文章编号:1001 - 9081(2008)06 - 1463 - 04

一种数据流中的频繁模式挖掘算法

朱 琼,施荣华
(中南大学 信息科学与工程学院, 长沙 410083)
(dido2nono@163.com)

摘 要:时序数据流的无限性、流动性和不规则性使得传统的频繁模式挖掘算法难以适用。针对时序数据流的特点,提出了一类特殊非规则数据流频繁模式挖掘的新算法。新算法采用时序数据分段的思想,逐段挖掘局部频繁模式,然后依据局部频繁模式有效地挖掘出所有的全局频繁模式。将新算法应用于电信领域的收入保障项目之中,结果表明,新算法具有良好的性能,能有效发现挖掘时序数据流中的频繁模式。

关键词:数据流; 频繁模式; 非规则; 局部频繁项集; 全局频繁项集
中图分类号: TP311.13 **文献标志码**:A

Algorithm of frequent-patterns mining in data stream

ZHU Qiong, SHI Rong-hua
(*School of Information Science and Engineering, Central South University, Changsha Hunan* 410083, *China*)

Abstract: The limitlessness, mobility, and irregularity of time series data stream make the traditional frequent-pattern mining algorithms difficult to extend to the mining problem of time series data stream. According to the characteristics of time series data stream, a new algorithm for mining the frequent-pattern from a kind of special irregular data stream was proposed, in which, time series data stream was partitioned firstly, and then the local frequent items were mined step by step. Finally, the global frequent items could be mined efficiently based on these local frequent items. After applying the new algorithm in the revenue assurance project of telecommunication field, the results show that the new algorithm has good performance, and can mine frequent-patterns effectively from the irregular data stream of telecommunication field.

Key words: data stream; frequent pattern; irregular; local frequent item; global frequent item

0 引言

数据流的无限性和流动性使得传统的频繁模式挖掘算法难以适用,因此,基于数据流的频繁模式挖掘技术面临着十分艰巨的挑战[1-3]。目前,发现数据流中的频繁模式正成为数据挖掘领域的一个新的研究热点,并已经取得了一些研究成果,但仍存在着许多不足。

目前,在已有的数据流中,任意长度频繁模式挖掘算法可分为两类:批处理方法和启发式方法[4-5]。批处理的方法主要思想是对数据流分段,通过分段上的局部频繁模式来求解全局近似频繁模式,其中代表性的基于批处理方法的数据流中的频繁模式挖掘算法有 Lossy Counting[6]、FP-Stream[7]、FP-DS[8]等。这类算法一般采用 FP-tree(Frequent Pattern tree)结构来捕捉数据流中频繁元组集信息,查询粒度通常较粗且实时性不高。启发式方法的主要思想是随着流数据的不断到达,由已知频繁模式逐步生成新的频繁模式。其中典型的基于启发式方法的数据流中的频繁模式挖掘算法有 Carma[9]、estDec[10]等。启发式方法能够随流数据的到达直接进行处理,在一定程度上可以实时地反映频繁模式的变化,但模式估计与查询精度较低。

本文基于时序数据流的无限性、流动性和不规则性等特点,提出了一种数据流频繁模式挖掘的新算法:新算法采用时序数据分段的思想,逐段挖掘局部频繁项集,然后依据局部频繁项集有效地挖掘出所有的全局频繁项集。通过将新算法应用于电信领域的实际项目之中,结果表明,该算法具有良好的性能,能有效发现挖掘时序数据流中的频繁模式。

1 相关定义

设 IDS 是非规则数据流序列,X 是一个项目集,其中 IDS 中包含项目集 X 的事务数成为 IDS 中项目集 X 的支持数,记为 $f_{IDS}(X)$。

定义 1 相同时段。假定非规则数据流 IDS 按时序可分段为 $IDS_1, IDS_2, \cdots, IDS_j, \cdots$,则 $IDS = IDS_1 \cup IDS_2 \cup \cdots \cup IDS_j \cup \cdots$,其中 $IDS_j = IDS[(j-1)t, jt]$,即 IDS_j 为非规则数据流序列 IDS 中从 $(j-1)t$ 时刻到 jt 时刻的所有数据。对给定的时间周期值 T,时段 $[(j-1)t, jt]$ 与 $[pT+(j-1)t, pT+jt]$ 称为相同时段。

