基于格论的关联规则挖掘算法的研究

蒋 震[1] 葛 垚[2] 黄 剑[3] 文俊浩[4]

(重庆交通学院计算机与信息学院 重庆 400074)[1] (重庆大学计算机学院 重庆 400030)[2]

(重庆大学自动化学院 重庆 400030)[3] (重庆大学软件学院 重庆 400030)[4]

摘 要 本文通过对关联规则挖掘中由候选项集生成频繁项集算法的分析,引入了格论的一些思想来改进算法,其中心思想是:通过在属性集和事务数据库的基础上进行建格,然后在格的基础上直接进行规则提取。在实验的基础上对Apriori算法和改进的算法进行了比较,实验结果表明,在特定的数据库中,改进的算法在挖掘效率上优于Apriori算法。

关键词 数据挖掘,关联规则,候选项集,频繁项集,闭项集,格论,闭项集格

Research on Association Rules Mining Algorithm Based Lattice Theory

JIANG Zhen[1] GE Yao[2] HUANG Jian[3] WEN Jun-Hao[4]

(Computer and Information Department of Chongqing Communication College, Chongqing 400074)[1]

(College of Computer Chongqing University ,Chongqing 400030)[2] (College of Automation,Chongqing University,Chongqing400030)[3]

(College of Software Engineering,Chongqing University,Chongqing 400030)[4]

Abstract By analyzing the algorithm from candidate itemsets to frequent itemsets in the association rules, we introduce the lattice theory and its some conceptions and present a improved algorithm called close algorithm which prevent from generat from a large number of candidate itemsets, and deduce the number of database passes. Based on the empirical result, we compare the Apriori algorithm with this improved algorithm. Experimental results demonstrate the more efficiency of the novel algorithm to the special database.

Keywords Data mining, Association rule, Frequent itemset, Closed itemset, Candidate itemset, Lattice theory, Closed itemset lattice

1 前言

关联规则挖掘就是从大量的数据中挖掘出有价值的描述数据项之间相互联系的有关知识。随着大量数据不停地收集和存储,人们对从数据库中挖掘关联规则越来越感兴趣。近几年,关联规则的研究一直是数据挖掘研究领域中的一个重要的课题。

Apriori算法是R. Agrawal本人在他们自己的AFS算法基础上于1994年提出的[1]。它是挖掘产生布尔关联规则所需频繁项集的基本算法,也是一个很有影响的关联规则挖掘算法。围绕着怎样精简候选项集 C_k 的大小和减少对数据库的扫描次数,已经提出了不少Apriori算法的变形,例如有基于哈希表技术的DHP算法;减少后面循环中所扫描的视图记录数的事务压缩法;Savasere等提出的划分数据法;H. Toivonen使用抽样来改进的方法;动态项集计数法;发现时间变化的序列模式法等。更进一步的研究涉及到分布式和并行环境下挖掘关联规则[2]。

本文中提出了通过格论的原理来提高算法的性能,也就是怎样减少候选项集 C_k 的数目。因为我们通过研究发现属性集的幂集与事务数据库中的事务的幂集是格,那么利用格论的知识可以在很大程度减少候选项集 C_k 的数量,这与关联规则挖掘研究中减少候选项集的研究方向是一致的。

2 基本定义

格论来源于数论、逻辑、几何等领域,已成为数学的一个重要分支,是许多数学领域研究工作的重要工具。首先,引入格论的一些基本概念和定义,作为关联规则挖掘算法基础。引入格论的目的,是为了将候选项集做到尽可能小,以减小系统开销,减少了对数据库的扫描次数。

2.1 闭项集格

定义1 假设给定信息表 $D=(O,I,R)$,其中 O 是对象集合,I 是属性集合,R 是 O 和 I 之间的一个二元关系[3],记为 $R\subseteq O\times I$,且这个信息表也称作数据挖掘环境。每一个 $(o,i)\in R$ 代表 $o\in O$ 与 $i\in I$ 相关。

定义2 设 $D=(O,I,R)$ 为一个数据挖掘环境,对于 $O\subseteq O$ 和 $I\subseteq I$,我们定义:

$f(O):2^O\to 2^I \quad g(I):2^I\to 2^O$

$f(O)=\{i\in I\mid \forall o\in O,(o,i)\in R\}$

$g(I)=\{o\in O\mid \forall i\in I,(o,i)\in R\}$

$f(O)$ 表示对象集合 O 中所有对象 o 中包含的属性的最大公共部分。$g(I)$ 表示包含属性集 I 中属性的所有对象 o 的集合。

这里定义 $\{f,g\}$ 为在 O 的幂集 2^O 和 I 上的幂集 2^I 上的一个Golois联络,同时我们也知道幂集 2^O 和幂集 2^I 是格,因而可以推出 $h=g(f(O))$ 和 $h'=f(g(I))$ 分别是格 2^O 和 2^I 上

