基于预判筛选的高效关联规则挖掘算法

赵学健*①②③ 孙知信①② 袁 源③
①(南京邮电大学物联网学院 南京 210003)
②(南京邮电大学江苏省通信与网络技术工程研究中心 南京 210003)
③(江苏省邮电规划设计院有限责任公司 南京 210006)

摘 要：关联规则分析作为数据挖掘的主要手段之一，在发现海量事务数据中隐含的有价值信息方面具有重要的作用。该文针对 Apriori 算法的固有缺陷，提出了 AWP (Apriori With Prejudging) 算法。该算法在 Apriori 算法连接、剪枝的基础上，添加了预判筛选的步骤，使用先验概率对候选频繁 k 项集集合进行缩减优化，并且引入阻尼因子和补偿因子对预判筛选产生的误差进行修正，简化了挖掘频繁项集的操作过程。实验证明 AWP 算法能够有效减少扫描数据库的次数，降低算法的运行时间。

关键词：数据挖掘；关联规则；事务数据库；预判筛选；Apriori
中图分类号：TP391 **文献标识码**：A **文章编号**：1009-5896(2016)07-1654-06
DOI: 10.11999/JEIT151107

An Efficient Association Rule Mining Algorithm Based on Prejudging and Screening

ZHAO Xuejian①②③ SUN Zhixin①② YUAN Yuan③
①(*School of Internet of Things*, *Nanjing University of Posts and Telecommunications*, *Nanjing* 210003, *China*)
②(*Engineering Research Center of Communication and Network Technology*, *Nanjing University of Posts and Telecommunications*, *Nanjing* 210003, *China*)
③(*Jiangsu Posts & Telecommunications Planning and Designing Institute Co. LTD*, *Nanjing* 210006, *China*)

Abstract: Association rule analysis, as one of the significant means of data mining, plays an important role in discovering the implicit knowledge in massive transaction data. To overcome the inherent defects of the classic Apriori algorithm, this paper proposes Apriori With Prejudging (AWP) algorithm. AWP algorithm adds a pre-judging procedure on the basis of the self-join and pruning progress in Apriori algorithm. It reduces and optimizes the k-frequent item sets using prior probability. In addition, the damping factor and compensating factor are introduced to revise the deviation caused by pre-judging. AWP algorithm simplifies the operation process of mining frequent item sets. Experimental results show that the improvement measures can effectively reduce the number of scanning databases and reduce the running time of the algorithm.

Key words: Data mining; Association rules; Transaction database; Prejudging; Apriori

1 引言

在大数据技术发展如火如荼的今天，人们逐渐意识到数据即是财富，尤其是对商业数据的分析更具有巨大的实用价值。关联规则分析作为数据挖掘的主要手段之一，是数据挖掘技术中不可或缺的一个重要组成部分，主要用于发现大型事务数据库中隐含的有价值的令人感兴趣的联系及规则[1,2]。因此，对关联规则算法的研究具有非常重要的意义。

早在 1993 年，IBM 的计算机科学家 Agrawal 等人在顾客交易数据库中发现了顾客在购买商品时的购买规律，提出了事务之间的相关性模式，即最初的关联规则。关联规则通常是一种不复杂但实用性却很高的规则。通过关联规则分析，我们可以将事务项集与项集之间的关系挖掘出来。关联规则分析最典型的应用是购物篮数据分析，比如经典的{啤酒}→{尿布}规则。除了可以应用于购物篮数据之

收稿日期：2015-09-29；改回日期：2016-02-26；网络出版：2016-04-14
*通信作者：赵学健 zhaoxj@njupt.edu.cn
基金项目：国家自然科学基金(61373135, 61401225, 61502252, 61201160)，江苏省基础研究计划(自然科学基金)(BK20140883, BK20140894, BK20131377)，中国博士后科学基金(2015M581844)，江苏省博士后科研资助计划项目(1501125B)，南京邮电大学校级科研基金(NY214101, NY215147)
Foundation Items: The National Natural Science Foundation of China (61373135, 61401225, 61502252, 61201160), Natural Science Foundation of Jiangsu Province of China (BK20140883, BK20140894, BK20131377), China Postdoctoral Science Foundation Funded Project (2015M581844), Jiangsu Planned Projects for Postdoctoral Research Funds (1501125B), NUPTSF (NY214101, NY215147)

