

一种改进的基于 MPI/OpenMP 混合编程关联规则算法

王 勇¹, 张 哲¹, 陶晓玲²

(1 桂林电子科技大学 计算机科学与工程学院, 广西 桂林 541004;

2 桂林电子科技大学 信息与通信学院, 广西 桂林 541004)

摘 要: 设计并实现了一种基于高性能计算集群的并行关联规则挖掘方法. 该算法采用分离的策略仅需对本地数据库进行一次访问, 能有效降低 I/O 开销, 同时利用 MPI/OpenMP 混合编程模式, 最大限度降低数据在网络上的通信开销. 在普通 PC 搭建的高性能计算集群的实验结果证明: 基于高性能计算集群的关联规则算法效率较高, 具有较好的加速比.

关键词: 并行关联规则; 高性能计算; 频繁项集增长算法; MPI/OpenMP

中图分类号: TP393

文献标识码: A

文章编号: 1000-7180(2014)11-0156-04

An Improved Association Rule Algorithm Based on MPI/OpenMP Hybrid Programming

WANG Yong¹, ZHANG Zhe¹, TAO Xiao-ling²

(1 Department of Computer Science and Engineering, Guilin University of Electronic Technology, Guilin 541004, China;

2 Department of Information and Communication, Guilin University of Electronic Technology, Guilin 541004, China)

Abstract: We design and implement a parallel association rules mining method based on high-performance computing clusters(HPC clusters). The algorithm adopts the separation strategy to simply visit a local database only once, thus, the inter-processor communication I/O overhead is reduced. What's more, that using MPI/OpenMP hybrid programming mode can minimize data communication overhead on the network. By using ordinary PC structures, HPC cluster experimental results verify that the proposed algorithm based on HPC offers higher efficiency and has a good speedup.

Key words: parallel association rules; HPC; FP-growth; MPI/OpenMP

1 引言

关联规则挖掘通过收集数据库中大量的事物记录进行分析, 查找海量数据中有趣的关联关系. 其中, 频繁模式(frequent pattern, FP)挖掘已经成为一种非常重要的组成部分^[1-4]. Apriori 算法^[5]是第一种“产生-测试”的频繁模式算法, 但它有两个主要限制: 需要频繁访问数据库和占用内存. 另一种是由 Han 等人^[5]提出的 FP-growth 算法, 通过使用一个高度紧凑的树形存储结构(称为 FP-tree), 仅需对数据库访问两次, 它对于挖掘不定长的频繁项集, 是有效的和可规模化的, 并且

比 Apriori 算法快一个数量级.

在文献[1-4]中 PFP-growth 算法的通信成本以及任务调度进行了优化改进, 但都没有彻底解决计算性能瓶颈问题^[6]. 同时, 随着网络应用的不断增加和网络速度的不断提高, 挖掘频繁项集的种类和数量也在不断增长, 而计算资源是有限的, 面对大样本数据集时, 其计算复杂度和通信成本也会成倍增长, 这些在很大程度上限制了 FP-growth 算法的应用^[7-8]. 因此, 本文提出了一种基于高性能计算集群 FP-growth 算法, 利用 MPI/OpenMP 混合编程模式, 最大限度的解决传统并行 FP-growth 算法的瓶颈问题. 实验表明, 该方法在保

收稿日期: 2014-01-03; 修回日期: 2014-02-28

基金项目: 国家自然科学基金项目(61163058); 广西自然科学基金项目(2011GXNSFB018076)

证数据挖掘结果正确性的同时,极大地提高了挖掘的效率,具有较好的加速比。

2 MPI/OpenMP 混合编程模式

MPI 是集群计算中广为流行的编程模式,但在很多情况下,完全采用 MPI 消息传递编程模式并不能在由多处理器构成的集群上取得理想的性能。OpenMP 是共享存储编程的实际工业标准, MPI 则能被用在集群的节点与节点之间消息的传递。因此,通过 MPI 与 OpenMP 混合编程模式,能够更好地实现分布式/共享内存层次结构化。这种混合编程模式提供节点内和节点间的两级并行,能充分利用共享存储模型和消息传递模型的优点,有效地改善系统的性能^[9]。MPI 与 OpenMP 优点对比如表 1 所示。

