Apriori 改进算法综述

何云峰
(泉州医学高等专科学校,福建 泉州 362011)

摘　要：介绍了近十几年中国学者对 Apriori 算法的宽度优先算法的改进研究。

关键词：Apriori 算法；Apriori 改进算法；宽度优先算法
中图分类号：TP311　　**文献标识码**：A　　**文章编号**：1674-7720(2013)06-0001-03

A survey of improved Apriori algorithm

He Yunfeng
(Quanzhou Medical College, Quanzhou 362011, China)

Abstract: This paper introduces Chinese scholars' research on the improvement of width priority algorithm of Apriori in recent ten years.

Key words: Apriori algorithm; improved Apriori algorithm; width priority algorithm

AGRAWL R 等人于 1993 年提出了挖掘关联规则最具影响力的 Apriori 算法。该算法的基本思想是先找出事务数据库中具有最小支持度的项目集(即最大项目集),再根据最大项目集生成关联规则。其中生成最大项目集是核心问题,其思想为:第一步,统计所有含一个元素项目集出现的频率,并找出不小于最小支持度的项目集,从第二步开始循环处理直到再没有最大项目集生成。循环过程是:在第 k 步中,根据第 $k-1$ 步生成的 $k-1$ 维最大项目集产生 k 维候选项目集,然后对数据库进行搜索,得到候选项目集的项集支持度,并与最小支持度比较,从而找到 k 维最大项目集。

之后,AGRAWL R 等人又提出了 Apriori 算法的改进算法 AprioriTid 算法和 AprioriHybird 算法。AprioriTid 算法对 Apriori 算法做了调整,它的特点是在第一次遍历数据库之后,就不再扫描数据库,而是用上次扫描生成的候选项目集,扫描的同时还会计算出频繁项目集的支持度。该算法以候选项目集来代替原数据库,从而减少了总是要扫描原数据库统计支持计数的开销。AprioriHybird 算法则是 Apriori 和 AprioriTid 的结合,初始扫描数据库时采用Apriori 算法,当生成的候选项目集大小可以存放到内存中进行处理时采再转成 AprioriTid 算法。1995 年 PARK 等人提出了 DHP 算法,即在生成频繁 2-项目集时由于运算量大而引入 Hash 技术来产生频繁 2-项目集。以上 4 种算法属于宽度优先算法,还有深度优先算法(如 FP-growth 算法、OP 算法、TreeProjection 算法)、数据集划分算法、采样算法、增量式更新算法等,由于后几种算法本质上已不同于 Apriori 算法,所以本文对其不再详述。

我国学者开始研究关联规则挖掘较晚,约在 2000 年左右。起初是跟着国外学者的思路先研究 Apriori 的改进算法 AprioriTid[1]、AprioriHybird 和 DHP,随后从 Apriori 算法的性质、扫描数据库的次数、消减数据库容量、转换数据库存储方式及与其他技术(如 Hash)和算法联合等方面对 Apriori 算法进行了改进。下面以改进内容为序,予以详述。

1 利用 Apriori 算法性质的改进

2002 年有学者根据 Apriori 算法中生成 k 维数据项集的一个推论:T_k 是 k 维数据项集,如果所有 $k-1$ 维高频数据项集集合 L_{k-1} 中包含 T_k 的 $k-1$ 维子集的个数小于 k,则 T_k 不可能是 k 维最大数据项集,从而在原 Apriori 算法从 C_k 中取元素,然后求该元素的子集,判断该子集是否在$|C_k|$中需进行的计算次数减少,即在判断某一项集是频繁时减少了判断次数[2]。

2004 年有学者根据性质:若 k 维数据项目集 $X=\{i_1, i_2,\cdots,i_k\}$中,存在一个 $j\in X$ 使得 $|L_{k-1}(j)|<k-1$,则 X 不是频繁项目集。其中 $|L_{k-1}(j)|$ 表示 $k-1$ 维频繁项目集的集合 L_{k-1} 中所包含 j 的个数。在修剪频繁集时进行了改进。又在连接步骤引入头、尾结点生成函数和优化连接函数改

