大数据环境下关联规则并行分层挖掘算法研究

张忠林 田苗凤 刘宗成

(兰州交通大学电子与信息工程学院 兰州 730070)

摘 要 为满足大数据实时处理的需求,提出了一种基于划分的关联规则并行分层挖掘算法(Parallel Hierarchical Association Rule Mining,PHARM)。首先,将整个数据库 D 随机分割成若干个非重叠区域,并行挖掘出局部频繁项集;然后利用先验性质,连接局部频繁项集得全局候选项集;再次扫描 D 统计出每个候选项集的实际支持度,以确定全局频繁项集。最后,建模分析了该算法的高效性。

关键词 大数据,划分,关联规则,并行分层挖掘,高效性

中图法分类号 TP311.5 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.1.061

Parallel Hierarchical Association Rule Mining in Big Data Environment

ZHANG Zhong-lin TIAN Miao-feng LIU Zong-cheng

(School of Electronics and Information Engineering, Lanzhou Jiaotong University, Lanzhou 730070, China)

Abstract To deal with big data's demand of real-time processing, we proposed the parallel hierarchical association rule mining algorithm based on partitioning. First, the algorithm divides the transactions of D into n nonoverlapping partitions randomly, and all the local frequent itemsets mining is parallelized. Second, apriori property is utilized to collect frequent itemsets from all partitions and form the global candidate itemsets with respect to D. Then the actual support of each candidate is counted to determine the global frequent itemsets. At last, the algorithm's high efficiency was analyzed by modeling.

Keywords Big data, Partition, Association rule, Parallel hierarchical mining, High efficiency

目前,基于数据驱动的决策支持系统得到广泛认可,使得"大数据"[1,2]成为焦点。大数据处理需要满足极高的时效性,数据量大意味着计算开销大,数据多样性意味着算法可扩展性需要更强,二者制约着大数据处理技术的时效性[2],大数据关联分析[3]开始崭露头角。

为了解决数据挖掘效率低的问题,研究者对数据的收集、存储、管理、处理、分析、共享和可视化技术进行了一系列研究。其中,Hadoop 平台的 MapReduce[4,5]架构备受青睐,文献[4]中通过 MapReduce 把数据划分为很多块,启动多个 map 同时处理来实现并行计算;基于 Hadoop 实现的并行程序能处理节点失效,并能做到负载均衡,但文章对分批大小、如何处理并行挖掘结果及结果的完整性没有提及。文献[6]提出的高效的多层关联规则挖掘方法,通过映射缩小了事务数据库规模,并采用压缩的 AFOPT 结构,有效地节省了算法的 I/O 时间,提升了多层关联规则的挖掘效率;但利用先验知识进行层次聚类不利于新模式的发现。文献[7]于集群环境下在每个节点上执行 CLAP 算法,虽提高了挖掘模型的效率,但由于大数据价值密度低[8]的特点,容易造成模式挖掘不完整的问题。文献[9]采用两个最小支持度值,min_supL1 用于本地头表剪枝,min_supL2 用于本地 FI-tree 剪枝,由于 min_supL1 的取值和数据集的划分,这种粗略剪枝可能会造成很大误差。

基于划分的、减小数据库扫描次数的挖掘关联规则的 Apriori 改进算法有很多,本文采用相对支持度阈值进行本地和全局剪枝,不失挖掘结果的精确度;此外,由于本文提出的 PHARM 算法在并行的同时考虑分层性,使得在大数据环境下 PHARM 算法的高效性更加突出。

1 研究背景分析

电力公司目前已实现对输变电设备状态的实时监控,但监控人员对视频图像的分析能力有限,无法对设备状态变化做出实时响应,会影响设备检修、运维。采用大数据技术对视频图像数据进行智能分析,能为智能电网的经济、高效运行提供辅助决策。通过设备状态变化检测,可准确评估设备当前状态,挖掘设备潜在异常,预测设备未来状态趋势,提高运维人员定位故障设备的工作效率,丰富故障设备检测手段,进一步推动无人值守变电站建设,为输变电设备状态检修工作提供辅助决策,提高公司视频监控智能化水平。

