依赖ER模型的多关系频繁模式发现方法

刘 波
(暨南大学计算机科学系 广州 510632)

摘 要 为了解决多关系频繁模式挖掘面临的统计偏斜问题和效率问题,提出了基于ER(实体-联系)概念模型的方法。其以ER模型的联系集为核心,利用扩展的关系数据库SQL统计原语,在用户给定数据约束和兴趣度约束的情况下,减少多关系频繁模式的产生数量,既不需要将相关关系表做物理连接,也不会产生统计偏斜。与相关研究工作的比较,说明了利用关系数据库管理系统和ER模型实现多关系频繁模式挖掘的有效性及正确性。

关键词 数据挖掘,多关系,ER模型,频繁项集
中图法分类号 TP301 **文献标识码** A

Multi-relational Frequent Pattern Mining Method Relying on the ER Data Model

LIU Bo
(Department of Computer Science, Jinan University, Guangzhou 510632, China)

Abstract In order to solve statistical skew and efficiency problems, it presented a method based on the ER(Entity-Relationship) concept model. The method takes relationship sets of an ER model as the core, and applies expanded SQL statistical primitives of relational databases to produce multi-relational frequent patterns based on data and interestingness constraints which are provided by users, such that the patterns' number may be greatly reduced. Not only no physical joining of relational tables is needed, but also no statistical skew is produced. The comparison with some related research works explains the effectiveness and correctness of the method, which is performed by making use of the relational database management system and the ER model.

Keywords Data mining, Multi-relation, ER model, Frequent pattern

1 前言

现有的数据挖掘方法大多是针对数据库中单个关系表来实现的,将它们直接应用到多个关系表的数据挖掘存在诸多问题[1],如多表连接构造一个连接表十分费时,导致算法性能问题;数据经过连接操作之后,有些信息被放大了,有些则被缩小了,偏离了原来的数据分布状况,产生了统计上的偏斜[2]。

发现多关系频繁模式是完成关联规则挖掘、频繁子句发现、序列模式发现、时序模式发现等其他多关系挖掘任务的核心步骤[3]。已有一些关于发现多关系频繁模式(或项集)的研究工作,典型的方法有基于归纳逻辑程序设计(ILP, Inductive Logic Programming)技术的WARMR算法[4]和FARMER算法[5]、基于星型模式的JSApriori算法[6]、masl算法和masb算法[7]。这些算法存在以下问题[1]:基于ILP的方法主要面向演绎数据库,不能直接处理关系数据库,计算复杂度高,解决统计偏斜问题的方法依赖于一个新增的参数(即关键原子);基于星型模式的方法重点解决算法的性能问题,没有考虑可能出现的统计偏斜问题,因而不适合于具有普遍意义的ER(实体-联系)模型数据库。如何使用关系数据库理论和技术提高多关系数据挖掘效率,已引起国内外研究者的关注。如杨炳儒等根据关系数据库理论与技术提出了面向语义的精化多关系频繁模式发现方法[8],在先验背景知识存在的情况下,解决了结果集中语义冗余问题;文献[9,10]利用元组ID传播的思想,避免了连接操作;文献[11,12]基于树连接的方式,提高了挖掘效率。

本文与上述及其他相关工作均是研究多关系频繁模式的方法,但研究背景与思路不同,具体存在以下几方面区别:(1)本工作在用户给定数据约束、兴趣度约束以及给定的实体集存在主键的情况下,基于ER概念模型研究多关系频繁模式的发现;(2)以ER模型的联系集为核心,在联系表及所参与的实体表之间挖掘频繁项集;(3)针对多个单表的查询视图(下面称非物化视图),使用所提出的扩展SQL统计原语,产生多关系频繁模式,不会产生统计偏斜,而且利用关系数据库管理系统,可以提高挖掘效率。

2 问题描述

2.1 相关定义

许多数据库设计都采用ER数据模型,该模型反映现实世界事实的含义和相互关联,由实体集、联系集和属性等组成。可以将ER数据模型映射为关系数据库模型,每个实体集和联系集都有唯一的表与之对应,表中各列对应相关属性。

到稿日期:2011-05-15 返修日期:2011-09-16 本文受广东省科技计划项目(2010B010600026)资助。
刘 波(1965—),女,副教授,主要研究方向为信息集成、数据挖掘、人工智能,E-mail:ddxllb@163.com。