进了连接步骤，同时按事务压缩技术原理压缩了数据库容量[3]。

2006 年有学者发现性质：生成候选项集 C_k 时，在 L_{k-1} 中的一个项集 I 与 L_{k-1} 中所有项集进行连接，把连接得到的不同 $k\_$ 项集存入 TQ，然后立即确定包含项集 I 的所有符合剪枝后的候选 $k\_$ 项集。根据这一性质省略了在 L_{k-1} 中寻找 $k\_$ 项集的所有 $(k-1)\_$ 子集的费时剪枝操作[4]。

2008 年有学者根据 $k\_$ 维频繁项集所有 $k-1$ 维子集均是频繁的且子集个数为 k 这一性质提出两点改进：(1)如果 C_{k-1} 中存在不符合最小支持度的元素，则删除它；而且将项数等于 $(k-1)$ 的事务与 k 项事务有交集的事务删除。(2)二次扫描数据库，分别产生频繁 1、2 项集 L_1、L_2，在生成频繁 3 项集时，首先由频繁 2 项集自乘生成候选 3 项集 C_3，依次取出 L_2 中各元素，检查其是否为 C_3 的子集，若是则计数加 1，扫描完 L_2 中各元素后，看 C_3 中各元素的计数，最终计数等于 3 的则为 3 项频繁的。更多项频繁集也是这种方法的重复[5]。

2 数据库扫描次数和消减容量的改进

2003 年有学者以减少扫描数据库的次数为目的，引入了概率估算候选频繁项集的思想[6]。

2005 年有学者利用频繁项集 L_{k-1} 对数据库进行筛选，如果在 L_{k-1} 没有包含它们的集合则从数据库中删掉这部分不符合最小支持度的元素，而且将项数等于 $k-1$ 的事务删除，从而减少了数据库容量[7]。

2008 年有学者通过第一次扫描数据库及最小支持度确定频繁 1 项集，之后根据频繁 1 项集重新组织数据库，再次扫描，把每个子集出现的次数统计出来，再根据最小支持度筛出频繁 $k\_$ 项集[8]。该方法仅扫描 2 次数据库，节约了时间，但在处理数据库中每项事务(即拆成子集)、统计其次数等上需花费一定的空间和时间。

2009 年有学者利用如果某事务项目数小于 $k\_$ 频繁项目集的项目个数，则在更新频繁项目集时可以不扫描的性质，压缩了数据库事务集；为提高剪枝效率，首次支持度裁剪后，比较非频繁项集项目数和频繁项集项目数，取小值进行剪枝操作[9]。

2011 年有学者引入了用户兴趣项，从而可以比较大范围地缩减数据库容量；同时使用数组方式表示数据库，减少了数据库的扫描次数[10]。

3 转换数据库存储方式

2004 年有学者把数据库转换成矩阵表示，事务为行，具体的项目为列，若第 i 条事务在第 j 列有项目，则该处记为 1，否则为 0。扫描数据库时，该矩阵的对应项也随之以加 1 的频率改写。最后考察矩阵的对应项与支持度的关系[11]。

2006 年有学者在以往学者提出的把数据库转为矩阵的基础上[11]，使用自定义的矩阵运算，产生新的数据库矩阵及完成相应的剪枝步骤。在生成关联规则时使用了概率论的基本性质，减少了计算量[12]。

2007 年有学者在以往学者提出的把数据库转为矩阵的基础上[11]，使用行向量内积的方法搜寻频繁项集，该方法仅扫描一次数据库，但要多次使用矩阵相乘获取频繁项集[13]。

2009 年有学者提出了一种事务的二元组表示法，该二元组直接用字段的值串和串的出现次数来替换原始事务数据库，并在此基础上扫描一遍数据库。例如通过扫描、处理后得到项目串和支持数 $[I_1I_2,2]$。该表示法所占内存大小只取决于数据库的基，即各元素取值种类的乘积。例，数据库有 4 个字段，每个字段的取值个数分别为 (2,3,5,3)，则该二元组数目不大于 $2\times3\times5\times3=90$。同时用链表结构来表示该二元组，能加快一定速度[14]。

