

# 关联规则中 Apriori 算法的研究与改进

赵志璞, 邹书蓉

(成都信息工程大学计算机学院, 四川 成都 610225)

**摘要:** Apriori 算法是挖掘关联规则中频繁项集的算法, 其算法思想就是利用上一步生成的候选项集挖掘频繁项集, 该算法在网络安全、电商等领域都有应用。Apriori 算法虽然是应用非常广泛的数据挖掘算法, 但是还存在一些缺陷, 比如频繁扫描事务数据库和产生不必要的候选项集。通过分析 Apriori 算法的执行步骤, 找出算法中的不足, 提出两种改进算法: 缩减事务数据库和压缩事务数据库映射的矩阵。通过实验比较两种改进算法和经典 Apriori 算法, 分析结果, 可以看出改进的两种算法较经典算法有较高的效率提高。

**关键词:** 关联规则; Apriori 算法; 频繁项集; 事务数据库

## 0 引言

在计算机和数据库技术高速发展的今天, 怎样在数以亿计的大量数据中发掘有价值的知识信息成为需要解决的主要问题, 数据挖掘技术也就随之出现。数据挖掘技术中的关联规则挖掘技术是其主要内容。关联规则挖掘最早是随着购物篮分析而出现的, 到后来它的研究方向慢慢演化为寻找关系、事务数据库以及其它的信息存储海量数据之间的规律。关联规则这样定义: 令  $I = \{I_1, I_2, \dots, I_m\}$  为项的集合。对于一个已知的事务数据库  $D$ , 每个事务都包含于  $I$  项集。如果  $X \in I, Y \in I$ , 且  $X \cap Y = \phi$ , 则关联规则  $X \rightarrow Y$  在事务数据库  $D$  中出现的频率定义为支持度, 支持度又定义了最小支持度 Minsup; 事务数据库出现  $X$  的事务中再出现  $Y$  的频率, 即条件概率, 称为关联规则  $X \rightarrow Y$  的置信度, 置信度表示此关联规则关联性强弱; 如果某一事务集在事务数据库中出现的频率不小于给定的 Minsup, 此事务集就是频繁项集。关联规则挖掘的主要步骤如下: 从给定的事务数据库  $D$  中找出所有的频繁项集; 找出这些频繁项集的关系。

通过分析找出 Apriori 算法的不足, 根据其瓶颈问题, 提出相对应的改进优化方法, 并给出 Apriori 算法的一些应用方向 and 未来发展前景。

## 1 Apriori 算法

### 1.1 算法概述

Agrawal 等发现的 Apriori 算法是关联规则挖掘算法中应用最为广泛的算法, 也是最经典的算法。Apri-

ori 算法利用递归的方法, 在事务数据库中通过对项集的挖掘, 找出相应的关联规则, 通过对候选项集的连接和剪枝, 得到新的项集, 再和 Minsup 比较, 就能得到频繁项集。

### 1.2 算法步骤

(1) 给定最小支持度 (Minsup) 并且令参数  $k = 1$ 。

(2) 扫描事务数据库  $D$ , 产生候选的  $k$ -项集, 此时需要判断  $k$  项集的支持度是否满足给定的支持度阈值 Minsup, 如果  $k$ -项集的支持度  $\geq$  Minsup, 则  $k$ -项集为频繁  $k$ -项集。若频繁  $k$ -项集不存在, 结束算法并返回结果。

(3) 根据 (2) 中得到的频繁  $k$ -项集, 通过自然连接得到候选 ( $k+1$ )-项集, 此时需要对候选 ( $k+1$ )-项集进行剪枝。剪枝过程是根据 Apriori 算法的先验原理进行的, 先验原理这样描述: 候选项集的所有子集都是频繁项集时, 此候选项集才可以是频繁项集。

(4) 令  $k = k + 1$ , 重复 (2)、(3) 步骤, 一直到找到事务数据库中所有的频繁项集。

### 1.3 算法分析及缺陷

分析算法步骤, 可推出 Apriori 算法的流程。首先, 遍历事务数据库, 得到每项出现的频率, 计算出每项的支持度, 通过和 Minsup 比较, 得到符合要求的项组成频繁 1-项集; 然后通过自然连接频繁 1-项集得到候选 2-项集, 此时再次扫描事务数据库  $D$ , 再次得出所有候选 2-项集在数据库中出现的频率, 计算支持度, 通过和 Minsup 比较, 得到符合要求的项组成频繁 2-项集, 反复执行以上步骤, 得到所有频繁项集, 输出结果。

通过以上分析可以发现, Apriori 算法缺陷主要有以下两点:

在寻找频繁项集的过程中, 必须多次扫描事务数

数据库,在得到候选  $k$ -项集后,需要对候选  $k$ -项集中的每一项和数据库中的数据进行匹配和计数,才能得到频繁  $k$ -项集;频繁项集自然连接生成候选  $(k+1)$ -项集后,还需要对候选  $(k+1)$ -项集进行剪枝,剪枝过程需要对候选  $(k+1)$ -项集的  $k$  项子集和频繁  $k$ -项集进行匹配,检查频繁  $k$ -项是否都包含其  $k$ -项子集。这样做会造成巨大的计算量,并且造成过大的 I/O 负载,使算法效率降低。