本研究的目标是依据若干实体表、联系表和用户感兴趣的属性集，挖掘满足最小支持度阈值的频繁项集。

基于多表的支持度、置信度、频繁项集等概念与基于单表的相关定义相似，但不同的算法对于支持度的定义也不尽相同[1]。有些算法将支持度定义为相对于连接表的支持度，有些定义为相对于所在表的支持度。本文采用如下的基于单表和跨表的有关定义[1]。

定义 1(项、单表项集、跨表项集) 一个表中非主键和非外键属性的每个不同取值称为一个项，记为形如 $A=v$ 的属性-值对。若干项的集合构成一个项集，若一个项集中的项全部来自于一个表，则称为单表项集；若来自于多个(大于 1 个)表，则称为跨表项集。

定义 2(单表项集支持度、跨表项集支持度) 一个项集 X 在一个表中出现的次数除以该表的总行数，称为该项集相对于该表的支持度，记为 Sup(X)。若 X 为单表项集，则相对于单表计算支持度，称为单表项集支持度；若 X 为跨表项集，则相对于连接表计算支持度，称为跨表项集支持度。

定义 3(单表频繁项集、跨表频繁项集) 用户给定一个最小支持度，支持度不小于该阈值的项集称为频繁项集。若频繁项集为单表项集，则称为单表频繁项集；若频繁项集为跨表项集，则称为跨表频繁项集。

定义 4(子跨表项集) 给定跨表项集 $I=(t_1.\text{attrset}, t_2.\text{attrset}, \cdots, t_m.\text{attrset})$，其中，$t_k.\text{attrset}$ 为属于表 t_k 的属性集。I 的子跨表项集为 $\text{Sub\_}I=(t_1.\text{subset}, t_2.\text{subset}, \cdots, t_m.\text{subset})$，其中，$t_k.\text{subset} \subseteq t_k.\text{attrset}$。

2.2 统计偏离问题

在基于单表的频繁项集发现算法中，通常利用下面两条重要的 Apriori 性质[13]。

性质 1 频繁项集的所有非空子集都是频繁的。

性质 2 如果一个项集不是频繁的，则它的所有超集也都不是频繁的。

值得注意的是，在多个单表连接得到的连接表中，部分信息可能偏离原单表数据分布，产生统计偏离问题[2]，所以上述性质对于跨表频繁项集并非总是成立的。但可以归纳以下两个性质：

(1)假设 X 是一个跨表频繁项集，且 Y 是 X 的一个非空子集，若 Y 为单表项集，则 Y 不一定是单表频繁项集；若 Y 是跨表项集(且与 X 涉及相同的关系表)，则 Y 是跨表频繁项集。

(2)假设 X 是非频繁单表项集，且 Y 是 X 的一个超集，若 Y 为单表项集，则 Y 也不是单表频繁项集；若 Y 是跨表项集，则 Y 可能是跨表频繁项集或非跨表频繁项集。

上述两点性质可以通过实例验证[1,2]。

2.3 研究目标描述

假设一个多关系数据库的 ER 模型(非扩展 ER 模型)映射为一组实体表和联系表，其中 E 表示实体表集合，R 表示联系表集合，A 表示所有实体表和联系表的属性并集，A_In 表示用户感兴趣的属性集，$Minsup$ 表示最小支持度阈值，本文研究的挖掘任务可以描述为如下：

给定：$E=\{E_1, E_2, \cdots, E_n\}$，$R=\{R_{ij} \mid E_i \in E, E_j \in E\}$，$A\_In=\{A_1, A_2, \cdots, A_m \mid A_k \in A\}$，$Minsup$；

发现：$X=\{X_1, X_2, \cdots, X_m\}$，$m \geqslant 1$，$X_i$ ($1 \leqslant i \leqslant m$) 为形如 $A_i=v_j$ 的项，$A_i \in A\_In$，$\text{Sup}(X) \geqslant Minsup$。

本文主要关注跨表频繁模式的挖掘。由于用户给定了感兴趣的属性集，因此能够限制搜索与计算空间，产生的模式长度取决于 A_In 中给定的属性数目。

3 ER 模型引导的跨表频繁模式挖掘相关概念

3.1 实例描述

为了便于分析与说明，图 1 给出了一个应用的 ER 数据模型，包括 3 个实体集(Client, Service, Employee)，两个联系集(SC, SE)；其对应的关系表如表 1－表 5 所列，包括 3 张实体表(Client, Service, Employee)，两张联系表(SC, SE)，标有下划线的属性为主键。