1.1 关联规则挖掘

作为大数据的一种关键技术,关联规则挖掘[10,11]旨在从大量事务数据库中发现事务之间有趣的关联关系,以揭示隐藏其中的客户行为模式,产生能应用于多种决策支持系统的

到稿日期:2014-11-17 返修日期:2015-03-06 本文受国家自然科学基金项目(61163010),甘肃省自然科学基金(1308RJZA194)资助。

张忠林(1965-),男,博士,教授,CCF 会员,主要研究方向为智能信息处理、软件工程;田苗凤(1987-),女,硕士生,主要研究方向为数据挖掘;刘宗成(1988-),男,硕士生,主要研究方向为数据挖掘。

规则。一般而言，关联规则的挖掘过程[12]分为两步：

(1)找出所有的频繁项集。根据定义，这些项集的每一个频繁出现次数至少与预定义的最小支持度计数 *min_sup* 一样。

(2)由频繁项集产生强关联规则。根据定义，这些规则必须满足最小支持度和最小置信度。

由于第二步的开销远低于第一步，挖掘关联规则的总体性能由第一步决定，因此本文重点研究产生频繁项集的过程。

1.2 划分的概念

划分[11]作为 Apriori 算法[13,14]的一种变形，只需两次数据库扫描，就能挖掘频繁项集(见图 1)，它包含两个阶段。阶段 I，算法把 D 中的事务划分成 n 个非重叠的分区。如果 D 中事务的最小支持度阈值为 *min_sup*，则每个分区的最小支持度计数为 *min_sup*×该分区中的事务数。对每个分区，找出所有的局部频繁项集(即在该分区内的频繁项集)。局部频繁项集可能是也可能不是整个数据库 D 的频繁项集；然而，D 的任何频繁项集必须作为局部频繁项集至少出现在一个分区中。因此，所有局部频繁项集都是 D 的候选项集。来自所有分区的局部频繁项集作为 D 的全局候选项集。阶段 II，第二次扫描 D，评估每个候选的实际支持度，以确定全局频繁项集。

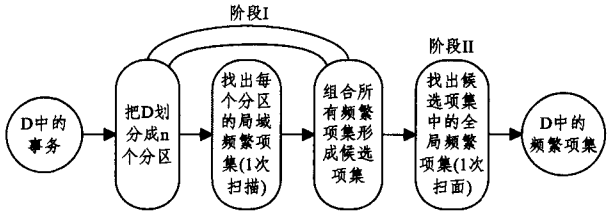

图 1 通过划分挖掘

2 基于划分的关联规则并行分层挖掘算法

2.1 基本思想

该算法同样只需两次数据库扫描，就能挖掘频繁项集，其基本思想及数据流向见图 2。预处理阶段：将整个事务数据库 D 随机分割成 n 个非重叠区域且尽量使各个分区大小相似。随机分割使得算法避免了依赖聚类或先验知识划分数据库的处理开销；分区非重叠保证了每个事务的同等重要性、被挖掘的机会均等，同时避免了重复挖掘的低效性，提高了挖掘结果的可靠性和挖掘效率；此外，所有分区挖掘所需时间由并行部分最费时的分区决定，每个分区大小相似，使得相同硬件条件下的分区挖掘时间近似、全局挖掘时间最短，提高挖掘效率。

