

# 基于 PFP-Growth 算法的海量频繁项集挖掘

江雨燕,李 平

(安徽工业大学 管理科学与工程学院,安徽 马鞍山 243032)

**摘 要:**随着互联网技术的发展,网络数据变得越来越巨大,如何从中挖掘有效信息成为人们研究的重点。近年来频繁项集挖掘由于其在关联规则挖掘、相关挖掘等任务中的相关重要作用,越来越受到人们的重视。文中针对分布式计算环境下频繁项集挖掘算法的研究,对 PFP-Growth 算法进行了改进,通过 MapReduce 编程模型对改进的 PFP-Growth 算法进行了实现和应用,使用户可以从海量数据中高效地获得所有需要的频繁项集。实验结果表明算法在针对海量数据时具有较高的效率和伸缩性。

**关键词:**频繁项集;海量数据;PFP-Growth

中图分类号:TP31

文献标识码:A

文章编号:1673-629X(2013)09-0063-03

doi:10.3969/j.issn.1673-629X.2013.09.016

## Mining Massive Frequent Items Based on PFP-Growth Algorithm

JIANG Yu-yan, LI Ping

(College of Management Science and Engineering, Anhui University of Technology, Maanshan 243032, China)

**Abstract:** As the development of Internet, the data on it becomes more massive. How to mine useful information from the Internet is the key of study. In recent years, frequent item mining which plays an important role in associations rule mining and correlations mining becomes popular among researchers. By the study of mining frequent itemsets based on cloud computing, the PFP-Growth algorithm is improved. Run the algorithm under the MapReduce model which allows users to obtain all required frequent itemsets efficiently from massive data, the results of experiment shows the algorithm has good efficiency and flexibility.

**Key words:** frequent itemset; massive data; PFP-Growth

## 0 引 言

关联规则由 R. Agrawal<sup>[1]</sup>等人提出,并且相关技术在近几年越来越受到人们的重视,现已经发展成为数据挖掘的重要内容之一。作为挖掘关联规则的一项重要任务,频繁项集的提取是关联规则挖掘算法性能的主要影响因素。R. Agrawal 和 R. Srikant 于 1994 年提出的 Apriori<sup>[2]</sup>算法,是一种通过迭代搜索的方式产生频繁项集的方法。在 2000 年 J. Han 等人提出了 FP-Growth<sup>[3]</sup>算法,它是一种利用频繁模式构建属性结构的算法。它不仅在数据压缩方面要远远优于其他算法,而且减少了多次扫描数据集所带不必要的 I/O 传输,从而有效提高了算法的运行效率。随后出现了大量关于 FP-Growth 算法的改进算法<sup>[4-6]</sup>。

随着网络数据的急剧增长,传统频繁项集的算法

已经不能满足人们的需求。近年来许多并行挖掘频繁项集的方法先后被提出,使得可以在面对海量数据的情况下,有效地挖掘频繁项集。R. Agrawal 等人提出了基于 Apriori 算法的并行算法 Count Distribution<sup>[7]</sup>等算法,但由于基于 Apriori 并行算法需要大量的 I/O 传输,因此这些算法都不能很好地满足挖掘海量频繁项集的需要。Haoyuan Li<sup>[8]</sup>等人提出了基于 FP-Growth 算法的分布式 FP-Growth 算法(PFP-Growth)。利用分布式编程框架 MapReduce,实现了 FP-Growth 算法的并行化。Le Zhou<sup>[9]</sup>通过改进 PFP-Growth 算法提出了 Balanced Parallel FP-Growth 算法,有效提高了 PFP-Growth 算法的运行速度。

但在 PFP-Growth 算法的运行过程中需要用户手动地设置支持度  $s$ ,但用户并不能确定这个支持度,这

收稿日期:2012-11-22

修回日期:2013-02-20

网络出版时间:2013-05-09

基金项目:安徽高校省级自然科学基金项目(kj2011z039);安徽工业大学硕士研究生导师创新基金项目(D2011024)

作者简介:江雨燕(1966-),女,副教授,硕士生导师,研究方向为计算机集成、数据挖掘;李 平(1987-),男,硕士研究生,研究方向为机器学习、数据挖掘。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20130509.1057.013.html>

样可能造成重要信息丢失,从而不能挖掘出全部的频繁模式<sup>[10]</sup>。

文中通过改进 PFP-Growth 算法使用户能够在不需要具体设置的支持度的情况下,完整地挖掘出有效的频繁模式。

## 1 FP-Growth 算法

### 1.1 FP-Tree 的结构

FP-Tree 是一种输入数据的压缩表示,它通过逐个读入事务,并把每个事务映像到 FP-Tree 中的一条路径来构造<sup>[11]</sup>。路径相互重叠的越多,使用 FP-Tree 结构获得的压缩效果越好。如果 FP-Tree 足够小,能够存放在内存中,就可直接从这个内存中的结构提取频繁项集,而不必重复地扫描存放在硬盘上的数据。