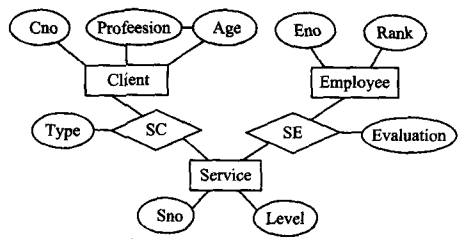

图 1 数据库实例 ER 图

表 1 Client

Cno	Profession	Age
C1	teacher	Young
C2	teacher	Middle
C3	worker	Old

表 2 Employee

Eno	Rank
E1	High
E2	Middle
E3	Middle

表 3 SC

Sno	Cno	Type
S1	C1	Vip
S2	C1	Ordinary
S3	C2	Vip
S4	C3	Vip

表 4 Service

Sno	Level
S1	A
S2	B
S3	A
S4	C

表 5 SE

Sno	Eno	Evaluate
S4	E1	Excellent
S1	E2	Good
S2	E3	Good
S3	E3	Good

3.2 ER 模型描述及相关概念

在本文考虑的 ER 模型中，实体集均有主码(即强实体集)，联系可以是二元的，也可以是多元的。在 ER 概念模型转换为数据库物理模型时，联系表一般建立两个或多个实体表的部分连接，包括参与联系的实体集的主码和联系集本身的属性。结合用户感兴趣的属性集，ER 模型对应的关系可以用实体表描述和联系表描述表示。

定义 5(实体表描述) 给定数据库 D，其实体表集 $E=\{e_1, e_2, \cdots, e_m\}$，实体表描述用 $E\_\text{Des}=\{e_1(e_1.\text{key}, e_1.\text{attrs}), e_2(e_2.\text{key}, e_2.\text{attrs}), \cdots, e_m(e_m.\text{key}, e_m.\text{attrs})\}$ 表示，其中 $e_i \in E$，$e_i.\text{key}$ 为表 e_i 的主码，$e_i.\text{attrs}$ 为用户对其感兴趣的属性集合(可为空集)。

定义 6(联系表描述) 给定数据库 D，其联系表集 $R=\{r_1, r_2, \cdots, r_p\}$，参与 r_i 的实体集 $r_i.E=\{e_1, e_2, \cdots, e_n\}$，联系

表描述用 $R_Des=\{r_1(r_1.key,r_1.attrs),r_2(r_2.key,r_2.attrs),\cdots,r_p(r_p.key,r_p.attrs)\}$表示，其中 $r_i\in R$，$r_i.key=\{e_1.key,e_2.key,\cdots,e_n.key\}$，$e_i.key(1\leqslant i\leqslant n)$表示 e_i 的主码，$r_i.attrs$ 为用户对 r_i 感兴趣的属性集合。

例 1　对于表 1—表 5 中的关系表，假设用户感兴趣的是属性集为 $A\_In=$(Client. Profession, Client. Age, SC. Type, Service. Level, SE. Evaluate, Employee. Rank)，则有

$E\_Des=$ {Client(Cno, Profession, Age), Service(Sno, Level), Employee(Eno, Rank)}

SC. $E=$ {Client, Service}

SE. $E=$ {Employee, Service}

$R=$ {SC, SE}

$R\_Des=$ {SC(Client. Cno, Service. Sno, Type),

SE(Employee. Eno, Service. Sno, Evaluate)}

ER 模型指明了一个实体集通过一个联系集能与另一个实体集相联系，同时反映实体集之间的约束以及实体集与联系集之间的约束，即映射基数和参与约束。显然，当若干个实体集和联系集之间存在这两种约束时，跨表的频繁模式才可能存在。在部分参与或全部参与的条件下，联系集可以反映出映射基数。所以，以联系集为中心引导多关系频繁模式的挖掘是一种有效途径。

定义 7(间接联系，传递属性)　给定两个联系表 r_1 和 r_2，且 $r_1.key\cap r_2.key\neq\Phi$，称 $r_1.E$ 中的实体与 $r_2.E$ 中的实体存在间接联系，记为 $r_1.E\backsim r_2.E$；$r_1.key\cap r_2.key$ 称为间接联系的传递属性。

4　跨表频繁项集的挖掘算法

4.1　相关统计原语及定理

在多关系数据挖掘研究中，某些工作利用查询图(Select Graphs)[14]获取相关统计值，加快挖掘过程。借鉴这种思想，对于实体描述集和联系描述集可以定义一系列挖掘原语，采用 SQL 语言结构进行统计操作，获取数据库中的计数信息。