频繁  $k$ -项集自然连接生成候选  $(k+1)$ -项集时,候选项集的数量越来越多,增长速度越来越快。10 个频繁 1-项集会产生 45 个候选项集,100 个频繁 1-项集就产生  $C_{100}^2$  个候选项集,以此类推,之后要挖掘的数据非常大。

## 2 Apriori 算法的改进算法

### 2.1 缩减事务数据库的改进方法

由 Apriori 算法的缺陷可知,由候选  $k$ -项集  $C_k$  得到频繁  $k$ -项集  $L_k$  的过程中,总进行事务数据库的扫描,其实在后面的扫描中,有些事务数据库中的数据已经对产生  $L_k$  没有用处,所以在每次扫描事务数据库时,删除一些不必要的事务会提高算法的效率。文献[1]提到 Apriori 算法的改进算法是对  $C_k$  计算支持度后,把事务数据库中不包含  $C_k$  中任一项集的事务记为删除项,之后对于其他候选项集进行支持度计算时,就不用再考虑。此算法虽然对事务数据库有一定的减少,但还是有些不尽人意之处,比如在对  $C_k$  进行事务数据库匹配时,即使某一事务包含  $C_k$  中的某一项集,但是此项集的支持度  $< \text{Minsup}$ ,这个事务对于产生频繁  $k$ -项集  $L_k$  也没有用处了,它也可以从事务数据库中删除。当事务数据库的规模很大时,删除掉的事务就会更多,对于候选项集的支持度计算就会相对减少,算法的效率相对于文献[1]提高了。

改进后的算法步骤如下:

(1) 给定最小支持度 (Minsup) 并且令参数  $k=1$ 。

(2) 扫描事务数据库  $D$ , 产生候选的  $k$ -项集,此时需要判断  $k$ -项集的支持度是否满足给定的支持度阈值 Minsup, 如果  $k$ -项集的支持度  $\geq \text{Minsup}$ , 则  $k$ -项集为频繁  $k$ -项集,此时把事务数据库中不包含  $k$ -项集中任一项集的事务记为删除项,同时把包含  $k$ -项集中某一项集但是支持度小于 Minsup 的事务也从数据库中删除。若频繁  $k$ -项集不存在,结束算法并返回结果。

(3) 根据(2)中得到的频繁  $k$ -项集,通过自然连接得到候选  $(k+1)$ -项集,此时需要对候选  $(k+1)$ -项集进行剪枝。剪枝过程是根据 Apriori 算法的先验原理进行的,先验原理这样描述:候选项集的所有子集都

是频繁项集时,此候选项集才可以是频繁项集。

(4) 令  $k=k+1$ , 重复(2)、(3)步骤,直到找到事务数据库中所有的频繁项集。

分别用经典 Apriori 算法和此改进算法对 10000 条交易数据,在不同的 Minsup 下进行测试,得到以下实验结果,如图 1 所示。

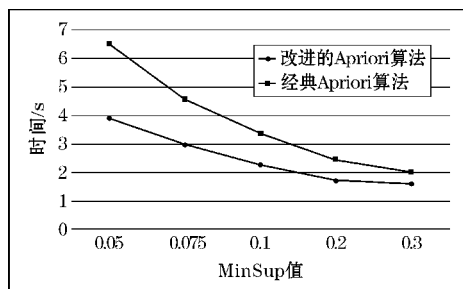


图1 实验数据图

从图 1 看出,在实验数据相同的情况下,通过给定不同的 Minsup 进行试验,改进算法执行时间明显减少,原因就是改进算法在扫描数据库时,删除一些无用事务,从而减少之后扫描数据库的工作量,算法效率提高了。

### 2.2 压缩事务数据库映射矩阵的改进方法

文献[2]提出把对应事务数据库分层的思想,具体方法为:把事务数据库中的每一事务都作为数据库的一行,把每一具体的项都作为数据库的一列,这样在统计项集支持度时,只需要扫描事务数据库中对应的项集列即可。在寻找频繁项集的过程中,扫描的只是事务数据库的一部分,不用反复全盘扫描数据库,算法效率相对于经典的 Apriori 算法有一定的提高。此算法虽然减少了对事务数据库的扫描,提高了效率,但是每次寻找频繁项集还是要扫描数据库,对于事务数巨大的数据库,此算法还是不够理想。基于文献[2]提出的算法,提出一种把事务数据库映射为矩阵,再对矩阵进行操作的方法。具体方法如下:扫描事务数据库  $D$ , 建立二维矩阵  $R_{m \times n}$ , 其中矩阵的  $m$  行代表事务数据库中的  $m$  个事务,矩阵的  $n$  列代表事务数据库中的  $n$  个项,如果事务中包含某一项,矩阵中对应的这个元素值就设为 1, 如果事务中不包含某一项,矩阵中对应元素就设为 0。矩阵中,除代表事务和项的行列外,还需要添加一个辅助行和辅助列,分别统计事务数据库中项的支持度和每个事务的个数,之后再求候选项集和频繁项集时,就可以只对此矩阵操作就行,不必要反复扫描事务数据库。在建立过矩阵后,改进算法的具体步骤如下:

