# 数据流中频繁模式的评估与维护*

宋国杰+, 唐世渭, 杨冬青, 王腾蛟

(北京大学 信息科学技术学院,北京 100871)

# Estimation and Maintenance of Frequent Pattern on Data Streams

SONG Guo-Jie+, TANG Shi-Wei, YANG Dong-Qing, WANG Teng-Jiao

(School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

+ Corresponding author: Phn: +86-10-62756374, E-mail: sgj@db.pku.edu.cn, http://www.pku.edu.cn

Received 2003-05-10; Accepted 2004-07-16

**Song GJ, Tang SW, Yang DQ, Wang TJ. Estimation and maintenance of frequent pattern on data streams. *Journal of Software*, 2004,15(Suppl.):20~27.**

**Abstract**: In this paper, the methods are investigate for online, frequent pattern mining of stream data, with the following contributions: (1) based on heuristic methodology and sample theory, step-by-step data stream mining method is used to estimate potential pattern set; (2) will find any length pattern not only single item pattern; (3) to find more appropriate length of each segment satisfying accuracy requirement, Hoeffding bound theory was introduced and revised to make it more suit for pattern mining; (4) a maintenance approach for estimating frequent patterns is developed for on-line analysis. Based on this design, estimation and maintenance algorithms are proposed for efficient analysis of data streams. This performance study compares the proposed algorithms and identifies the most accuracy-, memory- and time- efficient algorithms for stream data analysis.

**Key words**: data stream mining; sample; frequent pattern; Hoeffding bounds; heuristic method

**摘 要**: 研究了数据流中频繁模式的挖掘问题,主要贡献在于:(1) 基于启发式思想方法和抽样理论的基础上,提出了基于数据流样本集的分步模式估计方法;(2) 算法求解所有长度的模式,而不仅仅是单项集模式;(3) 为了找到满足精度要求的恰当的数据流样本集长度,引入了 Hoeffding bound 理论,并进行了修正,从而使之更适合于这一问题;(4) 提出了对估计模式进行在线维护的方法.基于上述方法的基础上,提出了模式估计和维护算法.最后,通过和已有算法进行实验对比分析,结果表明,该算法在结果精度、空间、时间复杂性等方面都适合进行数据流的分析.

**关键词**: 数据流挖掘;抽样;频繁模式;Hoeffding bounds;启发式方法

所谓数据流[1]是指一系列连续且有序的点组成的数据序列,这个序列中的数据按照固定的次序到达,而且只能被读取一次或者几次.数据流有着广泛的应用,如网络路由管理以及网络的入侵检测,电信中的欺诈分析,

* Supported by the National Grand Fundamental Research 973 Program of China under Grant No.G1999032705 (国家重点基础研究发展规划(973))

作者简介: 宋国杰(1975－),男,河南人,博士生,主要研究领域为数据库,信息系统;唐世渭(1939－),男,教授,博士生导师,主要研究领域为数据库,信息系统;杨冬青(1945－),女,教授,博士生导师,主要研究领域为数据库,信息系统;王腾蛟(1974－),男,博士,讲师,主要研究领域为数据库,信息系统.

金融检测及金融仲裁,商务点击流分析及个性化分析等.对数据流的研究主要包括数据流管理系统和数据流算法.系统如 Aurora[2],Hancock[3],STREAM[4],Telegraph[5]等,算法如聚类[6,7]、分类[8,9]、回归分析[10]、模式挖掘[11,12]等.目前研究较多的是数据流管理系统,对基于数据流系统上的数据分析与应用的研究较少.

本文重点研究频繁模式的挖掘.频繁模式有着广泛的应用.如在冰川查询、关联规则、冰川数据方以及路由管理和 IP 包计数等方面.基于传统数据集的频繁模式挖掘开展了广泛而深入地研究,但基于数据流的模式挖掘却甚少.文献[11]引入新的 Count Sketch 数据结构用来对数据流中的频繁项进行计数,结果表明无论时间复杂性还是空间复杂性都优于常规的基于抽样的算法.文献[12]给出了两个算法 Sticky Sampling 和 Lossy Counting,在设定支持度阈值 $s$ 和误差因子阈值$\varepsilon$下,主要给出了求解单个频繁项的有效算法.