(1)CountNode 原语(参数 t, attr)

Select count(t. attr)

from t

功能：按照属性 attr 的不同取值，计算单表 t 的元组数目。

(2)CountAttrs 原语(参数 t, attrs)

Select t. attrs, count(t. attrs)

from t

group by t. attrs

功能：计算单表 t 感兴趣属性集 t. attrs 的支持计数。

(3)CountStarTable 原语(参数 r)

Select count(*)

from r, $r.E$

where r. JoinCon

功能：从联系表 r 和与它联系的实体表，计算满足 r. JoinCon 的元组数目，其中 $r.E$ 为参与联系 r 的实体表集，r. JoinCon 为查询条件或 r 与 $r.E$ 虚拟连接的条件(由联系表与实体表之间语义相同的属性对构成)。

(4)CountMultiItem 原语(参数 I, r)

Select count(*)

from r, $r.E$

where r. JoinCon, I

功能：从联系表 r 和与它联系的实体表，计算项集 I 的支持计数，其中 $r.E$ 为参与联系 r 的实体表集，r. JoinCon 为查询条件或 r 与 $r.E$ 虚拟连接的条件。

定理 1　从一个联系表与参与它的实体表中，以实体表主键作为连接属性，进行连接查询，产生的非物化视图不会产生统计偏斜。

证明：令联系 r 用(r. key, r. attrs)表示，其中 $r.key=\{e_1.key,e_2.key,\cdots,e_n.key\}$，$e_i.key(1\leqslant i\leqslant n)$表示参与联系 r 的实体表 e_i 的主码，r. attrs 为 r 的非主码属性集合，则以 $e_1.key,e_2.key,\cdots,e_n.key$ 为连接属性，r 与 $e_1,e_2,\cdots,e_n$ 连接查询产生的非物化视图 V 的模式为(r. key, r. attrs, e_1. attrs, e_2. attrs, $\cdots$, e_n. attrs)，其中 $e_i.attrs(1\leqslant i\leqslant n)$表示实体表 e_i 的非主码属性集合。

因为 r. key 对应唯一的 r. attrs，(e_1. attrs, e_2. attrs, $\cdots$, e_n. attrs)，所以 r. key 也是 V 的主码，V 的实例表没有放大与缩小信息，也就不会产生统计偏斜。

定理 2　若两个联系集 r_1 和 r_2 之间存在传递属性 x，且 $x=r_1.key$ 或 $x=r_2.key$，则 r_1 与 r_2 之间以 x 为连接查询属性，产生的非物化视图不会产生统计偏斜。

证明：对于联系 $r_1(r_1.E,r_1.attr_1,r_1.attr_2,\cdots,r_1.attr_k)$ 和 $r_2(r_2.E,r_2.attr_1,r_2.attr_2,\cdots,r_2.attr_k)$，由于 r_1 和 r_2 之间存在传递属性 x，根据定义 7，x 为 r_1 或 r_2 的主键，因此以 x 为连接查询属性产生的非物化视图不会放大与缩小信息，也就不会产生统计偏斜。

4.2　算法描述

ER 图表现了实体集之间的联系语义，依据定理 1 和定理 2，在挖掘跨表频繁项集中，为保证跨表频繁项集的统计没有失真，采用下面两个关键步骤：

(1)在一个联系表 r 与参与它的实体表之间，进行统计查询，得到频繁项集 $F-r$；

(2)对于两个联系表 r_1 和 r_2，若 $r_1.E\backsim r_2.E$，且存在传递属性 $z=r_1.key\cap r_2.key$：

①如果 $z=r_1.key$ or $z=r_2.key$，则针对两个联系表进行统计查询，检测$\{x,y\}$是否频繁，其中 $x\in F-r_1$，$y\in F-r_2$，$F-r_1$和 $F-r_2$ 分别是 r_1 和 r_2 的频繁项集；

②若 $z\neq r_1.key$，且 $z\neq r_2.key$，但 CountNode(r_1, $r_1.key$)=CountNode(r_1, z)或 CountNode(r_2, $r_2.key$)=CountNode(r_2, z)，也针对两个联系表进行统计查询，检测$\{x,y\}$是否频繁，其中 $x\in F-r_1$，$y\in F-r_2$。

