第 37 卷 第 14 期
Vol.37 No.14

计 算 机 工 程
Computer Engineering

2011 年 7 月
July 2011

• 软件技术与数据库 •　　文章编号：1000—3428(2011)14—0059—03　　文献标识码：A　　中图分类号：TP311

基于 GPU 的闭合频繁项集挖掘方法

李海峰

(中央财经大学信息学院，北京 100081)

摘　要：提出一种采用图形处理器挖掘闭合频繁项集的方法，用二进制数据表示项集，利用单指令多数据的体系结构实现并行计算，结合项集索引树，可以提高项集支持度计算和项集查找的速度。在 2 种数据集上的实验结果表明，该方法能够用更少的空间保存频繁项集的全部信息，并减少挖掘时间。

关键词：图形处理器；频繁项集；闭合频繁项集；索引树；数据挖掘

Closed Frequent Itemset Mining Method Based on Graphic Processing Unit

LI Hai-feng

(School of Information, Central University of Finance and Economics, Beijing 100081, China)

【Abstract】This paper considers a problem that to the best of the knowledge has not been addressed, namely, how to use Graphic Processing Unit(GPU) for mining closed frequent itemsets. The method employs a single-instruction-multiple-data architecture to accelerate the mining speed using a bitmap data representation of frequent itemsets, a further memory-based index tree is used to make search faster. Experimental results show that the algorithm can store all the information of frequent itemsets using less space, and achieve better performance in running time.

【Key words】Graphic Processing Unit(GPU); frequent itemset; closed frequent itemset; index tree; data mining

DOI: 10.3969/j.issn.1000-3428.2011.14.018

1　概述

频繁项集挖掘是一种重要的数据挖掘类型。传统频繁项集挖掘方法主要考虑的是在静态事务数据库上的挖掘问题，事务可以被保存到硬盘上实现多次扫描。具体算法分为 3 种类型：基于读的算法，基于写的算法和基于指针的算法。其中，基于读的算法主要通过迭代的数据读操作来完成，基于写的算法将源数据进行再加工，建立压缩的数据格式，然后再根据新建的数据格式进行数据挖掘；而基于指针的算法以基于写的算法为基础，将建立的压缩数据格式中的某些数据直接用指针代替，减少数据存储的代价。

文献[1]对频繁项集挖掘的算法进行了综述，并讨论了今后的研究方向。文献[2]提出了闭合频繁项集的概念和最早的闭合项集挖掘算法 CLOSE，在层次模型下通过事务之间的交操作获得结果。文献[3]提出一种基于 DSFCI_tree 的数据结构，对动态变化的数据进行更新和剪枝，实现对闭合项集的挖掘。文献[4]提出了 LCM 算法在实现级别的改进版本，对数据库中的数据进行预处理，将支持度过小或过大的项删除，进而合并类似的事务，从而能够高性能地处理高阈值下的密集型数据库。

本文介绍了频繁项集和闭合项集的相关知识及 GPU 的属性，描述算法的数据结构和实现方法，给出了算法的验证结果。

本文基于图形处理器(Graphics Processing Unit,GPU)的闭合频繁项集挖掘的实现方法主要有 3 个特点：

(1)采用垂直表达方式将项集用二进制数据进行表示，能够充分利用 GPU 并行计算的特点，提高项集操作和支持度计算的速度。

(2)基于索引树提出了一种并行的闭合频繁项集挖掘的方法，可以提高项集查找的速度，降低冗余计算。

(3)通过在 2 种数据集上的实验，与最新的闭合频繁项集挖掘算法进行了比较，验证了本文算法的有效性。

2　频繁项集与闭合频繁项集

2.1　频繁项集

令 $\Gamma=\{i_1,i_2,\mathrm{L}\ ,i_n\}$ 表示一系列不同项目的集合，用 $|\Gamma|=n$ 表示 Γ 的大小，则称 Γ 的一个子集 X 为一个项集。假定 X 的大小为 k，则称 X 为 k 项集，其精简表示方式为 $x_1,x_2,\mathrm{L}\ ,x_k$。

在一个数据库 $D=\{T_1,T_2,\mathrm{L}\ ,T_v\}$ 中，每个事务都可以看作是一个包含 id 的项集。对于一个项集 X 来说，其绝对支持度是 D 中包含 X 的事务的数量，表示为 $\Lambda(X)=\{|\,T\,||\,T\in D\wedge X\subseteq T\}$，其相对支持度是绝对支持度与 D 的大小的比值，表示为 $\Lambda_r(X)=\dfrac{\Lambda(X)}{|D|}$。给定最小支持度 λ，如果 $\Lambda_r(X)>\lambda$，则称 X 是频繁项集。

