An Apriori Improving Algorithm Based on Encoding and Scale Compression of Items

Yan YANG, Yongquan ZHOU, Yong WANG

College of mathematics and computer Science Guangxi University for Nationalities, Nanning 530006, China

E-mail: yongquanzhou@126.com

Abstract: On the base of analyzing the shortcomings of Apriori Algorithm, using the encoding and scale compression strategy of items to improve the Apriori Algorithm is advanced. The main idea of the algorithm is followed. In dealing with encoding the item sets we take the encoding and compression method, by comparison between support and min-support of items, deleting ones that are less than min-upport items, containing ones whose frequency is equal or greater than min-support then obtain the frequent-k item sets by concentrated "and" operation among the inquiring projects. The mining results are the ones whose frequency is equal or greater than min-support, and the said "k" deprived from the automatically searching by computers. This algorithm need only once scanning over database. The result of experiment shows that the improved Apriori Algorithm improves the efficiency of mining, both in time and space.

Keywords: data mining; association rules; Apriori algorithm; coding

一种基于项编码和项规模压缩的 Apriori 改进算法

杨　艳，周永权，王　勇

广西民族大学数学与计算机科学学院，南宁，中国，530006

E-mail: yongquanzhou@126.com

摘　要: 在分析 Apriori 算法存在的不足之基础上，提出了采用项编码和项规模压缩策略的 Apriori 改进算法。该算法主要思想是采用项编码和项规模压缩方法对项目集编码，通过项的支持度与最小支持度的比较，删除小于最小支持度的项，保留满足最小支持度的频繁项，通过候选项目集中项目“与”运算得到频繁 k-项集。挖掘得到的结果是“满足最小支持度的长度最大的频繁k-项集”，其中的 k 是由计算机自动搜索得到的。本算法只需扫描一次事务数据库。实例验证结果表明，改进后的 Apriori 算法在时间和空间方面都提高了挖掘的效率。

关键词: 数据挖掘；关联规则；算法；编码

1 引言

Apriori 算法[1]是 R.Agranal 和 R.Srikant 于 1994 年提出的一种关联规则算法。它主要用频繁项集的性质，采用逐层搜索迭代先来生成候选频繁项目集，然后扫描数据库，累积每个项的计数，并收集满足最小支持度计数的项，显然 Apriori 算法在时间和空间上的耗费量都很大。针对 Apriori 算法存在的不足，许多基于矩阵的关联规则算法被提出，用于提高算法效率。如李超提出了基矩阵的 Apriori 改进算法[2]，通过 0-1 矩阵仅需扫描一次数据库来减少数据库扫描次数，节省算法执行时间。胡慧蓉提出了 SLIG 算法[3]，他是通过引入项目可辨识向量及其“与”运算，将频繁项目集的产生过程转化为项目集的关系矩阵中向量运算过程，扫描一次数据库，提高挖掘效率。然而，以上两种改进的关联规则算法均只从时间复杂度上对 Apriori 算法进行了改进，减少了算法的时间复杂度。若事务的项目集规模很大，则其需占据很大的内存空间，算法空间复杂度仍很大。而丁艳辉提出 BOM 算法[4]，该算法通过矩阵来表示基础数据，并通过矩阵的运算直接得到频繁k-项集。该文中所指的k是由用户给定的，而不是由计算机自动搜索得到的。也就是说，是用户指定要得到“频繁k-项集”，然后是计算机帮助寻找“频繁k-项集”。然而，人们事先不可能知道“频繁 k-项集”中的 k 到底有多大，k 过大了，事务数据库中可能不存在“频繁k-项集”，　k 太小了，事务数据库中

