第 47 卷第 1 期
2015 年 2 月

南 京 航 空 航 天 大 学 学 报
Journal of Nanjing University of Aeronautics & Astronautics

Vol. 47 No. 1
Feb. 2015

DOI:10.16356/j.1005-2615.2015.01.017

一种改进的隐私保护关联规则挖掘算法

顾 铖 朱保平 张金康

(南京理工大学计算机科学与工程学院,南京,210094)

摘要:部分隐藏的随机化回答方法是基于关联规则数据挖掘的隐私保护算法,针对该算法在重构频繁项集支持度上的指数级时间复杂度导致算法执行效率下降的不足,采用分治策略和集合运算方法对该算法进行改进,消除重构数据的指数级运算。改进算法降低了算法的时间复杂度并有效提高了执行效率。仿真实验与分析表明了改进算法的有效性。

关键词:关联规则;隐私保护;分治策略;集合运算

中图分类号:TP311 **文献标志码**:A **文章编号**:1005-2615(2015)01-0119-06

Improved Algorithm of Privacy Preserving Association Rule Mining

Gu Cheng, *Zhu Baoping*, *Zhang Jinkang*

(School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, 210094, China)

Abstract: Randomized response with partial hiding is a privacy preservation algorithm based on association rule mining. For the insufficient that the exponential time complexity of the algorithm in the reconstruction of frequent item set support reduces the execution efficiency of the algorithm, the divide and conquer strategy and the set operations are used to improve the algorithm, and the exponential computation in the reconstruction is eliminated. The improved algorithm can reduce the time complexity and promote the efficiency effectively. Experiments and analysis indicate the effectiveness of the improved algorithm.

Key words: association rules; privacy preservation; divide and conquer strategy; set operations

近年来,随着信息技术的高速发展,针对海量数据的管理与分析,数据挖掘技术的应用发挥了非常积极的作用。与此同时,隐私数据的保护问题逐渐呈现,例如医院患者的病历挖掘,可以发现各种疾病之间的关联,但在挖掘过程中也很容易暴露病人的隐私。类似的情况还有很多,所以隐私保护数据挖掘已经成为数据挖掘应用研究的重点[1]。

隐私保护数据挖掘常用的基本策略为数据干扰[2]和查询限制[3]。数据干扰就是通过对原始数据进行变换、离散化和添加噪声等方法进行干扰,从而保护隐私数据,再根据干扰后的数据进行挖掘,得到所需的模式、模型和规则;查询限制就是通过对原始数据进行隐藏、抽样和划分等方式,避免数据挖掘者获得完整的原始数据,然后数据挖掘者通过概率统计或者分布式计算的方法得到挖掘的模式和规则。但是,这两种隐私保护策略都存在自身的不足。数据干扰策略中,所有干扰的数据均与真实的原始数据直接相关;而查询限制策略中,数据挖掘者得到的数据都是真实的数据,这些都会降低对隐私数据的保护程度。另外,针对分布式数据挖掘[4]中,通常采用安全多方计算[5]的方法,对数据进行隐私保护。目前又有一种通过建立差分隐私[6]保护模型对隐私数据进行保护的方法,这也是目前研究的重要方向之一。

收稿日期:2014-08-29;**修订日期**:2014-10-22

通信作者:朱保平,男,副教授,E-mail:zbp2068@126.com。

关联规则[7]数据挖掘是数据挖掘的一个重要研究领域,在其相关的隐私保护取得了一定的研究成果。在数据干扰方面,文献[8]等提出了一种基于随机变换的 MASK 方法。该方法通过数据干扰和分布重构实现了隐私保护的关联规则挖掘。虽然该方法很好地兼顾了高度的隐私和准确的挖掘结果,但运行的时间效率较低。所以基于 MASK 算法的改进优化算法相继提出。文献[9]提出了改进的 EMASK 算法,该算法的改进非常有效,但是重构原始数据项的支持度是指数级运算,仍然影响执行效率。

本文主要针对部分隐藏的随机化回答方法(Randomized response with partial hiding, RRPH)[10]在重构项集支持度上指数级时间复杂度不足进行改进,采用分治策略[11]与集合运算的方法,消除重构数据的指数级运算,提高算法的执行效率。

