快速挖掘全局最大频繁项目集*

陆介平[1+], 杨 明[1], 孙志挥[1], 鞠时光[2]

[1](东南大学 计算机科学与工程系,江苏 南京 210096)

[2](江苏大学 计算机科学与通信工程学院,江苏 镇江 212013)

Fast Mining of Global Maximum Frequent Itemsets

LU Jie-Ping[1+], YANG Ming[1], SUN Zhi-Hui[1], JU Shi-Guang[2]

[1](Department of Computer Science and Engineering, Southeast University, Nanjing 210096, China)

[2](Department of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212013, China)

+ Corresponding author: Phn: +86-511-4424674, Fax: +86-511-4424674, E-mail: zjjplu@yahoo.com.cn, http://www.seu.edu.cn

Received 2004-06-03; Accepted 2004-07-02

Lu JP, Yang M, Sun ZH, Ju SG. Fast mining of global maximum frequent itemsets. *Journal of Software*, 2005,16(4):553−560. DOI: 10.1360/jos160553

Abstract: Mining maximum frequent itemsets is a key problem in data mining field with numerous important applications. The existing algorithms of mining maximum frequent itemsets are based on local databases, and very little work has been done in distributed databases. However, using the existing algorithms for the maximum frequent itemsets or using the algorithms proposed for the global frequent itemsets needs to generate a lots of candidate itemsets and requires a large amount of communication overhead. Therefore, this paper proposes an algorithm for fast mining global maximum frequent itemsets (FMGMFI), which can conveniently get the global frequency of any itemset from the corresponding paths of every local FP-tree by using frequent pattern tree and require far less communication overhead by the searching strategy of bottom-up and top-down. Experimental results show that FMGMFI is effective and efficient.

Key words: distribute database; data mining; frequent pattern tree; global maximum frequent itemset

摘 要: 挖掘最大频繁项目集是多种数据挖掘应用中的关键问题.现行可用的最大频繁项目集挖掘算法大多基于单机环境,针对分布式环境下的全局最大频繁项目集挖掘尚不多见.若将基于单机环境的最大频繁项目集挖掘算法运用于分布式环境,或运用分布式环境下的全局频繁项目集挖掘算法来挖掘全局最大频繁项目集,均会产生大量的候选频繁项目集,且网络通信代价高.为此,提出了快速挖掘全局最大频繁项目集算法 FMGMFI(fast mining global maximum frequent itemsets),该算法采用 FP-tree 存储结构,可方便地从各局部 FP-tree 的相关路径中得到项目集的频度,同时采用自顶向下和自底向上的双向搜索策略,可有效地降低网络通信代价.实验结果表明,FMGMF 算法是有

* Supported by the National Natural Science Foundation of China under Grant No.70371015 (国家自然科学基金); the National Natural Science Foundation of Jiangsu Province under Grant No.BK2004058 (江苏省自然科学基金)

作者简介: **陆介平**(1959−),男,江苏镇江人,教授级高工,主要研究领域为知识发现,数据挖掘;**杨明**(1964−),男,博士,教授,主要研究领域为知识发现,数据挖掘,粗集理论与应用;**孙志挥**(1941−),男,教授,博士生导师,主要研究领域为复杂系统信息集成,数据库系统及应用;**鞠时光**(1955−),男,教授,博士生导师,安全数据库,可视化技术.

效、可行的.

关键词: 分布式数据库;数据挖掘;频繁模式树;全局最大频繁项目集
中图法分类号: TP311 文献标识码: A

关联规则挖掘是数据挖掘的重要内容之一[1].Agrawal 于 1993 年首次提出布尔型关联规则问题及相应的 Apriori 算法[2,3]后,数据挖掘领域的研究者在关联规则挖掘与更新上做了大量的工作[4,5].其中,韩家炜等人提出的频繁项目集挖掘算法(FP-growth)采用压缩 FP-tree 存储结构,打破了 Apriori-like 框架,可以有效地提高频繁项目集的挖掘效率.由于对一个维数为 L 的频繁项目集,其每个子集都是频繁项目集,即有 2^L-1 个子集,当 L 很大时,将形成一个 NP-hard 问题,使得对于含有较长频繁项目集的情况,已有的频繁项目集挖掘算法将产生大量的频繁项目集,挖掘效率低.此外,某些数据挖掘应用(如挖掘最小主键等)仅需挖掘最大频繁项目集,而不必挖掘所有的频繁项目集.因而挖掘最大频繁项目集对数据挖掘具有重要意义.

