第35卷 第12期 计 算 机 工 程 2009年6月
Vol.35 No.12 Computer Engineering June 2009
·软件技术与数据库· 文章编号：1000—3428(2009)12—0059—02 文献标识码：A 中图分类号：TP311

基于移项的隐私保护关联规则挖掘算法

李 霞，陈子军，吕庆春
(燕山大学信息学院计算机科学与工程系，秦皇岛 066004)

摘 要：基于启发式规则的隐私保护关联规则挖掘算法中均通过删除项或增加项改变规则的支持度，对非敏感规则的支持度影响很大。针对上述不足，提出一种将删除项和增加项2种操作相结合的方法，在执行删除项操作后寻找合适的事务，对该事务执行增加项操作。实验结果表明，利用该算法清洗数据库所产生的规则丢失率和相异度均有所下降。

关键词：关联规则；隐私保护；数据挖掘

Privacy Preserving Association Rule Mining Algorithm Based on Item-moving

LI Xia, CHEN Zi-jun, LV Qing-chun
(Department of Computer Science and Engineering, Information Institute, Yanshan University, Qinhuangdao 066004)

【Abstract】All the heuristic approaches are realized by deleting an item or inserting an item, of which the negative effect on non-restricted rules is too much. Focusing on the shortcoming, this paper presents an algorithm combining those two operations, which insert an item into proper transaction after deleting that item. Experimental results show that privacy preserving data mining algorithm based on item-moving has lower miss rate and dissimilarity.

【Key words】association rule; privacy preserving; data mining

1 概述

近年来，隐私保护的数据挖掘成为数据库研究领域的一个热点。其中，关联规则中的规则隐藏更是被广为研究。关联规则数据挖掘中的规则隐藏就是在进行关联规则挖掘之前把数据拥有者不希望被数据使用者挖掘出来的敏感规则进行隐藏，但保证其他非敏感规则仍然能被挖掘出来[1]。

Algo1a[2], Algo1b[2], Algo2a[2], Algo2b[3], Algo2c[3], Naïve[4], MinFIA[4], MaxFIA[4], IGA[4], RRA[5], RA[5], SWA[6]等基于启发式规则的算法先后被提出。文献[2-3]假设所要隐藏的不同的敏感规则没有交集项，每次选择一个敏感规则，通过降低支持度或置信度隐藏该规则。文献[4-6]引入了冲突度的概念，通过降低支持度实现规则隐藏，可以隐藏有交集项的敏感规则集。然而，基于启发式规则的隐私保护算法均采用删除项或者增加项的方式减小敏感规则支持度，对非敏感规则支持度的影响很大，而在这些算法中，SWA算法呈现出相对较好的性能[7]。

为了减小对非敏感规则的影响，本文对SWA算法[6]做出改进，提出了一种基于移项的隐私保护关联规则挖掘算法，通过先删除项后插入项的方法降低对非敏感规则支持度的影响，在不增加虚假模式的前提下减小了非敏感规则的丢失率和相异度。

2 术语定义

文献[1]中用到的一些基本定义如下：

定义 1 敏感事务：数据库中至少支持一个敏感规则的事务。

定义 2 事务的敏感项：该事务支持的敏感规则中所包含的项。

定义 3 事务长度：事务包含的项的个数。

定义 4 规则丢失率(miss cost)：在隐藏敏感项集后合法的关联规则被隐藏的比例。

定义 5 相异度(dissimilarity)：清洗数据库后数据库中项的频度变化占总数据项频度之和的比例。

定义 6 虚假模式率(artifactual pattern)：在清洗数据库后引入的、原数据库不存在的频繁项集。

定义 7 公开度(disclosure)：用来衡量敏感项集所受保护的程度，具体指敏感事务可被用户发现的数量占敏感项集支持数的阈值。

本文提出下面的定义：

定义 8 移项候选事务：事务T是T1不包含I项的真子集，若把事务T1的I项移至T，可以使敏感项集R1支持数减1，则事务T就是事务T1的、项I针对R1的一个移项候选事务。

