ISSN 1009-3044
Computer Knowledge and Technology 电脑知识与技术
Vol.15, No.3, January. 2019

E-mail: wltx@dnzs.net.cn
http://www.dnzs.net.cn
Tel:+86-551-65690963 65690964

基于树堆的频繁项集挖掘算法

向春梅,陈超

(成都信息工程大学,四川 成都 610225)

摘要:近年来数据库信息越来越庞大,利用已有的算法来快速挖掘频繁项集已经变得越来越困难。为了解决这个问题,论文提出一种挖掘频繁项集的新算法。该算法首先需要为每一个项目设定一个不重复的优先级,然后采用最小优先级树堆的数据结构存储数据库中的每条事务,最后,从最小优先级树堆中寻找数据库中的各种频繁项集。通过实验测试,在相同的支持度下,使用该算法来挖掘频繁项集的运行效率的确比Apriori算法和FP-growth算法的运行效率要高。

关键词:数据挖掘;关联规则;频繁项集;树堆; Apriori ;FP-growth

中图分类号:TP312 文献标识码:A 文章编号:1009-3044(2019)03-0026-03

DOI:10.14004/j.cnki.ckt.2019.0128

A New Algorithm for Mining Frequent Items

XIANG Chun-mei,CHEN Chao

(Chengdu University of Information Technology,Chengdu 610225,China)

Abstract: In recent years, database information has become more and more huge. It is more and more difficult to use the existing algorithms to quickly mine frequent items.In order to solve this problem, the paper proposes a new algorithm for mining frequent items.First, the algorithm needs to set a non-repetitive priority for each item. Second, the data structure of the minimum priority tree-heap is used to store each transaction in the database, and finally, the minimum priority tree-heap is looked for various frequent items in the database.Under the same support threshold,the experiments show that using the algorithm to mine the frequent item sets is more efficient than the Apriori algorithm and the FP-growth algorithm.

Key words: data mining; association rules; frequent items; tree-heap; Apriori; FP-growth

关联规则挖掘是数据挖掘研究领域的一个重要研究方向[1],其目的是为了从大量数据中发现项与项之间的关联关系。传统的关联规则采用的是支持度—置信度框架,前者用于衡量关联规则在整个数据集中的统计重要性,后者用于衡量关联规则的可信程度。一般来说,只有支持度和置信度均较高的关联规则才可能是用户感兴趣、有用的关联规则[2]。关联规则的挖掘可以分成两个部分来实现[3]:挖掘数据库中的频繁项集以及从频繁项集中挖掘关联规则。在上面的两个步骤中,算法的总体性能主要由第一个部分决定[4]。因此,论文主要研究频繁项集的挖掘。

Apriori[5]算法是Agrawal等人首次提出的针对关联规则挖掘问题的算法。该算法采用广度优先的搜索策略,自顶向上的遍历思想,先产生候选集后获得频繁项集的思路。其缺点是多次扫描数据库,会增大IO负载;会产生大量不需要的候选频繁项集[6-8]。FP-growth[9]算法是由Jiawei Han等人提出的不产生候选集的挖掘频繁项集的算法。该算法的思路也可以分为两个部分[10]:构造FP-tree和通过FP-tree挖掘频繁项集。虽然该算法只需要扫描两次数据库,并且不产生候选集,但是当数据库很大时,构造基于内存的PF-tree也是不现实的[11]。因此论文提出一种新的挖掘频繁项集的算法记为MiningByTreap算法,该算法不产生候选项集,也不会存在无法构建树堆的情况。

1 MiningByTreap 算法

树堆是一种同时具有树和堆两种特性的数据结构。树堆中的节点包含两部分,数据值x和属于该节点的唯一的优先级p。树堆的节点有两个特性,对于数据值x而言,它遵循二叉查找树的属性,根节点数据值大于左子树的所有节点且小于右子树的所有节点;对于优先级而言,它遵循堆的属性,根节点的优先级大于左右子树中所有节点。就堆的数据结构而言,根据每个节点与其子节点的大小关系,可以分为大顶堆和小顶堆,所以树堆也可以根据每个节点优先级与其子节点的优先级的大小关系,分为最大优先级树堆和最小优先级顶树堆[12]。

论文选择最小优先级树堆作为研究的数据结构,提出的算法由三个主要的程序构成,首先通过计算优先级的子程序计算出数据库中每个变量的优先级。计算好优先级之后,通过创建树堆的子程序为每条事务创建一个树堆。最后,通过挖掘频繁项集的子程序,采用前序遍历的方法遍历树堆,从而找出频繁项集。论文以表1展示的数据作为研究对象,逐步阐述论文提出的算法过程。

收稿日期:2018-12-10

作者简介:向春梅(1992—),女,四川成都人,硕士研究生在读,主要研究方向为计算机网络与通信;陈超(1979—),男,副教授,硕士,主要研究方向为无线通信及移动互联网。