目前可用的最大频繁项目挖掘算法有 Max-Miner[6],Pincer-Search[7],DMFI[8]及 DMFIA[9]等算法,其中 DMFIA[9]采用 FP-tree 存储结构[5],仅需扫描数据库 2 遍,改进了传统算法的性能.但 DMFIA 和 Max-Miner, Pincer-Search 及 DMFI 等传统的最大频繁项目集挖掘算法均是基于单机环境的.现行可用的全局频繁项目集挖掘算法有 PDM[10],CD[11],FMD[12],FMAGF[13]及 FPM[14]等,主要针对全局频繁项目集长度相对较短的情况,专门用于挖掘全局最大频繁项目集的算法尚不多见.若将基于单机环境的最大频繁项目集挖掘算法运用于分布式环境,或运用分布式环境下的全局频繁项目集挖掘算法来挖掘全局最大频繁项目集,均会产生大量的候选频繁项目集,且网络通信代价高.故研究高效的全局最大频繁项目集挖掘算法是本文的主要目标.

在分布式环境中,有效地降低网络通信量是提高全局频繁项目集的关键.为此,提出了快速挖掘全局最大频繁项目集算法 FMGMFI(fast mining global maximum frequent itemsets),该算法采用 FP-tree 存储结构[5],由于 FP-tree 结构是一种压缩的存储结构,数据库中的记录可被压缩存储在建立的局部 FP-tree 的各路径中,使得 FMGMFI 可方便地从各局部 FP-tree 的相关路径中得到项目集的频度,同时采用自顶向下和自底向上的双向搜索策略,可以有效地降低网络通信代价.实验结果表明,FMGMFI 算法是有效、可行的.

1 相关概念和结论

1.1 全局最大频繁项目集

沿用文献[13]的记号,在本文的模型中,分布式数据库是水平划分的,且各部分的数据库模式逻辑同构.设有 n 个站点 $S^1,S^2,\ldots,S^n$,相应的成员数据库分别为 $DB^1,DB^2,\ldots,DB^n$, $DB=\bigcup_{i=1}^{n}DB^i$ .D 及 D^i 分别表示 DB 及 DB^i 中的交易数.对某一项目集 X,$X.count$ 及 $X.count^i$ 分别表示在 DB 及 DB^i 中含 X 的交易数,若 X 仅含一个项目,则 $X.count$ 及 $X.count^i$ 分别为对应项目的全局频度和局部频度.$card(X)$表示项目集 X 中含有的项目数.

定义 1[13]. 若 $X.sup(X.count/D)$及 $X.sup^i(X.count^i/D^i)$分别表示 X 在 DB 及 DB^i 中的支持度,则称 $X.sup$ 为 X 的全局支持度,$X.sup^i$ 为 X 在站点 S^i 的局部支持度.

定义 2[13]. 若 $X.sup\geq \min sup$(minimum support threshold,最小支持度阈值),则 X 为全局频繁项目集;若 $X.sup^i\geq \min sup$,则 X 为站点 S^i 的局部频繁项目集.

引理 1[13]. 若 X 为站点 S^i 上的局部频繁项目集,则 X 的所有非空子集均为站点 S^i 上的局部频繁项目集.

推论 1[13]. 若项目集 X 不是局部频繁项目集,则 X 的超集一定不是局部频繁项目集.

类似地,若 X 为全局频繁项目集,则 X 的所有非空子集均为全局频繁项目集.

性质 1[13]. 若 X 为全局频繁项目集,则存在一站点 $S^i(1\leq i\leq n)$,使得 X 及 X 的所有非空子集在站点 S^i 上为局部频繁项目集.

定义 3[13]. 若全局频繁项目集 X 的所有超集都是非全局频繁项目集,则称 X 为全局最大频繁项目集;将所有全局最大频繁项目集组成的集合称为全局最大频繁集,记为 GMFS(globally maximum frequent sets);将所有候选全局频繁项目集组成的集合称为全局最大频繁候选集,记为 GMFCS(globally maximum frequent candidate

sets).

为方便描述,称长度为 k 的全局频繁项目集为全局频繁 k-项目集,称全局频繁 1-项目集所包含的项目为全局频繁项目,称长度为 k 的局部频繁项目集为局部频繁 k-项目集,称局部频繁 1-项目集所包含的项目为局部频繁项目.

