵加入到 CBM 中，$U_1=U_1\cup\{x_i\}$。令 i=i+1，并转到 Step2。

Step5.7 将 x_i 与 U_1 中实例按要求依次建立增加的压缩二进制区分矩阵，在新建的压缩二进制区分矩阵中去除 ms_{ij} 为 0 的行并将剩余的矩阵加入到 CBM 中，此时 $U_2=U_2\cup\{x_i\}$。令 i=i+1，并转到 Step2。

Step6 求取核属性，并在 CBM 中删除相应行

Step6.1 根据更新后的 CBM，利用核属性判定定理得到此时的核属性 Core。

Step6.2 Reduction=Core。在 CBM 的 ms_{ij} 中，将核属性所在位置为 1 的行删除，得到删除后的 CBM，并将剩余 m((i,j),k) 的 i,j 保存在 U″中。

Step6.3 判断 CBM 是否为空，若为空则转到 Step9，若不为空则转到 Step7。

Step7 在剩余属性中挑选属性频率函数最大的属性，在 CBM 中删除其相应行。

Step7.1 在剩余的 CBM 中计算除 Reduction 外的剩余属性的属性频率函数。

Step7.2 挑选属性频率最大的属性 c=max{F(c_i)}，如果具有 max{F(c_i)}的 c_i 有多个，则 c=maxDis(IS″)，其中 IS″=(U″,(C−Reduction),D,V,f)。

Step7.3 在 CBM 的 ms_{ij} 中，将属性 c 所在位置为 1 的行删除，得到删除后的 CBM，并将剩余 m((i,j),k)的 i,j 保存在 U″中。Reduction=Reduction∪{c}。判断 CBM 是否为空，若为空则转到 Step9，若不为空则转到 Step7。

Step8 for(j=1;j≤|Reduction−Core|;j++)
{ if($POS_C''(D)=POS_{Reduction-\{c_j\}}''(D)$)
Then Reduction=Core−c_j}

Step9 输出约简集 Reduction。

其中 max $Dis(S'')$的描述为：根据 U''中的元素，取出频率相同的属性形成新的数据表，对数据表中的每个属性 c_k 求解 $POS''_{c_k}(D)$，选择 $POS''_{c_k}(D)$最大的 c_k 并返回，若满足要求的 max{$|POS''_{c_k}(D)|$}的属性有多个，则选择 U''/c_k 中具有最小等价类个数的 c_k 加入到约简集中。若满足要求的 max{$|POS''_{c_k}(D)|$}的属性 c_k 具有的最小等价类个数也相同，则任意选择 c_k 返回。

4.2 复杂度分析

(1)空间复杂度

本文算法 3 的空间复杂度主要消耗在存储简化二进制矩阵中，将传统二进制矩阵的空间复杂度 $O(|U^*||C|)$降为了 $O(3|U^*|)$，其中 $|U^*|$ 为形成实力对的个数，极大地缩减了二进制区分矩阵的存储空间，但仍大于文献[8]利用正域方法增量式求解属性约简时的空间复杂度。

(2)时间复杂度

当输入新实例集为 X 时，本文算法 3 的 Step3 在最坏情况下的时间复杂度为 $O((|U_1|+|U_2|)\times|C|\times|X|)$，Step4 的时间复杂度为 $O((|U_1|+|U_2|)\times|C|^2\times|X_1|)$，Step5 在最坏情况下的时间复杂度为 $O((|U_1|+|U_2|)\times|C|^2\times|X_2|)$，其中 $|X_1|+|X_2|=|X|$，Step6 的时间复杂度为 $O((|U_1|+|U_2|)\times|U_1|)$，Step7.1 在最坏情况下的时间复杂度为 $O((|U_1|+|U_2|)\times|U_1|\times|C|)$，Step7.2 和 Step7.3 的总时间复杂度为 $O((|U_1|+|U_2|)\times|U_1|\times|C|^2)$，Step8 在最坏情况下的时间复杂度为 $O((|U_1|+|U_2|)\times|U_1|\times|C|^2)$。因此，总的复杂度小于文献[8]在新加入实例集 X 下的复杂

度 $O(|U_1+U_2|^2\times|C|^2\times|X|)$。

4.3 实例计算

为了验证算法的有效性，以表 4 所列的决策表为例，通过增加新实例集 X 来进行计算说明。

表 4　决策表

	c_1	c_2	c_3	c_4	d
x_1	1	1	1	1	1
x_2	2	*	1	1	1
x_3	*	*	2	1	2
x_4	1	*	1	2	1
x_5	*	*	1	2	3
x_6	2	1	1	*	1