定义 2 周期性非规则数据流。假定数据流 IDS 按时序可分段为 $IDS_1, IDS_2, \cdots, IDS_j, \cdots$,则 $IDS = IDS_1 \cup IDS_2 \cup \cdots IDS_j \cup \cdots$,其中 $IDS_j = IDS[(j-1)t, jt]$,即 IDS_j 为数据流序列 IDS 中从 $(j-1)t$ 时刻到 jt 时刻的所有数据。若对任意 $j \in [1, K]$,满足:$|IDS_j| = |IDS_{j+pK}|$,其中:$K = T/L[(j-1)t, jt]$,$L[(j-1)t, jt]$ 表示时间区间 $[(j-1)t, jt]$ 的长度,$|IDS_j|$ 表示 IDS_j 中的事务数,则称数据流 IDS 为周期性非规则数据流。

假定 IDS 为定义 2 给出的一类周期性非规则数据流。令 N 表示采集得到的连续 L 个周期的数据流,则 N 可表示为如下矩阵形式:

收稿日期:2007 - 12 - 11;**修回日期**:2008 - 02 - 19。

作者简介:朱琼(1982 -),女,湖南长沙人,硕士研究生,主要研究方向:数据挖掘; 施荣华(1963 -),男,湖南安乡人,教授,主要研究方向:计算机网络、数据挖掘。

$$\boldsymbol{N}=\begin{bmatrix} IDS_1^1 & IDS_2^1 & \cdots & IDS_K^1 \\ IDS_1^2 & IDS_2^2 & \cdots & IDS_K^2 \\ \vdots & \vdots & & \vdots \\ IDS_1^L & IDS_2^L & \cdots & IDS_K^L \end{bmatrix} \tag{1}$$

显然,由定义2,式(1)中各 IDS_j^i 满足 $|IDS_j^p| = |IDS_j^q|$, $\forall j \in [1,K], p、q \in [1,L]$。

令 $N^i = \bigcup_{j=1}^{K} IDS_j^i$, $N_j = \bigcup_{i=1}^{L} IDS_j^i$, 则有:$\boldsymbol{N} = \bigcup_{i=1}^{L} N^i = \bigcup_{j=1}^{K} N_j$。

定义3 支持度。假定 *IDS* 为周期性非规则数据流,对给定的时间周期值 T,令 $\boldsymbol{N}$ 表示采集得到的连续 L 个周期的数据流。$\boldsymbol{N}$ 中项目集 X 的支持度记为 $X.sup(\boldsymbol{N})$,令 $X.sup(\boldsymbol{N}) = \mathrm{Max}\{f_{N_1}(X)/|N_1|, \cdots, f_{N_K}(X)/|N_K|, f_{N^1}(X)/|N^1|, \cdots, f_{N^L}(X)/|N^L|\}$,其中 $|N^i|$ 表示 N^i 中的事务数,$|N^j|$ 表示 N^j 中的事务数。

定义4 频繁项集。设预先给定的支持度为 ξ,允许偏差 ε,假定 *IDS* 为周期性非规则数据流。对给定的时间周期值 T,令 $\boldsymbol{N}$ 表示采集得到的连续 L 个周期的数据流。对项目集 X:

1) 满足:$X.sup(\boldsymbol{N}) \geqslant \xi - \varepsilon$,则称模式 X 为一个全局频繁项集。

2) 满足:$X.sup(\boldsymbol{N}) > \varepsilon$,则称模式 X 为一个局部频繁项集。

3) 若满足:$X.sup(\boldsymbol{N}) \leqslant \varepsilon$,则称模式 X 为一个非频繁项集。

给定一个周期性非规则数据流序列 *IDS* 和一个最小支持度阈值 ξ,允许偏差 ε,挖掘 *IDS* 中所有的全局频繁项集的问题就称之为周期性非规则数据流中的频繁模式挖掘问题。