频繁模式挖掘和数据流本身的特征决定了进行多项频繁模式挖掘是一个十分困难的工作.在数据流环境中,如果对所有的候选模式进行计数,那么会因为候选模式的数目巨大而使得数据流不能进行实时的处理.同样,如果采用多次迭代的方法进行频繁模式的挖掘,但由于对数据流的扫描只能是一次或者少数几次,从而使得这一方法不可用.因而必须采用新的处理方法来解决基于数据流的频繁模式挖掘问题.

基于启发式思想方法和抽样理论的基础上,本文提出了基于数据流样本集的分步模式估计方法.算法求解了多项模式集,而不仅仅是单项模式集.为了找到满足精度要求的恰当的数据流样本集长度,引入了 Hoeffding bound 理论,并进行了修正,从而使之更适合于这一问题.本文也提出了对估计模式进行在线维护的方法.

## 1 问题定义

**定义 1.1(数据流)**. 假设 $DS$ 是形如 $e_1,\ldots,e_i,\ldots,e_n,\ldots$的数据流序列;$I=\{I_1,I_2,\ldots,I_m\}$是 $m$ 个不同的属性组成的集合,其中每个属性称为一个项.数据流序列中的每个元素 $e_i\subseteq I$.

**定义 1.2(阈值参数)**. 用户设定的阈值参数有 3 个:$s\in(0,1)$表示用户设定的支持度阈值,$\varepsilon\in(0,1)$表示误差因子,满足$\varepsilon<s$,$1-\delta\in(0,1)$是置信度.

**定义 1.3(支持度)**. 假设 $p$ 是包含于 $I$ 的一个模式,$E$ 是数据流序列中所有包含模式 $p$ 的元素组成的集合,其中 $N$ 表示数据流的长度,则该模式的支持度为

$$Support(p)=|E|/N$$

基于上述描述的基础上,基于数据流的频繁模式挖掘定义如下:

**定义 1.4(约束)**. 在数据流中,算法在任何时刻输出的结果满足如下条件:

1. 支持度计数大于阈值 $sN$ 的所有频繁模式被输出.
2. 支持度计数小于阈值$(s-\varepsilon)N$ 的所有模式均不输出.
3. 一个模式的被估计的支持度计数值小于真实计数值最多为$\varepsilon N$.

本文的目的就是设计一个算法,该算法可以在允许$\varepsilon$偏差的范围内,可信度为 $1-\delta$的要求下,在尽可能小的空间复杂性和时间复杂性的基础上,输出所有满足定义 1.3 的所有模式.

## 2 频繁模式挖掘

频繁模式挖掘重要分为两个部分:模式估计和模式维护.模式估计是基于部分数据流估计出整个数据流中所有潜在的频繁模式.模式维护是对模式估计的结果在整个数据流上进行动态维护与调整.详述于第 2.1 节和第 2.2 节.

### 2.1 模式评估

#### 2.1.1 目 的

由于数据流本身是快速动态变化,因此很难精确的给出挖掘结果.因此以满足一定精度要求的近似求解,换取对实时数据流的有效处理,是当前对数据流进行分析处理的主流方法,也是本文采用的方法.同时考虑到数据流是无限长度的序列,因此在起始数据流范围内,以启发式迭代的方法,在若干个抽样的数据流样本集中逐步求出所有长度的频繁模式,这就是模式估计方法的基本思路.

模式估计基本思路如图 1 所示:从数据流中提取数据流样本集 $S_i$ 的方法是进行抽样,就是随机的从数据流中抽取一个小规模的数据流元素样本集合,该集合的数据能够代表整个数据流的特征,也是进行近似问题求解常用的基本方法.实用的经典抽样方法如 Reservoir Sampling,Concise sampling[13].每个数据流样本集 $S_i$ 需满足一定长度要求(第 1.1.2 节给出描述).每个抽样样本集标记为 $S_i(i\in 1,2,\ldots)$,起始样本集为 $S_1$.$N=S_1+S_2+\ldots+S_{l_{max}}$.