图 1 显示了一个由包含 10 个事务和 5 个项的数据集产生的 FP-Tree。

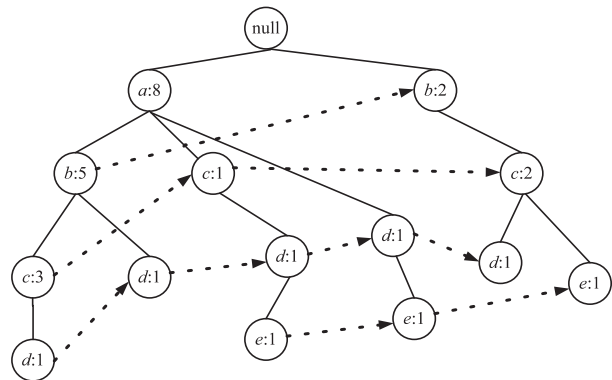


图 1 事务数据集 FP-Tree 表示图

### 1.2 由 FP-Tree 中挖掘频繁项集

FP 增长算法是一种以自底向上方式探索树,由 FP-Tree 产生频繁项集的算法。根据图 1 所示的树,算法首先查找以 *e* 结尾的频繁项集,接下来依次是 *d*, *c*, *b*, 最后是 *a*。这种用于发现以某一个特定项结尾的频繁项集的自底向上策略等价与基于后缀的方法。由于每一个事务都映像到 FP-Tree 中的一条路径,因而通过仅考虑包含特定节点的路径,就可以发现以 *e* 结尾的频繁项集。

### 1.3 PFP-Growth 算法

PFP-Growth 算法是一种基于 FP-Growth 算法的并行挖掘频繁项集的高效算法。在面对海量数据时, PFP-Growth 算法具有较高的准确性和伸缩性。PFP-Growth 算法具体过程如下:

(1) 分裂数据:把 DB 分裂成若干部分,并且把它们存储在不同的计算机 P 上。这被称为分片,每一个部分成为一个分片。

(2) 并行计算:运行一个 MapReduce 过程来计算在事务数据库 DB 中出现的所有项的支持度。每一个

mapper 的输入为一个 DB 的分片。这一步挖掘出每一个项的名称,这在大型数据库中经常是不知道的,结果被存储在 F-list 中。

(3) 把所有的项分组:把所有 F-list 中的项分成 *Q* 组。组的列表被称为 G-list,其中每一个组被分配一个唯一的 id(gid)。

(4) 并行运行 FP-Growth 算法:这是并行 FP-Growth 算法的主要步骤。其中 Mapper-生成以每一组对应的事务:每一个 Mapper 处理一个在第一步中生成的 DB 分片。在它一个一个地处理分片中的事务之前,首先读取 G-list。它输出若干个键值对,其中键为 gid 对应的值为对应的分组中的事务。Reducer-在每一个分片上运行 FP-Growth 算法。对于每一个分片,Reducer 构建一个本地的 FP-Tree 并且构建条件 FP-Tree 子树,在这个过程中,它将输出发现的频繁模式。

(5) 聚集融合:在这一步通过一个 MapReduce 过程融合在第(4)步中的输出,最终输出结果。

在 PFP-Growth 算法的运行过程中需要用户手动地设置支持度 *s*,但用户并不能确定这个支持度,这样可能造成重要信息丢失<sup>[12]</sup>,从而不能挖掘出全部的频繁模式。

## 2 PFP-Growth 算法改进策略

文中通过改进 PFP-Growth 算法使用户能够在不需要具体设置的支持度的情况下,完整地挖掘出有效的频繁模式。

### 2.1 并行计数过程

通过 Mapper 将输入的每条数据分解为单一的项,在 Reducer 中对每一项进行计数,将各项的名称及相应的频度 *f* 放入 F-List 中。在这个过程中仅对相同的项进行计数,而不考虑将频度较低的项删除,这样就保证 F-List 中包含所有项的频度信息。

### 2.2 分组过程

将 F-List 中的项进行分组,以供下一步各个节点分别处理不同分组的频繁项。这样不同分组的项分别由不同的节点处理,不同节点之间不会有信息的交换,在保证带宽的情况下实现 FP-Growth 算法的并行处理。假设每一项分为一组,即<第一组, *a*>、<第二组, *b*>、<第三组, *c*>、<第四组, *d*>、<第五组, *e*>,并且给每一组一个唯一的标志 gid。