定义5 FP-IDS 树。满足下述特征的树称为 FP-IDS 树:

1)由树根(记做 null)、局部频繁项集构成的前缀子树和局部频繁项头表组成;

2)除根节点之外的每一个节点由 4 个域(data,f,pnode,par)组成,其中:data 为该节点代表的项目名,f 表示包含从根节点(不含根节点)开始到该节点所构成项集的所有项集计数和,pnode 为指向具有相同项目名的下一个节点的指针,par 是指向父节点的指针;

3)头表中每个元组由项目名和指向树中有相同项目名的第 1 个节点的指针两部分组成。

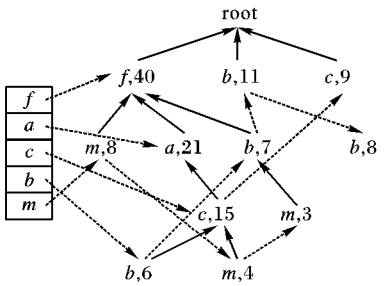

图1 FP-IDS 树结构

假定给定一个事务数据库 DB,支持数为 15,得到的频繁 1-项集的集合为 $L = [f:40, a:21, c:24, b:24, m:15]$(其中冒号后的数字代表支持度),则图 1 给出了其对应的 FP-IDS 树结构。由图 1 可知,FP-IDS 树与 FP-Tree 类似,但不同的是每一个节点仅由 4 个域(data,f,pnode,par)组成,所维护的信息更简洁,故所需要的存储空间更小。

基于以上描述,易证下述定理成立:

定理1 设预先给定的支持度为 ξ,允许偏差 ε,假定 *IDS* 为周期性非规则数据流。对给定的时间周期值 T,令 M_t 表示采集得到的第 t 个连续 L 个周期的数据流,令 $M = \bigcup_{i=1}^{S} M_i$。对任意项目集 X,若 $X.sup(M_t) \leqslant \varepsilon, t \in [1, S-1]$, $X.sup(M) \geqslant \xi$,则有:$X.sup(M_S) \geqslant \xi - \varepsilon$。

根据上述定理1,可以得出如下结论:删除当前的非频繁项集,即使它将来成为频繁项集,其支持度误差至多不会超过 ε,这种删除不会影响频繁项集的正确输出,因此,在实际应用过程中,只需保存局部频繁项集即可。

2 FP-IDS 算法

2.1 全局频繁 1-项集生成与更新

假定数据流分段为 $\boldsymbol{M}_t (t = 1,2,3,\cdots)$, 其中 $\boldsymbol{M}_t = \begin{bmatrix} IDS_1^{1+(t-1)L} & IDS_2^{1+(t-1)L} & \cdots & IDS_K^{1+(t-1)L} \\ IDS_1^{2+(t-1)L} & IDS_2^{2+(t-1)L} & \cdots & IDS_K^{2+(t-1)L} \\ \vdots & \vdots & & \vdots \\ IDS_1^{L+(t-1)L} & IDS_2^{L+(t-1)L} & \cdots & IDS_K^{L+(t-1)L} \end{bmatrix}$,记 $\boldsymbol{M}_t^i$ 为 $\boldsymbol{M}_t$ 的第 i 行,$\boldsymbol{M}_t^j$ 为 $\boldsymbol{M}_t$ 的第 j 列,则由定义2易证以下结论成立:

1) 对任意 $t = 1,2,3,\cdots$,有 $|\boldsymbol{M}_t^1| = |\boldsymbol{M}_t^2| = \cdots = |\boldsymbol{M}_t^L| = |\sum_{l=1}^{K} |IDS_l^1||} = C_0$,其中 C_0 为常数。