1 RRPH 算法

1.1 数据歪曲过程

对于 0-1 序列的事务集,给定随机化参数 p_1, p_2, p_3,其中 $0 \leqslant p_1, p_2, p_3 \leqslant 1$,且 $p_1 + p_2 + p_3 = 1$。对于事务集中的项 $a \in \{0,1\}$,设 $r_1 = a$, $r_2 = 1 - a$, $r_3 = 0$,定义随机函数 $r(a)$,函数以 p_j 的概率取值 r_j, $j = 1,2,3$。设事务集中项目的总个数为 k,则原始事务 $A = (a_1, a_2, \cdots, a_k)$ 干扰后转变为事务 $B = (b_1, b_2, \cdots, b_k)$ 可以通过 $B = R(A)$ 计算得到,其中 $b_i = r(a_i)$。即 b_i 以 p_1 的概率取值 a_i,以 p_2 的概率对 a_i 取反,以 p_3 的概率取值为 0。

1.2 项集支持度重构

RRPH 算法的数据集是根据真实数据集变换形成,所以需要通过重构项集获得真实支持度。设原始事务集 T,经过概率变换后的事务集为 D,对于事务集中任意一个项 c, c_0^T 和 c_1^T 分别表示项 c 在 T 中 0 和 1 的个数, c_0^D 和 c_1^D 分别表示项 c 在 D 中 0 和 1 的个数。由 p_1, p_2, p_3 的概率变换得方程组: $\begin{cases} p_1 c_1^T + p_2 c_0^T = c_1^D \\ (p_2 + p_3) c_1^T + (p_1 + p_3) c_0^T = c_0^D \end{cases}$,所以 $\boldsymbol{C}^T = \boldsymbol{M}^{-1} \boldsymbol{C}^D$,其中 $\boldsymbol{M} = \begin{bmatrix} p_1 & p_2 \\ p_2 + p_3 & p_1 + p_3 \end{bmatrix}$, $\boldsymbol{C}^T = \begin{bmatrix} c_1^T \\ c_0^T \end{bmatrix}$, $\boldsymbol{C}^D = \begin{bmatrix} c_1^D \\ c_0^D \end{bmatrix}$, $\boldsymbol{M}^{-1}$ 为 $\boldsymbol{M}$ 的逆矩阵,同时可求出 1-项集的支持度 $c_1^T = \dfrac{c_1^D - p_2}{p_1 - p_2} (p_1 \neq p_2)$。

k-项集支持度重构的方法与 1-项集类似,具体方法如下

通过等式 $\boldsymbol{C}^T = \boldsymbol{M}^{-1} \boldsymbol{C}^D$,此时 $\boldsymbol{C}^D = \begin{bmatrix} c_{2^k-1}^D \\ \vdots \\ c_1^D \\ c_0^D \end{bmatrix}$,

$\boldsymbol{C}^T = \begin{bmatrix} c_{2^k-1}^T \\ \vdots \\ c_1^T \\ c_0^T \end{bmatrix}$, $\boldsymbol{M}$ 是一个 2^k 阶矩阵,当 $\boldsymbol{M}$ 可逆时,假设 $\boldsymbol{M}^{-1} = (a_{i,j})$, $c_{2^k-1}^T$ 为需要计算的 k-项集的支持度,表达式为

$$c_{2^k-1}^T = a_{0,2^k-1} c_0^D + a_{0,2^k-2} c_1^D + \ldots + a_{0,0} c_{2^k-1}^D$$

通过以上方法可以估算出原始数据集中 k-项集的支持度,从而得到频繁项集。但是该算法也存在其执行效率上的不足。RRPH 算法在重构项集支持度时的计算是指数级时间复杂度,这对算法的执行效率有很大影响。其中重构项集支持度的时间消耗主要在计算转换矩阵 $\boldsymbol{M}$ 与 $\boldsymbol{M}^{-1}$ 和计算歪曲数据集中各个组合数(即 $\boldsymbol{C}^D$ 中各项的数量)。

2 RRPH 算法的改进

2.1 基于分治策略的改进

RRPH 算法需要通过公式 $\boldsymbol{C}^T = \boldsymbol{M}^{-1} \boldsymbol{C}^D$ 估算出 k-项集的真实支持度,可见需要构造阶数为 2^k 的转换矩阵 $\boldsymbol{M}$,然后再求出 $\boldsymbol{M}$ 的逆矩阵 $\boldsymbol{M}^{-1}$。随着候选项集 k 的增大,转换矩阵 $\boldsymbol{M}$ 的阶数以 2^k 的速度增大。在 RRPH 算法中,求解 $\boldsymbol{M}^{-1}$ 的时间复杂度为 $O(n^3)$,其中 $n = 2^k$,随着 n 的增大,算法的时间开销也越来越大。文中通过对转换矩阵的研究,采用分治策略的方法,可以得到 $\boldsymbol{M}^{-1}$ 满足递归关系。所以在求解 $\boldsymbol{M}^{-1}$ 时,可以将先求出 $\boldsymbol{M}$,再求其逆的方法改为直接求出 $\boldsymbol{M}_2^{-1}$,再根据递归关系最终求出 $\boldsymbol{M}^{-1}$。以下给出了求解 $\boldsymbol{M}^{-1}$ 的方法。