对表 4 求解 U_1 和 U_2。

首先根据算法 1 得到容差类集：$S_C(x_1)=\{x_1\}$，$S_C(x_2)=\{x_2,x_6\}$，$S_C(x_3)=\{x_3\}$，$S_C(x_4)=\{x_4,x_5\}$，$S_C(x_5)=\{x_4,x_5,x_6\}$，$S_C(x_6)=\{x_2,x_5,x_6\}$。

根据定义 8 求得正域 $POS_C(D)=\{x_1,x_2,x_3\}$，因此$U_1=\{x_1,x_2,x_3\}$，$U_2=\{x_4,x_5,x_6\}$。

根据 U_1 和 U_2 建立压缩二进制区分矩阵，如表 5 所列。

表 5　压缩二进制矩阵 CBM

exampair	$totalones_{ij}$	ms_{ij}
(x_1,x_3)	1	0010
(x_2,x_3)	1	0010
(x_1,x_4)	1	0001
(x_1,x_5)	1	0001
(x_1,x_6)	1	1000
(x_2,x_4)	2	1001
(x_2,x_5)	1	0001
(x_3,x_4)	2	0011
(x_3,x_5)	2	0011
(x_3,x_6)	1	0010

新加入的实例集 X 为(1，*，0，0，1；2，*，*，1，2)，X 中包含实例 x_7 和 x_8。先取出 x_7，根据算法 2 得到更新后的容差类：$S_C(x_1)=\{x_1\}$，$S_C(x_2)=\{x_2,x_6\}$，$S_C(x_3)=\{x_3\}$，$S_C(x_4)=\{x_4,x_5\}$，$S_C(x_5)=\{x_4,x_5,x_6\}$，$S_C(x_6)=\{x_2,x_5,x_6\}$，$S_C(x_7)=\{x_7\}$。

此时$|S_C(x_7)|=1$，根据算法 3 中的 Step4 得到的新增加的 CBM 如表 6 所列。

表 6　新增加的 CBM

exampair	$totalones_{ij}$	ms_{ij}
(x_3,x_7)	2	0011
(x_7,x_4)	2	0011
(x_7,x_5)	2	0011
(x_7,x_6)	2	1010

此时取出 x_8 并更新 CBM，根据算法 2 得到加入 x_8 后更新的容差类：$S_C(x_1)=\{x_1\}$，$S_C(x_2)=\{x_2,x_6,x_8\}$，$S_C(x_3)=\{x_3,x_8\}$，$S_C(x_4)=\{x_4,x_5\}$，$S_C(x_5)=\{x_4,x_5,x_6\}$，$S_C(x_6)=\{x_2,x_5,x_6,x_8\}$，$S_C(x_7)=\{x_7\}$，$S_C(x_8)=\{x_2,x_3,x_6,x_8\}$。

此时$|S_C(x_8)|\neq1$，根据定义 8 依次判断 $S_C(x_8)$ 中除了 x_8 外的$\{x_2,x_3,x_6\}$是否仍然属于 U_1 或者 U_2，并相应地按照算法 3 中的 Step5 更新 CBM，此时 $U_1=\{x_1,x_3,x_7\}$，$U_2=\{x_2,x_4,x_5,x_6\}$。完成更新后，根据 $S_C(x_8)$ 判断 x_8 是属于 U_1 还是属于 U_2，得到 $U_2=\{x_2,x_4,x_5,x_6,x_8\}$，并建立新增加的 CBM，最终得到的更新后的压缩二进制矩阵如表 7 所列。

表 7　更新后的 CBM

exampair	$totalones_{ij}$	ms_{ij}
(x_1,x_3)	1	0010
(x_1,x_4)	1	0001
(x_1,x_5)	1	0001
(x_1,x_6)	1	1000
(x_3,x_4)	2	0011
(x_3,x_5)	2	0011
(x_3,x_6)	1	0010
(x_3,x_7)	2	0011
(x_7,x_4)	2	0011
(x_7,x_5)	2	0011
(x_7,x_6)	2	1010
(x_1,x_2)	1	1000
(x_3,x_2)	1	0010
(x_7,x_2)	3	1011
(x_1,x_8)	1	1000
(x_7,x_8)	2	1001

根据更新后的 CBM，求得属性约简 $Reduction=\{x_1,x_3,x_4\}$。

4.4 实验仿真及分析

在与 3.3 节相同的实验环境下，采用 MATLAB 编程实现本文算法 3 以及文献[8]的增量式属性约简算法，用算法 C 表示文献[8]中的算法。选用 UCI 机器学习数据库数据源 anneal，soybean-large，mammographic，hepatitis 及 dermatology 进行算法测试。用 *total_n* 表示总共需要判断的实例数，n 表示已有实例数，*incre_n* 表示增加实例数，$|C|$ 表示条件属性个数，t 表示算法处理增量部分的执行时间，r 表示约简后剩余属性的个数。增量式属性约简运行结果的对比如表 8 所列。