2) 对任意 $i \in [1,K]$,有 $|\boldsymbol{M}_t^i| = |\boldsymbol{M}_t^i| = |\boldsymbol{M}_t^i| = \cdots = L|IDS_i^1| = C_i$,其中 C_i 为常数。

全局 1-项集的数据集合 *GFI*-1 是一个数组,每个 1-项集是数组中的一个元素,数据类型是一个有 2 个数据域(*data*, *f*) 的结构体,其中,*data* 是1-项集的取值,*f* 是1-项集的计数。下面给出算法描述。

算法1 全局频繁 1-项集生成与更新算法

目的:扫描 $\boldsymbol{M}_t$ 段数据流,生成与更新全局频繁 1-项集。

输入:$\boldsymbol{M}_t$ 段数据流,允许偏差 ε;

输出:到 $\boldsymbol{M}_t$ 段为止的全局频繁 1-项集。

1)在 $\boldsymbol{M}_1$ 段,初始化数组 *GFI*-1,*GFI*-1(0),*GFI*-1(1),*GFI*-1(2),…,*GFI*-1(*K*) 为空集。

/* 数组 *GFI*-1 用于存储全局频繁1-项集,*GFI*-1(0) 用于存储数据流中 $\bigcup_i \bigcup_{j=1}^{K} IDS_j^i$ 段数据流中的 1-项集,*GFI*-1(1), *GFI*-1(2),…,*GFI*-1(*K*) 分别用于存储数据流中 $\bigcup_i IDS_1^i, \cdots, \bigcup_i IDS_K^i$ 段数据流中的1-项集 */

2) FOR each $e_j \in \boldsymbol{M}_t$ { // 对每个新到的 $\boldsymbol{M}_t$ 段中的数据流元素 e_j

(1) 初始化数组 $temp(1), temp(2), \cdots, temp(L), temp(L+1), temp(L+2), \cdots, temp(L+K)$ 为空集。

(2) FOR ($i = 1, i \leqslant L, i++$) { // 生成1-项集

IF $e_j \in M_t^i$ { IF $e_j \in temp(i)$ { $e_j.f = e_j.f + 1$ }

ELSE {插入 e_j 到 $temp(i)$,令 $e_j.f = 1$ } } }

(3) FOR ($i = 1, i \leqslant K, i++$) { //生成1-项集

IF $e_j \in M_j^i$ { IF $e_j \in temp(L+i)$ { $e_j.f = e_j.f + 1$ }

ELSE {插入 e_j 到 $temp(L+i)$,令 $e_j.f = 1$ } } } }

3) FOR ($i = 1, i \leqslant L, i++$) {

FOR each $e_j \in temp(i)$ {

IF $e_j.f > \varepsilon * C_0$ { // 删除非频繁1-项集

IF $e_j \in GFI$-1(0) { // 更新频繁1-项集的支持度

IF e_j 在 *GFI*-1(0) 中的支持度 < e_j 在 $temp(i)$) 中的支持度 {用 e_j 在 $temp(i)$ 中的支持度更新 e_j 在 *GFI*-1(0) 中的支持度。} }

ELSE 插入 e_j 到 *GFI*-1(0)。} } } // 插入新频繁1-项集

FOR ($i = 1, i \leqslant K, i++$) {

FOR *each* $e_j \in temp(L+i)$ {

IF $e_j \in GFI$-1(i) {

λ_1 = e_j 在 $temp(L+i)$ 中的支持度

λ_2 = e_j 在 $GFI\text{-}1(i)$ 中的支持度。

IF $\lambda_1 > t*\varepsilon*C_i - \lambda_2$ { // 删除非频繁 1-项集

用 $\lambda_1 + \lambda_2$ 更新 e_j 在 $GFI\text{-}1(i)$ 中的支持度。} }

// 更新频繁 1-项集

ELSE IF $e_j.f > \varepsilon*C_i$ { // 删除非频繁 1-项集

插入 e_j 到 $GFI\text{-}1(i)$。} } } // 插入新频繁 1-项集

4) $GFI\text{-}1 = GFI\text{-}1(0) \cup GFI\text{-}1(1) \cup GFI\text{-}1(2) \cup \cdots \cup GFI\text{-}1(K)$。