算法 ERMiner 基于上述策略及联系表反映的实体间强语义连接，利用有关 SQL 统计原语，发现跨表频繁项集。其描述如下。

算法 1　ERMiner

输入：$E=\{e_1,e_2,\cdots,e_m\}$，实体表集合；

$e_i.key,e_i.attrs$，实体表 $e_i(1\leqslant i\leqslant m)$的主键及用户对其感兴趣的属性集；

R={r_1,r_2,…,r_k},联系表集;

r_1.E,r_2.E,…,r_k.E,r_1,r_2,…,r_k 联系表的实体表集合;

r_i.key,r_i.attrs,联系表 r_i($1\leqslant i\leqslant k$)的主键及用户对其感兴趣的属性集;

J={(t.a,s.b)|t∈E,s∈R,a 为 t 的主键,b 属于 s 的主键属性},实体表与联系表之间语义相同的属性对集合;

minsup,最小支持计数阈值。

输出:跨表频繁项集集合 F。

方法:

```
(1)F=Φ;
(2)for R 中每一联系集 r do
(3)  r.JoinCon={(t.a=r.b)∈J|t∈r.E};
(4)  List={(x,y)|x∈e.attrs,e⊆r.E,y∈r.attrs};
(5)  ComFreSets(List,r); //调用 ComFreSets 过程,计算出 List
     中的跨表频繁项集
(6)Endfor;
(7)while(|R|)>1) do //R 包括 1 个以上联系集
(8)  R′=Φ;
(9)  Combine(R,F);//调用 Combine 过程,合并联系,计算 F
(10) if(R′≠ Φ and R′≠R) R= R′ // R′为新产生的联系集集合
(11)  else break;
(12) Endwhile;
(13) Output(F); //输出 F
```

4.3 算法主要过程分析

算法 ERMiner 调用的主要过程分析如下。

(1)ComFreSets(L,r)

该过程的功能是:求联系表 r 及与它相联系的实体集之间的跨表频繁项集,其中 $L=\{(x,y)|x\in e.\text{attrs},e\subseteq r.E,y\in r.\text{attrs}\}$,其描述如下:

```
ComFreSets(L,r) {
  for L 中的每一跨表项集 I
    if CountMultiItem(I,r)/CountStarTable( )≥minsup; //检测 I
是否频繁
    { F-r=F-r∪I; F=F∪F-r;}
  Endfor;
}
```

ComFreSets 过程可以利用跨表频繁项集的子跨表项集也频繁的性质,对 L 中的跨表项集按长度由大到小的顺序检测其是否频繁,若某一跨表项集频繁,则在 L 中未检测的其子跨表项集就不需要检测了,它们一定频繁。

复杂度分析:设参与联系 r 的实体集为 $e_1,e_2,\cdots,e_m$,对于联系表和各实体表,用户感兴趣的属性取值数目分别为 $n_r,n_1,n_2,\cdots,n_m$(若对某实体表或联系表无感兴趣的属性,则令其值为 1),过程 ComFreSets 的复杂度为 $O(n_r*n_1*n_2*\cdots*n_m)$。

(2)Combine(R,F)

该过程的功能是合并 R 中的联系集,并计算新联系集及与它联系的实体集之间的跨表频繁项集,其描述如下:

```
Combine(R)
 {for R 中每两个联系集 r1 与 r2 do
   if(r1.key∩r2.key=r1.key or r1.key∩r2.key=r2.key or Count-
     Node(r1,r1.key)=CountNode(r1,r1.key∩r2.key) or Count-
     Node(r2,r2.key)= CountNode(r2,r1.key∩r2.key))
       //满足两个联系合并的条件
     {r.key= r1.key∪r2.key; //创建新的联系 r
     r.attrs= r1.attrs∪r2.attrs;
     r.E= r1.E∪r2.E;
     r.JoinCon={(t.a=r.b)∈J|t∈r1.E∪r2.E};
   R′= R∪r-r1-r2; // R′为新的联系集集合
   F-r={I=(x,y)|x∈F-r1,y∈F-r2,CountMultiItem(I,r)/
        CountStarTable(r)≥minsup};
   F=F∪F-r;
  }
}
```

假设 ER 模型的联系集数目为 m,则 Combine 过程的复杂度为 $O(m^2)$。

4.4 算法正确性证明

基于 ER 模型,在实体集与联系集之间存在主外键约束的情况下,算法正确性证明如下。

(1)在计算跨表项集支持度时,实体表与联系表之间以 $r.\text{JoinCon}=\{(t.a=r.b)\in J|t\in r.E\}$ 为连接查询条件,其中 a 为 t 的主键,按照定理 1,不会产生统计偏斜。