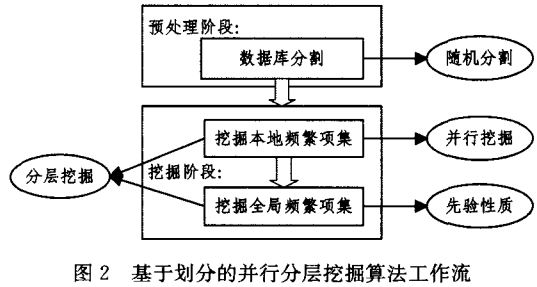

图 2 基于划分的并行分层挖掘算法工作流

挖掘阶段：扫描 D 的各个分区，并行挖掘出每个分区的局部频繁项集，局部频繁项集可能是也可能不是整个数据库 D 的频繁项集；然而，D 的任何频繁项集必须作为局部频繁项集至少出现在一个分区中。因此，所有局部频繁项集都是 D 的候选项集。来自所有分区的局部频繁项集作为 D 的全局候选项集。第二次扫描 D，评估每个候选项集的实际支持度，以确定全局频繁项集。其中挖掘的分层性体现在：D 的每个分区可看做一个小型的数据库继续分割成若干个更小的分区。

2.2 算法描述及其性能分析

2.2.1 PHARM 算法

输入：事务数据库 D；最小支持的阈值 min_sup；事务数据库 D 被划分成的分区数目 n

输出：L，D 中的频繁项集

方法：

(1)for(k=1;k<=n;k++){

(2)L_{Dk}=find_frequent_itemsets(D_k);//扫描 D_k，得到 D_k 的局部频繁项集 L_{Dk}

(3)} //end for

(4)C=$L_{D1} \bowtie L_{D2} \bowtie L_{D3} \bowtie \cdots\cdots \bowtie L_{Dn}$;//连接步：产生候选项

(5)扫描 D，统计每个候选项的实际支持度计数；

(6)删除实际支持度计数小于最小支持度 min_sup×|D|的候选项，得全局频繁项集 L;//剪枝步

(7)return L;

如上所述，PHARM 做了 3 个动作：局部挖掘、连接和全局剪枝。在进行局部挖掘时，每个分区 D_k 还可继续划分为若干个子分区 $D_{k1}, D_{k2}, \cdots, D_{km}$，体现了分层挖掘的概念；$D_1, D_2, \cdots, D_n$ 可并行执行，子分区 $D_{11}, D_{12}, \cdots, D_{1s}, D_{21}, D_{22}, \cdots, D_{2g}, \cdots, D_{k1}, D_{k2}, \cdots, D_{km}, \cdots, D_{n1}, D_{n2}, \cdots, D_{nd}$ 之间均可并行执行，体现了分层并行的思想。在连接部分，利用先验性质将 $L_{D1}, L_{D2}, \cdots, L_{Dn}$ 连接，产生候选项。在全局剪枝部分，通过扫描全局数据库，统计每个候选的实际支持度计数，将实际支持度计数小于最小支持度的候选剪掉。

2.2.2 算法演示

现将关联规则的并行分层挖掘算法 PHARM 应用于一个具体的例子。该例基于表 1 All-Electronics 的事务数据库 D。该数据库有 9 个事务，即 $|D|=9$。如下为使用 PHARM 算法发现 D 中频繁项集的过程。

表 1 All-Electronics 某分店的事务数据库

TID	商品 ID 列表	TID	商品 ID 列表	TID	商品 ID 列表
T100	I1,I2,I5	T400	I1,I2,I4	T700	I1,I3
T200	I2,I4	T500	I1,I3	T800	I1,I2,I3,I5
T300	I2,I3	T600	I2,I3	T900	I1,I2,I3

(1)将 D 随机分割成 3 个分区 $D1=\{T100, T700, T800\}$，$D2=\{T300, T400, T500\}$，$D3=\{T200, T600, T900\}$。