定理 2　按算法 1 得到的 $GFI\text{-}1$ 是一个局部频繁 1-项集的集合。

证明　由算法 1 可知,1)、2)分别用于将 M_t 段数据流中的 $\bigcup_{j=1}^{K} IDS_j^{1+(t-1)L}, \cdots, \bigcup_{j=1}^{K} IDS_j^{L+(t-1)L}, \bigcup_i IDS_1^i, \cdots, \bigcup_i IDS_K^i$ 段数据流中的 1-项集保存到 $temp(1), temp(2), \cdots, temp(L), temp(L+1), temp(L+2), \cdots, temp(L+K)$ 中。而 3)用于删除 $temp(1), temp(2), \cdots, temp(L), temp(L+1), temp(L+2), \cdots, temp(L+K)$ 中的非频繁 1-项集,故只需要证明 3)所删除的 1-项集均为非频繁 1-项集即可。

1) 针对 $temp(1), temp(2), \cdots, temp(L)$ 中的 1-项集:只有当 e_j 在 $temp(i)$ 中的支持度 $\leqslant \varepsilon*C_0$ 时才会被删除,即在 $temp(i)$ 中删除 1-项集 e_j 的条件为:$e_j.f \leqslant \varepsilon*C_0 \Rightarrow e_j.f/C_0 \leqslant \varepsilon$。由定义 3 可知,若 $e_j.sup(M) = e_j.f/C_0 = e_j.sup(M_t)$,则 $e_j.sup(M) \leqslant \varepsilon$,故 e_j 为非频繁 1-项集。而若 $e_j.sup(M) \neq e_j.f/C_0$,则若 e_j 为频繁 1-项集,即 $e_j.sup(M) \geqslant \xi - \varepsilon$。由定义 3 可知,显然存在 $M_s(M_s \neq M_t)$ 使得 $e_j.sup(M) = e_j.sup(M_s)$,故在 $temp(i)$ 中删除 1-项集 e_j 将不会影响 $e_j.sup(M)$。

2) 针对 $temp(L+1), temp(L+2), \cdots, temp(L+K)$ 中的 1-项集:仿照 1)同理可证。　证毕。

如果 1-项集是非频繁项集,则其超集频繁多项集也一定为非频繁项集。因此,根据上述定理 2,可以得出如下结论:在算法 1 中所删除的 1-项集将不会导致频繁多项集的丢失。

2.2　FP-IDS 树的生成与更新

定义 6 由 FP-DS 树的某一节点 e 出发到树根所形成的路径上各个节点所组成的项集称为 e 的一个模式,简称为模式 e。所有的模式 e 组成的集合称为 e 的模式集。

FP-IDS 树压缩存储局部频繁项集,第 i 个分段首先新生成一棵初始 FP-IDS 子树,然后将第 $i-1$ 个分段生成的 FP-IDS 子树加入到第 i 个分段新生成的初始 FP-IDS 子树,最终构建出一棵到第 i 个分段的 FP-IDS 树。

算法 2　第 i 个分段的 FP-IDS 子树生成算法

输入:数据流分段 M_i,数据流分段 M_{i-1} 的 FP-IDS 子树,允许偏差 ε;

输出:第 M_i 分段的 FP-IDS 子树。

1)调用算法 1 得到全局频繁 1-项集 $GFI\text{-}1$。

2)依据定义 5 构造第 M_i 分段的初始 FP-IDS 子树。

3)IF $(i>1)$ {

(1)调用 Insert-FP-IDS $(i-1)$ 函数。

/*将第 $i-1$ 段的 FP-IDS 树保存的临界频繁项集插入到第 i 段的 FP-IDS 树中*/

(2)删除第 $i-1$ 段的 FP-IDS 树。

}

Insert-FP-IDS $(i-1)$ {

1)按 M_{i-1} 段 FP-IDS 树头表的逆序取项目 e_i;