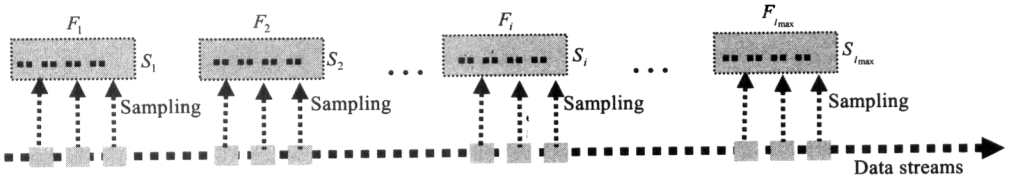


图 1 模式估计过程

模式估计从数据流样本集 $S_1$ 开始.在数据流样本集 $S_i$ 的基础上,估计出长度为 $i(i=1,2,\ldots,l_{max}$,$l_{max}$ 表示最长模频繁式的长度)的所有满足阈值要求的模式.值得说明的是,模式估计算法不是在一个数据流样本集 $S_i$ 上求得所有长度的频繁模式全集,而是只求取当前长度为 $i$ 的模式集,原因是频繁模式全集的求解需要对数据流样本集进行 $l_{max}$ 次扫描和算法迭代,这就违反了数据流的特性:① 数据流只能进行一次或者少数几次访问;② 数据流速度到达很快,必须进行实时处理.因此,本文方法只在数据流样本集 $S_i$ 上求解长度为 $i$ 的模式集,这样做的优点在于:① 因为只对长度为 $i$ 的模式计数,因此对数据流只进行线性扫描;② 由于采用启发式的方法,在数据流样本集 $S_i$ 之间 Apriori 特性同样有效;③ 可以提高数据流样本集 $S_i$ 长度的估计的准确性(见第 1.1.2 节);④ 在求解了 $l_{max}$ 个样本集后,就得到了全部的频繁模式集,这样在以后数据流中只需进行维护调整,无须重新求解.缺点是丢失了传统方法百分之百准确无误的精度,但实验结果表明,本方法能够得到满足用户设定精度要求的解.

本算法在求解过程中,维护的模式集是最大模式集.也就是说维护了模式全集的一个较小的,且与全集具有等价关系的边界子集.最大模式是指它的所有子模式也是频繁模式,它的所有超模式皆为非频繁模式.这样不但可以减少模式存储的空间,而且使得在模式维护过程中无须维护全部模式,而只需维护模式的边界子集即可.

**定义 2.1(Hoeffding Bound 估计)**. 给定一个数据流样本集 $S_i$ 和阈值 $s$,$\varepsilon$和$\delta$,则 $S_i$ 中潜在的模式分为 3 类:

1. 频繁模式 $\hat{f}_i$.满足 $\hat{f}_i$ 是频繁模式的可能性是 $1-\delta$,且 $\hat{f}_i.count>(s+\varepsilon)S_i$.
2. 临界模式 $\bar{f}_i$.满足 $\bar{f}_i$ 是临界模式的可能性是 $1-\delta$,且$(s-\varepsilon)S_i\le \bar{f}_i.count\le(s+\varepsilon)S_i$.
3. 非频繁模式 $\breve{f}_i$.满足 $\breve{f}_i$ 是临界模式的可能性是 $1-\delta$,且 $\breve{f}_i.count<(s-\varepsilon)S_i$.

其中,$\hat{F}_i$ 表示长度为 $i$ 的频繁模式集合,$\hat{f}_i\in\hat{F}_i$ 是长度为 $i$ 的一个频繁模式.$C_i$ 表示长度为 $i$ 的候选模式集合,$c_i\in C_i$ 是长度为 $i$ 的一个候选模式.$\bar{F}_i$ 表示长度为 $i$ 的临界模式集合,$\bar{f}_i\in\bar{F}_i$ 是长度为 $i$ 的一个临界模式.

### 2.1.2 Hoeffding Bound 估计与修正

如前所述,要求每个数据流样本集 $S_i$ 具有一个合适的长度值边界,使得该数据集能够近似的代表整个数据流数据的分布,并且基于该它的求解能够满足用户的要求.Hoeffding Bound[8]边界理论能够胜任这一要求.

假设任意一个模式 $X$ 是一个随机变量,它的概率范围是 $R$(随机变量 $X$ 的概率范围 $R$ 定义为最大概率值与最小概率值的差.本文中 $R$ 的取值为 1,因为最大概率值为 1,最小概率值为 0).假定对于随机变量 $X$,我们有 $n$ 个独立的观察值,它的均值是$\mu$.Hoeffding bound 边界理论指出,随机变量取值至少为$(\mu-\varepsilon)$的可能性为 $1-\delta$.其中

$$n=\frac{R^2}{2\varepsilon^2}\ln(1/\delta) \tag{1}$$

因此,基于式(1)就可以根据设定的参数值$\delta$和$\varepsilon$估计出数据流样本集 $N_i$ 的边界.