外，关联规则分析在其它领域的应用也十分广泛，如电子商务个性化推荐，金融服务，广告策划，生物信息学及科学数据分析等[3−5]。比如说在电子商务个性化推荐中，关联规则可以帮助电子商务网站向具有相似消费行为的顾客进行一些他们可能感兴趣的商品推荐，这样有助于电子商务网站提升用户体验，增加盈利等。

关联规则分析算法较多，其中最经典实用性最好的是 Apriori 算法及其改进算法。Apriori 算法是由文献[6]于 1994 年提出的第 1 个关联规则算法，应用广泛，该算法通过重复循环执行连接、剪枝生成频繁项目集，从而建立关联规则。基于 Apriori 算法，文献[7]提出了 Apriori-TFP 算法，该算法在关联规则挖掘过程中，将原始数据进行预处理并存储在局部支持树中，最后生成关联规则。该算法通过有效的预处理，降低了关联规则挖掘的时间，但是需要扫描数据库的次数仍然较多。文献[8]提出了 GP-Apriori 算法，GP-Apriori 算法采用图形处理器(Graphical Processing Unit, GPU)进行并行化的支持度计数，并将垂直交易列存储为线性有序阵列。GPU 通过遍历该有序阵列，并执行按位交叉实现支持度计算，并将结果复制回内存。与传统 CPU 上运行的 Apriori 算法相比，GP-Apriori 算法由于采用了先进的 GPU 提高了运行速率，但是复杂性反而有所增长。文献[9]提出了 Apriori 的改进算法(Apriori Mend Algorithm)。该算法使用哈希函数生成项目集，用户必须指定最小支持度以删除不需要的项集。该算法具有比传统 Apriori 算法更好的效率，但是执行时间有所增加。文献[10]基于 MapReduce 框架实现了 Apriori 算法的并行化。该算法在处理海量数据集时具有良好的可扩展性和效率，但是该算法需要强大的计算和存储能力支撑，通常运行在集群环境中。文献[11]尝试将 Apriori 算法应用于多维数据分析，探讨了在多维数据中建立关联规则更加具体有效的方法。文献[12]对 Apriori 算法进行了改进，引入了事务尺寸和事务规模的概念以消除非重要项目的影响。文献[13]提出了一种基于矩阵的 Apriori 算法，该算法通过矩阵有效地表示数据库的各种操作，并用基于矩阵的 AND 操作得到最大的频繁项目集。算法虽然大大减少了扫描数据库的次数，但是空间复杂度较高。文献[14]对 Apriori 算法进行了改进，提出了 M-Apriori 算法，该算法在寻找频繁项目集的过程中，对包含 k 项集的事务集合进行记录，在验证 $k+1$ 项集是否频繁时，不再扫描整个数据库，而是仅需扫描数据库中部分事务，从而提高时间效率。文献[15]提出了一种 Map-Reduce 框架下的分布式幂集 Apriori 算法。实验结果表明，该算法并行运算能力强，在低虚警率和漏检率的情况下，具有较好的检测率。文献[16]提出一种基于 Apriori 的改进算法，通过减少候选 2 项集的剪枝操作从而提高效率，但实验结果表明算法性能提升非常有限。

上述基于 Apriori 的改进算法大多继承了 Apriori 算法需要大量扫描数据库和占用大量内存的缺点[17]。本文针对经典 Apriori 算法的固有缺陷，提出了 AWP(Apriori With Prejudging) 算法，该算法在 Apriori 算法连接、剪枝的基础上，添加了预判筛选的步骤，通过使用先验概率对候选频繁 k 项集集合进行缩减优化，并且引入阻尼因子和补偿因子对预判筛选产生的误差进行修正，以减少扫描数据库的次数，降低算法的运算时间，提高算法的运算效率。

