文章编号：2096-1472(2017)-06-08-04

基于上三角矩阵构造多叉树的多维关联规则挖掘算法

叶 涛[1]，于利霞[2]，张亚平[2]
(1.青海民族大学计算机学院，青海 西宁 810007；
2.天津大学计算机学院，天津 300072)

摘 要：针对基于Iapriori算法的多维关联规则数据挖掘存在I/O负载过大，候选项集指数倍增加，优化算法随机性强，容易陷入局部最优解等问题。本文提出一种基于上三角矩阵和多叉树结合(UTMTU)的多维关联规则挖掘算法，算法对原数据编码筛选后映射为上三角矩阵，再映射为频繁项集树，实现整个过程只扫描一次数据库而不产生候选项集，将时间和空间成本尽量降到最低，并利用有效属性层次数提高内存和I/O的利用率。通过UTMTU与Iapriori对比分析表明，其算法的效率和精度得到显著地提高，有效改善原始算法的两个瓶颈问题。

关键词：多维关联规则；上三角矩阵；频繁项集树；有效属性层次数
中图分类号：TP391 **文献标识码**：A

A Multi-Dimensional Association Rules Mining Algorithm Based on Upper Triangular Matrix-Tree Union

YE Tao[1],YU Lixia[2],ZHANG Yaping[2]
(1.*College of Computer,Qinghai Nationalities University*,*Xining* 810007,*China*;
2.*School of Computer Science and Technology*,*Tianjin University*,*Tianjin* 300072,*China*)

Abstract: Many problems exist in the multi-dimensional association rules mining algorithm based on IApriori.The I/O load is too heavy,the size of the candidate set is increased exponentially,and the optimization algorithm is random and easy to fall into the local optimal solution.Accordingly,the paper proposes a multi-dimensional association rules mining algorithm based on Upper Triangular Matrix-Tree Union (UTMTU).The algorithm filters and maps the original coding data to the upper triangular matrix,and then maps it into the frequent item sets.In the whole process,UTMTU only scans the database once and does not generate the candidate item sets,which reduces the time and space cost to the minimum.The utilization rate of memory and I/O is improved by using the number of effective attribute layers.Compared with the algorithm based on IApriori,the efficiency and accuracy of the algorithm based on UTMTU has been effectively improved.Consequently,the UTMTU-based algorithm is more suitable for multi-layer and multi-attribute MARP.

Keywords:multi-dimensional association rules;upper triangular matrix;frequent item sets tree;the effective layers of attributes

1 引言(Introduction)

单维关联规则[1](Single-dimensional Association Rule Problem，简称SARP)的经典算法是由R.Agrawal和R.Srikant于1994年提出的Apriori算法，该算法存在两个重要的性能瓶颈：(1)多次扫描事物数据库，严重增加I/O负载；(2)可能按照指数倍增加产生庞大的候选项集。针对这两个不足，研究界提出了诸多改进算法，如AprioriTid、FP-Tree[2]、IUBM、SLIG等。多维关联规则(Multi-dimensional Association Rule Problem，简称MARP)[3]沿用“支持度-置信度”框架，研究两个方面问题：(1)属性的处理。目前较为有效的方法是归纳划分类别属性，聚类划分连续属性，如MAQA算法。(2)频繁项集挖掘。目前研究方法主要有：①改进Apriori算法：FARMBT[4]算法基于模糊集理论离散属性，再用Apriori分析。计数器算法对属性层次计数找频繁项集。②映射算法：Opportunistic[5]算法建立频繁项集树来投射交易子集。位图矩阵技术[6]缩小频繁项集的搜索范围，但仅限于两维。QARMM[7]算法对属性布尔位标注，再进行向量运算，但编码冗长，内存利用率低，仅限少量属性且属性层次较少的集合。③智能算法：引入AIS免疫算法[8]自体与非自体识别技术多点随机搜索最优解作为频繁项集。IGA利用免疫算法来改进遗传算法构造染色体，通过迭代选种，交叉和突变找最优解。GACM算法[9]将蚁群算法的信息素更新和遗传算法有机结