Hoeffding Bound 理论有一个很好的性质就是它是与生成这些样本值的概率分布无关的.由于通常我们不清楚求解模式的概率分布,所以这个性质显得尤为重要.但是要得到这种与概率分布无关的性质就要付出一定的代价,就是用这种方法求得的 $n$ 值比那些与概率分布有关的方法求得的要大一些.所以要想得到同样的值,就需要更多的样本.由于$\varepsilon$是样本容量的函数,所以它对结果的影响很大.因此,我们就需要使$\varepsilon$的值尽量小.为了提高估计的准确性,下面进行修正.

前面假定随机变量 $X$ 出现的最大概率为 1,从而得到 $R$=1.但由 Apriori 性质可知,任何一个模式的支持度总是小于或者等于它的任何子集的支持度.那么对于频繁模式集 $F_i$ 和 $F_{i+1}$,则 $F_{i+1}$ 中任何模式的支持度一定小于等于 $F_i$ 中模式支持度的上界(最大值).因此得出修正概率范围定义 2.2.

**定义 2.2(修正概率范围 $R$).** 与长度为 $\ell$ 的频繁模式 $F_\ell$ 对应的数据流样本集 $S_\ell$ 的概率范围 $R$ 的大小为

$$R=\mathrm{Max}\{Support(f_{\ell-1})\,|\,\text{for all } f_{\ell-1}\in \hat{F}_{l-1}\cup \bar{F}_{l-1}\},$$

其中 $Support\ (f_{\ell-1})$ 是长度为 $l-1$ 的频繁模式或者临界模式的支持度.

这样,$R$ 的值就成为一个与频繁模式长度相关的一个函数,可以表示为 $R=\Theta(l)$.这样,式(1)修正为

$$n=\frac{\Theta(l)}{2\varepsilon^2}\ln(1/\delta) \tag{2}$$

例:假有集合 $\hat{F}_2=\{ab(0.2),ac(0.15),ad(0.13)\}$ 和 $\bar{F}_2=\{bc(0.08)\}$.那么可以断言,对于将在 $\hat{F}_3$ 和 $\bar{F}_3$ 出现的任何模式,其支持度一定小于 0.2.也就是说与数据流样本集 $S_3$ 对应的参数 $R$ 可以设定为 0.2,而不是 1.

### 2.1.3 模式估计算法

**算法 1.** Pattern_Estimation.

算法输入:数据流样本集 $DS=S_1,S_2,\ldots,S_{t\max}$;阈值参数 $s,\varepsilon,\delta$.

算法输出:最大频繁模式集 $\hat{F}$,最大临界模式集 $\bar{F}$.

1) $(\hat{F}_1,\bar{F}_1)=find\_1\_itemsets(S_1)$ // get frequent and critical itemsets
2) For ($k$=2;$L_{k-1}\langle\rangle\varnothing$;$k$++)
3) $C_k=apriori\_gen(\hat{F}_{k-1},s,\varepsilon,\delta)$ // get candidate itemsets and filter them
4) For each element $e\in S_k$ // scan $N_k$ for count
5) $C_t=subset(C_k,e)$ // get subsets of t that are candidates
6) For each candidate $c\in C_t$
7) $c.count$++
8) $(\hat{F}_k,\bar{F}_k)=get\text{-}k\text{-}itemsets(C_k,s,\varepsilon,\delta)$
9) Filter($\hat{F}_k,\bar{F}_k$) // get longest itemsets
10) Return $\hat{F}=\cup_k\hat{F}_k,\bar{F}=\cup_k\bar{F}_k$

算法解释:

步骤 3)通过 Apriori-Gen 方法得到候选模式 $C_k$,并进行过滤.过滤方法是:对于任一候选项 $c_k\in C_k$,如果存在一个非频繁模式 $\bar{f}_k\subseteq c_k$,则过滤掉 $c_{i+1}$.因为根据我们前面的数据流数据分布的约定,数据段内模式的分布是近似相等的,因此 Apriori 性质依然成立.

步骤 4)是模式计数.为了提高计数的效率,采用哈希技术将 $C_{k+1}$ 散列到哈希树当中.当一个数据流元素 $e_i$ 到来时,如果 $e_i$ 的长度小于 $i$,则忽略该元素,否则将该元素利用哈希计数对哈希树中的候选模式进行计数.对于任意 $c_{k+1}\subseteq e_k$,则使其 $c_{k+1}.count=c_{k+1}.count+1$.直至 $S_{k+1}$ 数据流结束.

