基于极大团和 FP-Tree 的挖掘关联规则的改进算法*

陈安龙[1+], 唐常杰[1], 陶宏才[2], 元昌安[1,3], 谢方军[1]

[1](四川大学 计算机学院,四川 成都 610064)

[2](西南交通大学 计算机与通信工程学院,四川 成都 610031)

[3](广西师范学院 信息技术系,广西 南宁 530001)

An Improved Algorithm Based on Maximum Clique and FP-Tree for Mining Association Rules

CHEN An-Long[1+], TANG Chang-Jie[1], TAO Hong-Cai[2], YUAN Chang-An[1,3], XIE Fang-Jun[1]

[1](College of Computer Science and Engineering, Sichuan University, Chengdu 610064, China)

[2](College of Computer and Communication Engineering, Southwest Jiaotong University, Chengdu 610031, China)

[3](Department of Information Technology, Guangxi Teachers' College, Nanning 530001, China)

+ Corresponding author: Phn: +86-28-85466105, E-mail: chenanlong@126.com, http://cs.scu.edu.cn/~tangchangjie

Received 2003-10-28; Accepted 2004-04-27

Chen AL, Tang CJ, Tao HC, Yuan CA, Xie FJ. An improved algorithm based on maximum clique and FP-tree for mining association rules. *Journal of Software*, 2004,15(8):1198~1207.

http://www.jos.org.cn/1000-9825/15/1198.htm

Abstract: This paper integrates the advantage of the FP-Tree algorithm for mining association rules and the maximum clique theory of graph. The main contributions include: (1) An improved method to mine frequent 2-itemset by adjacency matrix is proposed. (2) The concept of maximum ordered frequent itemset is proposed, and the equivalence of Head Relation is proved as along with the theorems about Local Complexity and Merge Convergence Range. (3) The MaxCFPTree algorithm based on Maximum-clique partition is proposed and implemented with complexity $O(n^2)$. (4) The algorithms are validated by extensive experiments. The conflict between memory and huge number of items is resolved, and the system efficiency and scalability are improved.

Key words: association rules; FP-tree; maximum clique; adjacency matrix; merge convergence

摘 要: 融合了关联规则挖掘的 FP-Tree 算法和图论的极大团理论的优势,做了以下主要工作:(1) 提出了用邻接矩阵的产生频繁 2-项集的改进方法;(2) 提出了极大有序频繁集的概念,证明了 Head 关系的等价性、划分定理、局部复杂性定理和归并收敛值域定理;(3) 提出并实现了基于极大团划分的 MaxCFPTree 算法,扫描时间复杂性小于

* Supported by the National Natural Science Foundation of China under Grant No.60073046 (国家自然科学基金); the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No.20020610007 (国家教育部博士点专项基金); the Natural Science Foundation of Guangxi Province of China under Grant No.0339039 (广西自然科学基金)

作者简介: **陈安龙**(1971－),男,四川仪陇人,博士生,主要研究领域为数据挖掘;**唐常杰**(1946－),男,教授,博士生导师,主要研究领域为数据挖掘;**陶宏才**(1964－),男,副教授,主要研究领域为网络安全,数据库,数据挖掘;**元昌安**(1964－),男,博士生,副教授,主要研究领域为数据库,空间数据挖掘;**谢方军**(1973－),男,博士生,主要研究领域为数据挖掘,网络安全.

$O(n^2)$;(4) 做了相关实验,以验证算法的正确性.新方法缓解了项目数量巨大而内存不足的矛盾,提高了系统效率和伸缩性.

关键词: 关联规则;FP-Tree;极大团;邻接矩阵;归并收敛

中图法分类号: TP311 **文献标识码**: A

1 问题背景

关联规则(association rules)的挖掘是数据挖掘研究的重要内容之一.Agrawal 等人在 1993 年首先提出了从交易数据库中发现用户模式的相关性问题,并提出了基于频繁集的 Apriori 算法[1,2].该算法的主要优点是算法思路比较简单,以递归统计为基础,剪切生成频繁集.其主要缺点是,在产生频繁模式的过程中,需要产生大量的候选项和多次遍历数据库,占用大量的内存空间和 CPU 处理时间,难以适应海量数据挖掘[3,4].

J. Han 提出了用频繁模式树产生频繁集的方法[5,6],其主要思想是将用产生频繁集的数据压缩到一棵频繁模式树 FP-Tree 中,用 FP-Tree 存储项目的关联信息,然后对模式树产生频繁集.FP-Tree 算法主要优点是:(1) 不需要产生候选项,仅需要构造 FP-Tree 和条件 FP-Tree,通过递归地访问 FP-Tree,产生频繁模式;(2) 对事务数据库仅需两次遍历,第 1 次遍历产生频繁 1-项集,第 2 次遍历用于创建 FP-Tree,从而极大地降低了访问数据库的次数.FP-Tree 算法的主要缺点是需要占用大量内存(与 FP-Tree 的深度和宽度成比例).树的深度一般是单个事务中所含项目数量的最大值;树的宽度是平均每层所含项目的数量.如果数据库中的频繁 1-项集的数量很大,且内存不能装入库中所有项目在 FP-Tree 的映射信息时,算法将不能有效地工作.

本文在继承 FT-Tree 算法不需要产生候选项的优点的基础上,寻求一种合理的方法,将数据库的项集分解成若干子集,分别对子集使用 FP-Tree 算法得到频繁模式.这些频繁模式的并集为数据库的所有频繁模式.

2 本文的主要工作

(1) 使用邻接矩阵法产生频繁 2-项集,并以频繁 2-项集中的项目作为图的顶点,把属于同一个频繁 2-项集的两个项目对应的顶点用边相连的方法生成图.(2) 提出了极大有序频繁集的概念,证明了 Head 关系的等价性、划分定理、局部复杂性定理和归并收敛值域定理;用极大团的方法将频繁 2-项集中的项目划分为多个子集.(3) 在 FP-Tree 算法中融合了邻接矩阵和极大团划分的思想,提出了 MaxCFPTree 算法,其扫描的时间复杂性为 $O(n^2)$.(4) 对算法的正确性进行了相关的实验验证,对改进前后的 FP-Tree 算法与经典 Apriori 算法进行了比较.本文的创新点在于:融合了邻接矩阵方法、极大团的划分思想与 FP-Tree 算法,从深层性质导出新的算法,以缓解内存不足的矛盾,达到算法在时间和空间上的平衡,改善系统的效率,提高可伸缩性.