2009 年有学者改变了数据库的存储形式，即由原来的第几条事务包含有哪几个元素(例如“T1 A,B,E”)变为哪个元素被哪些事务包含 (例如 “A T1 T4 T8 T9”)。根据最小支持度可生成 $1\_$ 项频繁集，再通过交集运算生成 $2\_$ 项频繁集(例如 “AB T1 T8”)。又根据连接时的一个定理，即 L_{k-1} 和 L_1 连接时只需考察 L_{k-1} 的最后一项与 L_1 中各项在 L_1 中索引的大小关系，从而减少了不必要的重复连接[15]。

2009 年有学者把数据库的事务转为十字链表方式存储。该方法仅扫描一次数据库，节约了时间，但十字链表结构复杂，其生成也需消耗时间[16]。

4 Hash 技术与数据库扫描次数及数据库消减的合用

2003 年有学者用散列技术把生成的 $(k+1)\_$ 项集散列到散列表中并计数，同时考察支持度；同时使用性质：不包含任何 $k\_$ 项集的事务不可能包含任何 $(k+1)\_$ 项集[17]，来压缩事务数据库。

2004 年有学者提出在扫描数据库时引入散列技术，以达到降低数据库的扫描次数，同时根据支持度的要求减少不可能成为频繁项目集的候选项，从而了提高数据项集频度的统计速度[18]。

2007 年有学者提出在产生 $1\_$ 频繁项目集和 $2\_$ 频繁项目集时，使用 Hash 技术；在产生 $k\_$ 频繁项目集时使用事务压缩优化方法[19]。

5 转换数据库存储方式与 Apriori 性质或其他方面的联合使用

2008 年有学者在原数据库转成布尔矩阵的基础上，根据交易记录各项是按字典排序的，从而生成的候选项集和频繁项集也是有序的这一性质，减少了判断次数；同时利用 $k\_$ 维数据项目集的频繁项集的必要条件使它

的所有 $k-1$ 维子集均是频繁项目集这一性质，在一定程度上优化了频繁项集的修剪[20]。

2011 年有学者对 2_ 项集使用了散列表技术，能较快速地获得频繁 2_ 项集。同时对数据库生成候选 k_ 项集($k \geqslant 3$)时转为前学者[15]的矩阵方式存储，如 ABC 出现在第 1、3、4 条事务中则表示为(1011)，再根据支持度就可生成频繁 k_ 项集[21]。

2012 年有学者在前学者[16]十字链表的基础上，利用 k 维频繁项集的所有 $k-1$ 维子集均是频繁项集这一性质，优化了候选频繁项集的生成和数据库的扫描[22]。同时也引出了其他学者对十字链表的改进。

6 Apriori 算法与其他算法的联合使用

2007 年有学者提出 Apriori 算法与聚类算法相结合应用于 IDS 日志分析中[23]。

2010 年有学者提出用遗传算法对数据库进行性编码、评估和遗传，再使用 Apriori 的连接、剪枝和提取步骤完成整个挖掘过程[24]。

2012 年有学者把 Apriori 算法应用于矩阵聚类法中[25]。

2012 年有学者把云计算的两个重要步骤：Map 函数(映射)和 Reduce 函数(归约)，分别引入到 Apriori 算法的连接和剪枝步骤中，该思想丰富了 Apriori 的内容[26]。

从上面的论述中可以看到，起初对 Apriori 算法的改进着重于算法本身，比如利用其性质改进频繁项集的生成、缩减数据库容量、扫描次数等。后来算法本身的改进点基本都被挖掘出来了，就转向了数据库存储方式的改进，如转成布尔型矩阵、链表，数据库存储方式的改进可谓是开创了 Apriori 算法改进的一个新纪元。可是接下来似乎有点山穷水尽的味道，研究转向了与其他算法的合作，如与遗传算法、云计算的合作。Apriori 算法的宽度优先算法改进前途是否依然光明，我们拭目以待。

参考文献

[1] 颜雪松,蔡之华.一种基于 Apriori 的高效关联规则挖掘算法的研究[J].计算机工程与应用,2002,32(10):209-211.