广西自然科学基金(0832082; 0991086); 广西民族大学研究生教育创新计划（gxun-chx2009091）资助项目。

又存在大量的“频繁 k-项集”。这与数据挖掘的“自动发现”不一致。

本文提出的采用项编码和项规模压缩方法的 Apriori 改进算法，则是采用项编码和项规模压缩方法对项目集编码，通过项的支持度与最小支持度的比较，删除小于最小支持度的项，保留满足最小支持度的频繁项，通过候选项目集中项目“与”运算得到频繁 k-项集。整个挖掘过程均由计算机自动实现，用户只给定最小支持度即可，所得到的结果是“满足最小支持度的长度最大的频繁 k -项集”，其中的 k 是由计算机自动搜索得到的。本算法只需扫描一次事务数据库。这不仅节约了扫描数据库的时间，还减少用于存储候选项集的空间。

2 基于项编码和项规模压缩的 Apriori 改进算法

2.1 Apriori 算法

Apriori 算法的基本思想是通过对数据集进行多步处理来寻找最大频繁项目集.首先,简单统计所有含一个元素项目集出现频率,并找出那些不小于最小支持度的项目集, 即频繁 1-项集. 然后, 循环处理直到再没有最大频繁项目集生成. 循环过程是: 第 k 步,根据第 $k-1$ 步生成的频繁 $(k-1)$ -项集产生侯选 k -项集, 然后对数据库进行搜索, 得到侯选项目集的项集支持度, 与最小支持度比较, 从而找到频繁 k-项集.该算法在每次生成频繁项目集的过程中均需对数据库进行全扫描。

2.2 算法改进的思想

对所有项在事务出现的记录上编码，编码同时统计出项的支持度。给项集中的每个项设一个标志位 delete，初值设位 false。每次计算的项的支持度与最小支持度进行比较，若不小于最小支持度，则将标志位置 false，否则置位 true，并将该项从候选项集中删除。这样一个候选项集中就只包含满足支持度大于或等于最小支持度的项。以此循环直到得出符合关联规则的频繁 k-项集，此算法只扫描了一次数据库，且随着候选项集数目的增多，此方法大大节省了存储空间，提高了整个算法的效率。

2.3 项编码规则

项编码是此改进算法的第一要点。项编码主要是对事务中各项进行 0-1 编码，假如共有五个事务（T1,T2,T3,T4.T5），每个事务对应编码中的一个位置，如果某个项出现在第一个事务中，就相应的将编码的第一个位置设为“1”，否则设为“0”，依次类推。例如，某一项的记录出现在事务 T1,T3.T5 中，那么据编码规则可得到该项的编码为（10101）。

2.4 项规模压缩原理

对每个项进行编码后，通过计算编码中“1”的个数来得到支持度，支持度大于或等于最小支持度的项的集合就是频繁 1-项集。将频繁 1-项集的各项两两“与”运算，例如项 I_1 的编码为 100110111，项 I_2 的编码为 111101011，那么 $I_{1\cap 2} = I_1 \cap I_2 = (100110111) \cap (111101011) = 100100011$; 并计算新编码中“1”的个数，即支持度计数，若支持度小于最小支持度，则将该项标志位 delete 置为 true，即删除该新编码所对应的项，否则将该项标志位置 false，即将该项暂时保留并继续对下两个项进行“与”运算，得到候选二项集，并通过判断标志位的值得到频繁 2-项集。依次类推，直到得到频繁 k-项集，由于内存中不保存临时项目集，故最终内存只需分配相应空间保存频繁 k-项集。

2.5 改进 Apriori 算法的描述

输入：事务数据库 D；最小支持度计数阀值 min_sup.

输出：D 中的频繁项目集 L.

步骤 1: 扫描事务数据库，对每个商品 I_i（i=1,2,3…）进行 0/1 编码。设最小支持度为 min_sup；

步骤 2：将项目 I_i 的编码两两进行“与”运算，得到候选 l-项集的集合 C_l，通过支持度和 min_sup 的比较来确定标志位的值；

步骤 3：针对步骤 2 中得到的候选项集，将标志位为“true”的项集删除，标志位为“false”的项集保留，得到频繁 l-项集，并将之存储到内存；

步骤 4：将步骤 3 得到的频繁项集的项两两进行“与”运算，得到候选 m-项集的集合 C_m，通过支持度和 min_sup 的比较来确定标志位的值，将标志位为“true”的项集删除，标志位为“false”的项集保留，得到频繁 m-项集；

步骤 5：循环步骤 4，依次得到候选 k -项集。判断候选项目集的标志位，若标志位全为“true”，则结束算法，并将结果存储到内存中，否则返回步骤 5。

3 实例验证

978-1-935068-10-5 © 2010 SciRes.