由于 $\boldsymbol{C}^D = \boldsymbol{M}\, \boldsymbol{C}^T$,则 1-项集所对应的公式为 $\boldsymbol{C}^D = \boldsymbol{M}_2\ \boldsymbol{C}^T$,其中转换矩阵 $\boldsymbol{M}$ 可表示为 $\boldsymbol{M}_2 = \begin{bmatrix} p_1 & p_2 \\ p_2 + p_3 & p_1 + p_3 \end{bmatrix}$。

同理,2-项集所对应的公式为 $\boldsymbol{C}^D = \boldsymbol{M}_4\ \boldsymbol{C}^T$,其转换矩阵 $\boldsymbol{M}$ 表示为

$$\boldsymbol{M}_4=\begin{bmatrix} p_1^2 & p_1p_2 & p_1p_2 & p_2^2 \\ p_1(p_2+p_3) & p_1(p_1+p_3) & p_2(p_2+p_3) & p_2(p_1+p_3) \\ p_1(p_2+p_3) & p_2(p_2+p_3) & p_1(p_1+p_3) & p_2(p_1+p_3) \\ (p_2+p_3)^2 & (p_2+p_3)(p_1+p_3) & (p_2+p_3)(p_1+p_3) & (p_1+p_3)^2 \end{bmatrix}$$

采用分治策略的思想对 $\boldsymbol{M}_4$ 进行划分

$$\boldsymbol{M}_4=\left[\begin{array}{cc:cc} p_1^2 & p_1p_2 & p_1p_2 & p_2^2 \\ p_1(p_2+p_3) & p_1(p_1+p_3) & p_2(p_2+p_3) & p_2(p_1+p_3) \\ p_1(p_2+p_3) & p_2(p_2+p_3) & p_1(p_1+p_3) & p_2(p_1+p_3) \\ \hdashline (p_2+p_3)^2 & (p_2+p_3)(p_1+p_3) & (p_2+p_3)(p_1+p_3) & (p_1+p_3)^2 \end{array}\right]=\begin{bmatrix} p_1\boldsymbol{M}_2 & p_2\boldsymbol{M}_2 \\ (p_2+p_3)\boldsymbol{M}_2 & (p_1+p_3)\boldsymbol{M}_2 \end{bmatrix}$$

同理可求得,对应的 3-项集的转换矩阵 $\boldsymbol{M}$ 表示为

$$\boldsymbol{M}_8=\begin{bmatrix} p_1\boldsymbol{M}_4 & p_2\boldsymbol{M}_4 \\ (p_2+p_3)\boldsymbol{M}_4 & (p_1+p_3)\boldsymbol{M}_4 \end{bmatrix}$$

依次类推,转换矩阵 $\boldsymbol{M}$ 有如下递归关系

$$\boldsymbol{M}_n=\begin{bmatrix} p_1\boldsymbol{M}_{n/2} & p_2\boldsymbol{M}_{n/2} \\ (p_2+p_3)\boldsymbol{M}_{n/2} & (p_1+p_3)\boldsymbol{M}_{n/2} \end{bmatrix}$$

$$n=2^k,k=1,2,\cdots$$

对 $\boldsymbol{M}_n$ 求逆矩阵

$$\boldsymbol{M}_n^{-1}=\begin{bmatrix} p_1\boldsymbol{M}_{n/2} & p_2\boldsymbol{M}_{n/2} \\ (p_2+p_3)\boldsymbol{M}_{n/2} & (p_1+p_3)\boldsymbol{M}_{n/2} \end{bmatrix}^{-1}=\begin{bmatrix} \boldsymbol{M}_{n/2} & \\ & \boldsymbol{M}_{n/2} \end{bmatrix}^{-1}\begin{bmatrix} p_1\boldsymbol{E}_{n/2} & p_2\boldsymbol{E}_{n/2} \\ (p_2+p_3)\boldsymbol{E}_{n/2} & (p_1+p_3)\boldsymbol{E}_{n/2} \end{bmatrix}^{-1}$$

由分块对角矩阵的性质可知

$$\begin{bmatrix} \boldsymbol{M}_{n/2} & \\ & \boldsymbol{M}_{n/2} \end{bmatrix}^{-1}=\begin{bmatrix} \boldsymbol{M}_{n/2}^{-1} & \\ & \boldsymbol{M}_{n/2}^{-1} \end{bmatrix}$$