表 1 MPI 与 OpenMP 优点对比

MPI	OpenMP
允许静态任务分配	相对易于实现
显式并行,易于获得好的并行性能	更适合于共享内存架构
具有一系列优化的聚合通信的程序	允许运行时任务调度
可实现通信和计算重叠	粗细粒度均有效
没有数据分布问题	支持增量并行
通信自然的包含同步,降低了开销	

使用混合编程模式的模型结构图如图 1 所示。每个 MPI 进程中可以在 `#pragma omp parallel` 编译指导所标示的区域内产生线程级的并行,而在区域之外仍然是单线程。混合编程模型可以充分利用两种编程模式的优点, MPI 可以解决多处理器间的粗粒度通信,而 OpenMP 提供轻量级线程可以很好地解决每个多处理器计算机内部各处理器间的交互。大多数混合模式应用是一种层次模型, MPI 并行位于顶层, OpenMP 位于底层。比如处理一个二维数组可以先把它分割成节点个子数组,每个进程处理其中一个子数组,而子数组可以进一步被划分给若干个线程。这种模型很好地映射了多处理器计算机组成的集群体系结构, MPI 并行在节点间, OpenMP 并行在节点内部。也有部分应用是不符合这种层次模型的,比如说消息传递模型用于相对易实现的代码中,而共享内存并行用于消息传递模型难以实现的代码中,还有计算和通信的重叠问题等^[10]。

3 基于 MPI/OpenMP 混合编程关联规则算法改进

3.1 基于 MPI/OpenMP 混合编程关联规则算法总体框架

本文采用 MPI/OpenMP 混合编程思想,设计

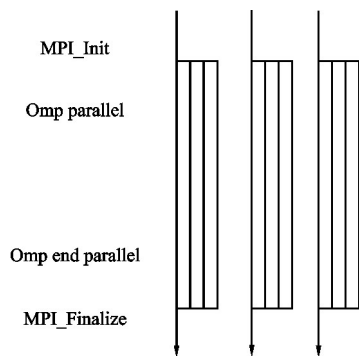


图 1 MPI/OpenMP 混合编程模型

并实现了改进的基于高性能计算集群的 FP-growth 算法(HPC FP-growth 算法,即 HFP 算法)。HFP 算法执行流程如图 2 所示。其算法遵循的设计原则如下:

- 尽可能减少节点之间的通信;
- 在必要的情况下,应该选择用计算换通信策略;
- 为避免数据竞争,采用不同进程写入不同的文件。

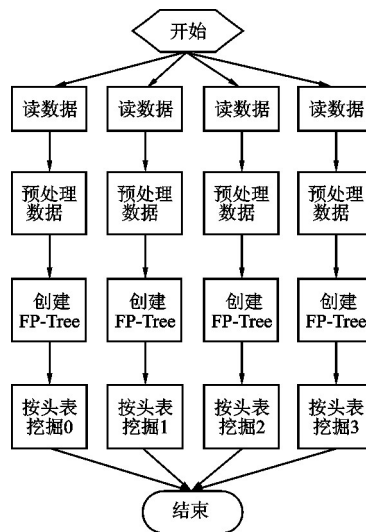


图 2 HPC 算法执行流程

HFP 算法其主要包括建立 Transaction 数据库、建立 FP-tree、频繁项挖掘以及频繁项写入磁盘这四个部分。具体步骤如下。

(1)并行建立 Transaction 数据库。主要完成对原始数据的读入, item 筛选, frequent 计算, 单个 transaction 排序, 对整个内存中的 transaction 数据库进行降序排列。使用 OpenMP 在节点内使用多核并行, 对串行算法进行改进。

(2)建立 FP-tree。使用前一步创建的数据库, 建

立局部 FP-tree.

(3)并行挖掘局部频繁项集.挖掘频繁项集的递归操作进行 MPI 并行化处理,然后将任务分配给各个进程来完成.

(4)频繁项写入磁盘.针对传统的串行程序中,使用类似全局变量的文件指针来统一在查找频繁项的过程中对磁盘文件执行写操作,本文用不同进程写入不同文件的方法来避免对一个文件的写操作的竞争,在计算结束后得到若干个不同进程写成的文件,最后,通过 bash 将几个产生的文件合并起来,得到最终的频繁项集.