2　AWP 算法

2.1　理论基础

定义 1 (支持度)　数据集 D 中包含项目集 X 的事务数称为项目集 X 的支持数，记为 σ_X。项目集 X 的支持度记为 $\mathrm{support}(X)$：

$$\mathrm{support}(X)=\sigma_X/|D| \tag{1}$$

其中 $|D|$ 是数据集 D 的事务数，若 $\mathrm{support}(X)$ 大于等于用户所指定的最小支持度，则称项目集 X 为频繁项目集，否则称 X 为非频繁项目集。

定义 2 (虚警率)　假设存在数据集 D，指定的最小支持度为 min_support，客观存在的频繁项目集为 L，成员数记为 N，运行算法检测所得频繁项目集为 L_a，L_a 成员数记为 N_a，L_a 中有 N_f 个成员的支持度小于指定的最小支持度 min_support。算法的虚警率记为 FAR (False Alarm Rate)：

$$\mathrm{FAR}=N_f/N \tag{2}$$

定义 3 (漏检率)　假设存在数据集 D，指定的最小支持度为 min_support，客观存在的频繁项目集为 L，成员数记为 N，运行算法检测所得频繁项目集为 L_a，成员数记为 N_a，尚存在 N_o 个项目集属于集合 L 但不属于集合 L_a，算法的漏检率记为 OF (Omission Factor)：

$$\mathrm{OF}=N_o/N \tag{3}$$

定理 1　设 X，Y 是两个非空项目集，且 $X\cap Y$ 为空集，项目集 X，Y 在数据库某事务中出现的概率分别为 $P(X)$，$P(Y)$，则项目集 $X\cup Y$ 在数据库某事务中出现的概率 $P(X\cup Y)$ 取值范围为 $[P(X)P(Y)-0.25, P(X)P(Y)+0.25]$。

证明　因为

$$\begin{aligned}P(X)&=P(X\cup Y)+P(X\cup\overline{Y})\\&=P(Y)P(X/Y)+P(\overline{Y})P(X/\overline{Y})\end{aligned}$$

故

$$\begin{aligned}&P(X)-P(X/Y)\\&=P(Y)P(X/Y)+P\left(\overline{Y}\right)P\left(X/\overline{Y}\right)-P(X/Y)\\&=P\left(\overline{Y}\right)P\left(X/\overline{Y}\right)-P\left(\overline{Y}\right)P(X/Y)\\&=P\left(\overline{Y}\right)\left(P\left(X/\overline{Y}\right)-P(X/Y)\right)\end{aligned}$$

则有

$$\begin{aligned}P(X\cup Y)&=P(Y)P(X/Y)\\&=P(Y)P(X)-P(Y)P\left(\overline{Y}\right)\left[P\left(X/\overline{Y}\right)\right.\\&\quad\left.-P\left(X/Y\right)\right]\end{aligned}$$

式中，$\overline{Y}$ 表示 Y 的补集，$P(X/\overline{Y})-P(X/Y)$ 的取值范围为[−1,1]。$P(Y)P(\overline{Y})=P(Y)\,(1-P(Y))$，求导可知当 $P(Y)=0.5$ 时，$P(Y)P(\overline{Y})$ 取得最大值 0.25。

证毕

AWP 算法尝试在连接、剪枝步骤后，通过计算先验概率的方法对候选频繁 k 项集集合中的项目集进行缩减优化，而不是直接通过扫描数据库进行判断，从而减少扫描数据库的次数，降低算法运行时间，提高算法运算效率。由定理 1 可知，若直接采用独立事件概率公式对 $P(X\cup Y)$ 的概率进行计算，在预判筛选过程中难免出现虚警(把非频繁项目集视为频繁项目集)和漏检(把频繁项目集视为非频繁项目集)的情况。因此，AWP 算法引入阻尼因子和补偿因子对预判筛选产生的虚警率和漏检率进行控制。若阻尼因子和补偿因子严格根据定理 1 进行设置，可保证虚警率和漏检率均为 0，但是算法性能会受到严重影响。为追求更佳的算法性能，并且保证一定的虚警率和漏检率的前提下，算法将采用实验验证的方法对阻尼因子和补偿因子进行了设置。