### 2.3 事务数据分组

该过程通过一组 MapReduce 处理所有数据。Map 过程首先将每一条事务数据读入内存,根据并行计数过程中产生的频繁项的频度计数按照从大到小的顺序对事务项进行排序,输出到 Reducer。Reducer 根据分组过程产生的数据,按照支持度最大的项对事务进行

分组。整个处理过程如下。

```
Mapper(key, value = 事务数据 t)
提取 t 中的所有项保存到数组 a 中
for j = (t 包含项的数目 - 1) to 0 do
Call
Output(对应项 a[j] 的分组 gid; a[0] + a[1] + ... + a[j]);
End
Reducer 处理过程:
Reducer(key = gid, value = 对应 gid 组中的事务数据 DB
(gid))
创建用于保存一组事务的 FP-Tree->tree
foreach 事务数据 Ti in DB(gid) do
Call insert(Ti, tree) 将 Ti 保存到 tree 中;
End
CallOutput(gid; tree)
```

2.4 运行 FP-Tree 算法

该过程主要是运行 FP-Growth 算法从事务中找到频繁模式,并输出。

Map 过程把上一步输出的事务数据作为输入,其中 key 为 gid,而 value 为事务的内容。通过 MapReduce 的框架的自动处理机制,相同分组的事务将被分配到同一个 Reducer 上,进行后续的处理。

Reducer 过程在每一台计算机上运行本地的 FP-Growth 算法对相同分组的数据进行处理,产生频繁模式及相应的支持度。在这个过程中将 FP-Growth 算法进行了改写,在进行频繁模式挖掘时,没有频繁模式的支持度与最小支持度进行比较的过程,从而产生了所有存在的频繁模式。

2.5 融合过程

该过程主要把上一步产生的频繁模式输出作为输入,输出每个项所对应的频繁模式。整个过程如下。

```
Mapper(key, value = 频繁模式)
foreach 频繁项 i in 频繁模式 do
CallOutput(i; 频繁模式);
End
Reducer(key = i, value = 包含 i 的频繁模式 Vi)
foreach 模式 v in Vi do
CallOutput(i, v);
end
```

3 试验结果与性能分析

为了验证算法的有效性和实用性,使用 UCI 数据集对改进算法进行了测试,测试结果如表 1 所示。

表 1 UCI 数据下算法运行时间比较

频繁项集个数	运行时间/s
1 894 855	167.45
3 495 867	220.23
4 000 894	246.36
18 000 385	539.12

改进算法的精确度与原算法精确度比较如图 2 所示。可看出随着频繁项集的个数不断增大,运行时间与频繁项集的个数并不呈线性关系。证明算法在面对大量的频繁项集时具有较好的伸缩性和稳定性。

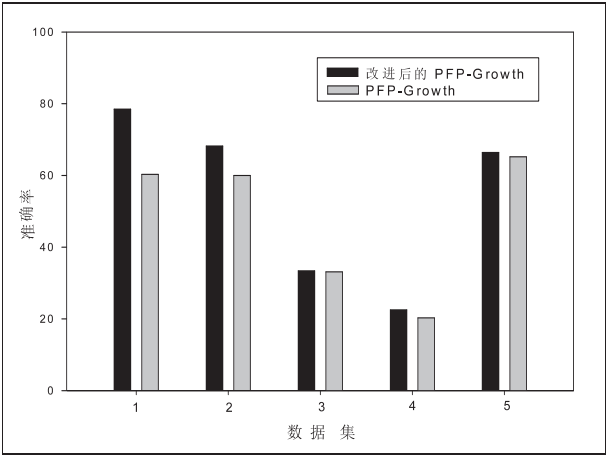


图 2 改进算法与原算法的精确度比较

改进算法与原 PFP-Growth 算法运行时间比较如图 3 所示。分别使用 UCI 的 Simulate 和 Chess 数据集对算法进行了测试,其中 Simulate 是一个比较稀疏的数据集,而 Chess 是一个稠密的数据集。算法首先在不同数据集下运行,然后计算出算法运行时间的平均时间。