3 用邻接矩阵求频繁 2-项集

本文借助于邻接矩阵研究频繁集的计数,以减少数据库的扫描次数.定义 1 描述了无向图的邻接矩阵[7].

定义 1(邻接矩阵). 设 $G=\langle V,E\rangle$是简单无向图,其中 $V=\{v_1v_2,\ldots,v_3\}$;$A=(a_{ij})$为 $n\times n$ 的矩阵,且满足

$$a_{ij}=\begin{cases}1, & \text{if } \langle v_i,v_j\rangle\in E\\ 0, & \text{if } \langle v_i,v_j\rangle\notin E\end{cases},$$

则称矩阵 A 为图 G 的邻接矩阵.

我们用类似于图的邻接矩阵来记录数据库中项目的 2-项集出现的频率;用与图 G 的定义相似的方法来定义 2-项集的邻接矩阵.

定义 2(2-项集的邻接矩阵). 设数据库中所有项目的集合为 $I=\{i_1,i_2,\ldots,i_n\}$,所有事务的集合为 T.设 $A=(a_{ij})$为满足下列条件的 $n\times n$ 的矩阵:(1) 如果 T 中有且仅有 k 个事务包含项目集$\{v_i,v_j\}$且 $i\neq j$,则矩阵中的元素 $a_{ij}=k$;(2) 如果 T 中有且仅有 k 个事务包含项目 v_i,则矩阵中的元素 $a_{it}=k$.称该矩阵是 2-项集的邻接矩阵 M.

用矩阵来记录 2-项集和 1-项集在数据库中出现的频率.如定义 2,矩阵中的元素 a_{ij} 记录项目集$\{v_i,v_j\}$且 $i\neq j$ 在事务集中出现的频率,对角线的元素 a_{ii} 记录项目在事务集中出现的次数.如果矩阵元素 $a_{ij}\geq minsup\times|D|$,项目

集为频繁 2-项集;如果对角线的元素 $a_{ii}\geq minsup\times|D|$,则项目集为频繁 1-项集.

例 1:表 1 是某个数据库中的事务记录,表 2 是项目的邻接矩阵,表 3 是频繁 2-项集,并且算法 1 描述了求事务数据库中的频繁 1-项集和频繁 2-项集的方法.

Table 1 List of item in database
表 1 数据库中的项目列表

TID	List of item ID
100	*A*,*C*,*D*,*F*,*G*,*I*
200	*A*,*B*,*C*,*D*,*F*
300	*B*,*F*,*H*,*J*
400	*B*,*C*,*J*
500	*A*,*C*,*E*,*F*

Table 2 The adjacency matrix of item
表 2 项目的邻接矩阵

Item	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*
A	3	1	3	2	1	3	1	0	1	0
B	1	3	2	1	0	2	0	1	0	2
C	3	2	4	2	1	3	1	0	1	0
D	2	1	2	2	0	2	1	0	1	0
E	1	0	1	0	1	1	0	0	0	0
F	3	2	3	2	1	4	1	1	1	1
G	1	0	1	1	0	1	1	0	1	0
H	0	1	0	0	0	1	0	1	0	1
I	1	0	1	1	0	1	1	0	1	0
J	0	2	1	0	0	1	0	1	0	2

Table 3 Frequent sets, *minsup*=2/5
表 3 频繁集,*minsup*=2/5

1-Item set	Support measure	2-Item set	Support measure
A	3/5	*AC*	3/5
B	3/5	*AD*	3/5
C	4/5	*AF*	2/5
D	2/5	*BC*	2/5
F	4/5	*BF*	2/5
J	2/5	*BJ*	2/5
		CD	3/5
		CF	2/5
		DF	2/5

算法 1. *MatrixF*2*Set*().

输入:数据库 *D*,最小支持度 *minsup*.

输出:频繁 1-项集和频繁 2-项集.

符号:t_k,数据库的第 k 条事务;v_i,事务中的项;L_1,频繁 1-项集构成的集合;L_2,频繁 2-项集的集合;

a_{ii},矩阵对角线上的元素;a_{ij},矩阵 i 行 k 列上的元素.

步骤:for all t_k do –

{ for $\forall v_i\in t_k$ do $a_{ii}=a_{ii}+1$;

for $\forall v_i\in t_k$ and $\forall v_j\in t_k$ and $i\neq j$ do $a_{ij}=a_{ij}+1$ }

for $i=1$ to n do { for $j=1$ to n do { if $a_{ij}\geq minsup\times|D|$ then if $i=j$ then $v_i\cup L_1$ else $\{v_i,v_j\}\cup L_2$ }}

Return L_1,L_2.

当数据库项目数量过大时,内存不可能装入 2-项集的邻接矩阵,则可采用改进算法 2.

算法 2. *MatrixF*2*SetbyK*().

输入:数据库 *D*,最小支持度 *minsup*,矩阵的最大行列数 *k*.

输出:频繁 1-项集和频繁 2-项集.

符号:*n*,数据库的项目总数;V_i,数据库项目分解的第 i 子集;L_{1i},第 i 子集产生的频繁 1-项集;

L_{2i},第 i 子集产生的频繁 2-项集,且 $1\leq i\leq\left\lceil\frac{n}{k}\right\rceil$;*AcrossMatrix*(),两个子集交叉产生频繁 2-项集.