显然,全局最大频繁项目集的所有非空子集都是全局频繁项目集,因而,可将挖掘所有全局频繁项目集的问题转化为挖掘全局最大频繁项目集的问题.

1.2 频繁模式树[5](frequent pattern tree,简称FP-tree)

一棵频繁模式树(FP-tree)为满足以下 3 个条件的树型结构:① 它由一个标为"null"的根结点(用 root 表示)作为根结点的孩子的项目前缀子树集合,以及频繁项目头表组成;② 项目前缀子树中的每一结点包含 4 个域:*item-name*,*node-count*,*node-link*,*node-parent*,其中,*item-name* 记录项目名,*node-count* 记录能到达该结点的路径所表示的交易的数目,*node-link* 为指向 FP-tree 中具有相同的 *item-name* 值的下一结点,当下一个结点不存在时,*node-link* 为 null;*node-parent* 为指向父结点的指针;③ 频繁项目头表的每一表项包含两个域:*item-name*,*head of node-link*,其中,*head of node-link* 为指向 FP-tree 中具有相同的 *item-name* 值的首结点的指针.

有关 FP-tree 建立及其上的频繁项目集挖掘算法等更详细的情况可参见文献[5].

2 全局最大频繁项目集挖掘算法

2.1 FMGMFI算法思路

如何在分布式环境中运用 FP-tree 结构及相应的全局最大频繁项目集挖掘算法来减小网络通信代价是本文研究的主要内容.为叙述方便,将 DB 对应的频繁模式树记为 FP-tree,DB^i 对应的频繁模式树记为 $FP\text{-}tree^i$ $(1\le i\le n)$,所有的全局频繁项目组成的集合记为 F.对给定的 minsup,全局最大频繁项目集的挖掘可分为 3 部分任务:① 建立 $FP\text{-}tree^i$;② 挖掘各 $FP\text{-}tree^i$ 获得各局部最大频繁集 $LMFI^i$;③ 依据各局部最大频繁项目集,采用自底向上和自顶向下的策略来挖掘全局频繁项目集.建立 $FP\text{-}tree^i$ 的过程类似文献[13].

定理 1. 对给定的最小支持度阈值 minsup,若 X 是全局最大频繁项目集,则存在一站点 S^i $(1\le i\le n)$,使得 X 为其上某一局部最大频繁项目集的子集.

证明:因 X 是全局最大频繁项目集,故 X 是全局频繁项目集.由性质 1,必存在一站点 S^i $(1\le i\le n)$,使得 X 在 S^i 上为局部频繁项目集,因而 X 为其上某一局部最大频繁项目集的子集. □

推论 1. 如果 X 不是任一站点 S^i 的局部最大频繁项目集的子集,则 X 一定不是全局最大集,这与已知条件矛盾.

由推论 1,全局最大频繁项目集可通过局部最大频繁项目集的子集得到.因而对于每一个站点,自顶向下搜索该站点上的局部最大频繁项目集的子集,若相应的子集 X 不是全局频繁项目集,则继续,否则,若在其他站点上的任意局部最大频繁项目集 Y 都不存在满足 $X\subseteq Z\subseteq Y$ 的全局频繁项目集 Z,则 X 是全局最大频繁项目集.

定理 2. 若 Y 为某站点上的局部最大频繁项目集,X 为 Y 的某个最大子集且为全局频繁项目集;对任意站点上的任意局部最大频繁项目集 Y',不存在满足 $X\subseteq Z\subseteq Y'$ 的全局频繁项目集 Z,则 X 是全局最大频繁项目集.

证明:反证法.若 X 不是全局最大频繁项目集,则由为全局频繁项目集可知 X 必为某个全局最大频繁项目集的子集.不妨设 $X\subset Z$ 且 Z 为全局最大频繁项目集,由定理 1 知,存在某个站点 S^i $(1\le i\le n)$使得 Z 为其上某一局部最大频繁项目集的子集,这与已知条件矛盾. □

实例 1. 设 3 个站点 S^1,S^2,及 S^3 上的交易数据库分别为 DB^1,DB^2 及 DB^3,见表 1.对 minsup=0.5,建立的 $FP\text{-}tree^1$,$FP\text{-}tree^2$ 及 $FP\text{-}tree^3$ 如图 1~图 3 所示.

