计算机科学1999Vol. 26№. 3

# 国际上关联规则发现研究述评*)

International Researches on Discovery of Association Rules

欧阳为民[1] 郑 诚[1,2] 蔡庆生[2]

(安徽大学计算中心 合肥 230039)[1] (中国科学技术大学计算机系 合肥 230072)[2]

**Abstract** This paper provides a survey of the itemset method for discovery of association rules. We discuss a number of variations of the association rules problem that have been proposed in the literature up to now. Some inherent weaknesses of the frequent itemset method of association rules discovery have been explored. We also discuss some other formulations of associations that can be viable alternatives to the traditional association rules generation method.

**Keywords** Knowledge discovery, Data mining, Association rules, Frequent itemset

## 1. 引言

近年来,数据发掘(Data Mining),亦称数据库中的知识发现(Knowledge Discovery in Databases,简称 KDD),受到当今国际人工智能与数据库界的广泛重视[1,2]。关联规则是 KDD 研究中的一个重要研究课题。该问题是 R. Agrawal 等人提出的,目的是要在交易数据库中发现各项目之间的关系[3,4]。例如,有这样一条关联规则:黄油、牛奶⇒面包(30%,2%)。其含义是购买了黄油和牛奶的顾客还将购买面包,30%、2%分别是该规则的信任度和支持度。

一般地,令 $I=\{i_1,i_2,\cdots,i_m\}$ 为项目集,$D$ 为事务数据库,其中每个事务 $T$ 是一个项目子集($T\subseteq I$),并另有一个唯一的顾客标识符 TID。我们说事务 T 包含项目集 $X$,如果 $X\subseteq T$。关联规则是形如 $X\Rightarrow Y$ 的逻辑蕴含式,其中 $X\subset T$,$Y\subset T$,且 $X\cap Y=\varnothing$。如果事务数据库中有 $s$%的事务包含 $X\cup Y$,那么我们说关联规则 $X\Rightarrow Y$ 的支持为 $s$;如果事务数据库中包含 $X$ 的事务中有 $c$%的事务同时也包含 $Y$,那么我们说关联规则 $X\Rightarrow Y$ 的信任为 $c$。

最著名的,也许是最重要的关联规则发现算法是 R. Agrawal 等人提出的 Apriori 算法。该算法将关联规则的发现分为两步。第一步是识别所有的频繁项目集(frequent itemset,过去称为 large itemset),即支持不低于用户最低支持的项目集。第二步是从频繁集中构造其信任不低于用户最低信任的规则。识别或发现所有的频繁项目集是关联规则发现算法的核心,是计算量最大的部分。如果有 $m$ 个项目,那么就有 $2^m$ 个可能的频繁集。这构成了 $I$ 上的可能解空间。关联规则的构造相对是容易的[5]。

为了生成频繁项目集,Apriori 算法首先遍历数据库,收集每个项目集的支持,取其支持不低于最低支持度的项目集构成频繁1-项目集的集合 $L_1$;然后,两两链接 $L_1$ 中项目集,形成候选2-项目集的集合 $C_2$,再次遍历数据库,收集每个候选项目集的支持,取其支持不低于最低支持度的项目集构成频繁2-项目集的集合 $L_2$;如此等等,不断迭代,直到新的候选集 $C_k$ 为空时为止。注意,由 $L_i$ 生成 $C_i$ 时,彼此链接的两项目集的前($i-2$)个项目是相同的。显然,该方法要对数据库作多次遍历。如果频繁项目集的长度最长为 $p$,就要遍历 $p$ 次。

## 2. 关联规则研究概况

R. Agrawal 等人在文[3]中首先提出关联规则发现算法之后,该问题得到了国际众多研究者的广泛研究,提出了许多形形色色的算法。在 R. Agrawal 等人的后继论文[4]中,他们提出了候选项目集 $C_k$ 的修剪技术,从而显著提高了频繁项目集的发现效