步骤:① if $n<=k$ then Return *MatrixF*2*Set*()

② else { for i=1 to $\left\lceil\frac{n}{k}\right\rceil$ do { if $i<\left\lceil\frac{n}{k}\right\rceil$ then $V_i=\{v_{k(i-1)+1},v_{k(i-1)+2},\ldots,v_{ki}\}$ else $V_i=\{v_{k(i-1)+1},v_{k(i-1)+2},\ldots,v_n\}$

for each V_i do { call *MatrixF*2*Set*(); $L_1=\bigcup L_{1i}$, $L_2=\bigcup L_{2i}$;}}

③ 取元素最少的两个集合 L_{1i}和L_{1j}; call *AcrossMatrix*(L_{1i}, L_{1j},*D*,*minsup*);

④ if $|L_{1i}|+|L_{1j}|<=k$ then $L_{1i}\bigcup L_{1j}$,goto ③

⑤ else for all L_{1i},L_{1j} and $i\neq j$ do call *AcrossMatrix*(L_{1i}, L_{1j},*D*,*minsup*);

Return L_1,L_2

function *AcrossMatrix*(L_{1k}, L_{1l},*D*,*minsup*) // L_{1k} 对应矩阵的行,L_{1l} 对应矩阵的列

{ for all t_k do {for $\forall v_i\in t_k$ and $\forall v_j\in t_k$ do $a_{ij}=a_{ij}+1$ }

for i=1 to 最大行的编号 do {for j=1 to 最大列的编号 do if $a_{ij}\geq minsup\times|D|$ then $\{v_i,v_j\}\cup L_2$}}

Return L_2.

上述算法需扫描数据库次数最多为 $\frac{\left\lceil \frac{n}{k} \right\rceil \left(\left\lceil \frac{n}{k} \right\rceil +1 \right)}{2}$,下文将给出证明.

4 用极大团划分项目集

为了方便讨论问题,本文引入有序序列来表示频繁集.下面将给出一系列定义以及性质[7,8].

定义 3(有序项目集). 设 $U=\{u_1,u_2,...,u_n\}$ 是构成数据库事务的项目集且 $k\in\{1,2,...,n\}$,如果对于 $\forall l$ 且 $1\le l<k$,都有 $u_l\le u_k$ 成立,则称项目集 U 为有序项目集.

定义 4(有序频繁集). 设 $U=\{u_1,u_2,...,u_n\}$ 为频繁集,$k\in\{1,2,...,n\}$,如果 U 为有序项目集,则称 U 为有序频繁集.

本文约定用有序串 $u_1u_2...u_n$ 表示 U.

定义 5(极大有序频繁集). 设 $U=\{u_1,u_2,...,u_n\}$ 为频繁集,且 $U\subseteq I$.(1) 对于 $\forall i_k\in I$ 且 $i_k\notin U$,$U\cup i_k$ 均为非频繁集,则称 U 为极大频繁集;(2) 如果 U 为有序频繁集,且满足 U 为极大频繁集,则称 U 为极大有序频繁集.

用 R 表示所有极大频繁有序集构成的集合.

命题 1. 设 R 是有限项目集 I 构成的所有极大频繁集的集合,U 是频繁集,则 $\exists W\in R$,使 $U\subseteq W$ 成立.

证明:(反证法)设 $\exists U$ 是频繁集,对于 $\forall W\in R$,有 $U\not\subset W$ 成立.由假设可知 U 不是极大频繁集,由定义 5 知,$\exists x_1\in I$,且 $x_1\notin U$,使得 $U\cup\{x_1\}$ 是频繁集.如果 $U\cup\{x_1\}$ 是极大频繁集,则 $U\subset(U\cup\{x_1\})\in R$,与假设矛盾,所以 $U\cup\{x_1\}$ 不是极大频繁集.因而 $\exists x_2\in I$,且 $x_2\notin U\bigcup\{x_1\}$,使得 $U\cup\{x_1,x_2\}$ 是频繁集,同理可知不是极大频繁集.如此进行下去,不能找到一个元素 x_n,使得 $U\cup\{x_1,x_2,...,x_n\}$ 是极大频繁集,则可以得到无限的项目序列 $x_1,x_2,\ldots,x_n,\ldots$.又因为项目集 I 为有限集,则出现矛盾.所以,总有 $\exists W\in R$,使得 $U\subseteq W$ 成立. □

命题 1 表明,如果找出了数据库中所有的极大频繁集,其他频繁集就隐含在其中.下面讨论利用团划分思想找出极大频繁集.为了利用团的性质寻找极大频繁集,下面给出形式化描述[7].

定义 6(极大团). 对于图 $G=(V,E)$,$\exists V'\subseteq V$,如果顶点集 V' 导出的子图 $G'=(V',E')$ 是完全图,则称 G' 为图 G 中的团;如果 $\neg\exists v\in V$ 且 $v\notin V'$,顶点集 $V'\bigcup\{v\}$ 的导出子图是 $G''=(V'\bigcup\{v\},E'')$ 是完全图,则称 G' 为图 G 中的极大团.

数据库的所有的有序频繁 2-项集构成集合 $\Psi=(\eta_1,\eta_2,...,\eta_n)$,以集合 $V=\eta_1\bigcup\eta_2\bigcup...\bigcup\eta_n$ 作为图的顶点,以边的集合 $E=\{uv\,|\,u,v\in\eta_i$ 且 $\eta_i\in\Psi\}$ 构造无向图,则称图 $G=(V,E)$ 是有序频繁 2-项集的生成图.

例 2:有序频繁 2-项集 {*AB*,*AD*,*AE*,*AF*,*AG*,*AH*,*AI*,*BC*,*BD*,*BH*,*BI*,*CD*,*DE*,*DF*,*EF*,*FG*,*GH*,*HI*,*HJ*,*IJ*}.其二元关系如图 1 所示.

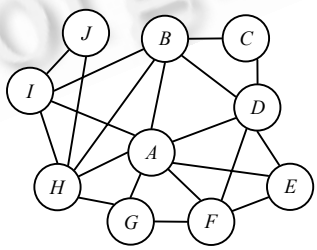

Fig.1 The generated graph of frequent 2-item sets

图 1 频繁 2-项集的生成图

命题 2. 在有序频繁 2-项集的生成图中,对于任何极大频繁 k-项集($k\ge 2$),总存在一个极大团的顶点所对应的项目集是该 k-项集的超集.