蒋 震 讲师,主要研究方向:数据挖掘、电子商务、网络信息系统;葛 垚 硕士,主要研究方向:数据挖掘、电子商务、信息安全技术。

的一个闭包运算[4]。

定义 3 设 $C\subseteq I$，当且仅当 $h(C)=C$ 时，称 C 为闭项集(closed itemset)。$h(I)$为包含项集 I 的最小闭项集。

定义 4 设 C 为采用 Galois 联络从 D 中得到的闭项集集合，$L_C=(C,\leqslant)$是完备格，也称其为闭项集格(closed itemset lattice)。它有如下性质：

(1)对任意元素 $C_1,C_2\in L_C$，C_1 与 C_2 存在偏序关系，即 $C_1\leqslant C_2$，当且仅当 $c_1\subseteq C_2$；

(2)所有 L_C 的子集都有一个最大下确界，一个最小上确界。

2.2 闭挖掘算法

闭挖掘算法是基于修剪闭项集格来发现频繁项集[5]的算法。所谓闭项集是指一组对象中最大的公共项集。对于确定关系的(对某一数据库而言)闭项集格与概念格(也称为 Galois 格)是对偶同构的[3]。使用闭项集格来找出频繁项集能够提高关联规则发现的效率。事实上，项集集合中既是闭项集又是频繁项集的集合的数目要比频繁项集集合的数目少得多。我们通过减小搜索空间，来减少数据库的扫描次数和生成频繁项集所带来的 CPU 的总开销。

闭项集格的框架可以得到以下性质：

(1)任意频繁项集的子集是频繁项集。

(2)任意非频繁项集的超集是非频繁项集。

(3)任意频繁闭项集的子集是频繁闭项集。

(4)任意非频繁闭项集的超闭集(频繁闭项集的超集)是非频繁闭项集。

(5)最大频繁项集与最大频繁闭项集相等。

(6)频繁项集 I，但不是频繁闭项集的支持度等于包含 I 的最小频繁闭集。

基于以上性质，从 $D=(Q,I,R)$中挖掘关联规则的步骤有以下三步：

1. 从 D 中找出所有频繁闭项集，即所有支持度大于最小支持度并且是闭项集。

2. 从步骤 1 得到的频繁闭项集中得到所有频繁项集。

3. 对于由步骤 2 得到的每一个频繁项集 I，生成置信度大于最小置信度的关联规则。

步骤 1 是算法中计算最集中的部分，需要对事务数据库进行扫描，但比 Apriori 算法中扫描事务数据库的次数大大减少了。除此步骤以外，不需要对事务数据库进行扫描，事实上，步骤 1 已经给出步骤 2 和步骤 3 中所需要的信息，特别是频繁闭项集的支持度能够被用来计算出频繁项集的支持度。

3 基于格论的关联规则挖掘定义

3.1 频繁项集

定义 5 设 I 是项集且 $I\subseteq I$，项集 I 的支持度为：$support(I)=\|g(I)\|/\|O\|$

定义 6 如果项集 I 支持度大于等于最小支持度，我们称其为频繁项集。D 中频繁项集集合 L 记为：$L=\{I\subseteq I\mid support(I)\geqslant minsupport\}$

定义 7 设 L 为频繁项集集合，定义集合 M 为最大频繁项集集合：$M=\{I\in L\mid\neg\exists I\in L,I\subset I\}$

性质 1 频繁项集的任意非空子集还是频繁项集。

性质 2 非频繁项集的任意超集还是非频繁项集。

3.2 频繁闭项集

定义 8 如果 D 中的闭项集 C 的支持度大于等于最小支持度，称其为频繁闭项集。定义集合 FC 为 D 中所有频繁闭项集的集合：$FC=\{C\in I\mid C=h(C)$且 $support(C)\geqslant minsupport\}$

定义 9 设 FC 为频繁闭项集集合，定义集合 MC 为最大频繁闭项集集合：$MC=\{C\in FC\mid\neg\exists C\in FC,C\subset C\}$

性质 3 频繁闭项集的任意非空子集都是频繁项集。

性质 4 非频繁闭项集的任意超闭集都是非频繁的。

性质 5 任意项集 I 的支持度等于包含项集的最小闭项集的支持度，即：

$Support(I)=support(h(I))$

性质 6 最大频繁项集集合中的元素与最大频繁闭项集集合中的元素是相同的。

3.3 关联规则

定义 10 关联规则是形如 $I_1\Rightarrow I_2$ 的蕴含式，I_1,I_2 均表示项集，且 $I_1,I_2\subset I$，$I_1\cap I_2=\varnothing$。I_1 称为规则的前项(antecedent)，I_2 称为规则的后项(consequent)。对于规则 $r:I_1\Rightarrow I_2$，我们利用 Galois 联络来给出它的支持度和置信度的定义：