表 8　增量式属性约简运行结果的对比

数据库	*total_n*	n	*incre_n*	$\|C\|$	算法 3		算法 4	
					t/s	r	t/s	r
anneal	798	750	48	38	31.7	9	1644.5	9
soybean-large	307	250	57	35	35.7	13	568.7	13
mammographic	961	900	61	5	15.2	5	2908.2	5
hepatitis	155	100	55	19	5.4	4	246.2	4
dermatology	366	300	66	34	54.6	6	484.1	6

从表 8 中可以看出，算法 3 的约简质量与算法 C 相同，从算法的运行时间可以看出，算法 3 的运行时间明显少于算法 C，这是因为算法 3 采用了增量式求取容差类的方法，当新实例加入时采用更新的方法得到新容差类，避免了重复计算已有容差类而造成的时间消耗。除此之外，当面对增加一组实例时，相比于算法 C 重复进行运行，算法 3 能够在一次算法调用中处理增加的一组数据，减少了使用静态属性约简方法的次数，从而极大地提高了算法的运行效率。

结束语　在不完备信息系统中，利用容差关系增量式求解属性约简是重要的研究内容之一。在增量式求解属性约简前，首先需要求解容差类。因此本文先给出了一种适应性较好的求解容差类方法，该方法能够在缺失值较多、较少或者适中的情况下都保持较好的效果；并根据该方法给出了一种增量式求解容差类方法，当加入新实例时，其能够快速得到所有实例的容差类；同时，针对不完备信息系统，提出了一种基于二进制区分矩阵的增量式属性约简算法，其能够快速且有效地处理增加多个实例的情况。实例计算及实验仿真均验证了算法的有效性。另外，本文只考虑了容差关系模型，研究在限制容差关系下的基于区分矩阵的增量式属性约简将是下一步的工作。

参 考 文 献

[1] HO H C, FANN W J D, CHIANG H J, et al. Application of Rough Set, GSM and MSM to Analyze Learning Outcome-An Example of Introduction to Education [J]. Journal of Intelligent Learning Systems and Applications, 2016, 8(1): 23-38.

[2] GOTLIB D, MARCINIAK J. Potential Application of the Rough Set Theory in Indoor Navigation [J]. Lecture Notes in Computer Science, 2014, 8537: 301-308.

[3] CHEN C S, LIANG W Y, HSU H Y. A cloud computing platform for ERP applications[J]. Applied Soft Computing, 2015, 27 (C): 127-136.

[4] CHEN X, WANG X M, HUANG Y, et al. Fault diagnosis for tilt-rotor aircraft flight control system based on variable precision rough set-OMELM [J]. Control and Decision, 2015, 35(3): 433-440. (in Chinese)
陈晓, 王新民, 黄誉, 等. 倾转旋翼机飞控系统的变精度粗糙集-OMELM 故障诊断方法[J]. 控制与决策, 2015, 35(3): 433-440.

[5] CHEN D, WANG C, HU Q. A new approach to attribute reduction of consistent and inconsistent covering decision systems with covering rough sets [J]. Information Sciences, 2007, 177 (17): 3500-3518.

[6] WU K, PAN W, WU L, et al. Incremental imputation method for incomplete decision system[J]. Industrial Marketing Management, 2011, 40(2): 267-277.

[7] YANG X P. Completing incomplete data based on maximum similarity in Rough sets [J]. Computer Engineering and Applications, 2012, 48(36): 164-166. (in Chinese)
杨小平. 粗集中最大相似度的不完备数据补齐[J]. 计算机工程与应用, 2012, 48(36): 164-166.

[8] SHU W, SHEN H. A rough-set based incremental approach for updating attribute reduction under dynamic incomplete decision systems[C]// Proceedings of the 2013 IEEE International Conference on Fuzzy Systems. FUZZ, 2013: 1-7

[9] ZHANG Q, ZHENG X, XU Z. Efficient Attribute Reduction Algorithm Based on Incomplete Decision Table[C]// 2009 Second International Conference on Intelligent Computation Technology and Automation. IEEE Computer Society, 2009: 192-195.

[10] LIU F, LI T R. Method for Attribute Reduction Based on Rough Sets Boundary Regions[J]. Computer Science, 2016, 43(3): 242-245, 284. (in Chinese)
刘芳, 李天瑞. 基于边界域的不完备信息系统属性约简方法[J]. 计算机科学, 2016, 43(3): 242-245, 284.