证明:在文献[6]已证明任何频繁集的子集也是频繁集,所以,极大频繁 k-项集的任何有序 2-项集也是频繁 2-项集,因此由极大频繁 k-项集所含的频繁 2-项集的生成图也一定是完全图.由定义 6 可知,任何完全图都是团;图的任何团总可能扩充成一个极大团. □

从命题 2 可知,如果找到有序频繁 2-项集的生成图的所有极大团,然后分别对极大团的顶点的对应项目所构成的集合可求出极大频繁集.下面介绍求生成图的极大团的方法.

对于非空有序频繁集 $I=i_1i_2...i_n$,引入函数: $Head(I)=i_1$; $Tail(I)=\begin{cases} i_2...i_n, & I=i_1i_2...i_n \\ \varnothing, & I=i_1 \end{cases}$.

定义 7(Head 关系). 设 F_2 为所有的有序频繁 2-项集构成的集合,二元关系 $\equiv_{\text{head}}$,称为 Head 关系,规定为 $X\equiv_{\text{head}}Y$,当且仅当 $Head(X)=Head(Y)$ 且 $X,Y\in F_2$.

命题 3. Head 关系为等价关系[9].

证明:对于 $\forall X,Y,Z \in F_2$ 且 $X \neq Y \neq Z$,

(1) 因为 $Head(X)=Head(X)$,所以 $X \equiv_{\text{head}} X$(满足自反性).

(2) 当 $X \equiv_{\text{head}} Y$ 时,则 $Head(X)=Head(Y)$,于是有 $Head(Y)=Head(X)$,$X \equiv_{\text{head}} Y$(满足对称性).

(3) 当 $X \equiv_{\text{head}} Y$, $Y \equiv_{\text{head}} Z$ 时,则 $Head(X)=Head(Y)=Head(Z)$,于是有 $X \equiv_{\text{head}} Z$(满足传递性).

综上所述,Head 关系为等价关系. □

由 Head 关系形成的类是等价类,如果等价类中的有频繁 2-项集 X,且 $Head(X)=x$,则该等价类中的频繁 2-项集的并集形成一个新的集合,用符号 $\delta_h(x)$ 表示.为了方便讨论问题,记集合中的元素为有序串.

例 3:由图 1 等价类形成如表 4 所示的有序集,ε表空串.将团的顶点的有序集构成集合簇记为Σ.

定义 8(CliqueHead 关系). 设Σ是团的顶点有序集构成的集合簇,二元关系 $\equiv_c$,简称 CliqueHead 关系,规定为 $X \equiv_c Y$,当且仅当 $Head(X)=Head(Y)$ 且 $X,Y \in \Sigma$.

命题 4. CliqueHead 关系为等价关系.

证明:用类似命题 3 的方法容易证明,在此略. □

由 CliqueHead 关系将Σ划分为多个等价类[9].当等价类中有 X 且 $Head(X)=x$ 时,用 $\psi_c(x)$ 表示等价类.

例 4:在图 1 中形成的等价类 $\psi_c(x)$ 见表 5.

Table 4 The order string is produced by $\equiv_{\text{head}}$
表 4 由$\equiv_{\text{head}}$形成的有序串

$\delta_h(x)$	Set of ∪	The order string
$\delta_h(A)$	*A,B,D,E,F,G,H,I*	*ABDEFGHI*
$\delta_h(B)$	*B,C,D,H,I*	*BCDHI*
$\delta_h(C)$	*C,D*	*CD*
$\delta_h(D)$	*D,E,F*	*DEF*
$\delta_h(E)$	*E,F*	*EF*
$\delta_h(F)$	*F,G*	*FG*
$\delta_h(G)$	*G,H*	*GH*
$\delta_h(H)$	*H,I,J*	*HIJ*
$\delta_h(I)$	*I,J*	*IJ*
$\delta_h(J)$	∅	ε

Table 5 The equivalence class of maximum clique
表 5 极大团的等价类

$\psi_c(x)$	The equivalence class of maximum clique
$\psi_c(A)$	*ABD,ABHI,ADEF,AFG,AGH*
$\psi_c(B)$	*BCD,BHI*
$\psi_c(C)$	*CD*
$\psi_c(D)$	*DEF*
$\psi_c(E)$	*EF*
$\psi_c(F)$	*FG*
$\psi_c(G)$	*GH*
$\psi_c(H)$	*HIJ*
$\psi_c(I)$	*IJ*
$\psi_c(J)$	*J*

下面首先探讨极大团的等价类之间的关系,然后利用团之间的关系和已知的极大团,找出其他的极大团.从等价类的定义可知,等价类中的团的有序串的 $Head(X)$ 值相同,为此给出如下定理:

定理 1(划分定理). 设 x 为图的顶点,求 x 将极大团的集合划分的等价类 $\psi_c(x)$ 的步骤为:(1) 求出 $\delta_h(x)$ 中的元素;(2) 对于 all $y \in \delta_h(x) \wedge y \neq x$ 递归求出 $\bigcup_{\text{all } y} \psi_c(y)$;(3) 对于 all $Z \in \bigcup_{\text{all } y} \psi_c(y)$,分别求 $\delta_h(x) \cap Z \cup \{x\}$;(4) 如有某集合是另一个的子集,则删除该子集,直到不存在这样的子集为止,得到集合簇 $\psi_c(x)$.

证明:第 1 步.因为极大团的集合以 x 划分的等价类 $\psi_c(x)$ 中的元素应是以 x 开头的团的顶点构成有序序列;构成这些有序序列的元素在 $\delta_h(x)$ 中.又因为 $\delta_h(x)$ 是以 x 开头的有序频繁集中的元素构成的集合,以 x 开头的团顶点的有序序列中的元素不出现在 $\delta_h(x)$ 的元素,就不能与 x 构成频繁集,于是证明了第 1 步.