*)本课题得到国家自然科学基金项目资助。欧阳为民 副教授,博士,主要研究方向:KDD,机器学习,人工智能及其应用。郑诚 讲师,在职博士生,主要研究方向:KKD,机器学习,人工智能及其应用。蔡庆生 教授,博士生导师,主要研究方向:机器学习,知识发现,协调式人工智能。

率，相应的算法称为 Apriori 算法。该算法所用的修剪技术是基于这样一个事实，即频繁项目集的任何子集必定也是频繁的。这样，如果某项目集 $I \in C_k$ 中存在一个不属于 $L_{k-1}$ 的 $(k-1)$-子集，那么该项目集 $I$ 就不可能是频繁的，因而可以从候选项目集 $C_k$ 中删除。于是，算法就不用计算项目集 $I$ 的支持。在同一论文[4]中，为便于高效计算项目集的支持，引入了 Hash 树数据结构。国际上关于关联规则发现的后继工作集中在如下几个方面：(1)削减遍历交易数据库的次数，以降低 I/O 代价；(2)改进频繁项目集的生成效率；(3)提出关联规则发现的并行算法；(4)引进抽样技术，以频繁项目集生成所需的 I/O 和计算代价；(5)扩展关联规则发现问题，如广义/多层(Generalized/Multiple Level)关联规则、定量(Quantitative)关联规则、循环(Cyclic)关联规则，等等。

### 2.1 若干典型的改进算法

首先简要回顾国际研究界在上述几方面所做的工作。为提高频繁项目集的发现效率，Park 等人提出了直接哈希修剪 DHP 算法[5]。研究指出在生成和发现频繁2-项目集上耗费的时间十分可观。DHP 算法试图通过直接哈希修剪技术快速发现频繁2-项目集，借以提高关联规则发现的效率。

Brin 等人提出了动态项目集计数 DIC 算法[6]。其基本思想是在对 k-项目集进行支持计数的同时也对某些(k+1)-项目集进行支持计数，以此提高每次交易数据库遍历的效益，从而减少对交易数据库的遍历次数。在大多数已经提出的频繁项目集发现算法中，$(k+1)$-项目集的支持计数要在 k-项目集的支持计数完成后再进行。然而，DIC 算法只要能初步确定某 $(k+1)$-项目集潜在频繁，就开始对其进行支持计数，这样，该算法能够尽可能早地在完成 $k$-项目集支持计数之前开始对某些 $(k+1)$-项目集进行支持计数。通常，该算法所需的遍历次数要比 Apriori 算法所需的遍历次数少。

分治法是一种常见而有效的算法设计思想。Savasere 等人提出了一种发现频繁项目集的划分算法[4]。该算法分为两个阶段。在第一阶段，算法将数据库划分为 $n$ 个互不相交的部分。每个划分的大小以其全部交易数据可以在内存中进行处理为原则。然后，在每个划分中独立地生成所有长度的局部频繁项目集。令 $P_i$ 为第 $i$ 个划分，在其中所发现的局部频项目集记为 $L_i^j (j=2,3,\cdots,k)$，$i$ 表示划分，$j$ 为项目集长度，$k$ 为该划分中局部频繁项目集的最大长度。在第二阶段，归并全部 $n$ 个划分的相同长度局部频繁项目集，从而生成全局候选项目集。长度为 $j$ 的候选项目集的集合为 $C_j^G = \bigcup_{i=1}^{n} L_i^j$。接着，再次遍历交易数据库，收集 $C_j^G$ 中每个候选的支持数。该算法的核心是基于这样一个观察：如果某项目集是全局频繁的，那么该项目集必定至少属于某局部频繁项目集 $L_i^j$。

上述算法具有良好的可并行性，例如为每个划分均指派一个处理器来生成局部频繁项目集。频繁项目集算法的每次循环结束时，各处理器需要彼此通讯，以发现候选 k-项目集的全局支持计数。这一通讯过程在算法运行时间方面成为瓶颈；而在其他算法框架下，为生成局部频繁项目集，各处理器的时间耗费将成为瓶颈。文[9]提出了在各处理器间共享 Hash 树的方法来生成频繁项目集。关于并行关联规则发现算法可进一步见其它有关参考文献。