2.2　闭合项集

给定 D 的事务集 T 和 Γ 的一个子集 X，闭合项集的概念可以用以下 2 个函数表示：

$$f(T)=\{i\in\Gamma\mid\forall t\in T,i\in t\}$$

$$g(X)=\{t\in D\mid\forall i\in X,i\in t\}$$

基金项目：中央财经大学“211 工程”三期基金资助项目

作者简介：李海峰(1979－)，男，讲师，主研方向：数据挖掘，商务智能管理

收稿日期：2010-12-20　**E-mail：**mydlhf@126.com

其中，$f(T)$函数计算属于事务集 T 的共有项目；$g(X)$计算包含项集 X 的事务集。根据这 2 个函数，可以得到闭合项集的公式化定义。

定义 1 如果项集 X 满足 $f(g(X))=X$，则称 X 为闭合项集。

从定义 1 可知，对于项集 X，如果包含 X 的其他项集的支持度与 X 的支持度不相等，则 X 为闭合项集。一个项集同时是频繁和闭合的，则称该项集为闭合频繁项集。从下例中可以看到，闭合频繁项集是频繁项集的一个子集。

给定表 1 的数据库和最小支持度 2，则频繁项集为{*a*, *b*, *c*, *d*, *e*, *ab*, *ac*, *ad*, *bc*, *bd*, *be*, *cd*, *ce*, *de*, *abc*, *abd*, *acd*, *bcd*, *cde*, *abcd*}，而闭合频繁项集为{*b*, *e*, *be*, *cd*, *bcd*, *cde*, *abcd*}。从该例中可以看到，5 个事务所包含的频繁项集的数量为 20 个，其中存在大量重复的数据，而闭合频繁项集的数量只有 7 个，压缩比例超过了一倍以上。

表 1 数据库的项集

id	项集				
1	*a*	*b*	*c*	*d*	*e*
2	*a*	*b*	*c*	*d*	
3		*b*	*c*	*d*	
4		*b*			*e*
5			*c*	*d*	*e*

3 图形处理器计算

GPU 是一种大规模的多线程多内核的高带宽的处理器，这种处理器可以看作一种特殊的并行架构的硬件。最新生产的 GPU 具有上百个内核，可以同时执行上千个线程；另外，GPU 与显存之间的带宽是 CPU 与内存之间带宽的十倍以上。这些特点非常适合数据的高效并行计算。

早期的 GPU 用于游戏和图形软件的设计和应用，最近则用于提高非图形处理软件的运行速度。如采用 GPU 进行数据仓库的 CUBE 计算，提高数据排序的速度，或应用于数据流挖掘和 k 中心聚类。文献[5]首次实现了基于 GPU 进行频繁项集挖掘的方法，使用位图减少前缀汇总，简化了 GPU 编程，同时，利用内建的容器结构进行局部优化和连接存储优化，能够有效地减少内存存取的次数。从该文献的实验可以得出结论：适当的利用 GPU 进行计算能够有效提高频繁项集挖掘的计算效率。

4 CG 算法

本文提出一种高效压缩的数据结构用来保存项集，然后利用该数据结构实现了基于 GPU 的闭合频繁项集挖掘方法 CG。与文献[5]中的算法 TBI 相比，CG 算法在数据存储和计算方面具有如下优势：

(1)TBI 算法是以最基本的频繁项集挖掘方法进行改进的，而 CG 算法针对性能最优的闭合项集挖掘方法进行基于 GPU 的优化，能够在提高计算性能的同时减少维护冗余数据的空间代价；

(2)尽管 TBI 算法与 CG 算法都利用索引提高性能，但 TBI 算法采用的是 Trie，而 CG 算法采用索引树，实现效率更高；

(3)TBI 在使用 GPU 时只是将数据按照整数所占的位数进行并行处理，CG 则直接使用 GPU 的着色器，不但实现简单，而且能进一步降低项集支持度计算和查找的次数。

4.1 数据结构

项集表示采用了数据的垂直表示结构。在垂直表示结构中，每个项目 i 由一系列包含该项目的事务 *id* 组成。垂直表示结构可以用二进制位图表示：当一个事务包含项目 i 时，在事务 *id* 的二进制位图中用 1 表示，否则，用 0 表示。表 1 中的数据集是水平表示方法，其垂直表示方式和位图表示方式如表 2 所示。这样，可以通过项目位图的与操作计算项集的支持度，例如，*ae* 的位图可以表示为 *a* 的位图和 *e* 的位图的与计算，即 11000&10011=10000，因为 10000 只包含一个 1，所以 *ae* 的支持度为 1。

表 2 垂直表示方式

项目	事务 id	二进制位图
a	1 2	11000
b	1 2 3 4	11110
c	1 2 3 5	11101
d	1 2 3 5	11101
e	1 4 5	10011