(1) 给定 Minsup 的值,然后和辅助行中每个元素进行比较,就可以得到频繁 1-项集,令  $k=1$ , 根据先验

原理(频繁项集的每个子集也是频繁项集)删除对应的矩阵列;

(2) 计算辅助列的值,把辅助列中值小于  $k$  的元素的行删除(因为要通过频繁  $k$ -项集得到候选  $(k+1)$ -项集,所以删除只有  $k$  项的事务),然后重新计算辅助行的每个元素值,再和 Minsup 比较,删除值小于 Minsup 的元素值对应的列,反复压缩矩阵,直到不能压缩;

(3) 求频繁- $(k+1)$ 项集:把矩阵中剩余的列的所有  $(k+1)$ 组合列出,分别对这  $(k+1)$ 列进行“与”运算别保存结果,把结果向量中出现“1”的个数大于等于 Minsup 的组合保留下来,此组合就是频繁- $(k+1)$ 项集;

(4) 若频繁- $(k+1)$ 项集的个数不小于  $(k+2)$ (假设存在频繁- $(k+2)$ 项集,那么频繁- $(k+2)$ 项集的  $(k+1)$ 项集就存在  $(k+2)$ 个,因为频繁项集的所有子集也都是频繁项集,所以频繁- $(k+1)$ 项集得个数最少有  $(k+2)$ 个),令  $k = k+1$ ,算法跳转至(2);否则算法结束。

对 10000 条交易数据,在 Minsup 分别为 5%、7.5%、10%、20%、30% 的情况下,用经典 Apriori 算法和此改进算法分别进行测试,可得到图 2。

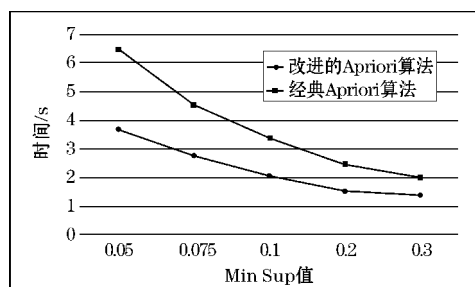


图2 测试结果图

在实验数据相同的情况下,对不同 Minsup 进行试验,此改进算法在算法的执行时间上明显减少,并且在 Minsup 较小的情况,相对于文献[2]的改进算法有更显著的效率提高。原因就是此改进算法整个寻找频繁项集的过程中只扫描了一次事务数据库,并且剪枝过程中都是对“0”、“1”进行的操作,所以大幅度提高了算法效率。

### 3 结束语

通过对 Apriori 算法和其一些改进算法的分析,发现一些缺陷和改进的局限性,从而提出两种新的改进算法,在通过缩减事务数据库事务和压缩事务数据库映射矩阵两方面,对寻找频繁项集和剪枝过程进行改进,并给出完整的算法描述。通过模拟数据进行实验对比,验证了真两种改进算法的有效性,提高了算法效

率,相信这种改进对于现今社会一些关联规则挖掘问题具有一定意义。

### 参考文献:

- [1] Peng Gong, Chi Yang; Hui Li. The application of improved association rules data mining algorithm Apriori in CRM[J]. 2nd International Conference on Pervasive Computing and Applications. 2007: 52-66.
- [2] Lu Lina, Chen Yaping, Wei Hengyi. Research on the algorithm Apriori of mining association rules. Mini-Micro System, 2002, 21(9): 940-943.
- [3] Michael O AKinde. Efficient OLAP Query Processing in Distributed Data Warehouses[J]. Lecture Notes in Computer Science, 2004, (3): 132-141.
- [4] Tomoaki Imamura, Shinya Matsumoto, Yoshiyuki Kanagawa. A technique for identifying three diagnostic findings using association analysis[J]. Medical and Biological Engineering and Computing. 2007: 51-59.
- [5] Kleinberg J, Papadimitriou C, Raghavan P. Segmentation Problems[A]. Proceedings of the 30th Annual Symposium on the Theory of Computing [C]. New York: ACM Press, 1998.
- [6] M Nyanchama, S L Osborn. The role graph model and conflict of interest[J]. ACM TISSEC, 1999, 2(1): 3-33.
- [7] LiX R, Jilkov V P. A survey of maneuvering target tracking-part III: Measurement models[C]//Proceedings of the Conference on Signal and Data Processing of Small Targets, CA, 2001, 4473(7/8): 423-446.
- [8] 吴斌,肖刚,陆佳炜. 基于关联规则领域的 Apriori 算法的优化研究[J]. 计算机工程与科学, 2009, 31(6): 116-118.
- [9] J Ming, Syan Chen. "Data Mining: An Overview from a Database Perspective" [J]. IEEE Transactions on Knowledge and Data Engineering, 1996, 8(6): 866-883.
- [10] Agrawal R, Srikant R. Fast algorithms for mining association rules in large database[C]//Proc of the 20th International Conference on Very Large Databases, 1994.