目前绝大多数关联规则发现算法均具有一个共同的特性，即都是基于 Apriori 算法所提出的"自底向上"的方法。数据库中的频繁项目集有多长，对数据库的遍历次数就有多少次，因而，这些算法的计算代价一般较高，其成功的关键在于数据库中的频繁项目集通常较短。

最近，Bayardo 提出了一种令人感兴趣的频繁项目集生成算法[10]。为了尽早发现较长的模式，该算法运用了明智的"向前看"技术。这些较长模式的子集就不必进一步考察，可以删除，文[10]的实验结果表明该算法较之 Apriori 算法在性能上有实质性的改进。

由于交易数据库的规模通常都很大，为了提高频繁项目集的生成效率，Toivonen 在文[11]中提出采用随机抽样技术。这样一来，可以节约相当可观的 I/O 代价。由于数据分布往往不均衡，随机抽样可能会引起数据倾斜(Data Skew)，因而，该方法的不足之处在于它有可能导致不精确的结果。为此，Lin 和 Dunham 提出了反数据倾斜的关联规则发现算法[12]。该算法将数据库的遍历次数降低为最多2次。为进一步削减候选项目集的数量并尽可能早地识别非繁项目集，该算法还在数据发掘过程中收集关于数据的知识，并采用了随机抽样技术。

因数据倾斜而产生的问题同样也出现在并行关联规则发现算法中。并行算法通过在不同处理器之间划分交易数据库的方法将负载分布到各个处理器上。每个处理器都会接受交易数据库的一个划分，而这个划分中数据之间的相互关系在整个数据库中没有代表性。并行关联规则发现算法中的有关数据倾斜的问题值得进行很好的研究。

### 2.2 关联规则问题的推广

关联规则最初是在超市数据环境下提出的，其动机是想发现顾客购买物品时各种物品之间是怎样彼此联系的。近年来，人们提出了许多令人感兴趣的扩展和新应用。文[13]提出了在关系型数据库中发现定量关联规则，指出在某种范围内类别和数值属性之间的关系，关系数据库中可以含有类别属性和数值属性。该算法将定量数据离散化为若干不相交的子区间，然后再构造与每个子区间相对应的项目-区间对，接着应用频繁项目集生成算法，从而发现定量关联规则，按照这种离散化方法，该算法通常会产生大量规则，其中可能有相当部分并不令人感兴趣。为此，该文还提出了一种兴趣度量方法。在文[14]中，B. Lent 等人提出了定量关联规则的聚类算法，将若干彼此相关的规则聚成一类，称为定量簇，该算法的目的是要生成以定量簇的形式更自然地表达的规则。

另一个令人感兴趣的扩展是在关联规则发现过程中引入各个项目的概念层次关系，从而提出了广义(Generalized)/多层(Multiple Level)关联规则发现算法[15,16]。其目的是希望在各种层次上，特别是在较高层次上，发现项目集间的关联。Savasere 等人在文[17]中将项目集间的关联分为积极的和消极的两种，积极的关联是指某项目的出现意味着另一项目的出现，而消极的关联是指某项目的出现意味着另一项目不出现。Savasere 等人称表达积极关联的规则为积极关联规则，而表达消极关联的规则为消极关联规则。在文[17]中，Savasere 等人提出了在项目的概念层次关系可利用的情况下，消极关联规则的发现算法，周期关联规则是对关联规则的另一个令人感兴趣的扩展。当数据包含时间成分时，也许可能存在周期性的变化规律，例如，商品的月销售额可能根据季节因素彼此相关。Ozden 等人在文[18]中提出了揭示数据随时间呈周期性变化规律的周期关联规则发现算法。