$Support(r)=\|g(I_1\cup I_2)\|/\|O\|$

$Confidence(r)=\|support(I_1\cup I_2)\|/\|support(I_1)\|=\|g(I_1\cup I_2)\|/\|g(I_1)\|$

定义 11 关联规则发现的任务是找出有效的关联规则，即那些支持度大于最小支持度、置信度大于最小置信度的规则。设 AR 是 D 中所有有效关联规则的集合，我们利用最大频繁闭项集 MC 中的集合 C 来定义 AR：

$AR(D,minsupport,minconfidence)=\{r:I_2\Rightarrow I_1\mid I_2\subset I_1,I_1\in L=\bigcup_{C\in MC}2^C and\ confidence(r)\geqslant minconidence\}$

4 挖掘算法

4.1 发现频繁闭项集

在 Apriori 算法中，项目是按词典序排列的。发现频繁闭项集的伪代码由算法 1 给出。表 1 给出了各符号及其代表的含义。

FCC_i 是由 FC_{i-1} 得来的，FCC_i 中域 *generators* 包含长度为 i 的项集，每一个长度为 i 的项集是一个 *generator*。域 *closure* 中的集合对应 *generators* 中每一个集合，并且是其闭项集，域 *support* 是域 *closure* 中每个闭项集的支持度。

集合	域	包含内容
FCC_i	generator	生成的长度为 i 的项集
	closure	通守闭包运算得到候选闭项集 closure=h(generator)
	suppor	上面候选闭项集的支持度 support=count(closure)=count(generator)
		根据性质 5
FC_i	generator	去掉 FCC_i 中支持度小于最小支持度的 generator
	closure	频率闭项集
	support	上面频繁闭项集的支持度

算法 1

(1)generators in $FCC_1\leftarrow\{1-$itemsets$\}$//生成 FCC_1 中的 generators，generators 包含所有长度为 1 的项集
(2)for(i←1;FCC_igenerator≠∅;i++) do begin
(3) closures in $FCC_i\leftarrow\varnothing$;
(4) supports in $FCC_i\leftarrow 0$
(5) $FCC_i\leftarrow$Gen-Closure(FCC_i);//得到每一个 generator 的闭项集及其支持度
(6) forall candidate closed itemsets c∈FCC_i do begin
(7) if (c. support≥ minsupport) then //如果 c 是频繁项集

```
(8)      FCi←FCi∪{c}; //将 c 并入 FCi 中
(9)   end
(10)     FCCi+1Gen-Generator(FCi);//产生 FCCi+1中的 generator
(11)  end
(12)  Answer FC←∪j=1^{i-1}{FCj. closure, FCj. support};
```

算法 2 Gen-Closure 函数

```
(1)forall objects o∈O do begin
(2)   G0 ←Subset(FCCi. generator, f({o}));//若 generator 是 f
        ({o})的子集,将其并入 G0
(3)   forall generators p∈G0 do begin
(4)     if (p. closure=∅) then
(5)       P. closuref←({o});
(6)     else p. closure←p. closur∩ef({o});
(7)      p. support ++
(8)   end
(9)end
(10)Answer←U{ c∈FCCi|c. closure≠∅};
```

算法 3 Gen-Generator 函数

```
(1)insert into FCCi+1. generator
(2)select p. item1, p. item2, …, p. itemi, q. itemi
(3)from FCi. generator p, FCi. generator q
(4) where p. item1 = q. item1, …, p. itemi-1 = q. itemi-1, p, itemi<
    q. item;
(5)for all generators p∈FCCi+1. generator do begin
(6)   for all i-subsets of p do begin
(7)     If (s∉FCi. generator) then
(8)       delete p from FCCi+1. generator;
(9)   end
(10)end
(11)for all generators p∈FCCi+1. generator do begin
(12)Sp←Subset(FCi. generator, p);
(13)  for all s∈Sp do begin
(14)    if (p⊆s. closure) then
(15)      delete p from FCCi+1. generator;
(16)  end
(17)end
(18) Answer←U{c∈FCCi+1};
```