第 2 步.因为等价类 $\psi_c(x)$ 中的有序序列的元素中的第 1 个是 x,由前面的证明可知,第 2 个元素应该是 $\delta_H(x)$ 中不等于 x 的某一个元素 y,集合中有多少个这样的元素就有多少种可能.如果该元素作为 $\psi_c(x)$ 中的某个团的序列的第 2 个元素,该序列形如 $X=xy...$,则存在序列 $Z \in \psi_c(y)$,使得 $Tail(X)=y... \subseteq Z$.假设 $Tail(X)=y...$ 序列中的元素不能构成团,则 $X=xy...$ 中的元素也不能构成团,出现矛盾,则序列 $Tail(X)=y...$ 的元素构成以 y 开头的团.该团一定是以 y 开头的极大团的序列的子序列.这就证明了第(2)步和第(3)步的正确性.第(4)步只是为了去掉不是极大团的序列. □

例 5:如果要计算 $Head(X)=B$ 的极大团,可以通过 $\forall y \in \delta_h(B)$且$y \neq B$ 中的 y,然后用 $\psi_c(y)$ 中的极大团分别与 $\delta_h(B)$ 相交而得到.这里,$\delta_h(B)=\{BCDHI\}$, $\psi_c(C)=\{CD\}$, $\psi_c(D)=\{DEF\}$, $\psi_c(H)=\{HIJ\}$, $\psi_c(I)=\{IJ\}$,则可

以求得:① $\delta_h(B)\cap\psi_c(C)\cup\{B\}=\{B,C,D\}$;② $\delta_h(B)\cap\psi_c(D)\cup\{B\}=\{B,D\}$;③ $\delta_h(B)\cap\psi_c(H)\cup\{B\}=\{B,H,I\}$;④ $\delta_h(B)\cap\psi_c(I)\cup\{B\}=\{B,I\}$.但②的集合是①的子集;式④求得的集合是式③求得的集合的子集,去掉式①和式④的集合.于是得到等价类 $\psi_c(B)=\{BCD,BHI\}$.定理 1 为计算频繁 2-项集的生成图的极大团提供了基本思路.求有序频繁 2-项集生成图的极大团的算法如下:

算法 3. *MaxClique*().

输入:有序频繁 2-项集构成数据库表.

输出:极大团的等价类.

步骤:(1) 对构成频繁 2-项集的项目逆序排列,且构成逆序集为 $\{u_n,u_{n-1},...,u_1\}$,令 $i=n$;

(2) if $i>0$ then 求出 $\delta_h(u_i)$ 中的元素且执行步骤(3) else 返回所有极大团的序列;

(3) if $\delta_h(u_i)\neq\varnothing$ then 执行(4) else $\psi_c(u_i)=\{u_i\}$,$i=i-1$,返回步骤(2);

(4) 对于 all $y\in\delta_h(u_i)\wedge y\neq u_i$,求出 $\bigcup_{\text{all }y}\psi_c(y)$;

(5) 对于 all $Z\in\bigcup_{\text{all }y}\psi_c(y)$,分别求 $\delta_h(u_i)\cap Z\cup\{u_i\}$;

(6) 如果某集合是另一个集合的子集,则删除该子集,直到不存在这样的子集为止,得到的集合簇记为 $\psi_c(u_i)$;

(7) 令 $i=i-1$,返回步骤(2).

定理 1 保证了算法 3 的正确性,但该算法主要存在两方面的缺陷:① 当边比较密集,且某些团之间相同元素过多时,会产生重复的频繁集,从而影响效率;② 如果划分粒度过小,扫描数据库的次数增多,对于海量数据库扫描,时间开销则迅速上升.为了改善性能,合并相同元素较多的极大团.

定义 9(团的相似度). 对于任意的 A 和 B 两个团,称值 $\frac{|A\cap B|}{|A\cup B|}$ 为团 A 和团 B 的相似度.显然,$0\le\frac{|A\cap B|}{|A\cup B|}\le 1$ 成立.对于给定的 $0\le\alpha\le 1$ 的常量:(1) 如果有 $\frac{|A\cap B|}{|A\cup B|}\ge\alpha$,则 A 和 B 是 α-相似;把 A 和 B 合并成一个团,该团为近似极大团;(2) 如果 $\frac{|A\cap B|}{|A\cup B|}<\alpha$,则称 A 和 B 是 α-相异;(3) 如果 $\frac{|A\cap B|}{|A\cup B|}=1$,则 A 和 B 两个团相等.

用求近似极大团算法代替算法 3,其改进算法如下:

算法 4. *NearMaxClique*().

输入:有序频繁 2-项集的数据库表,内存能装入 FP-Tree 的最大项目数为 k.

输出:近似极大团的等价类.

步骤:(1) 当项目数 $n\le k$ 时,返回项目集为近似极大团,否则进行步骤(2);

(2) 对构成频繁 2-项集的项目逆排序,且逆序集为 $\{u_n,u_{n-1},...,u_1\}$;

(3) 对于前 k 个元素 $u_n,u_{n-1},...,u_{n-k+1}$,令 $\psi_c(u_n)=\psi_c(u_{n-1})=...=\psi_c(u_{n-k+1})=\{u_{n-k+1}...u_n\}$ 且 $i=n-k$;

(4) if $i>0$ then 求出 $\delta_h(u_i)$ 中的元素且执行步骤(5) else 执行步骤(9);

(5) 对于 all $y\in\delta_h(u_i)\wedge y\neq u_i$,求出 $\bigcup_{\text{all }y}\psi_c(y)$;

(6) 对于 all $Z\in\bigcup_{\text{all }y}\psi_c(y)$,分别求 $\delta_h(u_i)\cap Z\cup\{u_i\}$;

(7) 若某集合是另一个的子集,则删除该子集,直到不存在这样的子集为止,记剩下的集合簇为 $\psi_c(u_i)$;

(8) 对于给定的常量 $0\le\alpha\le 1$,当 A和B 两个团满足 $\frac{|A\cap B|}{|A\cup B|}\ge\alpha$ 且$|A\cup B|\le k$ 时,合并 A和B,$i=i-1$,返回步骤(4);

(9) 检查所有的近似极大团,当 A和B 满足$|A\cup B|\le k$ 时,合并 A和B,且返回所有近似极大团的序列.