Table 1 The transaction DBs on S^1,S^2,S^3

表 1 S^1,S^2,S^3 上的交易数据库

	DB^1		DB^2		DB^3	
ID	10	20	30	40	50	60
Transactions	f,a,c,d,g,i,m,p	a,b,c,f,l,m,o	b,f,h,j,o	b,c,k,s,p	a,f,c,e,l,p,m,n	k,s

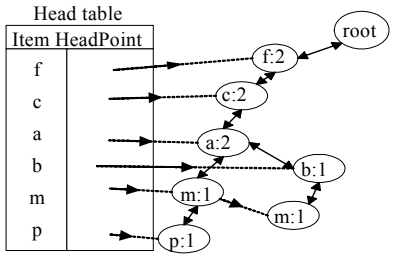

Fig.1 *FP-tree*[1]

图 1 *FP-tree*[1]

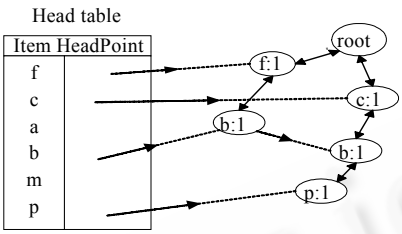

Fig.2 *FP-tree*[2]

图 2 *FP-tree*[2]

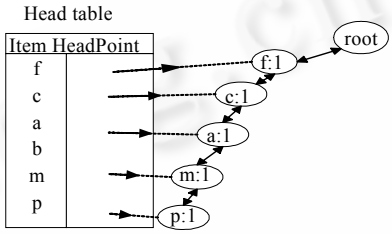

Fig.3 *FP-tree*[3]

图 3 *FP-tree*[3]

对于实例 1,若采用 FMD 算法挖掘全局最大频繁项目集,用 CL_k^i 和 GL_k^i 分别表示站点 S^i 产生的所有局部频繁候选 k-项目集(由在站点 S^i 既是全局频繁又是局部频繁的(k–1)-项目集生成)和所有全局频繁 k-项目集,用 $CL_k=\bigcup_{i=1}^{n}CL_k^i$ 及 $GL_k=\bigcup_{i=1}^{n}GL_k^i$ 分别表示所有全局频繁候选 k-项目集和所有全局频繁 k-项目集,则 FMD 算法逐层产生全局频繁和局部频繁项目集:

① $GL_1^1=\{\{f\},\{c\},\{a\},\{b\},\{m\},\{p\}\}$, $GL_1^2=\{\{f\},\{c\},\{b\},\{p\}\}$, $GL_1^3=\{\{f\},\{c\},\{a\},\{m\},\{p\}\}$;

② $CL_2^1=\{\{f,c\},\{f,a\},\{f,b\},\{f,m\},\{f,p\},\{c,a\},\{c,b\},\{c,m\},\{c,p\},\{a,b\},\{a,m\},\{a,p\},\{b,m\},\{b,p\},\{m,p\}\}$,

$CL_2^2=\{\{f,c\},\{f,b\},\{f,p\},\{c,b\},\{c,p\},\{b,p\}\}$,

$CL_2^3=\{\{f,c\},\{f,a\},\{f,m\},\{f,p\},\{c,a\},\{c,m\},\{c,p\},\{a,m\},\{a,p\},\{m,p\}\}$;

$GL_2^1=\{\{f,c\},\{f,a\},\{f,m\},\{c,a\},\{c,m\},\{c,p\},\{a,m\}\}$,

$GL_2^2=\{\{f,c\},\{c,p\}\}$,

$GL_2^3=\{\{f,c\},\{f,a\},\{f,m\},\{c,a\},\{c,m\},\{c,p\},\{a,m\}\}$.

重复过程②直到得到全局最大频繁集 $GMFS=\{\{b\},\{p,c\},\{f,c,a,m\}\}$.可见,FMD 算法依据 Apriori 算法框架在自底向上挖掘全局最大频繁项目集过程中将产生大量的候选频繁项目集,传送大量的候选项目集,网络通信代价高.而单机环境下的 DMFIA 算法,为发现全局最大频繁项目集$\{b\}$和$\{p,c\}$,DMFIA 在自顶向下的搜索过程中将传送大量全局频繁候选项目集,网络通信代价高.

为克服 FMD 及 DMFIA 存在的不足,FMGMFI 采用以下策略修剪全局频繁候选项目集:① 在自顶向下搜索时,采用较长的局部最大频繁项目集优先搜索,发现长最大全局频繁项目集;② 在自底向上搜索时,发现较短的最大频繁项目集.