采用高斯消元法求 $\begin{bmatrix} p_1\boldsymbol{E}_{n/2} & p_2\boldsymbol{E}_{n/2} \\ (p_2+p_3)\boldsymbol{E}_{n/2} & (p_1+p_3)\boldsymbol{E}_{n/2} \end{bmatrix}^{-1}$,过程如下

$$\left[\begin{array}{cc:cc} p_1\boldsymbol{E}_{n/2} & p_2\boldsymbol{E}_{n/2} & \boldsymbol{E}_{n/2} & 0 \\ (p_2+p_3)\boldsymbol{E}_{n/2} & (p_1+p_3)\boldsymbol{E}_{n/2} & 0 & \boldsymbol{E}_{n/2} \end{array}\right]\xrightarrow{r_2+r_1}$$

$$\begin{bmatrix} p_1\boldsymbol{E}_{n/2} & p_2\boldsymbol{E}_{n/2} & \boldsymbol{E}_{n/2} & 0 \\ \boldsymbol{E}_{n/2} & \boldsymbol{E}_{n/2} & \boldsymbol{E}_{n/2} & \boldsymbol{E}_{n/2} \end{bmatrix}\xrightarrow{r_2-\frac{r_1}{p_1}}$$

$$\left[\begin{array}{cc:cc} p_1\boldsymbol{E}_{n/2} & p_2\boldsymbol{E}_{n/2} & \boldsymbol{E}_{n/2} & 0 \\ 0 & \dfrac{p_1-p_2}{p_1}\boldsymbol{E}_{n/2} & \dfrac{p_1-1}{p_1}\boldsymbol{E}_{n/2} & \boldsymbol{E}_{n/2} \end{array}\right]\xrightarrow{\frac{p_1}{p_1-p_2}r_2}$$

$$\left[\begin{array}{cc:cc} p_1\boldsymbol{E}_{n/2} & p_2\boldsymbol{E}_{n/2} & \boldsymbol{E}_{n/2} & 0 \\ 0 & \boldsymbol{E}_{n/2} & \dfrac{p_1-1}{p_1-p_2}\boldsymbol{E}_{n/2} & \dfrac{p_1}{p_1-p_2}\boldsymbol{E}_{n/2} \end{array}\right]\xrightarrow{r_1-p_2r_2}$$

$$\left[\begin{array}{cc:cc} p_1\boldsymbol{E}_{n/2} & 0 & \dfrac{p_1(1-p_2)}{p_1-p_2}\boldsymbol{E}_{n/2} & \dfrac{-p_1p_2}{p_1-p_2}\boldsymbol{E}_{n/2} \\ 0 & \boldsymbol{E}_{n/2} & \dfrac{p_1-1}{p_1-p_2}\boldsymbol{E}_{n/2} & \dfrac{p_1}{p_1-p_2}\boldsymbol{E}_{n/2} \end{array}\right]\xrightarrow{\frac{1}{p_1}r_1}$$

$$\begin{bmatrix} \boldsymbol{E}_{n/2} & 0 & \dfrac{1-p_2}{p_1-p_2}\boldsymbol{E}_{n/2} \\ 0 & \boldsymbol{E}_{n/2} & \dfrac{p_1-1}{p_1-p_2}\boldsymbol{E}_{n/2} \end{bmatrix}$$

所以有,$\begin{bmatrix} p_1\boldsymbol{E}_{n/2} & p_2\boldsymbol{E}_{n/2} \\ (p_2+p_3)\boldsymbol{E}_{n/2} & (p_1+p_3)\boldsymbol{E}_{n/2} \end{bmatrix}^{-1}=\begin{bmatrix} \dfrac{1-p_2}{p_1-p_2}\boldsymbol{E}_{n/2} & \dfrac{-p_2}{p_1-p_2}\boldsymbol{E}_{n/2} \\ \dfrac{p_1-1}{p_1-p_2}\boldsymbol{E}_{n/2} & \dfrac{p_1}{p_1-p_2}\boldsymbol{E}_{n/2} \end{bmatrix}$

则

$$\boldsymbol{M}_n^{-1}=\begin{bmatrix} \boldsymbol{M}_{n/2}^{-1} & \\ & \boldsymbol{M}_{n/2}^{-1} \end{bmatrix}\begin{bmatrix} \dfrac{1-p_2}{p_1-p_2}\boldsymbol{E}_{n/2} & \dfrac{-p_2}{p_1-p_2}\boldsymbol{E}_{n/2} \\ \dfrac{p_1-1}{p_1-p_2}\boldsymbol{E}_{n/2} & \dfrac{p_1}{p_1-p_2}\boldsymbol{E}_{n/2} \end{bmatrix}=\begin{bmatrix} \dfrac{1-p_2}{p_1-p_2}\boldsymbol{M}_{n/2}^{-1} & \dfrac{-p_2}{p_1-p_2}\boldsymbol{M}_{n/2}^{-1} \\ \dfrac{p_1-1}{p_1-p_2}\boldsymbol{M}_{n/2}^{-1} & \dfrac{p_1}{p_1-p_2}\boldsymbol{M}_{n/2}^{-1} \end{bmatrix}$$