步骤 8)是根据设定的阈值,可以将 $c_{i+1}$ 按照定义 4.1 分为 3 类:频繁模式、临界模式和非频繁模式.

步骤 9)的过滤方法是:① 对于所有的频繁模式 $\hat{f}_{k+1}\in F_{k+1}$,过滤掉 $\hat{f}_k\in F_k$ 中所有满足条件 $\hat{f}_k\in\hat{f}_{k+1}$ 的频繁模式,也就是说仅仅保留最长的模式;② 对于所有的临界模式 $\bar{f}_{k+1}\in\bar{F}_{k+1}$,过滤掉 $\bar{f}_k\in\bar{F}_k$ 中所有满足条件 $\bar{f}_k\in\bar{f}_{k+1}$ 的所有临界模式,仅仅保留最长的临界模式.

## 2.2 模式维护

由于模式估计结果的近似性,所以需要在真实数据流中确认;由于数据流真实数据分布可能会随着时间而发生概念漂移,因此需对估计结果进行验证和调整,并进行维护.模式维护操作不是基于抽样结果集,而是在真实的数据流上进行,从而期望得到更为精确真实的结果.

模式维护对 $\hat{F}$ 和 $\bar{F}$ 采用不同的方法.由于 $\hat{F}$ 对应的阈值为 $s+\varepsilon$,但只有当计数小于阈值 $s-\varepsilon$ 才会被删除,因

此对其每隔一定距离 $Gap$ 进行一次"点"维护操作.$Gap$ 是上次维护到本次维护操作之间的距离.由于 $\overline{F}$ 的阈值范围是$[(s-\varepsilon),(s+\varepsilon)]$,所以其随时可能小于边界阈值,所以采用实时维护的方法,即对所有数据流进行"线"维护操作.

**定理 2.1.** 当 $Gap=2\varepsilon N/(s-\varepsilon)$时,对频繁模式集 $\hat{F}$ 进行一次维护操作($N$ 为从维护开始到当前数据流的长度).

证明:因为每个$f\in\hat{F}$ 其计数值最少为 $N(s+\varepsilon)$,那么假设在最坏情况下经过 $Gap$ 长度数据流,其中每个数据流元素都没有对 $f$ 进行计数,则这时其支持度小于阈值 $s-\varepsilon$而被删除.因为在点 $N$ 和$(N+Gap)$之间模式绝对计数是相等的,因此有等式$(N+Gap)(s-\varepsilon)=N(s+\varepsilon)$,则有 $Gap=2\varepsilon N/(s-\varepsilon)$. □

维护操作采用数据结构哈希树和线性表.$\hat{F}$ 被散列到哈希树中,从而提高计数效率.$\overline{F}$ 由于其规模较小且需要实时维护,所以插入线性表中.每个模式在数据结构中表示为$(f,count)$,其中 $f$ 表示模式,$count$ 表示与该模式对应的计数值.

**算法 2.** Pattern_Maintenance.

算法输入: 数据流 Data Streams,阈值参数 $s,\varepsilon,\delta$.

算法输出: 最大频繁模式集 $\hat{F}$ ,最大临界模式集 $\overline{F}$ .

```
1)  Gap=0;N_m=N;N=0;                                          // initialization;
2)  For each incoming e∈Data Streams do
3)      F̂ .count(e);                                          // F̂ counting;
4)      for each f∈ F̄ do                                      // F̄ maintenance;
5)          if f⊆e then
6)              f.count= f.count+1;
7)              if f.count>(s+ε) then { F̄ .delete(f); F̂ .insert(f);}
8)          else if f.count<(s−ε) then F̄ .decom_insert(f);
9)      if Gap=2εN_m/(s−ε) then                               // F̂ maintenance;
10)         for each f ∈ F̂ do
11)             N_m=N;Gap=0;
12)             if f.count∈[(s−ε),(s+ε)] then { F̂ .delete(f); F̄ .insert(f);}
13)             if f.count<(s−ε) then F̄ .decom_insert(f);
14)     Gap=Gap+1;
```

## 3 实验结果和性能分析

为了评估算法的各项性能指标,我们进行了一系列的实验.结果表明,本文设计算法的各项指标均呈现出较好的性能,能够满足在设定精度范围内的求解要求.

实验环境:OS 是 windows 2000,CPU 为 P4,主频为 1.4G,主存为 512M.算法实现的工具是 JB 6.0.