(2)两个联系表合并,分为以下情况:

①在满足 $r_1.\text{key}\cap r_2.\text{key}=r_1.\text{key}$ or $r_1.\text{key}\cap r_2.\text{key}=r_2.\text{key}$ 的条件下,根据定理 2,以 $r_1.\text{key}\cap r_2.\text{key}$ 作为连接查询条件,在实体表与联系表之间产生的非物化视图中不会产生统计偏斜。

②若 $r_1.\text{key}\cap r_2.\text{key}\neq r_1.\text{key}$,$r_1.\text{key}\cap r_2.\text{key}\neq r_2.\text{key}$,并且 $r_1.\text{key}\cap r_2.\text{key}$ 非空,CountNode(r_1,r_1.key)= CountNode(r_1,$r_1.\text{key}\cap r_2.\text{key}$)或 CountNode($r_2$,$r_2$.key)=CountNode($r_2$,$r_1.\text{key}\cap r_2.\text{key}$),则可将 $r_1.\text{key}\cap r_2.\text{key}$ 视为 r_1 或 r_2 的主键作用一样,以其作为连接查询条件产生的非物化视图也不会产生统计偏斜。

同理,由 r_1 和 r_2 合并为新的联系表 r 后,$r.\text{JoinCon}=\{(t.a=r.b)\in J|t\in r_1.E\cup r_2.E\}$,$a$ 为 t 的主键,按照定理 1,以 r.JoinCon 为连接查询条件,在实体表与新联系表之间产生的非物化视图也不会产生统计偏斜。

所以算法在不产生统计偏斜的情况下,依赖联系集与实体集的非物化视图计算得到的频繁项集正确。

5 实例与实验分析

例 2 根据图 1 的 ER 图及 3.2 节例 1 中实体表与联系表的描述,结合表 1—表 5,假设 $minsup=50\%$,求跨表频繁项集。

计算过程说明如下:

(1)在联系表 SC 与参与它的实体表之间挖掘,得到频繁项集{Profession=teacher,Type=Vip,Level=A}及其子跨表项集。

(2)在联系表 SE 与参与它的实体表之间挖掘,得到频繁项集{Level=A,Evaluate=Good,Rank=Middle}及其子跨表项集。

(3)由于 SC.key∩SE.key={Sno},CountNode(SC,SC.key)= CountNode(SC,Sno),因此可以产生一个新的联系集 SC-SE,进一步检测上面第(1)和第(2)步得到的频繁项集及其子集的组合是否频繁,从而得到最终频繁项集{Profession=

teacher, Type=Vip, Level=A, Evaluate=Good, Rank=Middle}及其子跨表项集。

在 Window XP、Oracle9.2 数据库管理系统、CPU 为 Intel Pentium Dual Core 的微机环境下，采用 PL/SQL 创建过程，进行了模拟实验。

Client, Service, Employee, SC, SE 等表记录数分别为 50000, 50000, 2000, 50000, 50000，感兴趣的属性为 Client. Profession, Client. Age, SC. Type, Service. Level, SE. Evaluate, Employee. Rank，在不同的支持度阈值 *minsup* 下，测得 ERMiner 算法平均运行时间如表 6 所列。同时，以 Cno, Sno, Eno 为连接属性，产生 Client, SC, Service, SE, Employee 的连接视图（仅包含感兴趣的属性），采用经典的 Aprior 算法[13]计算它的频繁项集，表 6 中将这种方法标为 Aprior_based。由于通过相关表的主键连接，因此该方法产生的连接视图没有产生统计偏斜，得到与 ERMiner 算法相同的结果，但平均运行时间不同。

分析表 6 的算法平均运行时间，可以得出如下结论。

表 6　模拟实验算法运行时间

	minsup	0.005	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6
ERMiner	time(ms)	516	531	547	558	563	796	1032	1031	1016
Aprior_based	time(ms)	1297	1276	1251	1245	1203	1078	734	718	703