表1　数据库D

TID	项目集
100	B C D
200	C D E
300	A B D F
400	B C D E
500	B D F

1.1 计算优先级

在数据库中，即使项目集不是频繁的，但在规则生成中它可能也具有一定的重要性。在所有关联算法中，这种不频繁的项目往往被省略并没有给出很多地分析。在计算优先级的算法中，首先找到所有项目的频率以确定该项目的优先级。如果只着眼于需要的高频繁项集，在后续挖掘规则时，可以根据最小支持度阈值，得到满足要求的频繁项集。当然也可以选择挖掘出所有的项集最优的组合。算法1是计算优先级的过程。

算法1：

输入：数据库D，包含n个不同项目和m条事务。

输出：具有n个项目及其优先级的Map集合。

步骤1：扫描数据库D并找出每个项目的频率f；

步骤2：将每个项目按照频率f依次增的顺序排列，并存入数组L[n]；

步骤3：将每个项目及其所在数组L[n]中的下标位置加1后，存入Map集合。

经过算法1后，可以得到每个项目的频率f和优先级数p，如表2所示。

表2　项目的信息

项目	频率f	优先级p
A	1	1
B	4	5
C	3	4
D	5	6
E	2	2
F	2	3

1.2 构建最小优先级树堆

论文使用的是最小优先级树堆的数据结构。它满足每个节点的优先级都比它的子节点的优先级小的特点，在此条件下再保证，满足每个节点数据值大于它左子节点且小于右子节点，这种特性相当于将树堆中的左右节点也做了排序[9]。

构建这种最小优先级树堆的程序必须是在每个项目的优先级已经确定的情况下才能执行。算法需要给定最小支持度阈值，将满足要求的节点按照其优先级大小构建最小优先级树堆。如果是挖掘所有项目集，可以将最小支持度阈值设为1，为了满足所构建的最小优先级树堆具有更低的深度，算法每次添加一个新节点是以完全二叉排序树进行存储，然后对该节点按照节点的优先级满足小顶堆的属性进行调整，最终得到的最小优先级树堆。算法2是构建树堆的过程。

算法2：

输入：具有n个项目及其优先级的Map集合，最小支持度阈值

输出：最小优先级树堆的List集合

步骤1：遍历每条事务中每个项目节点，如果满足最小支持度阈值则调用insert函数，添加该节点，创建一个完全二叉树；

步骤2：对每个完全二叉树的每个节点按照该节点的优先级进行调整；

步骤3：递归调整该节点的子节点；

步骤4：依次调整，直至调整到根节点；

步骤5：返回最小优先级树堆T。

设定minSupport=10%，算法2为每条事务创建一个最小优先级树堆，如图1所示。其中T300中的A频率为1，项目总数为17，经过计算可得，只有项目频率超过1才满足最小支持度阈值，所以节点A不会被加入最小优先级树堆中。

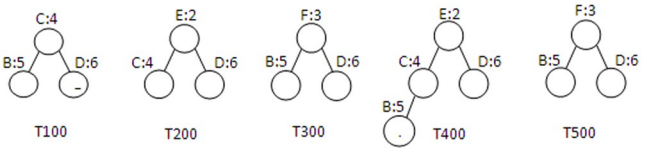

图1　最小优先级树堆

1.3 挖掘频繁项集

由于算法2中已经对最小支持度阈值并做出了相应的处理，因此在挖掘频繁项的算法中不需要输入最小支持度阈值。算法3采用前序遍历、自下而上地搜索频繁项集，直到离开节点并返回优先级值和其节点数据值。算法返回节点的父节点和兄弟节点。如果没有兄弟节点，则返回父节点和父节点的兄弟几点，依次类推，一直搜索到根节点，程序结束。

算法3：

输入：最小优先级树堆T

输出：频繁项集F

步骤1：前序遍历直到T中叶子节点的最左节点N；

步骤2：如果N节点存在兄弟节点则返回N节点的兄弟节、N节点和N节点的父节点，并加入频繁项集F中；

步骤3：如果N节点没有兄弟节点，则返回N节点的父节点的兄弟节点、N节点和N节点的父节点，并加入频繁项集F中；

步骤4：直到节点已经是根节点时，结束程序，返回频繁项集F。

通过算法3，得到频繁项集F，如表3所示：

表3　频繁项集F

TID	频繁项目集
100	{D B C }
200	{D C E}
300	{D B F}
400	{D B C } {D C E }
500	{D B F}

2 算法测试

为检验论文提出的算法的性能，将其与Apriori算法和FP-growth算法进行比较。实验环境：CPU为Intel Celeron 1.80GHz、内存4GB和Windows 7家庭普通版操作系统。算法

采用Java语言编写，在Eclipse开源软件上编译，分别实现了Apriori、FP-growth和Threap-mining三种算法，测试数据是从http://fimi.ua.ac.be/data/上下载的四种数据。通过对不同数据进行实验分析及测试，得到实验的结果如表4所示。分析表4可得，在相同支持度下，同一数据中，使用MiningByTreap算法挖掘频繁项的效率更高。同时测试了在同一数据不同支持度下使用论文提出的算法进行挖掘频繁项，实验结果如表5所示。分析表5可得，MiningByTreap算法在支持度较小时挖掘效率依然很高。