定理 3. 若 $X\in GL_k$ 是全局频繁项目集,且不存在 $Y\in GL_{k+1}$ 使得 $X\subset Y$,则 X 是全局最大频繁项目集.

证明:若 $X\in GL_k$ 是全局频繁项目集,且不存在 $Y\in GL_{k+1}$ 使得 $X\subset Y$,则 X 的所有超集都是非全局频繁项目集,因而 X 是全局最大频繁项目集. □

对于实例 1,依据 FMGMFI 算法的思路(详见第 2.2 节),对各 *FP-tree*^{i} (i=1,2,3)进行局部最大频繁项目集挖掘,得到局部最大频繁集 $LMFI^1$,$LMFI^2$,$LMFI^3$ 分别为$\{\{f,c,a,m,p\},\{f,c,a,m,b\}\}$,$\{\{f,b\},\{c,b,p\}\}$,$\{\{f,c,a,m,p\}\}$.若自顶向下得到全局最大频繁项目集$\{f,c,a,m\}$,则在自底向上搜索时无须传送项目集$\{f,c\}$,$\{f,a\}$,$\{f,m\}$,$\{c,a\}$,$\{c,m\}$,

{*a*,*m*},{*f*,*c*,*a*},{*f*,*c*,*m*},{*f*,*a*,*m*},{*c*,*a*,*m*}.若自底向上得到全局最大频繁项目集{*b*},则在自顶向下搜索时无须生成项目集{*b*,*f*,*c*,*a*,*m*},{*b*,*f*,*c*,*a*},{*b*,*f*,*c*,*m*},{*b*,*f*,*a*,*m*},{*b*,*c*,*a*,*m*},{*b*,*f*,*c*},{*b*,*f*,*a*},{*b*,*f*,*m*},{*b*,*c*,*a*},{*b*,*c*,*m*},{*b*,*a*,*m*}等.在站点S^2,由于{*f*,*p*},{*f*,*c*}等不是$LMFI^2$中任一项目集的子集,因而无须生成.可见,依据 FMGMFI 算法的策略,可有效地提高全局最大频繁集的挖掘效率,其性能优于 FMD.

2.2 算法FMGMFI描述

类似于 FMD 算法,采用计数站点(polling site)来优化网络通信,使得在含有 n 个站点的分布式环境中,信息的传送代价为 $O(n)$.由定理 1、定理 2 及定理 3,FMGMFI 算法描述如下.

FMGMFI 算法.

输入:

(1) DB^i 为站点 S^i 上的交易数据库,它含有 D^i 条交易,i=1,2,…,n;

(2) minsup 为最小支持度阈值,且各站点 S^i 的最小支持度阈值均为 minsup,i=1,2,…,n;

输出:全局最大频繁集 GMFI.

方法:按下列步骤挖掘全局最大频繁集 GMFS.

步骤 1. /* 建立各站点 S^i 的 $FP\text{-}tree^i$ */

(1) Scan DB^i once and Collect the set of local items F^i and their supports;

Collect the set of Global frequent items F from all F^i;

Sort F in support descending order as $\Re$,the list of Global frequent items.

Broadcast F. /* 将 F 传送到各站点 */

(2) Create the root of an $FP\text{-}tree^i$, T^i , and label it as "null"; /*类似文献[13]*/

步骤 2. /* 由各 $FP\text{-}tree^i$ 挖掘局部最大频繁集 $LMFS^i$ */

(3) $LMFS^i=\varnothing$; /* $LMFS^i$ 为由 $FP\text{-}tree^i$ 得到的局部最大频繁集 */

Call MiningLMFS($FP\text{-}tree^i$, S^i , minsup, $LMFS^i$);

步骤 3. /* 挖掘全局最大频繁集 */

(4) $GMFCS=\bigcup_{i=1}^{n} LMFS^i$;

$GMFS=\varnothing$; k=2; L=max({card(x)|$x\in GMFCS$}; /* L 表示 GMFCS 中含有项目数最多的项目集的项目数*/

generate GL_1^i from the head table of $FP\text{-}tree^i$;