定义 9 最佳移项候选事务：若多个事务T2是事务T1的移项候选事务，则长度与T1最接近的事务为事务T1的项I对R1的最佳移项候选事务。

3 IMBA算法

3.1 算法思想

在SWA算法中，采用删除项的方法使规则的支持度降到最小阈值以下，这样就认为敏感规则得到了隐藏。假设事务T1={a}∪B，a为事务中的一个项，B为一个项集，R1为和T1相关的敏感项集，则R1∈{Rr｜Rr={a}∪C，C为一项集且C⊂B}。

作者简介：李 霞(1984－)，女，硕士，主研方向：数据挖掘；陈子军，副教授、博士；吕庆春，硕士

收稿日期：2008-10-09 **E-mail：**zjchen@ysu.edu.cn

当SWA算法通过删除a项减小敏感项集R1的支持度时，会使所有非敏感项集{Rn｜Rn⊂T1 且 a∈Rn}的支持数都减小，当T1是一个长事务时，支持数减小的非敏感项集Rn会很多，导致对非敏感项集支持度的影响很大。

本文提出了用移项的方式减小项集 R1 的支持度，即寻找T1的项a针对R1的移项候选事务T2，根据定义有T2⊂B，把 a 从事务 T1 移至 T2，此时由于 a 与 T2 的结合，{Rn｜Rn⊂{a}∪C}中所有非敏感项集的支持数保持和清洗前一致。当C和B的项的个数相差最小，即事务T1和T2长度相差最小时，R1 支持数的减小对非敏感项集支持数的影响降至最低。当 T1 和 T2 的长度相差为 2 时，只有项集{IS｜R1⊂IS}支持数减小，而非敏感项集的支持度均不会受到影响。

例如，在数据库{T1: ABCD, T2: CD, T3: ABCDE, T4: BD}中，把敏感规则 AB 的支持度降低为小于等于阈值 25%，则按照 SWA 算法操作时，会选择事务 T1：ABCD 删除 A 项，这样规则 AB 支持度降低为 25%，从而得到隐藏。此时衡量删除项 A 所影响的模式有 AB, AC, AD, ABC, ABD, ACD, ABCD，由于删除了 A 项，因此 AC, AD, ACD 这 3 个非敏感项集支持数均减小了 1。其中，减小规则 AB 的支持数是数据处理的目的，但不希望影响到其他非敏感项集支持数。

本文提出用移项的方法减小规则 AB 的支持度，选择把T1中的A项移动至T2中，此时规则AB的支持度降为25%，而在包含 AB 的项集支持数减 1 的同时，其他项集支持度均未受到影响。其中，T2 是 T1 的项 A 针对敏感项集 AB 的最佳候选移项事务，当 T2 与 T1 的长度恰好小 2 时，能使移项时支持数受影响的规则最少。

3.2 算法描述与正确性证明

IMBA(Rr[1..n], Disclosure[1..n],D,K)

输入 敏感项集 Rr[1..n]，Disclosure[1..n]，数据库 D，窗口大小 *K*

输出 清洗后的数据库 D′

```
begin
while(D 中还有未处理过的事务)
begin
s := Read(D,K,Transaction[1..K]);
sup(Rr[1..n],Transaction[1..s],support[1..n]);
computeDelNum(DelNum[1..n],s,disclosure[1..n]);
totalDelNum := sum(DelNum[1..n]);
while(totalDelNum>0)
begin
t := findShortestTransaction(Transaction[1..s]);
I := mostFrequentItem(t,Rr[1..n]);
t2 := searchBestTrasaction(t,I);
move(t,t2,I);
end
end
end
```