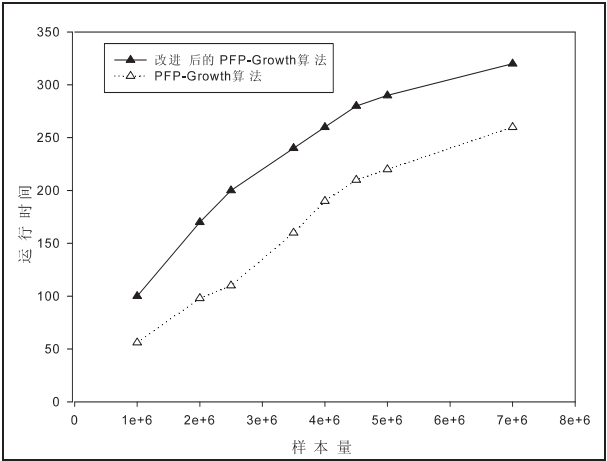


图 3 算法在不同数据集下的运行效率

改进算法的运行时间略高于原 PFP-Growth 算法,但频繁项集的准确率却明显高于原算法。同时改进算法在处理稀疏数据时具有明显优于 PFP-Growth 算法的性能。

4 结束语

文中针对分布式环境下频繁项集挖掘算法 PFP-Growth 进行了研究,并改进了 PFP-Growth 算法,通过 MapReduce 编程模型对改进的 PFP-Growth 算法进行实现和应用,结果表明研究的思想是可行的,研究结果

(下转第 198 页)

和描述符申请的内存空间,如果物理起始地址不能被 16 整除,需要进行对齐处理,否则 DMA 传输的数据会出错。

## 5 结束语

文中针对高速数据加密卡这一具体应用,对基于 PCI Express 的 SG DMA 高速数据传输系统的构成、工作流程以及驱动程序开发进行了介绍,利用现有的 Altera 公司的开发板作为硬件平台,在其上调试成功。基于该思想设计的高速数据加密卡采用 FPGA 的 PCIE 硬核,与采用专门芯片相比,可以更有效地发挥 PCIE 的高带宽优势,并且可以灵活配置,从而从整体上提高系统性能。采用 WinDriver 驱动开发平台,大大降低了 Windows 系统下程序开发的难度,缩短了程序开发与调试的时间。采用 DMA 方式提高了数据传送的能力,可以满足高速数据传输的可靠性,适用于大量数据的高速传输。

### 参考文献:

- [1] 马 萍,唐卫华,李绪志. 基于 PCI Express 总线高速数采卡的设计与实现[J]. 微计算机信息,2008,24(9-1):116-118.
- [2] 潘玉霞,马游春,熊继军. 基于 PCI Express 总线的高速数据传输卡设计与实现[J]. 电子技术应用,2010,36(8):92-95.
- [3] 刘秀萍. PCI 总线高速数据传输技术实现[J]. 电脑知识与技术,2009,5(9):2339-2340.

(上接第 65 页)

对云计算环境下频繁项集的挖掘是有效可扩展的。

### 参考文献:

- [1] Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large database [C]//Proc. of 1993 ACM SIGMOD Conf. on Management of Data. Washington DC:ACM Press,1993:207-216.
- [2] Agrawal R, Srikant R. Fast Algorithms for Mining Association Rules [C]//Proc. of the 20th Int'l Conference on Very Large Databases. Santiago, Chile: [s. n. ], 1994.
- [3] Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation [C]//Proc. of 2000 ACM SIGMOD Int'l Conf. on Management of Data. Dallas, TX, New York: ACM Press,2000:1-12.
- [4] 颜跃进,李舟军,陈火旺,等. 基于 FP-Tree 有效挖掘最大频繁项集[J]. 软件学报,2005,16(2):215-222.
- [5] 杨 云. FP-Growth 算法的改进[J]. 计算机工程与设计,2010,31(7):1506-1509.
- [6] 张玉芳,熊忠阳,彭 燕,等. 基于 FP-Tree 含正负项目的

- [4] Li B, Peng Y, Liu D T. A High Speed DMA Transaction Method for PCI Express Devices [J]. Journal of Electronic Science and Technology of China, 2009, 7(4):94-98.
- [5] PCI Express Base Specification Revision 1.1 [S]. PCI-SIG, 2005.
- [6] 项春雷. 基于 FPGA 的 PCI-E 接口设计 [D]. 郑州:信息工程大学,2011.
- [7] 王嘉良,赵曙光. 用 FPGA 实现 PCI-E 接口和 DMA 控制器设计 [J]. 计算机技术与发展,2011,21(6):181-184.
- [8] Arria II GX FPGA Development Kit User Guide [M]. [s. l.]: ALTERA Corporation, 2011.
- [9] PCI Express High Performance Reference Design [M]. [s. l.]: ALTERA Corporation, 2010.
- [10] IP Compiler for PCI Express User Guide [M]. [s. l.]: ALTERA Corporation, 2011.
- [11] Arria II GX FPGA Development Board Reference Manual [M]. [s. l.]: ALTERA Corporation, 2011.
- [12] 刘 波,库锡树,孙兆林. 基于 PCIE 总线协议的数据采集设备驱动程序实现 [J]. 工业控制计算机,2007,20(7):28-29.
- [13] 田 泽,刘 娟,王绮卉. 基于 WDM 的 PCIE 驱动设计和实现 [J]. 软件导刊,2010,9(4):9-10.
- [14] WinDriver PCI/ISA/CardBus User's Manual Version 10.10 [M]. [s. l.]: Jungo Ltd, 2009.
- [15] 刘晓光. 基于 PCI-E 接口数据采集系统软件设计与实现 [D]. 武汉:华中科技大学,2011.
- [16] 沈 辉,张 萍. FPGA 在 PCI Express 总线接口中的应用 [J]. 现代电子技术,2010(14):109-111.