[11] LI R, ZHANG D, ZHAO Y, et al. Incremental Core Computing for Incomplete Decision Tables [C]// International Symposium on Computational Intelligence and Design. IEEE Computer Society, 2008: 270-273.

[12] QIAN W, SHU W, XIE Y, et al. Feature Selection using Compact Discernibility Matrix-based Approach in Dynamic Incomplete Decision System [J]. Journal of Information Science & Engineering, 2015, 31(2): 509-527.

[13] QIAN W B, YANG B R, XU Z Y, et al. Efficient Algorithm for Computing Tolerance Classes of Incomplete Decision Table [J]. Journal of Chinese Computer Systems, 2013, 34(2): 345-350. (in Chinese)
钱文彬, 杨炳儒, 徐章艳, 等. 基于不完备决策表的容差类高效求解算法[J]. 小型微型计算机系统, 2013, 34(2): 345-350.

[14] SHU W, SHEN H. Updating attribute reduction in incomplete decision systems with the variation of attribute set [J]. International Journal of Approximate Reasoning, 2014, 55(3): 867-884.

[15] ZHANG T, YANG X, MA F. Improved algorithm for attribute core computing based on binary discernibility matrix [C]// 33rd Chinese Control Conference(CCC). 2014: 7400-7404.

[16] GE H, YANG C J, LI L S. An Improved Attribute Reduction Algorithm Based on Binary Discernibility Matrix [J]. Computer Technology and Development, 2008, 18(8): 12-15. (in Chinese)
葛浩, 杨传健, 李龙澍. 一种改进的基于二进制可分辨矩阵属性约简算法[J]. 计算机技术与发展, 2008, 18(8): 12-15.

[17] SHU W H, XU Z Y, QIAN W B, et al. Quick Attribution Reduction Algorithm Based on Incomplete Decision Table [J]. Journal of Chinese Computer Systems, 2011, 32(9): 1867-1871. (in Chinese)
舒文豪, 徐章艳, 钱文彬, 等. 一种快速的不完备决策表属性约简算法[J]. 小型微型计算机系统, 2011, 32(9): 1867-1871.

(上接第 209 页)

[10] ZHENG Y J. Water wave optimization : a new nature-inspired metaheuristic [J]. Computers & Operations Research, 2015, 55: 1-11.

[11] ZHANG B, ZHENG Y J. Convergence Analysis of Water Wave Optimization Algorithm [J]. Computer Science, 2016, 43(4): 41-44. (in Chinese)
张蓓, 郑宇军. 水波优化算法收敛性分析[J]. 计算机科学, 2016, 43(4): 41-44.

[12] ZHANG Y J, ZHANG B, XUE J Y. Selection of Key Software Components for Formal Development Using Water Wave Optimization[J]. Journal of Software, 2016, 27(4): 933-942. (in Chinese)
郑宇军, 张蓓, 薛锦云. 软件形式化开发关键部件选取的水波优化方法[J]. 软件学报, 2016, 27(4): 933-942.

[13] PAN Q K, SANG H Y, DUAN J H, et al. An improved fruit fly optimization algorithm for continuous function optimization problems [J]. Knowledge-Based Systems, 2014, 62(5): 69-83.

[14] RUEDA J L, ERLICH I. Testing MVMO on learning - based real-parameter single objective benchmark optimization problems [C]// 2015 IEEE Congress on Evolutionary Computation. New York: IEEE Press, 2015: 1025-1032.

[15] YANG X S. Firefly algorithm, stochastic test functions and design optimisation[J]. International Journal of Bio-Inspired Computation, 2010, 2(2): 78-84.

[16] MIRJALILI S, MIRJALILI S M, HATAMLOU A. Multi-verse optimizer: a nature-inspired algorithm for global optimization [J]. Neural Computing and Applications, 2015, 149(3): 29-38.

[17] ZHANG H, LI B, ZHANG J, et al. Parameter estimation of nonlinear chaotic system by improved TLBO strategy[J]. Soft Computing, 2016, 20(12): 4965-4980.

[18] ZHAO Z S, FENG X, LIN Y Y, et al. Improved rao blackwellized particle filter by particle swarm optimization[J]. Journal of Applied Mathematics, 2013, 10(4): 15-22.

[19] ZHAO Z S, FENG X, LIN Y Y, et al. Evolved neural network ensemble by multiple heterogeneous swarm intelligence [J]. Neurocomputing, 2015, 149(3): 29-38.

[20] LIU B, WANG L, JIN Y H. An effective PSO-based memetic algorithm for flow shop scheduling [J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 2007, 37 (1): 18-27.

[21] WOLPERT D H, MACREADY W G. No free lunch theorems for optimization [J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 67-82.