基金项目：天津大学-青海民族大学自主创新基金；国家自然科学基金项目(61572351)；天津市自然科学基金(15JCQNJC00200)。

合来找最大频繁项集。优化算法鲁棒性好，但随机性强，全局搜索能力差、易陷入局部最优解，且不能计算频繁项集的支持度，不利于后续决策。

本文提出一种基于上三角矩阵与多叉树结合的UTMTU(Upper Triangular Matrix-Tree Union association rules algorithm)算法。该算法对原数据编码筛选后映射为上三角矩阵，再映射为频繁项集树，整个过程只扫描一次数据库，将时间和空间成本尽量降到最低。

2 多维关联规则的基本知识(The basic knowledge of multidimensional association rules)

MARP的项目的集合为$I_r=I\times I\times P$，即$I_r=\{(x,l,u)\mid x\in I, l\in P,u\in P,l\le x\le P\}$，$(x,l,u\in I_r)$表示$X$取值在$l$和$u$之间。其中$I$为属性集合，$P$为正整数集合。对于任何的$X\subseteq I_r$，$attribute(X)=\{x\mid(x,l,u)\in X\}$。

定义2-1如果$\forall(x,l,u)\in X,\exists(x,v)\in T$，有$l\le v\le u$，称事务$T(T\in D)$支持$X(X\subseteq)I$。

定义2-2多值关联规则的蕴涵式为$X\Rightarrow Y$，$X\subset I_r,Y\subset I_r$而且$attribute(X)\cap arrtibute(Y)=\phi$，如果$D$中有$s$%的事务支持$X\cup Y$，且$c$%的支持$X$的事务同时也支持$Y$，则该规则的支持度和可信度为$s$和$c$。ARP的两个重要性质：

性质2-1若X是频繁项集，则其所有非空子集都是频繁项集。

性质2-2若X不是频繁项集，则其所有超集都是非频繁项集。

3 UTMTU多维关联规则算法描述(The algorithm description of UTMTU multidimensional association rule)

3.1 UTMTU算法过程描述：

UTMTU算法一共分为两大步骤：

(1)原数据编码：对每个事物每维每层的值用累加值表示。

(2)寻找频繁项集：

a.搜索频繁1项集$L1$：将编码的表标记为布尔向量矩阵。1表示Ti包含该属性，0则相反。通过行“与运算”[3]，将小于最小支持度项舍去，得到频繁1项集$L1$。

b.搜索频繁2项集$L2$：定义3-1：每个属性中包含的频繁1项的个数为该属性的有效属性层次个数，用$Layer_{effective}(Di)$表示。根据有效层次个数≺顺序扫描布尔矩阵，记录每个事物的所有属性的属于$L1$的层次信息，同时对相同信息计数，将结果用一维数组count[DimensionValues]，并依据性质2-2，将不属于$L1$的1项集去掉。

c.按照有效层次个数≺顺序将$L1$中的层次属性值分别作为矩阵的行和列标号并且扫描count[DimensionValues]计算A_{ij}形成$n\times(n\text{-Count}(D_{last}))$的上三角关系矩阵。若$A_{ij}>0$，$(i,j)$即为频繁2项集，sup=$A_{ij}$。$A_{ij}$取值用以下表示：

$$A_{ij}=\begin{cases}\sum_{i=1}^{m}\text{Count}_i(x,y), & \text{若}\sum_{i=1}^{m}\text{Count}_i(x,y)\ge\min\sup\\ 0, & \text{若}\sum_{i=1}^{m}\text{Count}_i(x,y)<\min\sup\end{cases}$$

d.搜索频繁k项集$L_k(k>2)$：①设root(空)为根节点；②扫描得到上三角矩阵为0的列项量，将列标i作为root子节点，value=0；③查询i行，if $A_{ij}\neq0$且$j\notin D_n(D_n$为count(D_n)的最大值对应的$D_n)$，则将j作为i的子节点value=A_{ij}；④查询第j行，找到$A_{jk}>0$，取布尔矩阵的列向量i、j、k作“与运算”，若$\text{Support}(a_i\wedge a_j\wedge a_k)\ge\min\sup$，则将$k$作为$j$的子节点；⑤判断完所有行得到频繁项集树，取$\min(\text{value}_i,\text{value}_j,\text{value}_k)$即为该频繁$k$项集的支持度。