(5) while $GMFCS\neq\varnothing$ do

{ /* 自顶向下 */

(6) for all $X\in LMFS^i$ do /* 删除全局最大频繁项目集的子集 */

if 存在 $Z\in GMFS$ 使得 $X\subseteq Z$ 或 $Z\subseteq X$ then $LMFS^i=LMFS^i-\{X\}$;

else

if $X\in LMFS^i$ and card(X)=L then

for j=1 to n do /* $CL_L^{i,j}$ 用来存放 S^i 中所有传送到 S^j 的项目集 */

if polling_site(X)=S^j and 存在 $Y\in GL_{K-1}^i$ 使得 $Y\subseteq X$ then add$\langle X,X.count^i\rangle$into $CL_L^{i,j}$;}

(7) for j=1 to n do send $CL_L^{i,j}$ to S^j; /* 发送候选项目集到计数站点 */

for j=1 to n do { receive $CL_L^{j,i}$; /* 计数站点接受候选项目集 */

for all $X\in CL_L^{j,i}$ do

store X in LP_L^i and update $X.larger\text{-}sites$ in LP_L^i; /* LP_L^i 记录已发送 X 到站点 S^i 的站点地址 */

broadcast polling requests for X to the sites S^j , where $S^j\notin X.larger\text{-}sites$;

/* $X.count^j$ 可快速地从 FP-tree^j 的各路径中得到 */

receive $X.count^j$ from the sites S^j , where $S^j\notin X.larger\text{-}sites$;

}

(8) for all $X\in LP_L^i$ do

{$X.count=\sum_{i=1}^{n} x.count^i$; /* 计算全局支持度 */

if $X.sup=X.count/D\geq \min sup$ then insert X into GL_L^i;

else /* 若 X 不是全局最大频繁项目集 */

for all item $x\in X$ do

if $X-\{x\}$ 不是 $LMFS^i$ 中某元素的子集 then $LMFS^i=LMFS^i\cup\{X-\{x\}\}$;

}

(9) $L=L-1$;

broadcast $GMFS$; broadcast $LMFS^i$;

receive $GMFS$ and $LMFS^j$ from all other sites S^j,$(j\neq i)$; $GMFCS=\bigcup_{i=1}^{n} LMFS^i$;

(10) /*自底向上*/

类似(6)~(9);

(11) } /* $GMFCS\neq\varnothing$ */

其中,过程 MiningLMFS 描述如下:

Procedure MiningLMFS(*LFP-tree*,*S*,min*sup*,*LMFS*)

/* *LFP-tree* 为站点 *S* 上的局部频繁模式树,min*sup* 为最小支持度阈值,LMFS 为局部最大频繁项目集 */

{ (1) *LMFCS*=F(*LFP-tree*); /* 初始化局部最大频繁候选项目集 */

(2) For all itemsets $X=\{x_1,\ldots,x_q\}\in LMFCS$ do{

(3) $LMFCS=LMFCS-X$;

(4) If X 不是 *LMFS* 中某元素的子集 then{

(5) call getcount(*LFP-tree*,*X*); /* 得到项目集 X 的支持度 */

(6) if support(X)$\geq$min*sup* then $LMFS=LMFS\cup X$;

(7) else for all item $x\in X$ do

(8) if $X-\{x\}$ 不是 *MFCS* 中某元素的子集 then

(9) $LMFCS=LMFCS\cup\{X-\{x\}\}$; }

可以看出,采用 FMGMFI 算法挖掘最大频繁集有以下优点:① 仅需扫描各局部数据库两遍,且各项目集的频度可由各局部 FP-tree 的相应路径快速得到而无须扫描局部数据库,无须记录非局部频繁项目集的局部频度;② 在自底向上搜索时无须传送已挖掘出的全局最大频繁项目集的子集,有效地减少了项目集的传送量,尤其是对含有较长全局最大频繁项目集的情况;③ 在自顶向下搜索时,无须传送非全局频繁项目集的超集,且无须生成全局最大频繁项目集的任何超集,因而降低了网络通信开销.