2.2 算法描述

AWP 算法为实现关联规则的挖掘，同样包含两个步骤：(1)找出所有的频繁项集；(2)由频繁项集产生强规则。

为了解决 Apriori 算法存在的技术问题，精简候选项目集中的元素，简化算法挖掘频繁项集以及规则的操作过程，减少扫描数据库的次数，AWP 算法根据定理 1 在 Apriori 算法连接、剪枝步骤后添加了预判筛选环节，算法找出所有频繁项集的过程如下：

步骤 1　扫描事务数据库 D 并计数，找出支持度大于预设最小支持度的频繁 1 项集集合 L_1，具体过程如下：对事务数据库 D 中包含项目 S_i 的事务数 N_i 进行统计，其中 $i\in\{1,2,\cdots,n_2\}$，n_2 为数据库包含的项目数量，则项目集 $X=\{S_i,i\in\{1,2,\cdots,n_2\}\}$ 的支持度为

$$\text{support}(X=\{S_i,i\in\{1,2,\cdots,n_2\}\})=N_i/|D|$$

其中，$|D|$ 为数据库 D 包含的事务数，若 support(X) 大于预设的最小支持度 min_support，则将项目集 X 加入频繁 1 项集集合 L_1；反之，不加入；

步骤 2　将所得到的频繁 $k-1$ 项集集合 L_{k-1} 与其自身连接产生候选 k 项集的集合，候选 k 项集的集合记作 C_k，其中，$k\in\{2,3,4,\cdots\}$，第 1 次执行时 K=2，每循环执行 1 次 k 取值加 1。连接过程如下：设 m_1 和是 m_2 频繁 $k-1$ 项集集合 L_{k-1} 的任意两个成员，成员中的项目按字典次序排序，即对于成员 m_i，有 $m_i[1]<m_i[2]<\cdots<m_i[k-1]$，其中符号 $m_i[j]$ 表示成员 m_i 中的第 j 个项目，$i\in\{0,1\}$，$j\in\{1,2,\cdots,k-1\}$，如果成员 m_1 和 m_2 中前 $k-2$ 个项目均相同，成员 m_1 的第 $k-2$ 个项目小于成员 m_2 的第 $k-2$ 个项目，即 $(m_1[1]=m_2[1])\wedge(m_1[2]=m_2[2])\wedge\cdots\wedge(m_1[k-2]=m_2[k-2])\wedge(m_1[k-1]<m_2[k-1])$，则判定 m_1 和 m_2 是可连接，连接 m_1 和 m_2 产生的结果是 $\{m_1[1],m_1[2],\cdots,m_1[k-1],m_2[k-1]\}$；

步骤 3　利用 Apriori 性质[5](任一频繁项集的所有非空子集也必须是频繁的，如果某个候选的非空子集不是频繁的，那么该候选肯定不是频繁的)对候选 k 项集集合 C_k 进行剪枝。剪枝过程如下：对候选 k 项集集合 C_k 的成员 c_i，$i\in\{1,2,3,\cdots\}$ 的所有非空子集的支持度进行判断，若该成员存在支持度小于预设的最小支持度 min_support 的非空子集，根据 Apriori 性质可判定该成员不是频繁项目集，将其从 C_k 中删除；反之，将该成员保留在候选 k 项集集合 C_k 中；