HFP 算法执行的伪代码如下:

```
MPI_Init(...);
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
MPI_Comm_size(MPI_COMM_WORLD, &size);
ReadData();
OMP_Create_Transaction_Database(); // openmp im-
plementation
Create_FP_Tree();
Find_frequent_items(rank, size); // different process
different job.
Write_to_files(rank);
MPI_Finalize();
```

3.2 并行建立 Transaction 数据库

针对串行算法主要存在两种并行改进:(1)对 transaction 数据库中不符合支持度条件的项删除,(2)对 transaction 数据库中不同的行进行排序.这两个操作都是基于单行数据进行处理的,故不同行之间的处理是独立的.因此使用 OpenMP 在节点内使用多核并行,以此来避免进程并行带来的通信开销.其伪代码如下所示.

```
#pragma omp parallel ...
{ tid = omp_get_thread_num(); numt = omp_get_
num_threads();
  naver = tracts_num/numt; nfirst = tracts_num -
naver * numt + naver;
  if (tid == 0){nstart = 0; nend=nfirst;}else{
  nstart=nfirst+(tid-1) * naver;nend=nstart+naver}
  for (i = nstart; i < nend; i++){t = tracts[i];if
(marks){
    removing_marked_items(t);
  }
  If (t->size > sums[tid])Sums[tid] = t->size;
  }}
  for (I = 0; I< THREADS_NUM; i++){
  if(sums[i] > sum) Sum = sums[i]
  }
```

3.3 并行挖掘局部频繁项集

在 FP-tree 建立之后,数据结构中会产生一个头表(header),头表包含 Transaction 中出现的所有

的数据项,同时按项集支持度降序排列,由于传统的 MPI 对 FP-growth 算法中频繁项挖掘采用递归操作,新的频繁项集的挖掘依赖于前一项建立树中的资源,这样给共享存储并行带来难度.因此,本文用 MPI 进程级并行来处理频繁项集挖掘的递归操作,将任务分配到各个进程中.同时,通过递归调用 rec_cmplx 函数把每个进程所查找的频繁项写入到不同的文件中.其伪代码如下所示:

```
// MPI parallel
Functionaban_rec_cmplx(cst,&rd)
Jobaver = cst->cnt/mpisize;
Jobfirst = cst->cnt - (jobaver) * (mpisize-1);
If(mpirank == 0) // master process.
  {i= 0; z = jobfirst;}
Else
  {i= jobfirst + (mpirank-1) * jobaver; z = i+
jobaver;}
  For (i = 0; i<z; i++){
    h = cst->heads + i;
    ...
    rec_cmplx(..);
  }
End function
```

4 实验与结果分析

为了对算法的性能进行评估,使用 IBM 公司的人工模拟数据生成器(Quest Synthetic Data Generator)^[11]来产生规则关联数据,以及 UCI KDD Archive(connect、pumsb、accident 数据集)^[11-12].实验采用 Master-Slave 模式,共有六个节点,其中包括一个主控节点.每个节点操作系统采用 Ubuntu ubuntu-12.04.3,MPI 版本采用 mpich-3.0.4,采用千兆网卡.其高性能计算集群硬件配置情况如表 2 所示.

表 2 高性能计算集群硬件配备情况

	CPU/GHz	RAM/GB	Disk/GB
主控节点	E8400 204	2	320
计算节点	P8600 204	2	160

将上述算法基于 MPI 和 OpenMP 混合编程模式在高性能计算集群上编程实现,采用的测试数据集为挖掘频繁项集领域常用的测试数据集中较大的数据集 connect、pumsb 和 accidents 分别进行实验,其中数据集大小 connect<pumsb<accidents.如图 3 所示,高性能计算集群在处理这三个规模不断上升的数据集时,随着集群节点的增加所用的时间呈线性减小,因此可见在增加集群节点数量能显著提高高性能计算集群对于相同规模作业的处理能力,