4.2 生成频繁项集

下面给出由频繁闭项集导出频繁项集的算法。

算法 4

```
(1)k←0;
(2)for all frequent closed itemsets c∈FC do begin
(3)  L‖c‖←L‖c‖∪{c};//将频繁闭项集按照长度插入 Li
(4)  If(k<‖c‖) then k←‖c‖;
(5)end
(6) for(i←k;k>1;i−) do begin
(7)  forall itemsets c∈Li do begin
(8)    forall (i−1)-subsets s of c do begin
(9)      if (s∉Li-1) then begin
(10)       s. support←c. support;
(11)       Li-1←Li-1∪{s}; //将 s 插入 Li-1的最后
(12)       end
(13)     end
(14)  end
(15)Answer←∪i=1^k Li
```

4.3 关联规则生成

关联规则生成这一部分我们仍采用 Apriori 算法中关于规则生成的算法，这里我们不再详细论述。

4.4 实验结果

我们在某市邮区中心局物资管理信息系统中所积累的数据的基础上对上述算法进行了比较，在已有的数据库上我们进行了一些转换工作，将事务数据库原始数据库的离散值进行分类和量化，转换后的数据库中的值是布尔型的。经过转换生成的事务数据库的规模为记录数 40000 余条，项目(属性)个数为 200 条(可以分为 6 类)，图 1 是两种算法的挖掘情况的对比图。

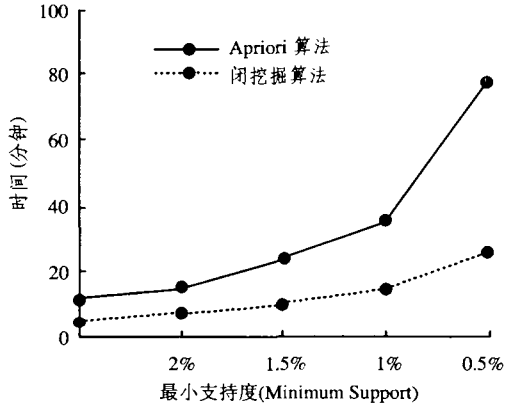

图 1 两种关联规则挖掘算法的对比图

从图中可以看到闭挖掘算法所耗用的时间比 Apriori 所耗用的时间要少，实验结果表明闭挖掘算法是优于 Apriori 的。

结束语 本文中通过引入格论，对关联规则的经典算法 Apriori 算法进行了改进，此算法通过减少候选项集的方法，既减少了寻找频繁项集的系统开销，又减少了数据库的扫描次数。我们对这两种算法进行了对比，实验结果表明，算法在特定的数据库中是有效的。

参 考 文 献

1 Keim D A. Visual techniques for exploring databases. In: Tutorial Notes, 3rd Int , Conf. on Knowledge Discovery and Data Mining (KDD97), Newport Beach, CA, Aug. 1997
2 何炎祥，等. 基于网络环境的分布式 KDD 及 Data Mining 研究. 小型微型计算机系统，1999(10)
3 王志海，等. 概念格上规则提取的一般算法及渐进式算法. 计算机学报，1999(1)
4 胡长流，宋振明. 格论基础. 河南大学出版社，1990
5 Pasquiter N, Bastide Y, Taouil R, et al. Efficient Mining of Association Rules Using Closed Itemset Latices. Information Systems, 1999, 24(1): 25～26
6 朱明，等. 数据挖掘. 中国科学技术大学出版社，2002
7 Zaki M J, Ogihara M. Theoretical Foundations of Assocation Rules. Computer Science Department, University of Rochester, Rochester NY 14627
8 Lauesen S. Software Requirements Styles and Techniques. 电子工业出版社，2002

(上接第 99 页)

该 URP 平台为“1＋7＋1”(一个门户、七(N)个应用、一个平台。1 个门户：是指让用户通过一个统一的门户/界面、一次认证，可以访问整个网络的应用。1 个平台：是指将所有的应用都集成在一个统一的 URP 平台上[7]。其中数据中心和门户服务的设计和构建采用了基于视图的方法，取得了良好的效果。

参 考 文 献

1 殷娜，傅明. 数字校园的建设发展与规划. 兰州铁道学院学报自然科学版，2002，4(8)：142～144
2 蒋东兴，史宗恺，等. 大学资源计划的方案研究. 清华大学学报自然科学版，2004，4(44)：572～576
3 蒋东兴，许庆红，谢矜，等. 高校信息化建设的一体化思路与实践. 清华大学，2004
4 黄卫卫，陈怀楚，高国柱，等. 基于交换的数据存储模式研究. 清华大学计算机与信息管理中心，2004
5 王家顺，胡耀光，王田苗，等. 企业分布式信息共享和集成技术. 微计算机应用，2001，7(4)：213～217
6 沈培华. 高校信息化研究与实践. 见：全国高校信息化研究 2003 年会
7 杨丹，等. 重庆大学虎溪数字化校园规划方案 1.5 版. 2005