5 基于团的 MaxCFPTree 算法

在融合图的邻接矩阵和极大团划分思想的基础上,吸收 FP-Tree 算法优势,给出了产生频繁项集的 MaxCFPTree 新算法如下:

算法 5. *MaxCFPTree*().

输入:数据库 D,最小支持度 *minsup*.

输出:所有的频繁项集.

步骤:(1) 使用算法 2 产生频繁 2-项集;

(2) 使用算法 4 求出所有的近似极大团;

(3) for all maxclique do {对近似极大团的频繁 1-项集建立 FP-Tree;利用 FP-Tree 构建条件模式基;利用条件模式基建立条件 FP-Tree;对此产生频繁项集 F_i }[5,6]

(4) 返回 $\cup F_i$.

6 MaxCFPTree 算法性能分析

对该算法作整体上定量分析存在较大难度,因此,仅给出局部的定量分析.先给出如下定理:

定理 2(局部复杂性定理). 设数据库中有不同项目数为 n,使用的 $k\times k$ 的邻接矩阵计数:

(1) 当 $k\geq n$ 时,算法 2 需要扫描数据库次数为 1;(2) 当 $k<n$ 时,算法 2 需扫描数据库次数最多为 $\frac{\left\lceil\frac{n}{k}\right\rceil\left(\left\lceil\frac{n}{k}\right\rceil+1\right)}{2}$.

证明:(1) 根据算法 2 的步骤①,(1)显然成立.

(2) 当 $k<n$ 时,数据库中存在项目为 n 个;根据算法 2 的步骤②,把 n 个项目分成 $\left\lceil\frac{n}{k}\right\rceil$ 个子集,这 $\left\lceil\frac{n}{k}\right\rceil$ 个子集需要分别使用算法 1 扫描数据库 1 次,步骤(2)共扫描 $\left\lceil\frac{n}{k}\right\rceil$ 次.

因为步骤③取元素最少的两个集合 L_{1i} 和 L_{1j} 组成新的矩阵后,每次执行步骤③,AcrossMatrix 函数扫描数据库一次.步骤④, 当 L_{1i} 和 L_{1j} 中元素个数之和不大于 k ,则将 L_{1i} 和 L_{1j} 合并为 L_{1i},矩阵个数减少 1;

因为如果步骤③④重复 t 次,则 AcrossMatrix 函数扫描数据库 t 次,合并后矩阵个数减少 t;

所以当进入步骤⑤时,矩阵个数为 $\left\lceil\frac{n}{k}\right\rceil-t$;步骤⑤的 $\left\lfloor\frac{n}{k}\right\rfloor-t$ 个集合分别两两组合,个数为 $C^2_{\left\lfloor\frac{n}{k}\right\rfloor-t}$,每个组合扫描数据库 1 次,则扫描数据库 $C^2_{\left\lfloor\frac{n}{k}\right\rfloor-t}$ 次;步骤③~⑤共扫描数据库 $C^2_{\left\lfloor\frac{n}{k}\right\rfloor-t}+t$ 次,显然有 $C^2_{\left\lfloor\frac{n}{k}\right\rfloor-t}+t<C^2_{\left\lfloor\frac{n}{k}\right\rfloor}$ 成立;综合上述知,算法 2 需要扫描数据库次数最多为 $\left\lceil\frac{n}{k}\right\rceil+C^2_{\left\lceil\frac{n}{k}\right\rceil}$,即为 $\frac{\left\lceil\frac{n}{k}\right\rceil\left(\left\lceil\frac{n}{k}\right\rceil+1\right)}{2}$. □

算法 4 的第(8)(9)步不断合并粒度较小的团,一直合并到不能再合并为止.下面的归并收敛值域定理给出了算法 4 在给定的相似度 α 的条件下,最终产生近似极大团的数量的值域.

定理 3(归并收敛值域定理). 设数据库的项目集 V ,$|V|=n$,算法 4 产生的集合 $V_1,V_2,...,V_p$ 是 V 的覆盖,当 $\forall V_i,V_j$ 满足条件:(1) $\frac{|V_i\cap V_j|}{|V_i\cup V_j|}<\alpha$;(2) $|V_i|\leq k$,$|V_j|\leq k$ 且 $|V_i\cup V_j|\geq k$ 时,则有下列结论成立:(1) 最多存在一个 $V_i\subset V$,$|V_i|\leq\frac{k}{2}$;(2) $\frac{n}{k}\leq p\leq\frac{n(1+\alpha)}{k}$.

证明:(1) 假设存在两个 V_i,V_j 满足 $|V_i|\leq\frac{k}{2}$ 且 $|V_j|\leq\frac{k}{2}$,则 $|V_i\cup V_j|\leq k$;与条件(2)相矛盾;则(1)成立.

(2) 因为集合 $V_1,V_2,...,V_p$ 是 V 的覆盖,则 $V=V_1\cup V_2\cup...\cup V_p$.

将集合 $V_1,V_2,...,V_p$ 从左至右依次排列为完全二叉树 T 的叶节点,内部节点为运算符 $\cup$,则序列 $V_1\cup V_2\cup...\cup V_p$ 可由中序遍历完全二叉树得到.不难证明完全二叉树的高度 h,则 $p\le 2^{h-1}$.

从树的根节点开始将 T 分解两棵子树 T_1和T_2,则 $T_1=V_1\cup V_2...\cup V_q$ 和 $T_2=V_{q+1}\cup V_2...\cup V_p$.

所以 $|V|=|V_1\cup V_2...\cup V_p|=|T_1\cup T_2|=|T_1|+|T_2|-|T_1\cap T_2|$.

又因为 $|T_1\cap T_2|\le(|T_1|+|T_2|)/2$,所以 $|V|=|V_1\cup V_2...\cup V_p|\ge(|T_1|+|T_2|)/2$.