3.2 UTMTU算法的具体流程图

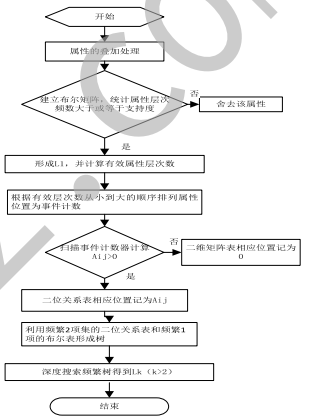

图1 UTMTU算法流程图

Fig.1 UTMTU algorithm flow chart

3.3 UTMTU算法的伪代码如下

输入：交易数据库D，最小支持度Minsup；

输出：D中所有频繁项集Lk

(1)编码：n个属性，每个属性层次有attrib_sum(i)个

for i←1 to n(最大属性种类数)

$layernum=\sum_{1}^{n-1}attribsum(i)+orinum$

(2)统计每个属性次数，除去<minsup的

for i←0 to 总属性个数

for j←0 to 条目数

cntnum[src[i][j]-1]++；

if(cntnum[i])<Minsup)throw层i；

(3)重新整理条目计数结果和总属性个数

if(count(属性i)<Minsup)

delete(含有属性i的条目中i所在位置)

delete(层次属性值表中的属性i)

(4)生成上三角矩阵并筛选L2

Findmax&min(整理后的属性的层次数)

Sort(attrib(i))(从少到多重排属性)

建立层次属性值表(上三角矩阵)

If(某个值<Minsup)此值←0

(5)建立多叉树 找到频繁n项集

for i←0 to 0值列向量的取值个数

for i←0 to L1属性取值个数之和

{AddNode();//依据广度优先算法

Recordtimes();//记录属性出现次数

Parent();//该节点的父节点}

BuildTree(){

Traverse(上三角矩阵)

If(mat2[i][j]>0){addNode()

Traverse(mat2[j][*])}}

for i←0 to MaxNum_Tree

Search(k)；k频繁项集

Search(){If((length==n) && (min(times))>Minsup) Return result;}

4 实例分析(Case analysis)

如表1所示，某一事物数据库中存在16个事物D={Ti|i=1,2,…,16}。每个事物对应的属性集A={Ai|i=1,2,…}。A1含5个属性层次，分别是1、2、3；A2为1—3的3个，A3为1—5的5个，A4为1—…}。A1含5个属性层次，分别是1、2、3；A2为1—3的3个，A3为1—5的5个，A4为1—4的4个，minsup=2。

表1 事物数据库

Tab.1 Transactions database

	A1	A2	A3	A4		A1	A2	A3	A4
T1	5	2	4	3	T9	2	3	2	2
T2	2	1	1	4	T10	5	2	4	3
T3	1	1	3	1	T11	4	2	4	3
T4	2	3	2	2	T12	3	1	3	2
T5	1	1	3	1	T13	2	3	2	2
T6	4	2	4	3	T14	1	1	3	1
T7	2	3	2	2	T15	5	2	4	3
T8	5	2	5	3	T16	3	1	3	2

(1)属性编码如：$a_{(|A1|+|A2|+2)}=\|A1\|+\|A2\|+2$，$\|A_n\|$表示该维层次总数。如$a10=a+b+2=5+3+2=10$。

表2 编码后的数据

Tab.2 The encoded data

	A1	A2	A3	A4		A1	A2	A3	A4
T1	5	7	12	16	T9	2	8	10	15
T2	2	6	9	17	T10	5	7	12	16
T3	1	6	11	14	T11	4	7	12	16
T4	2	8	10	15	T12	3	6	11	15
T5	1	6	11	14	T13	2	8	10	15
T6	4	7	12	16	T14	1	6	11	14
T7	2	8	10	15	T15	5	7	12	16
T8	5	7	13	16	T16	3	6	11	15