2)FOR each e_i {

(1)取得 FP-IDS 树中 e_i 的模式集,其模式分别是模式 e_{i1}, e_{i2},模式 $e_{i3}, \cdots$;

(2)FOR each e_{ij} { 按 M_i 段的 $GFI\text{-}1$ 排序 e_{ij} 模式中的各项目,插入到 M_i 段的 FP-IDS 树中} }

从算法 2 的实现过程可以看出,每个分段内只保留了局部频繁项集,根据定理 1,可以输出所有满足定义 4 的频繁项集。

2.3　频繁项集的生成

算法 3　频繁项集的生成算法

输入:M_i 段的 FP-IDS 树,支持度 ξ(设对应的支持数为 SF);

输出:频繁项集。

1)按 M_i 段 FP-IDS 树头表的逆序取项目 e_i;

2)FOR each e_i {

(1)取得 FP-IDS 树中 e_i 的模式集,其中的模式分别是模式 e_{i1}, e_{i2},模式 $e_{i3}, \cdots$;

(2)FOR each e_{ij} { IF $(e_{ij}.f > SF)$ {输出 e_{ij} 模式和它的计数}

IF $(e_{ij}.f \neq 0)$ { e_{ij} 模式中各节点的 f 值减去 $e_{ij}.f$。} } }

2.4　完整的 FP-IDS 算法

算法 4　FP-IDS 算法:使用 FP-IDS 树,挖掘频繁模式。

输入:非规则数据流 IDS,最小支持度阈值 ξ,允许误差 ε;

输出:频繁模式的完全集。

1) 数据流分段。数据流分段 $M_t(t=1,2,\cdots)$。

2)FOR $(t=1; t \leqslant$ 数据流的段数$; t++)$ {

(1) 调用算法 1,生成 M_t 段的 $GFI\text{-}1$。

(2) 调用算法 2,生成 M_t 段的 FP-IDS 树。

(3)调用算法 3,输出频繁项集。}

3　算法分析

为了测试 FP-IDS 算法的性能,在实验中与经典的 FP-GROWTH 算法进行了比较,具体采用文献[13]中的数据合成方法,合成一个事务数据库 T25. I20. D100K,项数为 10 KB。分别测试 FP-IDS 算法与 FP-growth 算法在不同事务数与支持度阈值情况下的运行时间。实验平台的操作系统为 Windows XP,CPU 为 P4 3.06 GHz,内存为 512 MB。测试程序使用 Java 编写。运行时间随支持度阈值变化的测试结果如图 2。当支持度阈值固定为 0.5% 时,运行时间随事务数变化的测试结果如图 3。

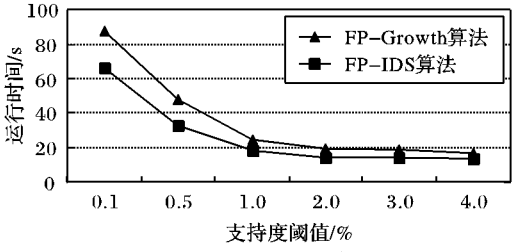

图 2　运行时间随支持度阈值变化情况

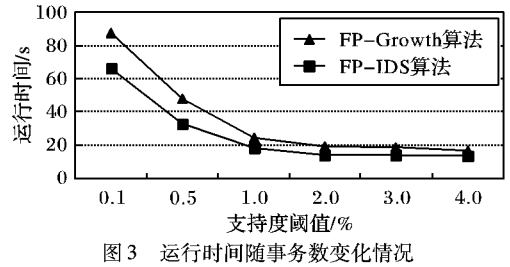

图 3　运行时间随事务数变化情况

由图 2、3 可知,显然,FP-IDS 算法要优于传统的 FP-growth 算法,具有更好的频繁模式挖掘性能。为了进一步实际验证算法性能,我们将算法应用到了电信企业 CDR 话单生成准确性稽核系统之中,该系统首先分别从 7 号信令系统和各类交换机采集到交换机所下话单 CDR 数据流 IDS_1 和 7 号信令消息 SS7 数据流 IDS_2,由于 7 号信令系统和交换机产生的话单数据均为典型的周期性非规则流数据,因此,在进行