步骤 4　计算剪枝后的候选 k 项集集合 C_k 中成员的预判支持度，进行预判筛选。预判筛选方法如下：通过独立事件概率公式计算候选 k 项集集合 C_k 中成员 c_i，$i\in\{1,2,3,\cdots\}$ 的先验概率，若 $P(c_i)>(1+\Delta_1)$min_support，则将该成员直接添加到频繁 k 项集 L_k 中；若 $P(c_i)<(1-\Delta_2)$min_support，则将该成员从 C_k 中删除；否则，该成员继续保留在候选 k 项集 C_k 中；其中，$P(c_i)=\sum P(c)P(c_i-c)/n$，$c$ 为成员 c_i 的单元素子集，即 c 中只包含一个项目，$P(c)$ 可由步骤 1 得到，$P(c_i-c)$ 可由获得频繁 $k-1$ 项集集合的循环轮次中步骤 5 得到，n 为成员 c_i 所包含的项目数；Δ_1 为阻尼因子，Δ_2 为补偿因子，为减少扫描数据库的次数，经实验验证，Δ_1 取值为

$$\Delta_1=\begin{cases}0.5, & |D|\le 10^3\\ 0.1[5-\lg(|D|/10^3)], & 10^3\le|D|\le 10^6\\ 0.2, & |D|\ge 10^6\end{cases}\tag{4}$$

Δ_2 取值为

$$\Delta_2=\begin{cases}0.25, & |D|\le 10^3\\ 0.05[5-2\lg(|D|/10^3)], & 10^3\le|D|\le 10^5\\ 0.1, & |D|\ge 10^5\end{cases} \quad (5)$$

步骤 5　通过扫描事务数据库 D，确定预判筛选后的候选 k 项集集合 C_k 中每个候选成员 c_i 的计数，判断该计数是否大于最小支持度计数，如果是，则判定该候选成员是频繁的，将该候选成员保留在频繁 k 项集集合 L_k 中，否则删除；

步骤 6　重复执行上述步骤 2~步骤 5，直到不能发现更大的频繁项目集为止；

步骤 7　最终获得的频繁项目集集合为 F。

AWP 算法由频繁项集产生强规则的过程与 Apriori 算法类似。若最终 AWP 算法获得的频繁项目集为 F，则可产生关联规则 $R=\{A\to B\}$，A 为频繁项目集集合 F 中任意成员 F_i 的非空子集，B 为 A 的补集，即 $F_i\in F$，$i\in\{1,2,\cdots,n_1\}$ 且 $A\cup B=F_i$，其中 n_1 为频繁项目集集合 F 包含的成员数目。比如说若集合 $\{I_1,I_2,I_3\}$ 是频繁项目集集合的成员，则可产生如下关联规则：$\{I_1\}\to\{I_2,I_3\}$，$\{I_2\}\to\{I_1,I_3\}$，$\{I_3\}\to\{I_1,I_2\}$，$\{I_1,I_2\}\to\{I_3\}$，$\{I_1,I_3\}\to\{I_2\}$，$\{I_2,I_3\}\to\{I_1\}$。

3　实验分析

本文采用 8 组事务数据库对 AWP 算法的性能进行了分析，8 组数据库分别包含 5×10^2, 1×10^3, 2×10^3, 5×10^3, 8×10^3, 1×10^4, 2×10^4, 4×10^4 个事务。由于 AWP 算法增加了预判筛选的步骤，该步骤通过独立事件概率公式计算候选 k 项集集合 C_k 中成员 c_i，$i\in\{1,2,3,\cdots\}$ 的先验概率，以减少扫描数据库的次数，这容易导致虚警和漏检的情况发生。为此，AWP 算法在预判筛选步骤中引入了阻尼因子 Δ_1 和补偿因子 Δ_2 两个参数，通过合理设置阻尼因子和补偿因子可有效降低虚警率和漏检率。根据虚警率和漏检率的概念可知，所谓虚警是指本不是频繁项目集却纳入到频繁项目集的范畴；所谓漏检是指本应是频繁项目集却没有纳入到频繁项目集的范畴。在现实情况中，为了尽可能地挖掘出所有的强关联规则，对算法的漏检率要求应该更加严格。因此，本文中要求算法的虚警率低于 5%，而漏检率低于 2%。在分析 AWP 算法性能之前，首先通过实验对阻尼因子和补偿因子的取值进行分析。