实验数据集:数据流数据可以看作快速到达的事务序列,因此实验采用的数据是将传统事务数据以流的形式输入到设计的算法进行处理.实验采用了两套数据 T10I6D500K 和 T20I4D500K.其中$|T|$表示数据集中事务的平均长度,$|I|$表示最大潜在频繁模式的平均长度,$|D|$表示总的事务数目.

实验方法:主要从如下几个角度进行了实验和性能分析:

1. 与 Apriori 算法运行结果比较,研究算法结果的准确度.
2. 支持度对算法的影响.
3. 修正概率范围 $R$ 与模式长度的关系分析.
4. 准确度与数据流样本集长度 $N_i$ 的关系分析.

### 3.1 准确度比较

本文算法是近似算法,在用户指定阈值 $s,\delta$和$\varepsilon$的范围内,输出满足用户要求的结果.准确无误的结果就是把

当前数据流数据作为静态数据,运用基于静态数据集的频繁模式挖掘算法,输出精确的结果.因此,本实验下载了采用著名的 Apriori 算法与我们的算法进行对比.

在保持支持度相同($s$=0.01)的前提下,通过变化精确度阈值$\delta$和$\varepsilon$,运行两算法,将其结果进行比较,得出他们之间的准确度情况.见表 1($T10I6D500K$).

表 1 参数$\delta$和$\varepsilon$对精确度的影响

| 置信度 $1-\delta$ | 误差因子$\varepsilon$ | 频繁模式 $\hat{F}$ 精确度*(%) | 临界模式 $\overline{F}$ 精确度(%) | $\lvert\overline{F}\rvert/\lvert\hat{F}\rvert$ (%) |
|---|---|---|---|---|
| 0.961 | 0.001 | 98 | 90.6 | 11 |
| 0.961 | 0.002 | 96.5 | 87 | 11.6 |
| 0.962 | 0.001 | 97.4 | 89 | 12 |
| 0.962 | 0.002 | 96 | 88 | 12.7 |
| 0.964 | 0.004 | 92 | 85 | 13 |
| 0.964 | 0.006 | 90 | 84 | 13.5 |
| 0.967 | 0.004 | 91.5 | 83 | 14 |
| 0.967 | 0.006 | 89.5 | 80 | 16 |

*模式精度的计算方法:($\hat{F}\cap$|Apriori 得到频繁模式集|)除以$|\hat{F}|$,临界模式与此类似.

由表 1 可知:① 结果中频繁模式的精度高于临界模式的精度.因为由于临界模式的支持度计数临近阈值边界,因此受误差影响后,可能被过滤或者分解掉;频繁模式则因其计数值较高,即使受到误差影响,但其计数的波动值较小,可能低于支持度阈值而改变;② 相比较而言,误差因子对结果精度的影响较大于置信度的影响.原因的本质是 $n$ 值随$\delta$和$\varepsilon$值的变化引起的.$n$ 值与$\delta$和$\varepsilon$都是反比函数,但从它们的函数曲线上看,同比例增长和减小,$\delta$对 $n$ 值的影响比$\varepsilon$大同时因为这里求得的 $n$ 值是临界值,所以 $n$ 值的改变对求得最终的结果的精度影响较大;③ 从模式分布看,$\overline{F}$ 远小于 $\hat{F}$ ,而且随着误差因子的增大而增大.原因在于临界区本身是一个小的区域,同时由于$[s-\varepsilon,s+\varepsilon]$是临界模式出现的区间,所以随着$\varepsilon$的增大而使得区间增大,所以临界模式占有的比例略有升高.

### 3.2 支持度影响分析

在保持精度参数值不变的情况下($\delta$=0.038 和$\varepsilon$=0.002),通过改变支持度 $s$ 的值,从如下几方面观察支持度的影响.如图 1 所示(T10I6D500K).

1. 对结果精确度的影响

图 2 通过调整支持度阈值运行算法,对估计最终产生的模式(频繁模式和临界模式)的准确度进行了分析.由图可知,支持度阈值对结果精度的影响不大.影响不大的原因是在保持参数$\delta$和$\varepsilon$不变的情况下,无论支持度如何变化,都不影响样本集 $S_i$ 的大小,影响的只是模式的多少而模式分布因子 $\varepsilon$ 并未改变,所以精度不会变化很大.