(1) 当 $minsup \leqslant 0.3$ 时，ERMiner 算法的时间效率高于 Aprior_based 算法的时间效率；

(2) 当 $minsup > 0.3$ 时，ERMiner 算法的时间效率低于 Aprior_based 算法的时间效率；

(3) 当 $minsup \leqslant 0.4$ 时，随着 *minsup* 的增大，ERMiner 算法的平均运行时间增加，而再随着 *minsup* 的增大，如 $minsup=0.5$、0.6 时，算法的平均运行时间逐渐减少；不同地，随着 *minsup* 的增大，Aprior_based 算法的平均运行时间一直是逐渐减少的。

究其原因，ERMiner 算法首先考虑联系表及其相关实体视图的频繁项集，若不存在结果，才考虑视图的子集；而 Aprior_based 算法先考虑 2-频繁项集，若存在结果，再考虑 3-频繁项集、4-频繁项集等。因此，ERMiner 算法随着 *minsup* 的增大，当联系表及其相关实体的视图不存在频繁项集时，需要更多的时间，但 *minsup* 增大到某一值时，由于频繁项集数目的减少，平均运行时间转为减少。

综上所述，*minsup* 越小，产生的频繁项越多时，ERMiner 算法越有优势。

结束语　在前言中已提到过一些关于多关系频繁项集挖掘的研究工作。但本研究工作与相关研究的背景不同，其基于用户给出的感兴趣的数据属性和 ER 模型，不仅简化了结果集，而且解决了效率和统计偏斜问题。其主要特色如下：面向关系数据库，提出了 ERMiner 算法，利用不同关系实体之间的直接联系、间接联系，采用扩展 SQL 的统计原语计算跨表项集支持度，获得较高的频繁项集挖掘效率，尤其适合于最小支持度较小、产生频繁项较多的情况；而且，基于具有普遍意义的 ER 模型，避免了实体表与联系表之间的物化连接操作以及统计偏斜。

参 考 文 献

[1] 何军，刘红岩，杜小勇. 挖掘多关系关联规则[J]. 软件学报，2007，18(11)：2752-2765

[2] Getoor L. Multi-relational data mining using probabilistic relational models: Research summary[C]// Proceedings of the 1st Workshop in Multi-relational Data Mining. 2001：6-17

[3] 张伟，杨柄儒，钱榕. 多关系频繁模式发现研究[J]. 计算机科学，2007，34(7)：158-164

[4] Dehape L, De Raedt L. Mining association rules in multiple relations[C]// Proceedings of the 7th Int'l Workshop on Inductive Logic Programming. Berlin: Springer-Verlag, 1997：125-132

[5] Nijssen S, Kok J. Faster association rules for multiple relations [C]// Proceedings of the 17th International Joint Conference on Artificial Intelligence. USA: Morgan Kaufmann, 2001：819-896

[6] Jense V C, Soparkar N. Frequent itemset counting across multiple tables[C]// Proceedings of the 4th Pacific-Asia Conference of Knowledge Discovery and Data Mining, Current Issues and New Applications. Berlin: Springer-Verlag, 2000：49-61

[7] Ng E K K, Fu A W, Wang K. Mining association rules from stars[C]// Proceedings of the 2002 IEEE International Conference on Data Mining. Maebashi. Washington: IEEE Press, 2002：322-329

[8] 杨炳儒，张伟，钱榕. 面向语义的精简化多关系频繁模式发现方法[J]. 中国工程科学，2008，10(9)：47-53

[9] 邓左祥，刘连芳，梁一平，等. 一种多关系频繁模式挖掘算法[J]. 计算机应用研究，2009(9)：3285-3288

[10] 郭景峰，吕庆春，李霞. 一种新型基于用户指导的多关系关联规则挖掘算法[J]. 微计算机信息，2009(24)：124-126

[11] Jiménez A, Berzal F, Cubero J C. Frequent Itemset Mining in Multirelational Databases[C]// Foundations of Intelligent Systems, 18th International Symposium. Lecture Notes in Computer Science, Vol. 5722, Berlin: Springer-Verlag, 2009：15-24

[12] Hou Wei, Yang Bing-ru, Xie Yong-hong, et al. Mining Multi-relational Frequent Patterns in Data Streams[C]// International Conference on Business Intelligence and Financial Engineering. Washington: IEEE Press, 2009：205-209

[13] Han Jia-wei, Kamber M. 数据挖掘概念与技术(第 2 版)[M]. 范明，孟小峰，译. 北京：机械工业出版社，2007：146-152

[14] Knobbe A J, Blockeel H, Siebes A, et al. Multi-relational Data Mining[C]// Proceedings of Benelearn'99. 1999