如表 1 所示，该表描述了最小支持度 min_support 为 0.04，阻尼因子 Δ_1 取值从 0.1 变化到 0.5 的过程中，事务数分别为 5×10^2, 5×10^3, 1×10^4, 2×10^4, 4×10^4 的 5 个事务数据库所对应的虚警率。由该表可以看出，对同一个事务数据库而言，当阻尼因子 Δ_1 逐渐增大时，虚警率呈现出逐渐减小的趋势；对不同的事务数据库而言，当阻尼因子取值相同时，虚警率随着数据库中事务数的增多同样有减小的趋势。为了确保虚警率小于 5%，事务数分别为 5×10^2, 5×10^3, 1×10^4, 2×10^4, 4×10^4 的 5 个事务数据库所对应的阻尼因子 Δ_1 的取值分别为 0.4, 0.4, 0.3, 0.2, 0.1。因此，在 AWP 算法中阻尼因子 Δ_1 的取值如式(4)所示。

表 1 阻尼因子-虚警率分析表(min_support=0.04)

事务数	虚警率(%)				
	Δ_1 =0.5	Δ_1 =0.4	Δ_1 =0.3	Δ_1 =0.2	Δ_1 =0.1
5×10^2	3.93	4.57	5.21	6.85	8.76
5×10^3	3.59	4.24	5.01	6.26	8.13
1×10^4	3.22	3.88	4.31	5.44	6.59
2×10^4	2.74	3.31	3.67	4.91	5.63
4×10^4	1.56	2.65	3.11	3.75	4.76

如表 2 所示，该表描述了最小支持度 min_support 为 0.04，补偿因子 Δ_2 取值从 0.05 变化到 0.25 的过程中，事务数分别为 5×10^2, 5×10^3, 1×10^4, 2×10^4, 4×10^4 的 5 个事务数据库所对应的漏检率。由表 2 可以看出，对同一个事务数据库而言，当补偿因子 Δ_2 逐渐增大时，漏检率呈现出逐渐减小的趋势；对不同的事务数据库而言，当补偿因子取值相同时，漏检率随着数据库中事务数的增多同样有减小的趋势。为了确保漏检率小于 2%，事务数分别为 5×10^2, 5×10^3, 1×10^4, 2×10^4, 4×10^4 的 5 个事务数据库所对应的补偿因子 Δ_2 的取值分别为 0.20, 0.15, 0.10, 0.10, 0.05。因此，在 AWP 算法中补偿因子 Δ_2 的取值如式(5)所示。

在确定了算法的阻尼因子和补偿因子分别如式(4)和式(5)所示后，第 2 组实验对算法的运行时间随事务数的变化情况进行了分析。在最小支持度 min_support 设置为 0.04 时，算法运行时间随事务数变化曲线如图 1 所示。由于数据库包含的事务数变化较大，因此图 1 横坐标采用了对数坐标轴，纵坐标依然为普通坐标轴。由图 1 可以看出，随着数据库中事务数的增多，Apriori 算法、M-Apriori 算法和 AWP 算法的运行时间均逐渐增大。但是，AWP 算法相对于 Apriori 算法和 M-Apriori 算法来说，运行时间得到了大幅缩短。当事务数据库包含 4×10^4 条记录时，AWP 算法的运行时间为 34.48 s，比 Apriori 算法和 M-Apriori 分别降低 48.9%和 17.1%。

表 2 补偿因子-漏检率分析表(min_support = 0.04)

事务数	漏检率(%)				
	Δ_2 =0.25	Δ_2 =0.20	Δ_2 =0.15	Δ_2 =0.10	Δ_2 =0.05
5×10^2	0.86	1.34	2.89	4.52	8.51
5×10^3	0.73	1.15	1.92	2.87	5.38
1×10^4	0.55	0.82	0.97	1.94	4.63
2×10^4	0.39	0.66	0.84	1.37	3.25
4×10^4	0.17	0.49	0.67	0.98	1.46