可见,$\boldsymbol{M}_n^{-1}$ 满足递归关系。因此,可以通过求出 $\boldsymbol{M}_2^{-1}=\begin{bmatrix} \dfrac{1-p_2}{p_1-p_2} & \dfrac{-p_2}{p_1-p_2} \\ \dfrac{p_1-1}{p_1-p_2} & \dfrac{p_1}{p_1-p_2} \end{bmatrix}$,再递归得出 $\boldsymbol{M}_n^{-1}$。然后,根据公式 $\boldsymbol{C}^T=\boldsymbol{M}^{-1}\boldsymbol{C}^D$ 求出各项集的真实支持度,最终还原出频繁项集。

根据递推公式,可以推算其时间复杂度如下

$$\begin{aligned} T(k)&=T(k/2)+S(k)=T(k/4)+S(k/2)+S(k)\cdots=T(2)+S(4)+\cdots+S(k/2)+S(k)= \\ &T(2)+2S(2)+2^2S(2)+\cdots+2^{n-1}S(2)= \\ &T(2)+S(2)(2+2^2+\cdots+2^{n-1})=T(2)+S(2)*\frac{2(1-2^{n-1})}{1-2}= \\ &T(2)+S(2)(2^n-2)= \\ &T(2)+S(2)(k-2) \end{aligned}$$

式中:$k=2^n,n=1,2,3,\cdots;T(2)$ 为求出 $\boldsymbol{M}_2^{-1}$ 所需要的时间;$S(2)$ 为生成矩阵 $\boldsymbol{M}_2^{-1}$ 所需要的时间;

$T(2)$ 和 $S(2)$ 均为常数，所以 $T(k)=O(k)$。而 RRPH 算法求解逆矩阵时的时间复杂度为 $O(n^3)$，递推的方法在时间复杂度上提高了两个数量级。

采用分治策略对 RRPH 算法进行改进，降低了算法的时间复杂度，有效地提高了算法的时间效率。

2.2 基于集合运算的改进

根据 RRPH 算法的具体步骤发现，该算法不仅在求解转换矩阵时间效率低下，而且在重构项集真实支持度时对歪曲数据集求各个组合数量的过程也是同样如此。

RRPH 算法在计算项集支持度时，是从干扰后的数据集估算出真实数据集的项集支持度，例如对于 2-项集，项集 11 经过概率变换后可能变为 00 或 01 或 10 或 11，所以在计算其支持度时必须考虑以上 4 种取值情况。同样对于任意的 k-项集，需要计算出 2^k 种不同的取值情况，所以 RRPH 算法在对干扰矩阵各个组合的计数过程时间开销很大。

针对以上问题，可以运用集合运算的原理进行相关计算。在文献[12]中，介绍了针对 MASK 算法采用集合运算原理进行优化的方法，RRPH 算法可以采用类似的方法进行改进。由于 RRPH 算法是在二维稀疏布尔矩阵的数据集上进行挖掘，根据该数据集的特性，可以发现在 2-项集$\{A,B\}$的计算中，A,B 取值为 0 或 1，只要查询出 A,B 取值都为 1 的个数，即 11 的个数，A 和 B 的取值在 1-项集挖掘中可以得到，其他的取值组合可以表示为

$$10:|A\cap\overline{B}|=|A|-|A\cap B|$$

$$01:|\overline{A}\cap B|=|B|-|A\cap B|$$

$$00:|\overline{A}\cap\overline{B}|=U-|B|-|A|+|A\cap B|$$

根据集合运算的原理可以推理

$$|\overline{A}_1\,\overline{A}_2\cdots\overline{A}_m B_1B_2\cdots B_n|=|B_1B_2\cdots B_n|-|B_1B_2\cdots B_n(A_1\cup A_2\cup\cdots\cup A_m)|$$

又可知

$$|A_1\cup A_2\cup\cdots\cup A_m|=\sum_{i=1}^{m}|A_i|-\sum_{i=1}^{m}\sum_{j>i}|A_iA_j|+\sum_{i=1}^{m}\sum_{j>i}\sum_{k>j}|A_iA_jA_k|-\cdots+(-1)^{m-1}|A_1A_2\cdots A_m|$$

由于事务内每一项都是项相互独立的，所以

$$|B_1B_2\cdots B_n(A_1\cup A_2\cup\cdots\cup A_m)|=|B_1B_2\cdots B_n|*|A_1\cup A_2\cup\cdots\cup A_m|$$