(2)假设最小支持度阈值为 30%，即 min_sup=30%(这里是指相对支持度)，则每个分区的实际支持度计数为0.9，分区 $D1$ 的局部频繁项集 $L_{D1}=\{\{I1\},\{I2\},\{I3\},\{I5\},\{I1,I2\},\{I1,I3\},\{I1,I5\},\{I2,I3\},\{I2,I5\},\{I3,I5\},\{I1,I2,I3\},\{I1,I2,I5\},\{I1,I3,I5\},\{I2,I3,I5\},\{I1,I2,I3,I5\}\}$。同理，$L_{D2}=\{\{I1\},\{I2\},\{I3\},\{I4\},\{I1,I2\},\{I1,I3\},\{I1,I4\},\{I2,I3\},\{I2,I4\},\{I1,I2,I4\}\}$，$L_{D3}=\{\{I1\},\{I2\},\{I3\},\{I4\},\{I1,I2\},\{I2,I3\},\{I2,I4\},\{I1,I2,I3\}\}$。所以 D 的全局候选项集 $C=\{\{I1\},\{I2\},\{I3\},\{I4\},\{I5\},\{I1,I2\},\{I1,$

$I3\},\{I1,I4\},\{I1,I5\},\{I2,I3\},\{I2,I4\},\{I2,I5\},\{I3,I5\},\{I1,I2,I3\},\{I1,I2,I4\},\{I1,I2,I5\},\{I1,I3,I5\},\{I2,I3,I5\},\{I1,I2,I3,I5\}\}$。

(3)由 min_sup 和$|D|$得全局支持度计数为 2.7。扫描 D,得全局频繁项集 $L=\{\{I1\},\{I2\},\{I3\},\{I1,I2\},\{I1,I3\},\{I2,I3\}\}$。

从挖掘过程可以看出,若实例中的最小支持度阈值取为40%,分区的挖掘效率会更高。令 $value=min_sup\times$该分区中的事务数,有如下推论:当 $value<1$ 时,PHARM 算法的挖掘效率比常规挖掘算法效率更低;当 $value>1$ 时,随着 $value$ 值的增加 PHARM 算法的挖掘速率会先升后降;当 $value$ 达到最大,即整个事务数据库不进行分割时,PHARM 算法与普通关联规则挖掘(ARM)算法的挖掘速率相同。

3 建模分析

以下通过建模分析 PHARM 算法挖掘的最佳速率与哪些因素有关,及各因素之间的相互关系,具体步骤如下。

3.1 模型假设与建立

假设事务数据库 D 被均匀分割;处理器采用常规 ARM 算法(如 Apriori 算法[11])时挖掘速率为定值 v_0(单位:事务数目/秒),即单个处理器的处理时间只与事务数据库的规模$|D|$有关;合并各分区频繁项集得全局候选项集,第二次扫描 D 时,忽略传输距离和分区内数据量的影响,假设任意两个处理器间的通信开销为 t_0(单位:秒)。由局部频繁项集连接得全局候选项集的时间开销可记为$(n-1)t_0$,采用 PHARM 算法进行挖掘所需时间为 $T_{(n)}=\frac{|D|}{nv_0}+(n-1)t_0+\Delta T$(其中,$1\leqslant n\leqslant|D|$取整数,指事务数据库被分割成的分区数目;$\Delta T$为扫描整个数据库 D 所费时间,只与数据库规模有关。),求 $T_{(n)}$ 何时取到最小?

3.2 模型分析

由于函数 $T_{(x)}=\frac{|D|}{xv_0}+(x-1)t_0+\Delta T$ 为连续函数,为求其最小值,对函数求导得:$T_{(x)}'=t_0-\frac{|D|}{x^2v_0}$,令 $T_{(x)}'=0$,得 $x=\sqrt{\frac{|D|}{t_0v_0}}$(其中$\frac{|D|}{v_0}$为常规 ARM 算法挖掘 D 所需时间,当数据量很大时其值大于 t_0),因此 $x>1$。

当 $x\in(1,\sqrt{\frac{|D|}{t_0v_0}})$时,$T_{(x)}'<0$;