频繁项集挖掘算法 [J]. 模式识别与人工智能,2008,21(2):246-253.

- [7] Agrawal R, Shafer J. Parallel mining of association rules [J]. IEEE Trans. on Knowledge and Data Engineering, 1996, 8(6):962-969.
- [8] Li Haoyuan, Wang Yi, Zhang Dong, et al. PFP: Parallel FP-Growth for Query Recommendation [C]//Proceedings of the 2008 ACM Conference on Recommender Systems. Lusanne, Switzerland: [s. n. ], 2008:125-137.
- [9] Zhou L, Zhong Z, Chang J, et al. Balanced Parallel FP-Growth with Map-Reduce [C]//Proceedings of 2010 IEEE Youth Conference on Information Computing and Telecommunications (YC-ICT). Beijing: [s. n. ], 2010:243-246.
- [10] 宋 威,刘文博,李晋宏. 基于动态裁剪频繁模式树的频繁项集并发挖掘算法 [J]. 山东大学学报:工学版,2011,41(4):49-55.
- [11] 邱 勇,兰永杰. 高效 FP-TREE 创建算法 [J]. 计算机科学,2004,31(10):98-100.
- [12] 谭峻松,首照宇. 一种分布式环境下动态挖掘频繁闭项集算法 [J]. 大众科技,2010(9):38-40.

作者：[江雨燕](#)，[李平](#)，[JIANG Yu-yan](#)，[LI Ping](#)  
作者单位：[安徽工业大学 管理科学与工程学院, 安徽 马鞍山, 243032](#)  
刊名：[计算机技术与发展](#)

ISTIC

英文刊名：[Computer Technology and Development](#)

年，卷(期)：2013(9)

参考文献(12条)

1. [Agrawal R](#), [Imielinski T](#), [Swami A](#) [Mining association rules between sets of items in large database](#) 1993
2. [Agrawal R](#), [Srikant R](#) [Fast Algorithms for Mining Association Rules](#) 1994
3. [Han J](#), [Pei J](#), [Yin Y](#) [Mining frequent patterns without candi-date generation](#) 2000
4. [颜跃进](#), [李舟军](#), [陈火旺](#) [基于FP-Tree有效挖掘最大频繁项集](#)[期刊论文]-[软件学报](#) 2005(02)
5. [杨云](#) [FP-Growth算法的改进](#)[期刊论文]-[计算机工程与设计](#) 2010(07)
6. [张玉芳](#), [熊忠阳](#), [彭燕](#) [基于FP-Tree含正负项目的频繁项集挖掘算法](#)[期刊论文]-[模式识别与人工智能](#) 2008(02)
7. [Agrawal R](#), [Shafer J](#) [Parallel mining of association rules](#)[外文期刊] 1996(06)
8. [Li Haoyuan](#), [Wang Yi](#), [Zhang Dong](#) [PFP:Parallel FP-Growth for Query Recommendation](#) 2008
9. [Zhou L](#), [Zhong Z](#), [Chang J](#) [Balanced Parallel FP-Growth with Map-Reduce](#) 2010
10. [宋威](#), [刘文博](#), [李晋宏](#) [基于动态裁剪频繁模式树的频繁项集并发挖掘算法](#)[期刊论文]-[山东大学学报\(工学版\)](#) 2011(04)
11. [邱勇](#), [兰永杰](#) [高效FP-TREE创建算法](#)[期刊论文]-[计算机科学](#) 2004(10)
12. [谭峻松](#), [首照宇](#) [一种分布式环境下动态挖掘频繁闭项集算法](#)[期刊论文]-[大众科技](#) 2010(09)

本文链接：[http://d.wanfangdata.com.cn/Periodical\\_wjz201309016.aspx](http://d.wanfangdata.com.cn/Periodical_wjz201309016.aspx)