3 实验结果

为测试算法的性能,采用文献[3]的合成数据方法生成实验测试数据,并将合成的交易数据库均分到 3 个站点,使各站点的 DB^i 记录数相等(i=1,2,3).为叙述方便,用 *DB* 表示合成的交易数据库;*D* 表示总的交易数;*N* 表示交易项目的个数;|*T*|表示交易数据记录的平均长度;|*I*|表示最大的潜在频繁项目集的平均长度;|*L*|表示最大的潜在全局频繁项目集数;min*sup* 表示最小支持度阈值;$T_x.T_y.DB_m$K 表示测试数据库实例,其中,|*T*|=*x*,|*I*|=*y*,|*DB*|=*D*=*m*K.我们在操作系统为 Windows 2000 server,CPU 为 PIII400,硬盘为 40G,内存为 256M 的实验环境下,用 VC++6.0 实现了算法 FPM,FMGFI 和 FMGMFI,并进行了以下 3 组实验:① 对 *D*=300K,*N*=1000,|*L*|=1500,|*T*|=10,|*I*|=8,用 5 个不同的 min*sup*(2%,1.5%,1%,0.75%,0.1%)进行测试,结果如图 4 和图 5 所示;② 保持①中的 *N*,|*L*|及 min*sup* 取值不变,而使 *D*=200K,|*T*|=14,|*I*|=5,对算法进行测试,结果如图 6 所示;③

对 min*sup*=1%,*N*=1000,|*T*|=10,|*I*|=6,用 5 个不同的 *D*(180K,210K,240K,270K,300K)进行测试,结果如图 7 所示.由图 4~图 7 可知,FMGMFI 的性能优于 FPM 和 FMGFI.

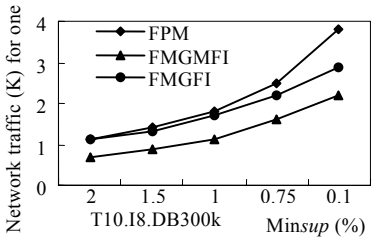

Fig.4 Network traffic

图 4 网络通信量

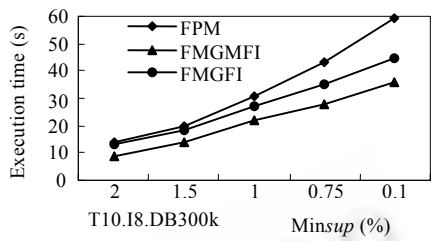

Fig.5 Execution times

图 5 算法执行时间

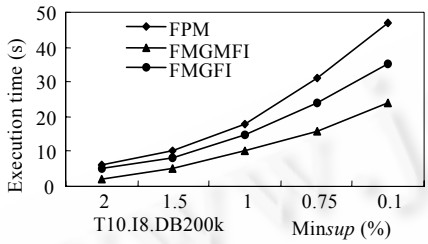

Fig.6 Execution times

图 6 算法的执行时间

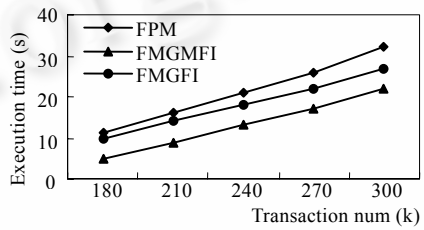

Fig.7 Scalability(min*sup*=1%)

图 7 算法的可扩展性(min*sup*=1%)

4 结 语

提出快速挖掘全局最大频繁项目集算法 FMGMFI,采用 FP-tree 存储结构,快速地从各局部 FP-tree 的相关路径中得到项目集的频度,同时采用自顶向下和自底向上的双向搜索策略,可以克服直接应用最大频繁项目集挖掘的单机算法和用全局频繁项目集挖掘算法来挖掘全局最大频繁项目集存在的不足,有效地减少生成和传送的全局频繁候选项目集数目,降低网络通信代价,因而提高了全局最大频繁项目集的挖掘效率.下一步的研究目标是在数据库动态改变情况下,研究全局最大频繁项目集的增量式更新算法.

References:

[1] Han J, Kamber M. Data Mining: Concepts and Techniques. Beijing: High Education Press, 2001.

[2] Agrawal R, ImielinSki T, Swami A. Mining association rules between sets of items in large database. In: Proc. of the ACM SIGMOD Int'l Conf. on Management of Data. Vol 2, Washington DC: SIGMOD, 1993. 207−216.

[3] Agrawal, R Srikant. Fast algorithms for mining association rules. In: Proc. of the 20th Int'l Conf. Very Large Data Bases (VLDB'94). 1994. 487−499.

[4] Yang M, Sun ZH. An incremental updating algorithm based on prefix ceneral list for association rules. Chinese Journal of Computers, 2003,26(10):1318−1325.

[5] Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation. In: Proc. of the 2000 ACM-SIGMOD Int'l Conf. on Management of Data. Dallas: ACM Press, 2000. 1−12.