结合上述的公式可以得到

$$|\overline{A}_1\,\overline{A}_2\cdots\overline{A}_m B_1B_2\cdots B_n|=|B_1B_2\cdots B_n|-\sum_{i=1}^{m}|A_iB_1B_2\cdots B_n|-\sum_{i=1}^{m}\sum_{j>i}|A_iA_jB_1B_2\cdots B_n|+\sum_{i=1}^{m}\sum_{j>i}\sum_{k>j}|A_iA_jA_kB_1B_2\cdots B_n|-\cdots(-1)^{m-1}|A_1A_2\cdots A_mB_1B_2\cdots B_n|$$

对于任意 k 次候选项集，只需要在歪曲数据集中查询一次，其他的各种组合个数(即支持数)可以通过之前在歪曲数据集中得到的频繁项集取值都为 1 的个数，并结合该公式求出。在整个挖掘过程中，哈希表用于存储各个取值都为 1 的频繁项集的个数，以供后面的挖掘计算使用。通过该方法可以显著改善算法在计算歪曲数据集支持度的时间复杂度。

2.3 改进算法描述

在改进的 RRPH 算法中，需要经过概率干扰的歪曲数据集 D，增加函数 sup(A)用于计算项集 A 在歪曲数据集 D 中的支持数，函数 cal(k)用于计算 k-项集各个组合的支持数，哈希表 hashtab 用于存储频繁项集在歪曲数据集中取值都为 1 的支持数。

输入：歪曲数据集 D，扰动概率 p_1,p_2,p_3，最小支持度阈值 s。

输出：原始数据集的频繁项集 L。

scan D，for each $i\in I$，count i.count；//对数据集 D 中项集 I 中各项 i 计数

$L_1=\{\{i\}\mid i\in I,(i.\text{count}-p_2)/(p_1-p_2)\geqslant s\}$；

计算矩阵 $\boldsymbol{M}_2$；

for($k=2;L_{k-1}\neq\phi;k++$){$C_k=$ apriori_gen(L_{k-1})；

递归计算矩阵 $\boldsymbol{M}_n^{-1},n=2^k$；

for each $c\in C_k$｛

scan D，if ($c=1$) count c.count;//查询数据集 D，得到 c 中取值全为 1 的各项的支持数 count；

cal(k);//利用集合运算的推导公式求出项集中各个组合的支持数

$$\text{sup\_real}(c)=\sum_{j=0}^{2^k-1}\boldsymbol{M}_{2^k}^{-1}[0][j]*\text{sup}(c)$$

if(sup_real(c)$\geqslant s$){

hashtab.add(c.count)；

}

$L_k=\{\{c\}c\in C_k,\ \text{sup\_real}(c)\geqslant s\}$

}

}

return $L=\bigcup_k L_k$；

3　实验与分析

3.1　实验方法

为了验证改进的算法的有效性,通过实验将改进的算法和改进前的算法进行比较。本文实验的硬件平台为是 Intel(R) Core(TM) i5 CPU M480 2.67 GHz 处理器和 4 GB 内存,操作系统为 Windows 8,开发软件为 Microsoft Visual Studio 2010。

本实验采用的数据集是由 IBM Almaden 的人工数据集生成器生成。生成的数据集 D 的参数为 T10,I4,D100k,N100,分别表示数据集中数据的数据平均长度为 10,频繁项集的平均长度为 4,数据集中数据总个数为 1 000 000,数据集总项目数为 100。

本文改进算法主要针对 RRPH 算法的时间复杂度进行优化,所以在对算法进行时间效率分析的相关实验中,原始数据集以参数为 p_1,p_2,p_3 的概率进行干扰,最小支持度为 3%。

由于改进算法对原始数据集的干扰方法没有改变,所以改进后的算法与 RRPH 算法在隐私保护度与准确度上理论上没有改变。假设 F 为原始数据集 T 中频繁项集的集合,P 为歪曲数据集 D 中频繁项集的集合,则频繁项集的误差为 $\mathrm{IE}=\frac{|F-P|}{|F|}$。又假设 f 为 F 中一个频繁项集,其真实支持度为 s_f,歪曲后的支持度为 s_p,则支持度的误差为 $\mathrm{SE}_f=\frac{|s_f-s_p|}{|s_f|}$,总的支持度误差为 $\mathrm{SE}=\frac{1}{|F|}\sum_f \mathrm{SE}_f$。实验选取参数 $p_1=p$,$p_2=p_3=(1-p_1)/2$,其中 p 的取值为 0.1,0.2,0.3,0.4,0.49,0.51,0.6,0.7,0.8,0.9。并且根据不同的最小支持度阈值将改进的算法与 RRPH 算法和 MASK 算法比较项集误差与支持度误差。