当 $x\in(\sqrt{\frac{|D|}{t_0v_0}},+\infty)$时,$T_{(x)}'>0$。

所以,当 $x=\sqrt{\frac{|D|}{t_0v_0}}$时,函数 $T_{(x)}$ 取得最小值。

又因为 n 取整数,所以当 $n=\left\lfloor\sqrt{\frac{|D|}{t_0v_0}}\right\rfloor$或$\left\lceil\sqrt{\frac{|D|}{t_0v_0}}\right\rceil$时,$T_{(n)}$取到最小值,算法 PHARM 的挖掘速率最快。理想情况下:

$$T_{(n)\min}\approx T_{(x)\min}=T_{(\sqrt{\frac{|D|}{t_0v_0}})}=2\sqrt{\frac{t_0|D|}{v_0}}+\Delta T-t_0$$

当 $n=1$,即不分割数据库直接进行挖掘时的效率为$T_{(1)}=\frac{|D|}{v_0}+\Delta T$。由此,当数据规模$|D|$很大,处理器间相互通信时间开销 t_0(单位:秒)很小时,$\frac{|D|}{v_0}\gg2\sqrt{\frac{t_0|D|}{v_0}}-t_0$,即 $T_{(1)}\gg T_{(n)\min}$。

3.3 大数据仿真

由于缺乏使用 PHARM 算法的超级云计算环境(众多处理器并行执行),如下仅从理论角度说明当数据规模$|D|$一定时,PHARM 算法理论上的最短挖掘时间 $T_{(n)\min}$,及数据规模$|D|$非常大时该算法的理论最短时间。此外,数据规模$|D|$很大时,数据分区数的变化区间也很大,难以用作图软件绘制,因此,用数据分区数 n 的对数 $\lg(n)$表示横坐标,同理,用理论最短挖掘时间 $T_{(x)\min}$的对数 $\lg(T)$表示纵坐标。

(1)当$|D|=10^{12}$,$t_0=1$,$v_0=10^8$,取 $\Delta T=10$。有

$$T_{(n)}=\frac{10^{12}}{10^8n}+(n-1)+10=\frac{10^4}{n}+n+9$$

其中,$1\leqslant n\leqslant10^{12}$,且 n 为整数。

当 $n=10^2$ 时,$T_{(n)\min}=110$。

Apriori 算法挖掘所需时间 $T_0=\frac{10^{12}}{10^8}+10\approx10^4$,两种情况的挖掘效率对比如图 3 所示。

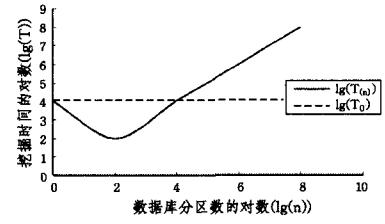

图 3 $|D|=10^{12}$时挖掘时间随分区数的变化趋势

通过图 3 大致可以看出,如果选择了合适的数据库分区数,PHARM 算法的最短挖掘时间 $T_{(x)\min}$远小于普通算法的挖掘时间 T_0,两者关系为 $T_0\approx T_{(x)\min}^2$。

(2)其他条件不变,当$|D|=10^{22}$时,取 $\Delta T=100$,则有

$$T_{(n)}=\frac{10^{22}}{10^8n}+(n-1)+10=\frac{10^{14}}{n}+n+99$$

其中,$1\leqslant n\leqslant10^{22}$,且 n 为整数。

当 $n'=10^7$ 时,得最短挖掘时间 $T_{(n)\min}\approx10^7$。

Apriori 算法挖掘此规模数据所需时间为 $T_0=\frac{10^{22}}{10^8}+10\approx10^{14}$。

设在以上条件下考虑一次分层的最短挖掘时间为 $T_{1(n)\min}$,由于 $n'=10^7$ 时,$T_{(n)}$ 取到最小,因此将数据划分成 10^7 个分区,每个分区的大小$|D|_0=\frac{|D|}{n'}=\frac{10^{22}}{10^7}=10^{15}$,对每个分区分别采用 PHARM 算法挖掘,各分区并行挖掘用时 $T_{(n_0)}=\frac{10^{15}}{10^8n_0}+(n_0-1)=\frac{10^7}{n_0}+(n_0-1)$。