图 1 为建立的基于内存的索引树，索引树中的每个结点用三元组 $<\theta,\Psi,\Lambda(\theta)>$ 表示：其中，θ 代表项集，Ψ 表示二进制位图，而 $\Lambda(\theta)$ 是项集 θ 的支持度。在该图中，采用虚线框表示非频繁项集，采用实线框表示闭合频繁项集。通过索引树，能够在 GPU 计算中提高数据查找的速度。

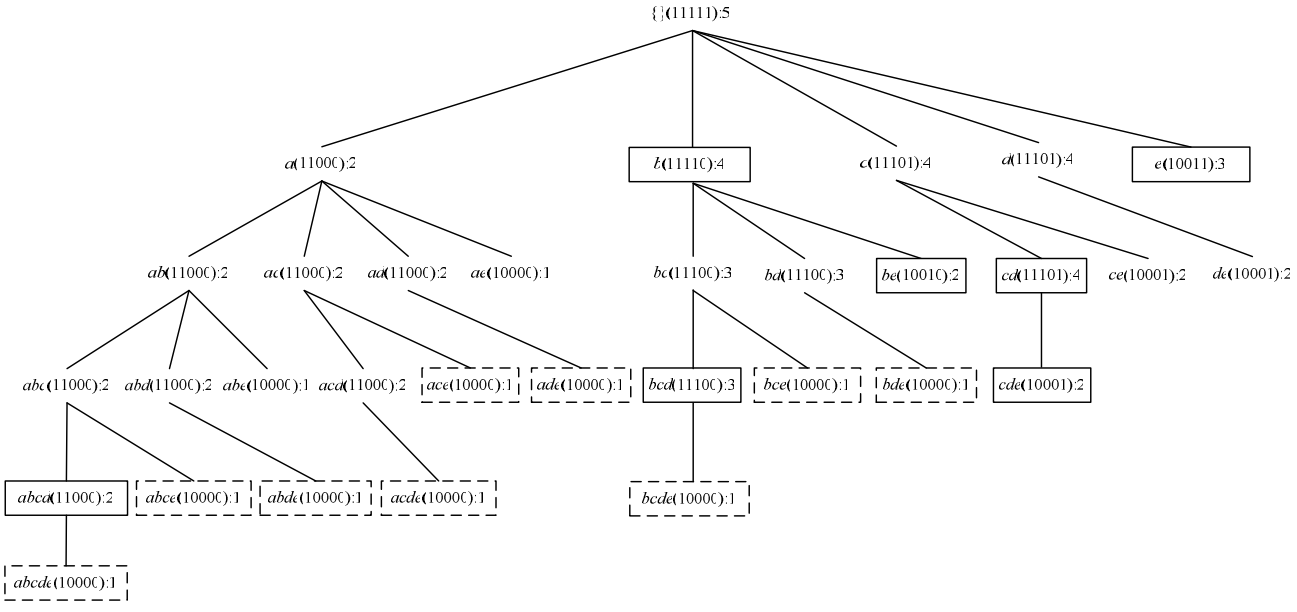


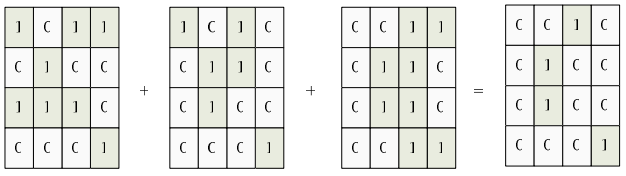
图 1 索引树结构

4.2　CG 算法实现

本文基于闭合频繁项集挖掘算法 LCM 第 3 版[4]实现了 GPU 计算。在 LCM 算法中，其项集支持度的计算采用 CPU 完成，由于支持度计算属于频繁操作，每次计算都需要进行大量的项集比较操作，该操作采用 CPU 计算会造成整个挖掘过程的瓶颈。另一方面，项集需要不断进行查找完成比较，从而完成非闭合项集的筛选，在没有索引的情况下也会降低运算速度。因此，充分利用 GPU 实现二进制位图中的与操作，以完成支持度计算，并辅以索引树提高项集查找速度，是解决问题的有效途径。

GPU 计算的 2 个主要特点是数据并行性和独立性，可以在不同的顶点和模型下进行相同的计算。因此，可以采用 GPU 的着色器完成位图计算：将每一个二进制位图看成是一个逻辑上的二维数组，二维数组中每一维的大小由二进制位图的长度决定；然后用一个像素表示一个二进制位，并用一种颜色表示该二进制值。