分别对子树 T_1和T_2 作同样的分解,T_1 的子树 T_{11}和T_{12},T_2 的子树 T_{21}和T_{22},则 $T_1=|T_{11}|+|T_{12}|-|T_{11}\cap T_{12}|\ge(|T_{11}|+|T_{12}|)/2$ 和 $T_2=|T_{21}|+|T_{22}|-|T_{21}\cap T_{22}|\ge(|T_{21}|+|T_{22}|)/2$ 成立.

于是得到 $|V|=|V_1\cup V_2...\cup V_p|\ge(|T_1|+|T_2|)/2\ge(|T_{11}|+|T_{12}|+|T_{21}|+|T_{22}|)/4$ …进行到所有子树有两个叶节点或只有一个叶节点的子树时,停止分解.设这些子树序列分别为 $V_1\cup V_2$,$V_3\cup V_4$,…,$V_{p-1}\cup V_p$,则有

$$|V|=|V_1\cup V_2...\cup V_p|\ge(|T_1|+|T_2|)/2\ge(|V_1\cup V_2|+|V_1\cup V_2|+...+|V_{p-1}\cup V_p|)/2^{h-2} \quad ①$$

因为 $\frac{|V_i\cap V_j|}{|V_i\cup V_j|}<\alpha$,所以 $|V_i\cap V_j|<\alpha\times(|V_i\cup V_j|)=\alpha\times(|V_i|+|V_j|-|V_i\cap V_j|)$,从而 $|V_i\cap V_j|<(|V_i|+|V_j|)\times\frac{\alpha}{1+\alpha}$,于是有

$$|V_i\cup V_j|\ge(|V_i|+|V_j|)\times\frac{1}{1+\alpha} \quad ②$$

由①②得:$|V_1\cup V_2\cup...\cup V_p|\ge(|V_1|+|V_2|+...+|V_p|)\times\frac{p}{(1+\alpha)\times 2^{h-2}}$.

由结论(1)可知,最多存在一个 V_i 满足 $|V_i|\le\frac{k}{2}$,则 $\forall V_j\ne V_i$ 满足 $\frac{k}{2}\le|V_j|\le k$;又由条件(2)可知,$\forall V_i,V_j$ 满足 $|V_i\cup V_j|\ge k$,从而得出 $|V_1\cup V_2\cup...\cup V_p|\ge\frac{kp^2}{(1+\alpha)\times 2^{h-1}}$;又由 $|V|=n$ 可知 $n\ge\frac{kp^2}{(1+\alpha)\times 2^{h-1}}$,由 $p\le 2^{h-1}$ 可知 $n\ge\frac{kp}{(1+\alpha)}$;从而推出

$$p\le\frac{n(1+\alpha)}{k} \quad ③$$

由条件(2)可知

$$\frac{n}{k}\le p \quad ④$$

成立.

综合③④,$\frac{n}{k}\le p\le\frac{n(1+\alpha)}{k}$ 成立. □

定理 4. 设数据库中的项目数为 n,邻接矩阵行列的最大值分别为 k,当内存中一次装入的项目数也为 k 时,则算法 5 扫描数据库的时间复杂性为 $O(n^2)$.

证明:由定理 2 和定理 3 容易得证(略). □

分析:MaxCFPTree 算法融合了频繁模式增长算法的优点,对 Apriori 算法的两个主要缺陷都有一定的改善:(1) Apriori 算法的每一中间过程要产生大量的候选项,MaxCFPTree 算法吸收了 FP-Tree 算法的优点,中间过程不需要产生候选项;(2) 克服了 Apriori 算法产生频繁模式需要多次遍历海量数据库的缺点.

MaxCFPTree 算法融入了极大团划分思想,将数据库的频繁 2-项集分解成多个子集,减少了扫描数据库的次数,克服了 FP-Tree 算法受限于内存的缺陷.该算法时间开销主要取决于与外存的数据交换和扫描次数.

(1) 基于极大团划分子集的方法是建立在频繁 2-项集的基础上的,使用邻接矩阵求频繁 2-项集,其时间开销主要取决于遍历数据库所使用的邻接矩阵的次数.当数据库中项目数为 n,邻接矩阵行列的最大值分别为 k 时,根据定理 2 可知,最多扫描数据库 $\frac{\left\lceil\frac{n}{k}\right\rceil\left(\left\lceil\frac{n}{k}\right\rceil+1\right)}{2}$ 次,就可以找出所有的频繁 2-项集.

(2) 算法 5 的步骤(3)改进了 FP-Tree 算法,当每个团分别扫描数据库一次就可以产生频繁模式时,步骤(3)扫描数据库的总次数与近似极大团的数目相等.定理 3 表明,取适当的 α 值,产生的子集个数 p 满足 $\frac{n}{k} \le p \le \frac{n(1+\alpha)}{k}$,则说明此阶段扫描数据库的次数最多为 $\frac{n(1+\alpha)}{k}$.

(3) 将频繁 2-项集的项目划分为若干近似极大团,在内存中需要多次循环,时间的开销与频繁 2-项集的数量以及划分的粒度有关,但相对于多次扫描百万级的数据库来讲,它不是主要因素.

MaxCFPTree 算法在不同阶段对空间的要求不同,可从如下几个阶段作定性分析:(1) 当产生频繁 2-项集时,空间的开销主要包括两部分,一部分是构造的邻接矩阵所占用的空间,另一部分用于扫描的数据库的记录所占用的空间,但不是本文讨论的重点;(2) 划分项目集为若干个子集,主要是用于生成图占用的空间,与频繁 2-项集的数量成正比;(3) 在使用 FP-Tree 算法时,FP-Tree 占用的空间取决于项目子集所含项目数和事务中所含项目平均数.

7 实验结果分析

该实验是对微软的 SQL SERVER 2000 系统的 Nothwind 数据库中的产品销售数据进行的模拟实验.其测试平台和参数如下:(1) CPU 为 PIII600;(2) 内存为 128M;(3) 操作系统使用 Window2000;(4) 对 Nothwind 数据库中的记录进行模拟扩充至约 10 万条左右,商品种类(即项目)扩充至 1 000 种左右.我们进行了如下实验:

7.1 频繁集的数目与最小支持度的关系

测试最小支持度对产生频繁集的数目的影响(不含 1-项集).在指定内存最大能装入项目数 k=200 的条件下,分别取最小支持度为 0.1,0.2,0.3,…,0.9 这 9 个不同值时的产生频繁集进行了实验.图 2 表明,频繁集的数目随着最小支持度的增加而减少.因为最小支持度越高,被淘汰的项目就越多.