当 $n_0'=\sqrt{10^7}$时,$T_{(n_0)\min}\approx10^4$。

假设每个分区挖掘后得到的结果规模为 10^5,全局挖掘时的数据规模为$|D|_1=10^5n'=10^{12}$,挖掘时间 $T_{(n_1)}=\frac{10^{12}}{10^8n_1}+(n_1-1)$,$T_{(n_1)\min}\approx10^2$,$T_{1(n)}=T_{(n_0)\min}+T_{(n_1)\min}\approx10^4$。

各算法的挖掘效率对比如图 4 所示。

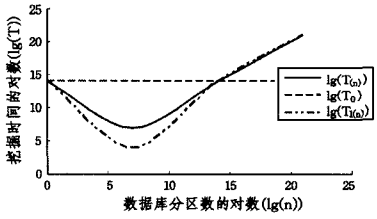

图 4 $|D|=10^{22}$时挖掘时间随分区数的变化趋势

通过对比曲线 $\lg(T_{(n)})$、$\lg(T_{1(n)})$可知，本算法的核心优势在于并行的基础上采用迭代式的分层思想。通过对比图 3、图 4 可知，随着数据规模$|D|$的增加，T_0、$T_{(x)\min}$均增加，但两者之间的关系 $T_0 \approx T^2_{(x)\min}$保持不变，即针对同一规模的数据进行挖掘，采用普通 ARM 算法所用的时间约等于采用 PHARM 算法所用时间的平方，且处理数据规模$|D|$越大，PHARM 算法越高效。

又 $value=min\_sup\times\frac{|D|}{n}$，得 $n=min\_sup\times\frac{|D|}{value}$，代入 $T_{(n)}$得 $T_{(value)}=\frac{value}{v_0\times min\_sup}+\frac{min\_sup\times|D|\times t_0}{value}-t_0+\Delta T$，从而验证了 2.2.2 节算法演示中推理的正确性。

4 数据分析

实验选用超市交易数据库中的 10 万条交易数据和 100 件商品。先对事务数据库 D 进行前期处理，将表中数据转变为整型数据，0 表示对应事务记录 T 没有选购此列商品 Item，1 表示对应事务记录 T 购买了此列商品 Item。实现算法的编程语言为 Java，选用 MyEclipse 作为集成开发环境。

固定交易事务表的事务数 $TD=10^6$ 和支持度 $sup=2\%$，考查不同分区数下 PHARM 算法的执行时间，图 5 是实验结果示意图。

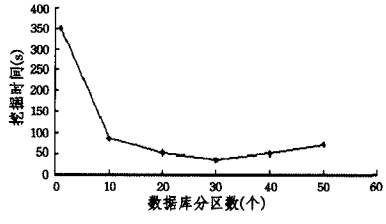

图 5 PHARM 算法挖掘效率随分区数的变化

图 5 中曲线的左起始点的横坐标为分区数 $n=1$，此时数据没进行分割，数据库全部作为一个整体进行挖掘，PHARM 算法和 Apriori 算法挖掘时间一致。但随着分区数的增加，PHARM 算法挖掘时间较 Apriori 算法明显降低，挖掘效率显著提高。

结束语 本文针对传统关联规则挖掘算法效率低，相关扩展并行算法挖掘不完整、挖掘结果精确度不高的问题，通过将划分的思想应用于关联规则挖掘过程中，提出了不丧失挖掘结果精确度的并行分层关联规则挖掘 PHARM 算法，并建模分析了算法的高效性及影响其挖掘效率的因素。在后续研究工作中，拟将 PHARM 算法移植到 Hadoop 平台进行测试，并进一步探讨其在大数据挖掘方面的应用。