(2)搜索频繁项集

①搜索频繁1项集L1：将表2映射为布尔向量矩阵表3。通过行“与运算”，项9、13、17的个数小于2,所以不是频繁的，得到的即是频繁1项集L1={1,2,3,4,5,6,7,8,10,11,12,14,15,16}。取10个结果展示如下：

表3 布尔向量矩阵

Tab.3 The boolean vector matrix

	1	2	3	4	5	6	7	9	8	10
T1					1		1			
T2		1				1		1		
T3	1					1				
T4		1							1	1
T5	1					1				
T6				1			1			
T7		1							1	1
T8					1		1			
T9		1							1	1
T10					1		1			
T11				1			1			
T12			1			1				
T13		1							1	1
T14	1					1				
T15					1		1			
T16			1			1				
count	3	5	2	2	4	6	6	1	4	4

②搜索频繁2项集L2：

根据定义3-1得A1有效层次数为5，A2、A3、A4各为3，则按A2、A3、A4、A1顺序扫描表2，得到count[DimensionValues] (i=1—7)的结果：

Count1[8,10,15,2=4,　　Count5[7,12,16,4]=2,

Count2[6,11,14,1]=3,　　Count6[6,2]=1,

Count3[7,12,16,5]=3,　　Count7[7,16,5]=1,

Count4[6,11,15,3]=2,

③依次扫描count表，按照行标6,7,8,10,11,12,14,15,16,1,2,3,4,5和列标分别6,7,8,10,11,12,14,15,16的顺序形成9*14的上三角矩阵表4。如(1,11)即为频繁2项集,sup=5。

表4 上三角矩阵图

Tab.4 Upper triangular matrix

	6	7	8	10	11	12	14	15	16	1	2	3	4	5
6					5		3	2		3		2		
7						5			6				2	4
8				4				4			4			
10								4			4			
11							3	2		3		2		
12									5				2	3
14										3				
15											4	2		
16													2	4

④搜索频繁k项集L_k(k>2)：根据2.4的方法组织树如图2。如查询表4，因6、7、8列向量为0，则将6、7、8作为root子节点，value=0；查询“6”这一行，11、14、15对应的A_{ij}均≠0且$\notin D_1$，将11、14、15作为6的子节点，value=5,3,2，同理得7和8子节点；查询第11行，找到(11,14)，$Sup(a_6 \wedge a_{11} \wedge a_{14}) \geq 2$，则14为11子节点，同理得该层其他值的子节点。查询(14,1)，由于$Support(a_6 \wedge a_{11} \wedge a_{14} \wedge a_1) \geq 2$所以1为14子节点，此时因为$1 \in A_1$，所以将1作为叶子节点。树的每个分枝都是频繁项集。如(6,11,14)是频繁3项集，sup=3，(6,11,14,1)是频繁4项集，

sup=3。以此类推。

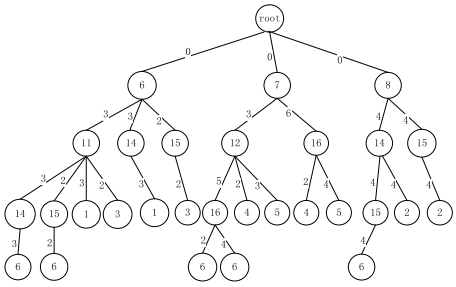

图2 频繁项集树图

Fig.2 The tree of frequent item sets

5 结果比较与性能评价(Results comparison and performance evaluation)

UTMTU算法只扫描一次数据库，复杂度主要在树的构建，假设有d维，L_1有m个，节点n对应的项数是m−n个，因此总共需要进行的比较次数是 $\sum_{n=1}^{m-D_{avr}}(m-n)$次，故建立这棵树的复杂度为$O(n^2)$，而改进的面向多维的IApriori复杂度为$O(2^k)$(k为最大频繁项集的维数)，因此维数数据量越大，本算法越优。

