相联规则发现的一般性算法研究*)

On General Algorithms of Association Rules Discovery

叶阳东　姬安明　潘玉英　范　明
(郑州大学计算机科学系　郑州450052)

摘　要　大型事务数据库中相联规则的发现是KDD中一个很重要的问题。本文描述了相联规则发现的一般性算法，对其核心问题进行了全面和较深入的探讨，并提出了一些提高算法效率的方法。

关键词　相联规则，事务，支持度，置信度，强项集

相联规则的发现是数据库应用领域KDD(Knowledge Discovery in Database)近期所研究的一个非常重要的问题。在实际的大型事务数据库(如超级市场的事务数据库)中，发现出规则：购买物品A和B的客户有95%的客户购买物品D和C，用规则R:A,B⇒C,D(95%)来表示。这在分类设计、商店的布局、隶属邮寄、产品的排放、市场分析等多方面大有应用。

相联规则发现算法研究中，一方面借鉴了许多其他发现算法的发现技术，其中包括函数依赖性、强规则、分类规则、因果规则、聚类分析等[1-3]；另一方面，算法在发现多种形式的规则方面进一步完善，这包括一般化规则[4]、分层规则[5]等。然而，所有这些发现算法的基本步骤都是相同的，其核心问题在于如何能快速有效地求出强项集。本文主要描述了相联规则发现的一般性算法，并对强项集的求得这一核心问题进行了较深入的探讨，提出用抽样、分区、引入领域知识等方法来提高算法的效率。

一、相关知识

·事务　$I=\{i_1,i_2,\cdots,i_m\}$是由m个不同点组成的集合，i_k称为项。事务是I上的一个子集，也就是由一组点$i_i,i_k,\cdots,i_p\in I$构成，每个事务均有一个唯一标识符Tid。不同事务一起构成了相联规则发现的事务数据库D。

·项集X的支持度Support(X)　表示项集X的重要性。设$X\subset I$为项集，B=D中包含X的事务的数量，A=D中所有事务的数量，项集X的支持度Support(X)=B/A。

·最小支持度　发现任务所要求的项集的最小支持度。只有满足于最小支持度的项集才有可能在相联规则中出现，这些项集称之为强项集(large itemset)。

·规则置信度(confidence)　规则可靠性的度量。对于规则R:$X\Rightarrow Y$，其中$X,Y\subset I$，规则R的置信度confidence(R)=support$(X\cup Y)$/support(X)。

·相联规则(association rule)　形式是$X\Rightarrow Y$ {F}，其中$X,Y\subset I$，并且$N\cap Y=\varnothing$，X称为先决条件，Y称为规则的结果，F为规则的置信度。

二、相联规则的发现步骤

相联规则的发现问题实质上是在满足于最小支持度的项集中，找出满足于最小置信度的相关联的规则。在D非常大的情况下，该问题显得尤其复杂，从而成为KDD近期研究的热点。所有的发现算法无论它采用什么数据结构，其复杂程度、效率如何，最终都可分为如下几步：

步骤1　预处理与发现任务有关的数据。根据具体问题的要求对数据库进行相应的操作，从而构成规格化后的事务数据库D。

步骤2　针对D，求出所有满足于最小支持度的项集L.set(即强项集)。在D很大、I中项数m非常大的情况下，该问题就构成了发现算法的核心。

步骤3　提取满足于最小置信度的规则集。方法为：

For all itemset∈ L.set and itemset. flag * =

*)本文得到国家自然科学基金项目和河南省自然科学基金项目部分资助

0and X⊂itemset and Y⊂X Do
{F = support(X)/support(X-Y); if F ≥ min confidence
THEN Rset=Rset ∪ {X-Y⇒Y{F}}}

步骤4 解释并输出Rset。

说明:L.set中的每一项itemset有三个域:set、sup、flag,分别表示项集、支持度、被包含标志。若flag=1,说明在L.set中存在有itemset的超集;flag=0就表示L.set中无itemset的超集。flag标志域的设置会提高步骤3的速度。

三、算法中的关键性问题--L.set的产生

由以上的发现步骤可知,在求取L.set时,由于不同的项集数量可达到2^m个,况且数据库中的事务可能很多,若对所有的不同项集都进行支持度的计算,几乎是不可能的。目前有关相联规则的发现算法主要是集中研究如何能快速有效地提取L.set,因此其焦点有二:一是能对项集进行充分地剪支,尽最大可能地最小化有用项集;二是最小次数地扫描事务数据库D,从而提高算法的效率。