3.2　实验结果分析

图 1 是改进后算法与 RRPH 算法的运行时间对比,原始数据集以参数 $p_1=0.6$,$p_2=0.2$,$p_3=0.2$ 的概率进行干扰,最小支持度为 3%。实验对比结果如图 1 所示。

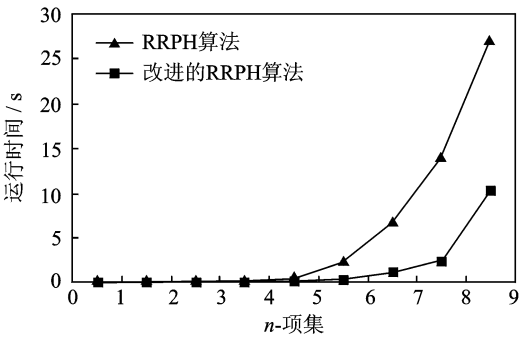

图 1　运行时间对比图

Fig. 1　Running time comparison chart

由图 1 实验结果可知,在设定的实验参数下,随着 n-项集的增大,算法运行的时间也逐步增加。n 越大,改进的 RRPH 算法的时间效率越明显,当 $n\geqslant 5$ 时,改进的算法还保持着比较好的运行效率。实验表明,改进的算法在时间性能上相比于改进前有了很大的提高。

图 2 是数据集以参数 $p_1=0.6$,$p_2=0.2$,$p_3=0.2$的概率进行干扰,最小支持度为 3%。其中数据集的数据总数分别取 50×10^3,75×10^3,100×10^3,125×10^3,150×10^3,175×10^3,200×10^3,225×10^3,250×10^3,275×10^3,时间对比均为计算 5-项集所需时间。根据运行时间的结果可以看出,RRPH 算法与改进的 RRPH 算法的运行时间,在计算同项集的情况下,随着数据总数的增加,运行时间逐步增加,从图中可以看出该增加规律,基本呈线性增加,并略有波动。

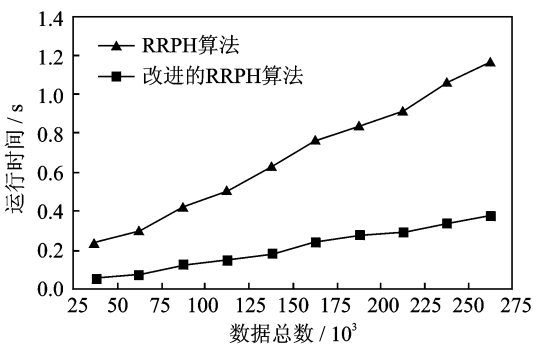

图 2　量化时间对比图

Fig. 2　Quantification of time comparison chart

图 3,4 分别给出改进前后的 RRPH 算法在最小支持度阈值为 3%时,随着参数 p 的变化,项集误差的变化情况,以及在 $p_1=0.6$,$p_2=0.2$,$p_3=0.2$ 时,随着最小支持度阈值 s 的变化,支持度误差的变化情况。由图 3 可知,改进前后的 RRPH 算法随着参数 p 的增加,项集误差不断减小,并趋于稳定;同时两个算法的误差基本一致,且隐私破坏相对较小。同样由图 4 可知,当 $p_1=0.6$,$p_2=0.2$,$p_3=0.2$ 时,改进前后的 RRPH 算法误差波动相对一致,并且误差较小。所以改进后的 RRPH 算法相比 RRPH 算法,在隐私保护度与准