表4 三种算法的效率比较

最小支持度	数据名称	数据条数 单位：条	Apriori 单位：ms	FP-growth(ms)单位：ms	MiningByTreap 单位：ms
25%	Chess	3196	1,866	1,740	989
25%	Mushroom	8124	2,925	2,283	1,086
25%	Accidents	20000	10,703	9,041	4,636
25%	Pumsb	49046	60,570	49,729	42,622

表5 不同支持度下的算法效率

最小支持度	数据名称	数据条数 单位：条	MiningByTreap 单位：ms
1%	Pumsb	49046	56,329
5%	Pumsb	49046	47,454
25%	Pumsb	49046	42,622
50%	Pumsb	49046	37,497

3 结束语

本文提出的利用最小优先级树堆来挖掘频繁项集的算法，在不同大小的数据上应用均可得到相对其他算法更高的效率，而且可以快速地挖掘支持度较小的频繁项集。同时，该算法还可以有效地挖掘非频繁项集。

参考文献：

[1] 穆罕默德·扎基,小瓦格纳·梅拉.数据挖掘与分析：概念与算法[M].吴诚堃,译.北京:人民邮电出版社,2017.

[2] 张全红.面向大数据的关联规则算法研究[D].西安:西安科技大学,2017.

[3] 张健.关联规则中频繁与高效用项集挖掘算法研究[D].厦门:华侨大学,2017.

[4] 孙金鑫.数据挖掘中的关联规则的研究[J].智能计算机与应用,2018,8(3):132-135.

[5] 潘燕.关联规则下的数据挖掘算法分析[J].信息记录材料,2018,19(7):212-213.

[6] 王国胤,刘群.大数据挖掘及应用[M].北京:清华大学出版社,2017.

[7] Dong Juan Gu,Lei Xia. A Novel and Improved Apriori Algorithm[J]. Applied Mechanics and Materials,2015,3748(721).

[8] 王建明,袁伟.基于节点表的FP-Growth算法改进[J].计算机工程与设计,2018,39(1):140-145.

[9] Chien-Hua Wang,Li Zheng,Xuelian Yu,XiDuan Zheng. Using Fuzzy FP-Growth for Mining Association Rules[P]. 2017 International Conference on Organizational Innovation (ICOI 2017),2017.

[10] 何恒松,李文明,李文峰.基于FP增长的数据关联改进算法[J].电子测量技术,2017,40(9):58-64.

[11] Yi Zeng,Shiqun Yin,Jiangyue Liu,et al. Gendelman. Research of Improved FP-Growth Algorithm in Association Rules Mining[J]. Scientific Programming,2015,2015.

[12] Mark Allen Weiss.数据结构与算法分析：Java语言描述 [M].冯舜玺,译.北京:机械工业出版社 ,2016.

【通联编辑：谢媛媛】

（上接第25页）

5 结论

本文重点以番茄病虫害为例从多方面具体介绍了Scrapy爬虫的方法。番茄病虫害的防治重点在于作物种植的有效监控和科学治理[8]。将爬虫技术合理应用于番茄病虫害防治的前期工作中，进一步推动番茄病虫害数据的信息化与规范化，以便为日后应用打下基础，从而也促进农业作物的数据化发展。与此同时，收集到的数据可应用于多方面。例如，利用图片特征结合名称标注，建立了能够完成病种判别的神经网络；利用文字信息归纳总结针对同种作物病害不同救治方法或不同作物同种救治方法；甚至可以考虑结合智能化机器，利用机器可以完成自动化监测、对农作物进行智能化保护、针对性变量喷药、病害远程诊断等，真正达到精准农业的目标。

参考文献：

[1] 吴军,倪萌,夏倩,等. 江苏无锡市设施番茄病虫害发生特点与绿色防控策略[J].中国园艺文摘,2017,33(10):195-197.

[2] 安子建. 基于Scrapy框架的网络爬虫实现与数据抓取分析[D].长春:吉林大学,2017.

[3] 孙小越,王超.基于Scrapy框架的电商数据分析平台[J].电脑知识与技术,2017,13(28):276-278.

[4] 鄂世嘉,林培裕,向阳.自动化构建的中文知识图谱系统[J].计算机应用,2016,36(4):992-996+1001.

[5] 李乔宇,尚明华,王富军,等.基于Scrapy的农业网络数据爬取[J].山东农业科学,2018,50(1):142-147.

[6] 肖庆都,屈亮亮,侯霞.基于Neo4j图数据库的课程体系知识图谱系统设计与实现[J].电脑知识与技术,2017,13(36):130-132.

[7] 杨君,陈春玲,余瀚.基于Scrapy技术的数据采集系统的设计与实现[J].计算机技术与发展,2018,28(10):177-181.

[8] 余世英. 病虫害防治技术在番茄种植过程中的应用分析 [J].农家科技,2016(11):105.

【通联编辑：谢媛媛】