预处理之后,将依据 FP-IDS 算法对 SS7 数据流和 CDR 数据流进行频繁模式挖掘,以判定交换机 CDR 话单生成过程是否存在收入漏洞。具体判定方法如下:

假定利用 FP-IDS 算法得到的数据流 IDS_1 的频繁项集为 $FP_{IDS1} = \{FP_{11}, FP_{12}, \cdots, FP_{1m}\}$,数据流 IDS_2 的频繁项集为 $FP_{IDS_2} = \{FP_{21}, FP_{22}, \cdots, FP_{2n}\}$,由于7号信令系统记录了用户通过过程产生的所有控制和计费消息,而交换机仅记录了用户通过过程产生的计费消息,因此选择以 IDS_2 作为参照,显然可得到如下交换机 CDR 话单生成准确性判定准则:

1)若 $FP_{IDS_1} = FP_{IDS_2}$,则交换机 CDR 话单生成过程正确。

2)若存在频繁模式 $FP_{2k} \notin FP_{IDS_1}$,则交换机 CDR 话单生成过程中存在频繁模式 FP_{2k} 类型的漏单。

3)若存在频繁模式 $FP_{1k} \notin FP_{IDS_2}$,则交换机 CDR 话单生成过程中存在频繁模式 FP_{1k} 类型的错单。

在实际频繁模式挖掘过程中,取 $K = 12, L = 3$,最小支持度阈值 $\xi = 10^{-4}$,允许误差 $\varepsilon = 10^{-5}$,项集 $I = \{a_1, a_2, \cdots, a_7\}$,其中:$a_1$ 为主叫号码,a_2 为被叫号码,a_3 为话单时段,一天按小时分为24个时段,数据的采集周期为从当天的0点整到24点,a_4 为计费类型,a_5 为套餐类型,a_6 为话单类型(POC,PTC,MOC,MTC 等),a_7 为交换机类型(NOKIA,NORTEL 等)。

最终,发现频繁模式 $\{a_2 = 1380013800, a_3 = 11, a_4 = MTC, a_7 = NOKIA\}$,$\{a_2 = 17990, a_3 = 11, a_4 = MTC, a_7 = NOKIA\}$,$\{a_2 = 17995, a_3 = 11, a_4 = MTC, a_7 = NOKIA\}$ 属于 FP_{IDS_2},而不属于 FP_{IDS_1},即发现交换机下单过程中存在以下类型漏单情形:在22:00到24:00之间,经常发生该运营商 NOKIA 用户拨打移动充值号码 1380013800,以及铁通 IP 卡充值号码 17990 与 17995,而这几类通话未被计费。

4 结语

提出了一种不规则数据流频繁模式挖掘算法 FP-IDS,用户可以连续在线获得当前所有的频繁项集,新算法不需列举交易的每个子集,也不需产生大量的频繁候选项,从而大大提高了算法的效率。在于电信行业的初步实际应用表明,新算法能有效挖掘出不规则数据流中的频繁模式,具有良好的性能。

参考文献:

[1] CHARIKAR M, CHEN K, FARACH-COLTON M. Finding frequent items in data streams [C]// Proceedings of the 29th International Colloquium on Automata, Languages, and Programming (ICALP'02), LNCS 2380. London: Springer-Verlag, 2002: 693 - 703.

[2] KARP R M, SHENKER S, PAPADIMITRIOU C H. A simple algorithm for finding frequent elements in streams and bags [J]. ACM Transactions on Database Systems, 2003, 28(1): 51 - 55.