### 2.3 关联规则的在线生成

由于交易数据库的规模通常都很大，关联规则发现算法的计算代价和 I/O 代价均较高，难以为用户查询提供快速响应。C. Aggarwal 等人在文[19]中讨论了关联规则的在线生成方法。该算法应用了 OLAP 的"preprocess-once-query-many"框架，在相邻格阵预先存储项目集，以快速生成关联规则。该工作令人感兴趣的特点是规则的生成独立于交易数据库的规模和预先存储项目集的数量，算法的运行时间与其输出结果的大小成正比。该方法也可以用于生成包含指定项目的数据发掘查询。另外，该文还讨论了关联规则生成过程中的冗余问题。对给定的支持与信任级别，某规则被认为是冗余的，如果存在另一条规则蕴涵该规则。例如有这样两条规则：(1) {Milk}⇒{Bread, Butter}；(2) {Milk, Bread}⇒{Butter}。显然，第二条规则是冗余的，因为第一条规则蕴涵了第二条规则，典型地，单前件规则可能会蕴涵结果中的许多规则。文[19]提出了生成最小必要规则集的方法。

## 3. 频繁项目集方法的缺点

Apriori 的频繁项目集方法已被证明是在大型数据集中发现关联规则的有用工具。然而，在许多情况下，该方法因计算原因难以推广到其它场合。此外，更重要的是，该方法可能会生成虚假的关联规则。

我们用一个例子来说明这个问题。考察某校5000名学生早晨活动情况。调查显示，有3000名学生打篮球，3750名学生吃点心，2000名学生既打篮球又吃早餐。如果最低支持度和最低信任度分别为40%和60%，我们将会发现关联规则：play basketball⇒eat nosh，其支持度与信任度分别为40%和66.7%。该规则实际上是在误导，因为吃点心的学生在学生总数中所占的比例高达75%，既超过了最低信任度60%，又超过了规则的信任度66.7%。该规则所传达的信息没有价值。事实上，打篮球与吃点心在一定程度上是相互否定的，即参加某项活动将会降低参加另一项活动的机会。考察消极关联规则：play basketball⇒not eat nosh，其支持度与信任度分别为20%和33.3%。尽管该规则的支持与信任均低于指定的最低支持度和最低信任度，但它所传达的信息比前述的积极关联规则更有价值。一方面，如果最低支持度和最低信任度设置得足够低，就会同时得到上述两条相互矛盾的关联规则；另一方面，设置得足够低，就会得到没有什么价值的关联规则。换言之，不论如何设置最低支持度和最低信任度，我们都不可能生成合适的关联规则集。

表1

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| X | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Y | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Z | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

(a)数据集

| 规则 | 支持度 | 信任度 |
|---|---|---|
| X⇒Y | 25% | 50% |
| X⇒Z | 37.5% | 75% |

(b)相应的支持度和信任度

更进一步的例子如表1所示，其中，有3个项目 $X$、$Y$ 和 $Z$。$X$ 和 $Y$ 之间的相关系数为0.577，$X$ 和 $Z$ 之间的相关系数为−0.378，因此，$X$ 和 $Y$ 是积极关联的，$X$ 和 $Z$ 是消极关联的。根据频繁项目集方法，可以发现关联规则 $X\Rightarrow Y$ 和 $X\Rightarrow Z$，其支持度与信任度如表2所示。奇怪的是，规则 $X\Rightarrow Z$ 的支持度与信任度均分别超过规则 $X\Rightarrow Y$ 的支持度与信任度，而且，我们还不可能找到合适的最低支持度和最低信任度，使得仅生成关联规则 $X\Rightarrow Y$，而不生成虚假关联规则 $X\Rightarrow Z$。我们再次陷入困境。

如果最低支持度和最低信任度设置得足够低，那么在很多情况下，特别是在项目分布十分不同的情况下，我们将会得到大量关联规则，其中大多数也许是虚假的关联规则。这与发现关联规则的初衷相悖。通常，最低支持度和最低信任度是根据计算约束和为了将可能生成的规则数控制在一个合理的范围内来设置的。所以，我们不可能肯定何时会遗漏有价值的关联规则，何时又不会。