参 考 文 献

[1] Jacobs A. The Pathologies of Big Data[J]. Communications of the ACM, 2009, 52(8): 36-40

[2] Fan Wei, Bifet A. Mining Big Data: Current Status, and Forecast to the Future[J]. SIGKDD Explorations, 2013, 14(2): 1-5

[3] Jin Zong-ze, Feng Ya-lan, Ji Bo, et al. Data Mining Association in the Data Analysis [J]. Computer & Digital Engineering, 2014, 10(42): 1295-1296(in Chinese)
金宗泽，冯亚兰，纪博，等. 大数据分析中的关联挖掘[J]. 计算机与数字工程，2014，10(42)：1295-1296

[4] Yu Chu-li, Xiao Ying-yuan. Parallel association rules algorithm based on Hadoop[D]. Tianjin: Tianjin University of Technology, 2011: 16-20(in Chinese)
余楚礼，肖迎元. 基于 Hadoop 的并行关联规则算法研究[D]. 天津：天津理工大学，2011：16-20

[5] Li Jian-feng, Peng Jian. Task scheduling algorithm based on improved genetic algorithm in cloud computing environment[J]. Journal of Computer Applications, 2011, 31(1): 184-185(in Chinese)
李建锋，彭舰. 云计算环境下基于改进遗传算法的任务调度算法[J]. 计算机应用，2011，31(1)：184-185

[6] Mao Yu-xing, Chen Tong-bing, Shi Bai-le. Efficient method for mining multiple-level and generalized association rules[J]. Journal of Software, 2011, 22(12): 2965-2980(in Chinese)
毛宇星，陈彤兵，施伯乐. 一种高效的多层和概化关联规则挖掘方法[J]. 软件学报，2011，22(12)：2965-2980

[7] Feng Zhong-hui, Zhou Bing, Shen Jun-yi. A parallel hierarchical clustering algorithm for PCs cluster system[J]. Neurocomputing, 2007, 70(4-6): 809-818

[8] Chaitan B, Milind B, Raghunath N. Big Data Benchmarking [C]// MBDS'12. San Jose, California, USA: ACM, 2012: 39-40

[9] Manakasemsak B, Benjamas N, Surarerks A, et al. Parallel Association Rule Mining based on FI-Growth Algorithm[C]// 2007 International Conference on Parallel and Distributed Systems. 2007: 1-8

[10] Meng Xiao-feng, Ci Xiang. Big data management: concepts, techniques and challenges[J]. Journal of Computer Research and Development, 2013, 50(1): 146-169(in Chinese)
孟小峰，慈祥. 大数据管理：概念、技术与挑战[J]. 计算机研究与发展，2013，50(1)：146-149

[11] Cuzzocrea A, Song I-Y, Davis K C. Analytics over Large-Scale Multidimensional Data: The Big Data Revolution[C]// DOLAP'11. Glasgow, Scotland, UK: ACM, 2011: 101-103

[12] Zhou Jin, Hu Liang, Wang Feng, et al. An Efficient Multidimensional Fusion Algorithm for IoT Data Based on Partitioning[J]. Tsinghua Science & Technology, 2013, 18(4): 369-378

[13] Han Jia-wei, Kamber M, Pei Jian. Data mining: concepts and techniques(3th ED)[M]. Fan M, Meng F, translated. Beijing: China Machine Press, 2012: 160-166(in Chinese)
Han Jia-wei, Kamber M, Pei Jian. 数据挖掘：概念与技术(第 3 版)[M]. 范明，孟小峰，译. 北京：机械工业出版社，2012：160-161

[14] Tang Jia-wei, Wang Xiao-feng. Design and Implementation of Apriori on GPU[J]. Computer Science, 2014, 41(10): 238-239 (in Chinese)
唐家维，王晓峰. 基于 GPU 的并行化 Apriori 算法的设计与实现[J]. 计算机科学，2014，41(10)：238-239