同样地，设置最小支持度 min_support 为 0.04，第 3 组实验对算法扫描数据库的次数随事务数的变化情况进行了分析。如图 2 所示，横坐标依然采用对数坐标轴，纵坐标为普通坐标轴。由图 2 可以看出，随着数据库中事务数的增多，Apriori 算法、M-Apriori 算法和 AWP 算法扫描数据库的次数均逐渐增大。但是，Apriori 算法扫描数据库的次数随着数据库包含事务数的增大增长较快，而 AWP 算法由于对候选项目集集合进行了预判筛，扫描数据库的次数得到了较大程度的减少。当事务数据库包含 4×10^4 条记录时，AWP 算法扫描数据库的次数为 862 次，比 Apriori 算法和 M-Apriori 分别降低 45.6% 和 15.4%。

最后一组实验，针对事务数为 10^4 的数据库，分析了算法运行时间随最小支持度 min_support 的变化情况，如图 3 所示。由图可以看出，3 种算法的运行时间均随最小支持度的增大而减小。这是因为当最小支持度增大时，候选项目集集合中的成员逐渐减少，因此扫描数据库进行比较的次数减少，运行时间自然降低。在最小支持度相同的情况下，AWP 算法比 Apriori 算法和 M-Apriori 算法降低了运行时间，并且当最小支持度越小的时候，效果愈发明显。

4 结论

本文提出一种基于预判筛选的高效关联规则挖掘算法 —— AWP 算法。该算法在 Apriori 算法连接、剪枝的基础上，增加了预判筛选的步骤，通过使用先验概率对候选频繁 k 项集集合进行缩减优化，从而减少关联规则挖掘过程中扫描数据库的次数。此外，算法引入阻尼因子和补偿因子对预判筛选引起的虚警率和漏检率进行控制，同时通过实验给出了满足虚警率低于 5%，而漏检率低于 2%的阻尼因子和补偿因子设置方法。经实验验证，AWP 算法有效减少了扫描数据库的次数，降低了算法的运算时间，提高了算法的运算效率。

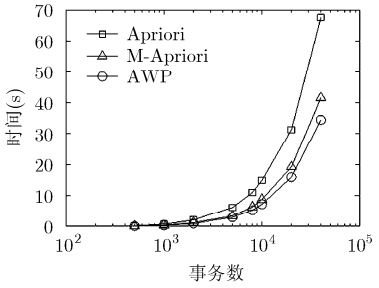

图 1 算法运行时间随事务数变化曲线

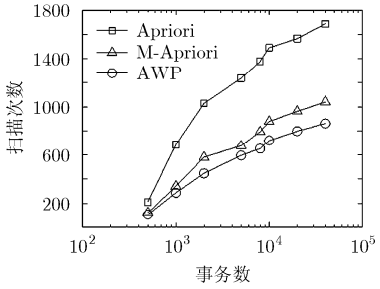

图 2 扫描数据库次数随事务数变化曲线

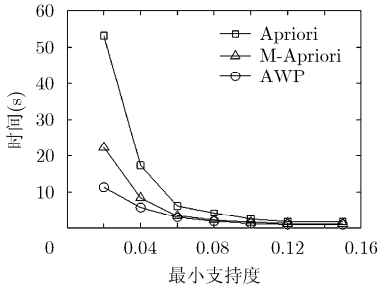

图 3 算法运行时间随最小支持度变化曲线

参 考 文 献

[1] SINGLA S and MALIK A. Survey on various improved Apriori algorithms[J]. *International Journal of Advanced Research in Computer and Communication Engineering*, 2014, 3(11): 8528–8531. doi: 10.17148/ijarcce.2014.31139.

[2] MINAL G I and SURYAVANSHI N Y. Association rule mining using improved Apriori algorithm[J]. *International Journal of Computer Applications*, 2015, 112(4): 37–42.

[3] RAJESWARI K. Improved Apriori algorithm — A comparative study using different objective measures[J]. *International Journal of Computer Science and Information Technologies*, 2015, 6(3): 3185–3191.

[4] ACHAR A, LAXMAN S, and SASTRY P S. A unified view of the Apriori-based algorithms for frequent episode discovery[J]. *Knowledge & Information Systems*, 2012, 31(2): 223–250. doi: 10.1007/s10115-011-0408-2.