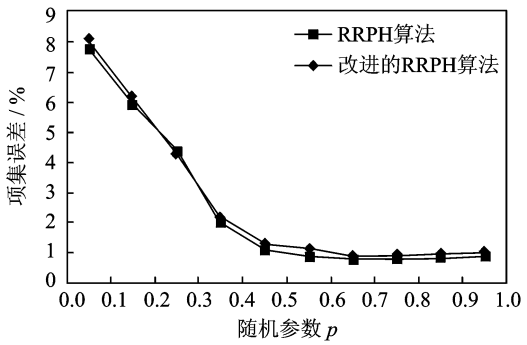

图3 项集误差对比图

Fig. 3 Item set error comparison chart

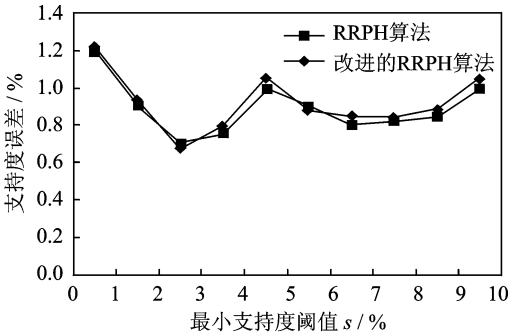

图4 支持度误差对比图

Fig. 4 Support error comparison chart

确度上基本没有变化。

4 结束语

本文针对RRPH算法执行效率低的问题，采用分治策略和集合运算的方法对该算法进行改进。改进的算法在重构原始数据支持度时，在计算$\boldsymbol{C}^T=\boldsymbol{M}^{-1}\boldsymbol{C}^D$时，通过分治策略降低了$\boldsymbol{M}^{-1}$的时间复杂度，并通过递归计算$\boldsymbol{M}^{-1}$的值。同时在对歪曲数据集进行支持数计数时，采用集合运算的方法，降低其运算的时间复杂度，从而使得算法得到优化。最后通过实验，验证了改进算法的有效性。

本文算法主要针对算法的时间效率，隐私保护度和准确度并没有得到提高。下一步工作将在保证算法的效率下，提高隐私保护度和准确度。

参考文献：

[1] Verykios V S, Bertino E, Fovino I N, et al. State-of-the-art in privacy preserving data mining[J]. SIGMOD Record, 2004,33(1):50-57.

[2] 陈晓明，李军怀，彭军，等. 隐私保护数据挖掘算法综述[J]. 计算机科学，2007,34(6):183-186.
Chen Xiaoming, Li Junhuai, Peng Jun, et al. Privacy preserving data mining algorithm overview [J]. Computer Science, 2007,34(6):183-186.

[3] Kantarcioglu M, Clifton C. Privacy-preserving distributed mining of association rules on horizontally partitioned data[J]. IEEE Trans on Knowledge and Data Engineering, 2004,16(9):1026-1037.

[4] 刘英华，杨炳儒，马楠，等. 分布式隐私保护数据挖掘研究[J]. 计算机应用研究，2011,28(10):3606-3610.
Liu Yinghua, Yang Bingru, Ma Nan, et al. State-of-the-art in distributed privacy preserving data mining [J]. Application Research of Computers, 2011,28(10): 3606-3610.

[5] 田宏，王亚伟，王秀坤. 基于可逆方阵的隐私保护关联规则挖掘[J]. 计算机工程，2009,35(7):153-155.
Tian Hong, Wang Yawei, Wang Xiukun. Invertible matrix-based privacy-preserving association rules mining [J]. Computer Engineering, 2009,35(7): 153-155.

[6] 熊平，朱天清，王晓峰. 差分隐私保护及应用[J]. 计算机学报，2014,37(1):101-122.
Xiong Ping, Zhu Tianqing, Wang Xiaofeng. A survey on differential privacy and applicatons [J]. Chinese Journal of Computers, 2014,37 (1): 101-122.

[7] 张有东，王建东，叶飞跃，等. 外关联规则挖掘[J]. 南京航空航天大学学报，2005,37(6):766-770.
Zhang Youdong, Wang Jiandong, Ye Feiyue, et al. Outside the association rules mining [J]. Journal of Nanjing University of Aeronautics & Astronautics, 2005,37(6): 766-770.

[8] Rizvi S J, Haritsa J R. Maintaining data privacy in association rule mining[C]// Proc of the 28th Int'l Conf on Very Large Data Bases. Hong Kong, China: Morgan Kaufmann Publishers, 2002:682-693.

[9] Agrawal S, Krishnan V, Haritsa J R. On addressing efficiency concerns in privacy preserving data mining [C]// DASFAA. Jeju Island: Springer-Verlay, 2004: 113-124.

[10] 张鹏，童云海，唐世渭. 一种有效的隐私保护关联规则挖掘方法[J]. 软件学报，2006,17(8):1765-1774.
Zhang Peng, Tong Yunhai, Tang Shiwei. A privacy preserving association ruleeffective mining method [J]. Journal of Software, 2006,17 (8): 1765-1774.

[11] 沈中林，崔建国. 改进的隐私保护关联规则挖掘算法[J]. 计算机工程与应用，2010,46(8):133-136.
Shen Zhonglin, Cui Jianguo. Improvement of privacy preserving association rulemining algorithm [J]. Computer Engineering and Applications, 2010, 46 (8): 133-136.

[12] 张长星，钱雪忠. 隐私保护数据挖掘算法 MASK 的优化[J]. 计算机工程与设计，2009,30(14):3316-3318.
Zhang Changxing, Qian Xuezhong. Optimization for MASK algorithm in privacy preserving data mining [J]. Computer Engineering and Design, 2009, 30 (14): 3316-3318.