图3—图5分别从不同最小支持度、事物数和维数三个方面来考察IApriori和UTMTU算法的处理效率。随事物数增加，IApriori耗时增幅大，UTMTU增幅缓慢，效率近似是IAPriori的2—3倍，随项数增加效果会更明显。而minsup越小，UTMTU效率越高，因为其对大维数据排序，挑选大维个别项作搜索，尤其适合大数据量处理，且每增加一维，IApriori耗时成倍数增加，到8维之后曲线开始陡升，UTMTU增幅缓慢，且4维后远远小于IAprapri。

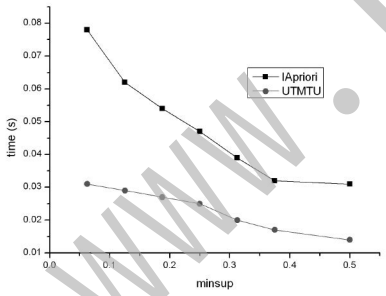

图3 不同最小支持度下的时间对比图

Fig.3 Comparison of time under different minimum support

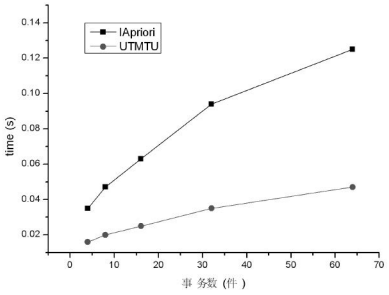

图4 不同事务数下的时间对比图

Fig.4 Comparison of time in different transactions

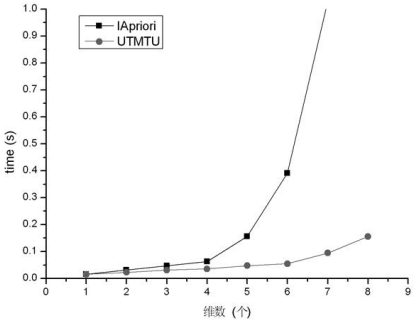

图5 不同维数下的时间对比图

Fig.5 Comparison of time in different dimensions

6 结论(Conclusion)

UTMTU算法整个过程只扫描一次数据库，不涉及候选项集，采用三角矩阵和树的储存方式，避免了瓶颈问题1和2。该算法特别定义并比较有效层次个数，按照≺的顺序来排列各属性的位置，把有效层次最多的属性放在最后，那么在上三角矩阵中就可以省去最多的行且在构建频繁项集树时，不用扫描最多的属性层。这对于属性层次多的多属性事物的处理是非常有效的。

参考文献(References)

[1] Czibula G,Marian Z,Czibula I G.Software defect prediction using relational association rule mining[J].Information Sciences,2014,264(183):260−278.

[2] Dandu S,Deekshatulu B L,Chandra P.Improved Algorithm for Frequent Itemsets Mining Based on Apriori and FP−Tree[J].Global Journal of Computer Science and Technology,2013,13(2):12−16.

[3] Hira S,Deshpande P S.Data analysis using multidimensional modeling,statistical analysis and data mining on agriculture parameters[J].Procedia Computer Science,2015,54:431−439.

[4] 董豆豆,李登峰,程春田.一种基于关系数据库的模糊关联规则算法[J].计算机工程与应用,2003(9):186−188.

[5] Junqiang Liu,Yunhe Pan.Mining Frequent Item Sets by Opportunistic Projection[C].The 8th ACM SIGKDD international conference on Knowledge discovery and data mining,2002:229−238.

[6] Gouda K,Hassaan M.Efficiently Using Prime−Encoding for Mining Frequent Itemsets in Sparse Data[J].Computing and Informatics,2013,32(5):1079−1099.

[7] 李国雁,沈夏炯.一种基于关系矩阵的多维关联规则挖掘算法[J].计算机工程与科学,2008(30):72−74;77.

[8] Mukhopadhyay A,et al.Survey of multiobjective evolutionary algorithms for data mining:Part II[J].IEEE Transactions on

(下转第4页)