为了最大可能地减少项集的组合,所有的发现算法在产生强项集时都遵循这样的原则:任何强项集的子集都是强项集;任何弱项集的超集都是弱项集。这样在求强项集时,算法首先求出项数为一项的强项集L.1.itemset,再由L.1.itemset产生项数为二的候选项集C.2.itemset,扫描D计算支持度求出L.2.itemset,依次类推产生C.k.itemset,扫描D求出L.k.itemset。其描述如下:

```
(1) For all Tid∈D Do{对一项集进行计数求支持度};
(2) L.1.itemset1=支持度大于最小支持度的所有一项集;
(3) L.set1=L.1.itemset1;
(4) k1=2;//k代表扫描D的次数
(5) While (L.k-1.itemset≠∅)Do
(6) {
(7) C.k.itemset1=generate.k.itemset from L.k-1.itemset1;
(8) For all Tid ∈ D Do{对Tid所包含的所有C.k.itemset中的项集计数求支持度}
(9)
(10) L.k.itemset1={itemset |itemset ∈ C.k.itemset and itemset.sup≥mini-support};
(11)
(12) L.set1=L.set ∪ L.k.itemset1;
(13) K1=K+1
(14) };
(15) For all itemset ∈ L.set and itemset.flag = 0 Do
     {Forall X⊂itemset and
(16) X ∈ L.set Do X.flag=1} //将L.set中的所有itemset的子集的flag标志置为1
(17)
```

四、提高算法效率的几种方法

由求取L-set算法的(7)、(8)、(5)—(14)知,要提高算法的效率必须注意以下几个环节:

1. 怎样快速地产生C.k.itemset。由算法的(6)可知C.k.itemset是由不同的L.k-1.itemset中的项集结合而产生的。为提高算法的效率可对L.k-1.itemset中的项集进行预处理,对其进行项排序必然会提高C.k.itemset的产生速度。

2. 减少扫描事务数据库的次数。由算法的(4)—(13)可知,若L.set中的最大项集的长度是K(含有K个项),则要对数据库扫描K次。怎样减少数据库的扫描次数是提高算法效率的重要途径之一。为减少数据库的扫描次数,可先对D进行预处理,例如分区,扫描D的分区产生所有的不同长度的候选项,最后一次性扫描D,计数并产生L.set。但必须要注意怎样分区,怎样提高项计数的速度。

3. 怎样提高项集的计数速度。每次扫描数据库都要对C.k.itemset中的项集进行测试计数,可通过对事务的编码和建位图(Bitmap)的方法,提高每个事务对项集的计数速度。

4. 最大程度地对候选项集进行剪支。该问题是目前许多算法所研究的热点,在分层相联规则和广义相联规则的发现算法中[4,5],利用抽样、预测和分层代码去进行候选项集的优化,从而使得项集的计数速度和数据库的扫描速度均有较大提高。在算法中引入领域知识去剪除毫无意义的项集是算法提高效率的途径之一。

结束语 出于相联规则的重要性和发现算法的不完善,今后一段时间内的许多研究工作仍将围绕着如何提高算法的效率而进行。如何在算法中引入领域知识对算法进行优化,以及对数据库进行抽样预测优化候选项集,是我们今后一段时期所工作的目标。

参考文献

[1] Gregory Piatetsky-Shapiro, Discovery, Analysis, and Presentation of Strong Rules, in G. Piatetsky-Shapiro and W. J. Frawley(eds), Knowledge Discovery in Database, AAAI/MIT Press, 1991

[2] Jiawei Han et al., Knowledge Discovery in Database: An Attribute-Oriented Approach. In Proc. of the 18th Intl. Conf. on VLDB, Canada, 1992

[3] Ashok Savasere et al., An Efficient Algorithm for Mining Association Rules in Large Databases, In Proc. of the 21th Intl. Conf. on VLDB, Swizerland, 1995

[4] Ramakrishnan Srikan, Rakesh Agrawal, Mining Generalized Association Rules, Same to[3]

[5] Jiawei Han, Yongjian Fu, Discovery of Multiple-Level Association Rules from Large Databases, Same to[3]