本文以文献[1]中 AllElectronics 事务数据库实例来分析和说明该算法的优越性。

Table 1. All Electronics sub-shop work data
表 1. AllElectronics 某分店的事务数据

	项集	编码	支持度计数	标志位
频繁 2-项集	{I1,I2}	100100011	4	false
	{I1,I3}	000010111	4	false
	{I1,I5}	100000010	2	false
	{I2,I3}	001001011	4	false
	{I2,I4}	010100000	2	false
	{I2,I5}	100000010	2	false

扫描数据库对各项进行 0-1 编码，通过“与”最终得到频繁 1-项集，如表二。该算法整个过程只扫描了这一次数据库。

Table 2. The generative process of rule mining and frequent 1-items
表 2. 规则挖掘及频繁 1-项集生成过程

	项集	编码	支持度计数	标志位
频繁 1-项集	{I1}	100110111	6	false
	{I2}	111101011	7	false
	{I3}	001011111	6	false
	{I4}	010100000	2	false
	{I5}	100000010	2	false

由最小支持度计数不小于 2，得到表二中的标志位均为“false”。频繁 1-项集的项编码两两进行“与”运算，得到候选 2-项集（如表 3），但该候选项集并不存储在内存中，而是通过标志位删除候选项目集中不满足最小支持度计数的项，剩下的即为频繁 2-项集。此过程不需要扫描数据库。

Table 3. The generative process of rule mining and candidate 2-items
表 3. 规则挖掘及候选 2-项集生成过程

	项集	编码	支持度计数	标志位
候选 2-项集	{I1,I2}	100100011	4	false
	{I1,I3}	000010111	4	false
	{I1,I4}	000100000	1	true
	{I1,I5}	100000010	2	false
	{I2,I3}	001001011	4	false
	{I2,I4}	010100000	2	false
	{I2,I5}	100000010	2	false
	{I3,I4}	000000000	0	true
	{I3,I5}	000000010	1	true
	{I4,I5}	000000000	0	true

通过表 3 中标志位的值删除相关项目集得到频繁 2-项集（如表 4）。此过程计算机自动完成，且不需要存储在内存，从而节省了内存的存储空间。

同理，频繁 2-项集中的项两两“与”运算得到候选 3-项集（如表 5）。此过程仍不需要扫描数据库。

同样的方法，通过表五中标志位的值删除相关项目集得到频繁 3-项集，如表 6 所示.

Table 4 The generative process of rule mining and frequent 2-items
表 4 规则挖掘及频繁 2-项集生成过程

TID	商品 ID 的列表
T100	I1,I2,I5
T200	I2,I4
T300	I2,I3
T400	I1, I2,I4
T500	I1,I3
T600	I2,I3
T700	I1,I3
T800	I1 ,I2,I3,I5
T900	I1, I2,I3

Table 5 The generative process of rule mining and candidate 3-items
表 5 规则挖掘及候选 3-项集生成过程

	项集	编码	支持度计数	标志位
候选 3-项集	{I1,I2,I3}	000000011	2	false
	{I1,I2,I5}	100000010	2	false
	{I1,I3,I5}	000000010	1	true
	{I2,I3,I4}	000000000	0	true
	{I2,I3,I5}	000000010	1	true
	{I2,I4,I5}	000000000	0	true

Table 6. The generative process of rule mining and frequent 3-items
表 6. 规则挖掘及频繁 3-项集生成过程

	项集	编码	支持度计数	标志位
频繁 3-项集	{I1,I2,I3}	000000011	2	false
	{I1,I2,I5}	100000010	2	false

由于生成候选项集及频繁项集的过程是计算机自动完成的，并且一开始计算机并不知道结果 k 为多少，所以计算机会自动循环运算，直到满足结束条件（候选项集中所有的项的标志位值均为“true”）。计算机运算得到的候选 4-项集如表 7 所示。