[2] 李绪成,王保保. 挖掘关联规则中 Apriori 算法的一种改进[J].计算机工程,2002,28(7):104-105.

[3] 徐章艳,刘美玲,张师超,等.Apriori 算法的三种优化方法[J].计算机工程与应用,2004,40(36):190-192.

[4] 胡吉明,鲜学丰.挖掘关联规则中 Apriori 算法的研究与改进[J].计算机技术与发展,2006,16(4):99-101.

[5] 杨启昉,马广平.关联规则挖掘 Apriori 算法的改进[J].计算机应用,2008,28(12):217-218.

[6] 陈江平,傅仲良,徐志红.一种 Apriori 的改进算法[J].武汉大学学报(信息科学版),2003,28(1):94-98.

[7] 冯兴杰,周谆.Apriori 算法的改进[J].计算机工程,2005,31(增刊):172-173.

[8] 郭健美,宋顺林,李世松.基于 Apriori 算法的改进算法[J].计算机工程与设计,2008,29(11):2814-2815.

[9] 吴斌,肖刚,陆佳炜.基于关联规则挖掘领域的 Apriori 算法的优化研究[J].计算机工程与科学,2009,31(6):116-118.

[10] 刘维晓,陈俊丽,屈世富,等.一种改进的 Apriori 算法[J].计算机工程与应用,2011,47(11):149-151.

[11] 马盈仓.挖掘关联规则中 Apriori 算法的改进[J].计算机应用与软件,2004,21(11):82-83.

[12] 李超,余昭平.基于矩阵的 Apriori 算法改进[J].计算机工程,2006,32(23):68-69.

[13] 刘以安,羊斌.关联规则挖掘中对 Apriori 算法的一种改进研究[J].计算机应用,2007,27(2):418-420.

[14] 张春生.改进的数据库一次扫描快速 Apriori 算法[J].计算机工程与设计,2009,30(16):3811-3813.

[15] 刘华婷,郭仁祥,姜浩.关联规则挖掘 Apriori 算法的研究与改进[J].计算机应用与软件,2009,26(1):146-148.

[16] 黄建明,赵文静,王星星.基于十字链表的 Apriori 改进算法[J].计算机工程,2009,35(2):37-38.

[17] 黄进,尹治本.关联规则挖掘的 Apriori 算法的改进[J].电子科技大学学报,2003,32(1):76-79.

[18] 王创新.关联规则提取中对 Apriori 算法的一种改进[J].计算机工程与应用,2004,40(34):183-185.

[19] 柴华昕,王勇.Apriori 挖掘频繁项目集算法的改进[J].计算机工程与应用,2007,43(24):158-161.

[20] 钱光超,贾瑞玉,张然,等.Apriori 算法的一种优化方法[J].计算机工程,2008,34(23):196-198.

[21] 栗晓聪,滕少华. 频繁项集挖掘的 Apriori 改进算法研究[J].江西师范大学学报(自然科学版),2011,35(5):498-501.

[22] 刘玉文.基于十字链表的 Apriori 算法的研究与改进[J].计算机应用与软件,2012,29(5):267-269.

[23] 朱金清,王建新,陈志泊.基于 APRIORI 的层次化聚类算法及其在 IDS 日志分析中的应用[J].计算机研究与发展,2007,44(增刊):326-330.

[24] 詹芹,张幼明.一种改进的动态遗传 Apriori 挖掘算法[J].计算机应用研究,2010,27(8):2929-2930.

[25] 陈立宁,罗可.基于 Apriori 算法的确定指定精度矩阵聚类方法[J].计算机工程与应用,2012,48(7):139-141.

[26] 吴琪.基于云计算的 Apriori 挖掘算法[J].计算机测量与控制,2012,20(6):1653-1655.

(收稿日期：2012-12-17)

作者简介：

何云峰，男，1975 年生，讲师，主要研究方向：数据挖掘，证券。