[6] Bayardo RJ. Efficiently mining long patterns from databases. In: Haas LM, Tiwary A, eds. Proc. of the ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 1998. 85−93.

[7] Lin D, Kedem ZM. Pincer-Search: A new algorithm for discovering the maximum frequent set. In: Proc. of the 6th European Conf. on Extending Database Technology. Heidelberg: Springer-Verlag, 1998. 105−119.

[8] Lu SF, Lu ZD. Fast mining maximum frequent itemsets. Journal of Software, 2001,12(2):293−297 (in Chinese with English abstract).

[9] Song YQ, Zhu YQ, Sun ZH, Chen G. An algorithm an its updating algorithm based on FP-Tree for mining maximum frequent itemsets. Journal of Software, 2003,14(9):1586−1592 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/14/1586.htm

[10] Park JS, Chen MS, Yu PS. Efficient parallel data mining for association rules. In: Proc. of the 4th Int'l Conf. on Information and Knowledge Management. 1995. 31−36.

[11] Agrawal R, Shafer J. Parallel mining of association rules. IEEE Trans. on Knowledge and Data Engineering, 1996,8(6):962−969.

[12] Cheung DW, Han JW, Ng VT. A fast distributed algorithm for mining association rules. In: Proc. of the IEEE 4th Int'l Conf. Parallel and Distributed Information Systems. Miami Beach: IEEE Press, 1996. 31−44.

[13] Yang M, Sun ZH, Ji GL. Fast mining of global frequent Itemsets. Journal of Computer of Research and Development, 2003,40(4):620−626 (in Chinese with English abstract).

[14] Cheung DW, Lee SD, Xiao YQ. Effect of data skewness and workload balance in parallel data mining. IEEE Trans. on Knowledge and Data Engineering, 2002,14(3):498−514.

附中文参考文献:

[4] 杨明,孙志挥.一种基于前缀广义表的关联规则增量式更新算法.计算机学报,2003,26(10):1318−1325.

[8] 路松峰,卢正鼎.快速开采最大频繁项目集.软件学报,2001,12(2):293−297.

[9] 宋余庆,朱玉全,孙志挥,陈耿.基于 FP-tree 的最大频繁项目集挖掘及更新算法.软件学报,2003,14(9):1586−1592. http://www.jos.org.cn/1000-9825/14/1586.htm

[13] 杨明,孙志挥,吉根林.快速挖掘全局频繁项目集.计算机研究与发展,2003,40(4):620−626.

第 12 届全国图像图形学学术会议

征 文 通 知

由中国图像图形学学会主办的“第 12 届全国图像图形学学术会议”将于 2005 年 10 月 12 日~10 月 14 日在北京召开。

一、会议征文范围

A) 图像处理和编码: 图像采集、获取及存储，图像重建，图像变换、滤波、增强、校正、恢复/复原，图像/视频压缩编码，图像数字水印和图像信息隐藏等

B) 图像分析和识别: 边缘检测、图像分割，目标表达、描述、测量，目标颜色、形状、纹理、空间、运动等的分析，目标检测、提取、跟踪、识别和分类等

C) 图像理解和计算机视觉: 立体成像，图像配准、匹配、融合、镶嵌，3-D 表示、建模、重构、场景恢复，图像感知、解释、推理等

D) 计算机图形学: 图形学基本理论和算法，真实感图形生成，图形系统、GIS 及图形数据库，几何造形基础理论和算法，自然景物模拟，计算机动画等

E) 虚拟现实和增强现实: 虚拟环境构建，基于图像的建模和绘制，场景建模和漫游，分布式虚拟现实，人工生命等

F) 数字娱乐与游戏设计: 游戏设计方法，游戏开发平台，游戏中的人工智能，游戏引擎，手机游戏，三维游戏等

G) 多媒体技术: 人机交互与用户界面，人体生物特征提取和验证，基于内容的图像和视频检索等

H) 图像图形技术应用: 图像图形技术在通信广播，生物医学，文档处理，遥感、雷达、测绘，工业自动化等领域的应用

二、论文投稿格式 详见会议网址 http://www.csig.org.cn/ncig2005

三、重要日期 2005 年 5 月 15 日：全文投稿　　2005 年 6 月 15 日：发录取通知

2005 年 7 月 15 日：上交最终稿　　2005 年 10 月 12 日~14 日：召开会议

四、邮件地址 ncig2005@ia.ac.cn