[5] 李鹏, 于晓洋, 孙渤禹. 基于用户群组行为分析的视频推荐方法研究[J]. 电子与信息学报, 2014, 36(6): 1484–1491. doi: 10.3724/SP.J.1146.2013.01225.
LI Peng, YU Xiaoyang, and SUN Boyu. Video recommendation method based on group user behavior analysis[J]. *Journal of Electronics & Information Technology*, 2014, 36(6): 1484–1491. doi: 10.3724/SP.J.1146.2013.01225.

[6] AGRAWAL R and SRIKANT R. Fast algorithms for mining association rules[C]. VLDB'94 Proceedings of the 20th International Conference on Very Large Data Bases, San Francisco, CA, USA, 1994: 487– 499.

[7] YANG Z, TANG W, SHINTEMIROV A, *et al.* Association rule mining-based dissolved gas analysis for fault diagnosis of power transformers[J]. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 2009, 39(6): 597–610. doi: 10.1109/TSMCC.2009.2021989.

[8] ZHANG F, ZHANG Y, and BAKOS J D. Gpapriori: Gpu-accelerated frequent itemset mining[C]. 2011 IEEE International Conference on Cluster Computing, Austin, TX, USA, 2011: 590–594. doi: 10.1109/CLUSTER.2011.61.

[9] ANGELINE M D and JAMES S P. Association rule generation using Apriori mend algorithm for student's placement[J]. *International Journal of Emerging Sciences,* 2012, 2(1): 78–86.

[10] LI N, ZENG L, HE Q, *et al.* Parallel implementation of Apriori algorithm based on MapReduce[C]. 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel Distributed Computing (SNPD), Kyoto, Japan, 2012: 236–241. doi: 10.1109/ SNPD. 2012.31.

[11] SULIANTA F, LIONG TH, and ATASTINA I. Mining food industry's multidimensional data to produce association rules using Apriori algorithm as a basis of business strategy[C]. 2013 International Conference of Information and Communication Technology (ICoICT), Bandung, Indonisia, 2013: 176–181. doi: 10.1109/ICoICT.2013.6574569.

[12] ABAYA S A. Association rule mining based on Apriori algorithm in minimizing candidate generation[J]. *International Journal of Scientific & Engineering Research*, 2012, 3(7): 1–4.

[13] WANG Feng and LI Yonghua. An improved Apriori algorithm based on the matrix[C]. Proceedings of 2008 International Seminar on Future BioMedical Information Engineering, Wuhan, China, 2008: 152–155. doi: 10.1109/ FBIE.2008.80.

[14] MAOLEGI M A and ARKOK B. An improved Apriori algorithm for association rules[J]. *International Journal on Natural Language Computing*, 2014, 3(1): 21–29. doi: 10.5121/ijnlc.2014.3103.

[15] 葛琳, 季新生, 江涛. 基于关联规则的网络信息内容安全事件发现及其 Map-Reduce 实现[J]. 电子与信息学报, 2014, 36(8): 1831–1837. doi: 10.3724/SP.J.1146.2013.01272.
GE Lin, JI Xinsheng, and JIANG Tao. Discovery of network information content security incidents based on association rules and its implementation in Map-Reduce[J]. *Journal of Electronics & Information Technology*, 2014, 36(8): 1831–1837. doi: 10.3724/SP.J.1146.2013.01272.

[16] TANK D M. Improved Apriori algorithm for mining association rules[J]. *International Journal of Information Technology and Computer Science*, 2014, 6(7): 15–23. doi: 10.5815/ijitcs.2014.07.03.

[17] RAO S and GUPTA R. Implementing improved algorithm over Apriori data mining association rule algorithm[J]. *International Journal of Computer Science and Technology*, 2012, 34(3): 489–493.

赵学健：　男，1982 年生，副教授，研究方向为数据挖掘、无线传感器网络.

孙知信：　男，1964 年生，教授，研究方向为大数据、物联网.

袁　源：　女，1976 年生，教授级工程师，研究方向为大数据、电信网络.