7.2 最小支持度和划分粒度与运行时间的关系

在项目总数不变,内存装入最大项目数 k 分别取 k=200,k=150,k=100 的条件下,分别在最小支持度为 0.1,0.2,0.3,…,0.9 这 9 个取值的情况下测试了 MaxCFPTree 算法的时间开销,同时与 Apriori 算法、FT-Tree 算法进行了比较.图 3 表明:

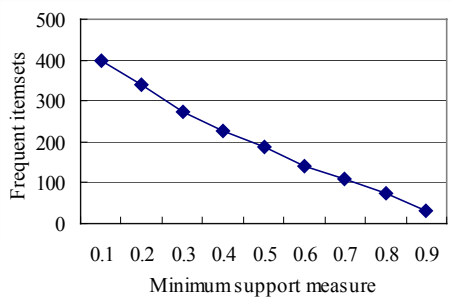

Fig.2 Minimum support measure versus frequent item sets

图 2 最小支持度与频繁集的关系

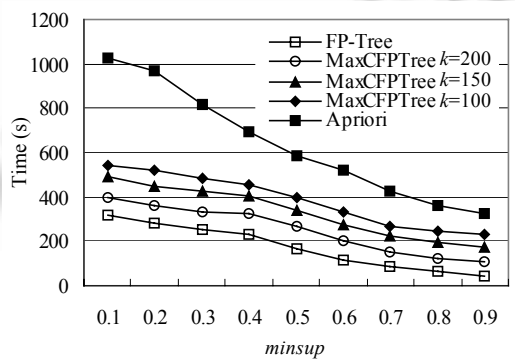

Fig.3 Support measure and granularity k versus time

图 3 支持度和划分粒度 k 对时间的影响

(1) 系统的时间开销随着最小支持度的增加而减少.因为最小支持度越高,淘汰的项目就越多.Apriori 算法产生的候选项将越来越少;MaxCFPTree 算法和 FT-Tree 算法产生的频繁 2-项集将越来越少,使处理频繁模式树和扫描数据库的次数的时间开销减少.

(2) 3 种算法的时间开销.Apriori 算法几乎对每一个候选项都要扫描数据库一次,时间开销最大;FT-Tree 算法只需扫描数据库两次就可产生所有的频繁项集,大部分工作是在内存中进行的,时间开销最小;MaxCFPTree

算法需要将项目划分多个子集用邻接矩阵求解频繁 2-项集,同时,将频繁 2-项集的项目划分为多个子集,扫描数据库的次数与这两个阶段划分的子集数相同,扫描次数比 FT-Tree 算法多,但远小于 Apriori 算法.实验说明了 MaxCFPTree 算法的可行性和有效性.

(3) MaxCFPTree 算法的时间效率.如图 3 所示,随着 k 的取值的减少,时间开销将逐渐增加.因为 k 的取值越小,邻接矩阵和近似极大团的划分粒度越小,所得到的子集数将越多,扫描数据库的次数也将越多,反之,扫描数据库次数则越少.k 的取值为 200,MaxCFPTree 算法和 FT-Tree 算法的曲线比较接近,主要原因是该实验选用的项目数不是特别大,划分的子集数不是特别多.

8 结束语

本文在用邻接矩阵求出的频繁 2-项集的基础上,融合了极大团的划分思想与 FP-Tree 算法,同时,提出并证明了两个有关扫描次数的局部复杂性定理和归并收敛域值定理.用分治策略解决了项目数量巨大而内存空间不足的矛盾,从而达到时间和空间的平衡.

References:

[1] Agrawa lR, Imielinski T, Swami A. Mining association rules between sets of items in large databases (C). In: Buneman P, Jajodia S,eds. Proc. of the ACM SIGMOD Conf. on Management of Data (SIGMOD'93). New York: ACM Press, 1993. 207~216.

[2] Agrawa lR, Srikant R. Fast algorithms for mining association rules in large databases. In: Bocca JB, Jarke M, Zaniolo C, eds. Proc. of the 20th Int'l Conf. on Very Large Data Bases. Santiago: Morgan Kaufmann, 1994. 478~499.

[3] Aly HH, Taha Y, Amr AA. Fast mining of association rules in large-scale problems. In: Abdel-Wahab H, Jeffay K, eds. Proc. of the 6th IEEE Symp. on Computers and Communications (ISCC 2001). New York: IEEE Computer Society Press, 2001. 107~113.

[4] Tsai CF, Lin YC, Chen CP. A new fast algorithms for mining association rules in large databases. In: Kamel AE, Mellouli K, Borne P, eds. Proc. of the 2002 IEEE Int'l Conf. on Systems, Man and Cybernetics (SMC 2002). IEEE Computer Society Press, 2002. 251~256.

[5] Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation. In: Chen WD, Naughton J, Bernstein PA, eds. Proc. of the 2000 ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD 2000). New York: ACM Press, 2000. 1~12.

[6] Han JW, Kember M. Data Mining. Concepts and Techniques. 2nd ed. Beijing: Higher Education Press, 2001. 240~243.

[7] Wang SH. Graph Theory and Algorithms. Hefei: University of Science and Technology of China Press, 1990. 246~250 (in Chinese).

[8] Zaki MJ. Scalable algorithms for association mining. IEEE Trans. on Knowledge and Data Engineering, 2000,12(3):372~390.

[9] Sun SL. Algebra Structure. Hefei: University of Science and Technology of China Press, 1990. 74~77 (in Chinese).

附中文参考文献:

[7] 王树和.图论及其算法.合肥:中国科学技术大学出版社,1990.246~250.

[9] 孙淑玲.代数结构.合肥:中国科学技术大学出版社,1990.74~77.