如图 2 所示，每个二进制位图可看作是一个矩阵。将数据投影到显存上，可实现位图与操作的并行计算；同时，其支持度也能够通过像素计数的方式以很小的代价计算得到。假设一个数据集有 n 个事务，每个项集可以用 n 个二进制位表示，如果系统中用 m 个字节表示一个整数，那么一个项集可以用 $\lceil n/m \rceil$ 个整数表示。在这种表示方式下，如果用 CPU 完成 2 个项集的与操作需要 $\lceil n/m \rceil$ 次整数计算的时间；而在 GPU 上只需要一次整数计算的时间即可。另外，在 CPU 上计算项集的支持度也需要耗费大量的时间，即使在建立二进制位图索引的情况下，也需要至少 n 次查找，而在 GPU 上只需要一次查找即可完成。



1011010011100001&1010011001000001&0011011001100011=0010010001000001

图 2　GPU 中的二进制位图计算

5　实验

本文算法采用 C++实现，在 Windows XP 的 Visual Studio 2005 上编译执行，并与 LCM 算法进行了比较。运行的硬件环境为 Core2 DUO CPU 2.0 GHz、2 GB 内存和 NVIDIA GTX 220 显卡(1 GB 显存)。采用 2 种标准数据集 T40I10D100K 和 BMS-POS 作为挖掘对象，其中，T40I10D100K 是采用 IBM 的数据生成器生成的模拟数据，BMS-POS 是 KDDCUP 2000 使用的商业销售数据，其具体属性如表 3 所示。

表 3　数据集属性

数据集	事务数量	平均事务长度	最小事务长度	最大事务长度	项目数量
T40I10D100K	100 000	39.6	4	77	942
BMS-POS	515 597	6.5	1	164	1 657

一方面，由于采用了并行计算，解决了在挖掘中的计算瓶颈，因此能够有效降低运行时间；另一方面，在 CPU 上进行计算，二进制位图越大，计算量越大。因此，事务集越大，采用 GPU 方法实现的效率越高。如图 3 和图 4 所示，CG 算法在 2 个数据集上的运行时间都低于 LCM 算法。

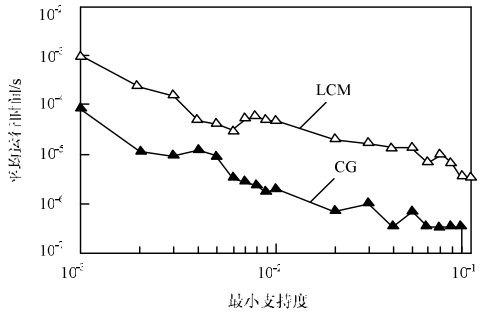


图 3　T40I10D100K 数据集上的运行时间比较

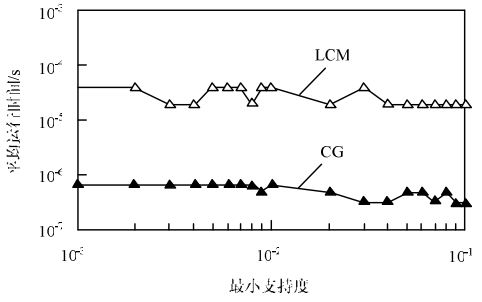


图 4　BMS-POS 数据集上的运行时间比较

另外，由于 BMS-POS 的事务数量是 T4I10D100K 的 5 倍，因此 CG 算法在 BMS-POS 数据集上效率更高：任取最小支持度为 0.01，CG 算法在 T40I10D100K 上的运行效率是 LCM 算法的 20 倍，而在 BMS-POS 上达到了 60 倍。这是因为事务数量越大会使得数据压缩比例越高，数据计算效率也越高。

6　结束语

本文考虑了在 GPU 上实现闭合频繁项集挖掘的问题，将项集用二进制位图表示，充分利用 GPU 的并行计算的体系结构，实现位图的与操作，进而对项集支持度进行计算，提高项集比较和查找的速度，将原来多次完成的计算一次完成。实验结果证明算法是可行有效的。

参考文献

[1] Han Jiawei, Cheng Hong, Xin Dong, et al. Frequent Pattern Mining: Current Status and Future Directions[J]. Data Mining and Knowledge Discovery, 2007, 14(1): 55-86.

[2] Pasquier N, Bastide Y, Taouil R, et al. Efficient Mining of Association Rules Using Closed Itemset Lattices[J]. Journal of Information Systems, 1999, 24(1): 1-12.

[3] 程转流, 胡学钢. 数据流中频繁闭合模式的挖掘[J]. 计算机工程, 2008, 34(16): 50-52.

[4] Uno T, Kiyomi M, Arimura H. LCM Ver. 3: Collaboration of Array, Bitmap and Prefix Tree for Frequent Itemset Mining[C]//Proc. of Workshop on Knowledge Discovery and Data Mining. Helsinki, Finland: [s. n.], 2005.

[5] Fang Wenbin, Lu Mian, Xiao Xiangye, et al. Frequent Itemset Mining on Graphics Processors[C]//Proc. of Workshop on Data Management on New Hardware. Rhode Island, USA: [s. n.], 2009.

编辑　张正兴