2. 对模式估计的数据流抽样长度($N_{\text{estimation}}$)的影响

图 3 给出在支持度变化时,进行模式估计的数据流长度的变化情况.模式估计数据流就是进行模式估计的所有数据流样本集长度的和($N_{\text{estimation}}=S_1+S_2+\ldots+S_{l\max}$ ).图示表明,随着支持度的升高,模式估计数据流长度呈现下降趋势.原因是随着支持度升高,最长模式的长度大大降低,而我们算法的循环次数等于最长模式的长度,因此.随着支持度的升高,最长模式的降低,进行估计的 $l$ 大大减小,$N_{\text{estimation}}$ 自然也随之减小.

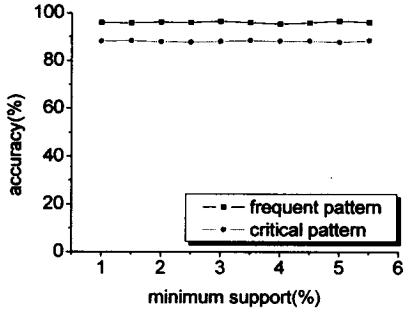


图 2 支持度阈值对结果精度的影响

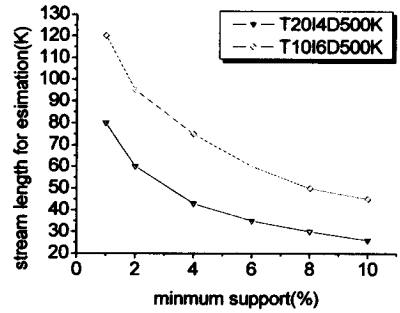


图 3 支持度阈值对 $N_{\text{estimation}}$ 的影响

3. 对运行时间的影响

图4给出了和Apriori算法对比的时间复杂性分析.结果表明,随着支持度的升高,算法花费时间降低,且小于算法 Apriori 所花费的时间.原因是随着支持度的升高,一方面产生的模式降低,进行计数所花费的时间减小,另一方面进行模式估计的数据规模降低.效率高于 Apriori 的原因是明显的:Apriori 是进行 $l$ 次迭代得到结果,而本文算法只是线性的扫描一遍数据集即可(如果需要进行 I/O 操作,本文算法的优越性将更加明显).

4. 对耗费空间的影响

图 5 给出了算法耗费空间和内存的比较.算法耗费空间的计算方法是产生模式的数目与模式长度以及每个项占用空间的乘积.结果表明:(1) 算法优于 Apriori,原因是我们存储的只是最长的模式,而 Apriori 存储的是所有的模式;(2) 随着支持度降低耗费空间降低,原因是算法产生的模式大为减小.

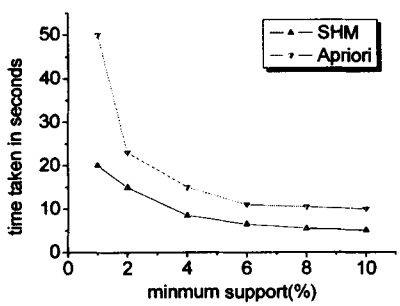


图 4 支持度对运行时间的影响

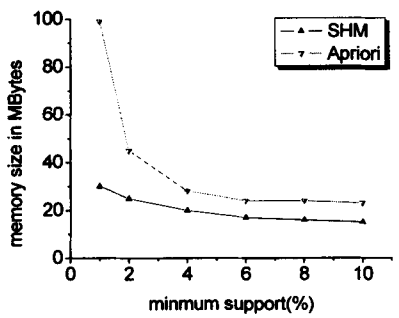


图 5 支持度对耗费空间的影响

### 3.3 修正概率范围$R$与模式长度的关系分析

本文对概率范围 $R$ 的值进行了修正,通常方法是把 $R$ 认为是 1(长度为 1 时认为 $R$=1,如图 6 所示),但是这样就使得估计的结果不是更为准确的临界值,造成对样本集长度 $S_i$ 的估计偏大,从而使得 $N_{\text{estimation}}$ 的结果偏大,使得进行模式估计面临更多的数据集,耗费更多的时间和空间.本文通过把模式支持度上界作为当前样本集估计 $R$ 的值,从而使得估计结果的 $S_i$ 大大降低.如图 6 所示.