其中，Read()函数读取 *K* 条事务并返回 *K*，若数据库中未处理事务不足 *K* 条，则返回读取事务的数量；sup()函数计算数据库中敏感规则的支持数；computeDelNum()函数根据敏感规则的支持数和公开度计算每个敏感规则需要减小的支持数；findShortestTransaction()函数查找最短事务；mostFrequentItem()函数查找该事务中支持敏感规则最多的项；searchBestTrasaction()寻找最佳移项候选；move()函数执行移项操作。

下面通过定理证明算法的正确性。

定理 IMBA 算法不会产生虚假模式。

证明 假设 B1, B2 为 2 个项集，Rr 为 1 个敏感项集，事务 T1={a}∪B1，事务 T2=B2，a∈Rr，敏感项集 Rr⊂T1，且 B2⊂B1，则事务 T2 是事务 T1 的项 a 针对 Rr 的移项候选事务。将 a 项从事务 T1 移至事务 T2 产生事务 T1′, T2′，因此，需证明没有因为项 a 的移动导致某个项集的支持数增大。

(1)假设任意项集 IS 在事务 T1, T2 中的支持数 sup(IS)=2，则 sup(IS)≤2=sup′(IS)，命题成立。

(2)假设 sup(IS)=1，由于 T2⊂T1，因此有 IS⊂T1 且 IS⊄T2。

1)若 a∈IS，因为 IS⊄T1′，所以 sup′(IS)≤1；

2)若 a∉IS，因为 IS⊄T2，所以 sup′(IS)≤1。

因此，sup(IS′)≤sup(IS)，命题成立。

(3)假设 sup(IS)=0，则 IS⊄T1，由于 T2⊂T1，得到 IS⊄T1∪T2 和 IS⊄T1′∪T2′，因此有 sup(IS′)=0≤sup(IS)。

综上可知，IMBA 算法对数据进行处理后不会产生虚假模式。

4 实验性能分析

为了对算法进行性能测试，将 IMBA 算法与 SWA 算法的规则损失率和相异度进行了对比。实验用 Java 语言实现，采用 Eclipse 平台，操作系统为 Windows XP，CPU 为 Pentium4 1.5 GHz，内存为 512 MB，数据集包含 50 个项和 10^5 条事务，敏感项集项数为 2 个~5 个，敏感项集为 3 个~6 个。而在 2 个算法操作的过程中，非敏感模式支持数减少所造成的规则损失率和相异度如图 1、图 2 所示。

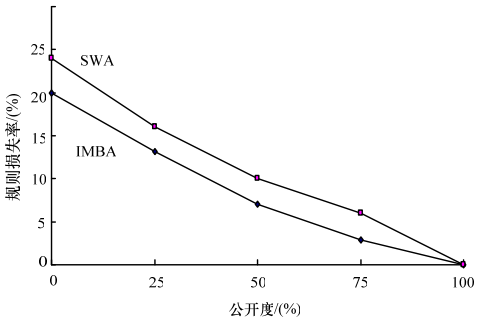

图 1 SWA 和 IMBA 的规则丢失率对比

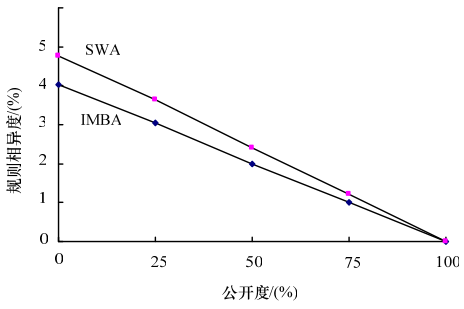

图 2 SWA 和 IMBA 的规则相异度对比

对比 2 张图可知，由于 IMBA 在减小敏感项集支持数的过程中对非敏感项集所受的负面影响做出了补偿，因此数据库的质量得到了提高。

5 结束语

本文在 SWA 算法的基础上提出利用移项的方法进行敏感知识隐藏，减小了非敏感模式在此过程中所受到的影响，并通过实验验证了本算法的有效性。

(下转第 63 页)