也从侧面反映了基于高性能计算集群的改进算法具有良好的加速比。

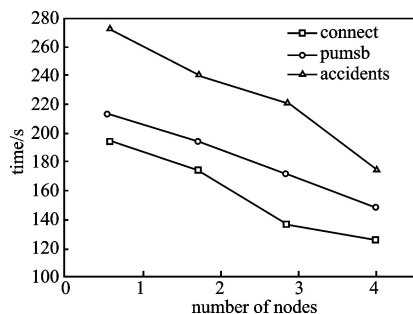


图3 不同数据集执行时间随着节点数量的变化情况

如图4所示,在 T20.15, N100K, D100K 数据集中 HFP 和串行 FP-growth 算法在支持度阈值从 0.01% 到 0.05% 执行时间的变化情况。可以看出虽然随着支持度阈值的增加,执行时间均呈减小的趋势发展,但 HFP 明显提高了串行算法的性能。这是因为串行 FP-growth 算法在执行大数据集时,由于反复的迭代,需要大量的 I/O 开销,占用大量内存,导致内存不足。而 HFP 算法基于 MPI/OpenMP 混合编程模式,通过 OpenMP 多线程并行处理代替程序的反复迭代,不但能减少传统并行算法对于计算结果消息传递过程中传输的时间消耗,也能有效的降低算法的 I/O 开销。同时,利用 MPI 进行进程间消息的并行传递,减少了建立 FP-tree 的等待时间。

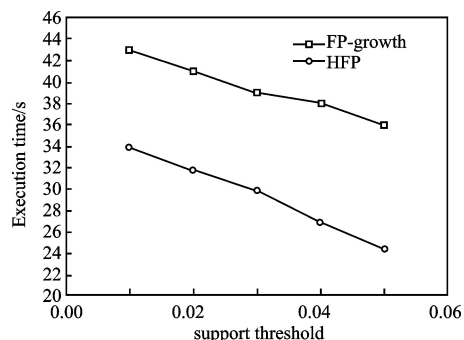


图4 在 T20.15, N100K, D100K 数据库中执行时间随着支持度阈值的变化情况

5 结束语

本文提出了一种基于 MPI/OpenMP 混合编程模式的改进关联规则算法,针对 FP-growth 算法的可并行部分,结合 MPI/OpenMP 混合编程模式对其进行改进,实验证明对于处理大数据集,较串行算法, HFP 算法能有效降低 I/O 开销。但是对于较小

数据集或者结点较少时,往往效果不是太理想。

参考文献:

- [1] Agrawal R, Srikant R. Fast algorithms for mining association rules[C]// Proc. 20th Int Conf Very Large Data Bases, VLDB. 1994, 1215: 487-499.
- [2] Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation[J]. ACM SIGMOD Record, 2000, 29(2): 1-12.
- [3] Javed A, Khokhar A. Frequent pattern mining on message passing multiprocessor systems[J]. Distributed and Parallel Databases, 2004, 16(3): 321-334.
- [4] Yu K M, Zhou J, Hsiao W C. Load balancing approach parallel algorithm for frequent pattern mining [M]. Berlin, Heidelberg: Springer, 2007: 623-631.
- [5] Zaiane O R, El-Hajj M, Lu P. Fast parallel association rule mining without candidacy generation[C]// Data Mining Proceedings IEEE International Conference on. [s. l.]: IEEE, 2001: 665-668.
- [6] Chen D, Lai C, Hu W, et al. Tree partition based parallel frequent pattern mining on shared memory systems[C]// Proceedings of the Parallel and Distributed Processing Symposium. Rhodes island, Greece.: IEEE, 2006: 8-18.
- [7] 彭硕, 吴昊. 基于 MDPI 的多维关联规则算法的研究[J]. 微电子学与计算机, 2011, 28(1): 78-82.
- [8] 刘芳. DF 关联规则及其挖掘算法的研究[J]. 微电子学与计算机, 2009 (10): 93-98.
- [9] 冯云, 周淑秋. MPI+OpenMP 混合并行编程模型应用研究[J]. 计算机系统应用, 2006(02): 86-89.
- [10] 王惠春. 基于 SMP 集群的 MPI+OpenMP 混合并行编程模型研究与应用[D]. 湘潭: 湘潭大学, 2008.
- [11] Agrawal R, Srikant R. Quest Synthetic Data Generator[DB/OL]. [2014-01-03] California: Almaden Research Center, San Jose. <http://www.almaden.ibm.com/cs/quest/syndata.html>.
- [12] UCI KDD Archive[DB/OL]. [2014-01-03] (2013-12-27). <http://archive.ics.uci.edu/ml/datasets/>.

作者简介:

王 勇 男, (1964-), 博士, 教授. 研究方向为计算机网络。

张 哲 男, (1988-), 硕士研究生. 研究方向为计算机网络。

陶晓玲 女, (1977-), 硕士, 副研究员. 研究方向为计算机网络。