### 3.4 准确度与数据流样本集长度$N_i$的关系分析

图 7 反映的是对于每个数据流样本集的长度取值对于其计算结果精度的影响.正常情况下,我们都取当横坐标为 1 时的值,也就是满足我们计算结果精度的临界值.图 7 表明,当数据流样本集值 $S_i$ 取值小于满足当前精度临界值 $n$ 的时候,其精度大大降低,当取值大于这个值时,其计算的结果精度的提高变缓.因此,正确的估计整个临界的值 $n$ 对算法的结果和性能影响很大.

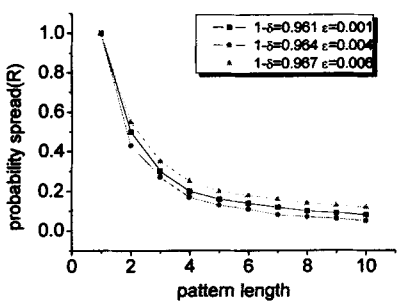


图 6 模式长度与修正概率 R 的关系

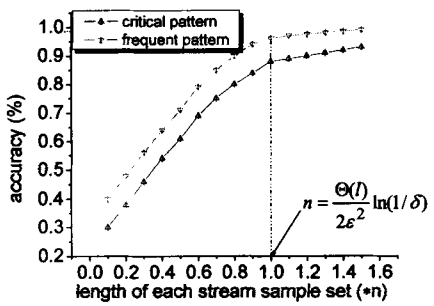


图 7 每个抽样集大小对结果精度的影响

## 4 结论与展望

本文解决了数据流上进行频繁模式挖掘的近似有效算法.该算法不仅解决了单个项的频繁模式求解,而且

对任意长度的模式,在满足一定精度范围内都可以进行有效求解.

本文虽然给出了解决这一问题的方法,但仍有许多问题有待进一步思考:如何解决噪音存在的情况?如何支持度动态变化的情况?如何解决序列的情况?这些都有待于进一步深入的研究.

**References:**

[1] Garofalakis M, Gehrke J, Rastogi R. Querying and mining data streams: You only get one look. In: Tutorial at 2002 ACM-SIGMOD Int'l Conf. on Management of Data (SIGMOD 02). Madison, WI, 2002.

[2] Carney D, Cetintemel U, Cherniack M, Convey C, Lee S, Seidman G, Stonebraker M, Tatbul N, Zdonik S. Monitoring streams—A new class of data management applications. In: Proc. of the 28th Int'l Conf. on Very Large Data Bases. 2002.

[3] Cortes C, Fisher K, Pregibon D, Rogers A, Smith F. Hancock: A language for extracting signatures from data streams. In: Proc. of the 2000 ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. 2000. 9~17.

[4] Arasu A, Babcock B, Babu S, Datar M, Ito K, Nishizawa I, Rosenstein J, Widom J. STREAM: The Stanford stream data manager. In: Proc. of the ACM Int'l Conf. on Management of Data (SIGMOD 2003). 2003.

[5] Avnur R, Hellerstein JM. Eddies: Continuously adaptive query processing. In: Proc. of the 2000 ACM SIGMOD Int'l Conf. on Management of Data. 2000. 261~272.

[6] Guha S, Mishra N, Motwani R, O'Callaghan L. Clustering data streams. In: Proc. of the Annual Symp. on Foundations of Computer Science (FOCS 2000). 2000.

[7] O'Callaghan L, Mishra N, Meyerson A, Guha S, Motwani R. High-Performance clustering of streams and large data sets. In: Proc. of the 2002 Int'l Conf. on Data Engineering (ICDE 2002). 2002.

[8] Domingos P, Hulten G. Mining high-speed data streams. In: Proc. of the 2000 ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. 2000. 71~80.

[9] Hulten G, Spencer L, Domingos P. Mining time-changing data streams. In: Proc. of the 2001 ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. 2001. http://citeseer.nj.nec.com/hulten01mining.html

[10] Chen Y, Dong G, Han J, Wah BW, Wang J. Multi-Dimensional regression analysis of time-series data streams. In: VLDB Conf. 2002.

[11] Charikar M, Chen K, Farach-Colton M. Finding frequent items in data streams. In: Proc. of the 29th Int'l Colloquium on Automata, Languages and Programming. 2002.

[12] Manku GS, Motwani R. Approximate frequency counts over streaming data. In: Proc. of the 28th Int'l Conf. on Very Large Data Bases (VLDB 2002). 2002.

[13] Gibbons PB, Matias Y. New sampling-based summary statistics for improving approximate query answers. In: Proc. of the 1998 ACM SIGMOD. 1998. 331~342.