[3] NAN JIANG, LE GRUENWALD. CFI-Stream: Mining closed frequent itemsets in data streams [C]// The 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 06). New York: ACM Press, 2006: 592 - 597.

[4] HAN JIA-WEI, PEI JIAN, YIN YI-WEN, *et al*. Mining frequent patterns without candidate generation: A frequent pattern tree approach [J]. Data Mining and Knowledge Discovery, 2004, 8(1): 53 - 87.

[5] CHENG J, KE YI-PING, NG W. Maintaining frequent itemsets over high-speed data streams [C]// Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2006), LNAI 3918. Berlin: Springer-Verlag, 2006: 462 - 467.

[6] MANKU G S, MOTWANI R. Approximate frequency counts over data streams [C]// Proceedings of the 28th International Conference on Very Large Data Bases. Hong Kong: Morgan Kaufmann Publishers, 2002: 346 - 357.

[7] GIANNELL A, HAN J, PEI J, *et al*. Mining frequent patterns in data streams at multiple time granularities [C]// Next Generation Data Mining. [S.l.]: AAAI/MIT Press, 2003: 191 - 202.

[8] 刘学军,徐宏炳,董逸生. 挖掘数据流中的频繁模式[J]. 计算机研究与发展, 2005, 42(12): 2192 - 2198.

[9] HIDBER C. Online association rule mining [C]// Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 1999). New York: ACM Press, 1999: 145 - 156.

[10] CHANG J H, LEE W S. Finding recent frequent itemsets adaptively over online data streams [C]// Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2003: 487 - 492.

(上接第1462页)

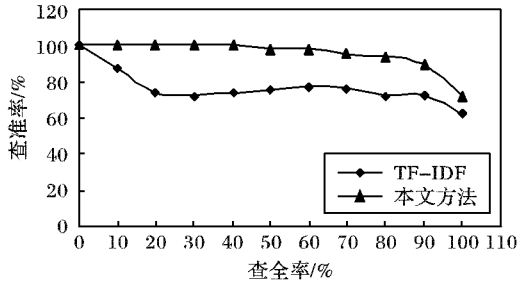

图3 五个主题下的平均查准率/查全率曲线

4 结语

本文针对基于 LSI 的文本检索中的重要优化过程——权重计算,提出了一种基于 Sigmiod 处理和位置因子的新权重方案,取得了较好的实验结果,但仍有很大的完善空间。影响文本特征词的因素并非只有词频、反文本频率和位置,文献[6]提到的"义长"、文献[7]提到的"受限自然语言理解技术"等都是确定权重时值得考虑的因素,若能多层次地综合考虑更多的因素,实验结果将会有更大的提高。此外,由于目前位置因子的比例都是经验值,还缺乏相应的理论依据,若能在以上两方面做进一步的工作,结果可能更为理想,这也正是下一步工作的重点。

参考文献:

[1] 焦玉英. 信息检索进展[M]. 2版. 北京:科学出版社, 2003: 17 - 18.

[2] PAPADIMITRIOU C, RAGHAVAN P, TAMAKI H. Latent Semantic indexing: A probabilistic analysis [J]. Journal of Computer and System Science, 2000, 61(2): 621 - 625.

[3] LIN HONG-FEI. The mechanism of text file classification based on examples [J]. Journal of Computer Research & Development, 2001, 38(9): 237 - 241.

[4] GAO J, ZHANG J. Clustered SVD strategies in Latent semantic indexing[J]. Information Processing & Management, 2007, 41(3): 1051 - 1063.

[5] 苏亮, 聂峰光, 郭力, 等. 隐含语义检索系统词条权重的处理[J]. 计算机与应用化学, 2005, 22(11): 971 - 976.

[6] 韩客松, 王永成. 一种用于主题提取的非线性加权方法[J]. 情报学报, 2000, 19(6): 650 - 653.

[7] 何新贵, 彭甫阳. 中文文本的关键词自动抽取和模糊分类[J]. 中文信息学报, 1999, 13(1): 9 - 15.