Table 7. The generative process of rule mining and frequent 4-items
表 7. 规则挖掘及候选 4-项集生成过程

	项集	编码	支持度计数	标志位
候选 4-项集	{I1,I2,I3,I5}	000000010	1	true

由上表可知，候选 4-项集中所有项的标志位值均为“true”，运算结束；且此时得到的候选项集为 $(k+1)$-项集，其上一个频繁 k-项集即为所求结果。将则频繁 k-项集存储到内存中；否则继续运算。从上

述过程可知，频繁 3-项集即为最终结果。整个过程中编码运算使数据库只扫描了一次，计算机自动运算直到得到候选 $(k+1)$ -项集，且只将频繁 k -项集的项存储到内存空间。该算法时耗减少的同时空间也大大减小。

4. 算法复杂度分析

基于项编码和项规模压缩的 Apriori 改进算法仅需扫描一次数据库，且不需花费时间来比较、修剪和排序矩阵中的项。其时间复杂度为 $O[nm(m-1)(m-2)(m-3)\cdots(m-k+1)]=O(nA_m^k)$ ，m 为候选项集中项集的行数，n 为候选项集的列数，k 为待发现的频繁项集的大小；BOM 算法的时间复杂度为 $O(n\,m^k)$ ，m 为修剪后矩阵的行数，n 为矩阵的列数，k 为待发现的频繁项集的大小；Apriori 算法的时间复杂度为 $O(|D|^{k+1})$ ，$|D|$ 为交易事务数据库中包含的交易数量。在实际应用中，交易数量一般远大于商品数量，即 $|D|\times m$ ，但 m 仍很大，$m^k > A_m^k$ 仍成立。所以，本文的改进算法比 Apriori 算法及其他几种关于矩阵的改进算法的时间特性都要好。Apriori 算法及文献[2]、[3]提出的关于矩阵的改进算法均会产生大量候选项集，且需占用大量内存空间。本文的改进算法及文献[4]提出的 BOM 算法只需存储频繁 k -项集，大大节省了存储空间。

5. 结束语

随着信息技术的不断推广应用，将人们带入了一个信息爆炸的时代。如何充分利用这些数据信息为决策者提供决策支持成为一个十分迫切的又棘手的问题，人们除了利用现有的关系数据库标准查询语句得到一般的直观的信息以外，必须挖掘其内含的、未知的却又实际存在的数据关系。著名的 Apriori 算法是一种挖掘关联规则的算法。文中通过项编码和规模压缩的方法对传统 Apriori 算法进行了改进，提出了利用项编码和项规模压缩的 Apriori 改进算法. 实例验证结果表明，改进后的 Apriori 算法在时间和空间两方面都提高了挖掘的效率减少了数据库扫描次数及数据项存储空间，提高了算法的效率。

References (参考文献)

[1] Fan Ming, Meng Xiao-feng. Data ming concept and technology, Beijing: Mechanical industry press,2003.
范明，孟小峰等译. 数据挖掘概念与技术[M].北京：机械工业出版社.2003.

[2] LI Chao; YU Zhaoping. Improved Method of Apriori Algorithm Based on Matrix. Computer Engineering ,2006,32(23):68-69
李超，余绍平. 基于矩阵的 Apriori 算法改进[J].计算机工程，2006,32(23):68-69

[3] HU Hui-rong; WANG Zhou-jing. Fast algorithm for mining association rules based on relationship matrix, Computer Applications, 2005,25(7):1577-1579.
胡慧蓉，王周敬. 一种基于关系矩阵的关联规则快速挖掘算法[J].计算机应用，2005,25(7):1577-1579.

[4] DING Yan-Hui WANG Hong-Guo GAO Ming GU Jian-Jun .A New Algorithm Based on Matrix for Mining Association Rules, Computer Science, 2006,33(4):188-189.
丁艳辉，王洪国，高明等. 一种基于矩阵的关联规则挖掘新算法[J]. 计算机科学，2006,33(4):188-189.