## 4. 规则的其它评价标准

兴趣度可以用于避免产生虚假关联规则。某规则的兴趣度为其实际强度与基于统计独立假设的期望强度之比。已有的工作一般都集中在利用兴趣度来删除输出中不令人感兴趣的规则。然而，只要在最初的项目集生成阶段支持度仍然是决定因素，我们就无法避免两难处境，即不论如何设置最低支持度和最低信任度，都存在我们不希望出现的问题。

这样，提出其它评价规则的标准就势在必行了。Brin 等人不用支持度和信任度参数，而运用统计学中的相关性概念，在文[7]中提出了相关规则。在文[6]中，他们还讨论了发现蕴涵规则而不是关联规则的问题。规则的蕴涵强度在0到∞之间，蕴涵强度为1表示规则强度恰好为统计独立假设的期望强度，超过1的蕴涵强度表示规则的强度超过了期望强度。这种度量标准比信任度好，因为它有利于发现期望强度的规则。

C. Aggarwal 和 P. S. Yu 两人在文[20]提出了与蕴涵强度类似但颇具特色的"集体强度"(Collective Strength)概念，其目的也在于通过利用"超过期望值"来发现令人感兴趣的关联规则。某项目集的集体强度定义为0到∞之间的一个数，值为0表示完全否定的关系，值为∞表示完全肯定的关系，值为1表示项目集恰好以其期望值出现。该文还提出了一个与项目集支持相反的"违背"概念，即如果某交易没有包含某项目集中的全部项目，我们就认为该项目集违背该交易。概念违背指出某顾客可能购买项目集中的某些项目，而可能不购买其它项目。

某项目集 $I$ 的违背率 $V(I)$ 定义为项目集 $I$ 的违背交易的次数与全部交易总数之比，项目集 $I$ 的集体强度 $C(I)$ 定义如下：

$$C(I)=\frac{1-V(I)}{1-E[V(I)]}\times\frac{E[V(I)]}{V(I)} \qquad (1)$$

其中，$E[V(I)]$ 为 $V(I)$ 的期望值。

从试图建立项目间高度相关关系的角度看，项目集违背某交易是件"坏事"，因此，$C(I)$ 代表的是出现"坏事"的比例，而 $(1-V(I))$ 就是出现"好事"的比例。上述关于集体强度的定义可直观地改写如下：

$$C(I)=\frac{GoodEvents}{E[GoodEvents)]}\times\frac{E[BadEvents]}{BadEvents} \qquad (2)$$

集体强度具有如下令人感兴趣的特性：

(1)集体强度以一种对称的方式处理值0和1。如果为了发现项目集，项目的出现与否均要考虑，那么应用集体强度概念就较为适宜；

(2)考察2-项目集 $I=\{i_1,i_2\}$ 的集体强度。如果项目 $i_1$ 和 $i_2$ 完全积极相关，那么相应2-项目集 $I$ 的集体强度为∞。这是因为"坏事"的出现机会为0。另一方面，如果项目 $i_1$ 和 $i_2$ 完全消极相关，"坏事"的出现机会为0，那么相应2-项目集 $I$ 的集体强度为0。如果项目 $i_1$ 和 $i_2$ 彼此独立，那么项目集 $I$ 的集体强度为1。

(3)集体强度运用了项目集在交易数据库中出现的相对次数。如果某项目集的出现无关紧要，该项目集在后继阶段就删除。项目集出现机会的级别是一项有用的信息，但与发现项目集中的项目是否彼此相关并不完全是一回事。支持仍然可以用于识别数据库中具有较高出现机会的项目集。

**小结** 本文评述了基于频繁项目集的关联规则发现算法及应用，特别是近年来的最新研究进展。通过修改 R. Agrawall 等人提出的 Apriori 算法，人们不断地提出新的高效频繁项目集方法。我们还讨论了频繁项目集方法的不足之处，指出有必要提出新的度量标准来生成项目集。Brin 和 C. Aggawal 分别对此作了一些较为有益的工作。这方面的工作还有待进一步地研究探讨。(参考文献共20篇，略)