# 基于权重的流数据频繁项挖掘算法的应用

杨 立
(运城学院 公共计算机教学部,山西 运城 044000)

**摘 要**:针对 Lossy Counting 算法,即一个基于计数的确定性方案,提出一种新的基于权重的流数据频繁项挖掘算法(Lossy Weight),扩展了流数据频繁项的作用域。Lossy Weight 算法不仅可用于传统的基于计数的频繁项挖掘,还可以挖掘出在整个流数据中所占权重比重大于门槛值的数据。实验数据分析证明该方案是有效的。

**关键词**:频繁项;数据挖掘;权值
**中图分类号**:TP311 **文献标识码**:A **文章编号**:1674-7720(2011)02-0106-03

## Weight frequent items mining algorithm flow of data in the financial database application

Yang Li
(Public Computer Education Department, Yuncheng University, Yuncheng 044000, China)

**Abstract**: Lossy counting algorithm is a program based on counting ncertainty. This paper proposed a new flow of data based on the weight frequent items mining algorithm. Expanded the scope of current frequent item of data. Lossy weight algorithm can be used in the traditional count-based mining frequent item, also excavated in the entire flow data is larger than the proportion of total weight threshold data, analysis of experimental data show that the program is effective.

**Key words**: frequent item; data mining; weight

基于计数的频繁项挖掘算法适用于每个数据元组所含知识相等或近似的情况,例如用户在网页上的点击流,搜索引擎的关键词流、路由器上的IP包流等情况。但在更多的情况下,每个事务代表的知识是不相等的。如电信系统中的通话记录,每个用户的电话用时是不相同的;在证券交易中心,每笔交易的金额也是不同的。许多小客户的事务数多,但每笔事务的权值很小;重要的大客户事务数虽少,但每笔事务的权值很大。如果此时用原有的频繁项挖掘算法,将不能很好地体现那些事务数少但重要性高的客户。而采用新的基于权重的算法,则可以很好地找出那些重要性高的元素。

本文提出的基于权重的新算法是对原有 Lossy Counting[1]的扩展。不仅可以解决基于计数的频繁项挖掘问题,还能解决基于权重的频繁项挖掘问题。并且 Lossy Counting 算法本质上是新算法的一个特例(窗口定长,权值为 1)。新算法在应用域上超出了原有算法,甚至可支持基于计数与权重的混合查询。

## 1 Lossy Counting 算法

Lossy Counting 算法是一个基于计数的确定性算法。它有两个用户指定的参数—门槛参数 $s$ 和误差参数 $\varepsilon$。将流数据划分为若干个大小为定长 $width=\left|\frac{1}{\varepsilon}\right|$ 的窗口,下标从 1 开始,当前窗口标记为 $b_{curr}$,值为 $\left|\frac{N}{width}\right|$。在窗口边沿运行裁剪算法,并最终得到计数的近似值。

算法数据结构为三元组$(e,f,\Delta)$的集合 $D$。其中 $e$ 表示为流中的元素,$f$ 为估计的计数,$\Delta$ 为 $f$ 可能的最大误差。$D$ 初始为空,每当有一个新元素 $e$ 到达时,首先在 $D$ 中查找是否已存在包含 $e$ 的元组。若存在则对应的该元组计数 $f$ 加 1;否则创建一个新的元组$(e,1,b_{curr}-1)$。当到达窗口边界时,对 $D$ 进行如下裁剪:若元组$(e,f,\Delta)$满足 $f+\Delta\leqslant b_{curr}$,则删除该元组。当查询到达时,返回所有 $f\geqslant(s-\varepsilon)N$ 的元组。

## 2 Lossy Weight 算法

本文提出的基于权重的频繁项挖掘算法(Lossy

Weight Algorithm)与原有算法有着相同的定义:根据用户定义的门槛参数 $s \in (0,1)$,输出在整个流数据中所占权重比重大于 $s$ 的所有元素。

新算法同样满足实时性的要求。在任意时间内,用户都可以提交查询,算法的结果满足以下的要求:(1)数据所有占权重比超过 $s$ 的元素都被输出;(2)所有占权重比小于 $s-\varepsilon$ 都不会被输出;(3)权重频繁项的误差至多为 $\varepsilon$。

新的算法保持了原有的 Lossy Counting 实现简单、处理速度快的特点。同样地,在误差的精确控制上有这样两个特点[2]:(1)存在误报可能(false positive);(2)误报的误差可控制。

### 2.1 Lossy Weight 算法实现

新算法有如下的定义:用户必须明确地指定门槛参数 $s$ 和误差参数 $\varepsilon$,并且定义流数据当前大小为 $N$。

流数据被划分为一个个窗口。新算法对窗口的大小不加限制,可根据实际需求来设定其大小。这一特性使得算法的适应能力更强。当系统负载高时,可以通过增大窗口尺寸缓冲更多的数据处理,起到批处理的作用,减少裁剪算法的调用次数,有效地提高系统的处理能力。划分后的每个窗口都拥有一个从 1 开始的下标,令当前窗口下标为 $b_{\text{current}}$。对于流数据中的元素 $e$,定义其到当前的实际权值为 $W_e$。$b_{\text{current}}$ 和 $W_e$ 会随着流数据的持续到达而不断增长。定义数据结构 $D$,由元组 $(e, W, \Delta)$ 组成。其中 $e$ 表示为流中的元素,$W$ 为估计的权值(数值上等于创建元组后到达的权值总和),$\Delta$ 为在创建元组前 $W$ 可能存在的最大误差。此外,分别定义两个权值计数器 $V_{b-1}$ 和 $V$。$V$ 为当前流数据所有元素的权重总和,有 $V=\sum_{i=1}^{N} W_i$。$V_{b-1}$ 是截止上一窗口结束时所有元素的权重总和。算法描述如下:

初始,$D=\phi$,$V_{b-1}=0$。

当一个新元素 $e$ 到达时,将 $e$ 的权值 $W_i$ 加入计数器 $V$,之后对 $D$ 进行更新操作。首先查找 $D$ 中是否存在 $e$。如果存在,将 $e$ 的权值加入 $W$。否则新建成员 $(e, W_i, \varepsilon V_{b-1})$。

在窗口的边界,对 $D$ 进行裁剪。裁剪的规则很简单,当 $W+\Delta \leqslant \varepsilon V$ 时,即从 $D$ 中删除该元组。裁剪后,更新 $V_{b-1}$ 的值为当前的 $V$。当查询到达时,返回所有 $W \geqslant (s-\varepsilon)N$ 的元组。

### 2.2 新算法的优势

在 Lossy Counting 算法的基础上改进的 Lossy Weight 算法保留了原有算法处理效率高、占用空间少、误差精确可控的优点。同样地,算法实现简明,很容易应用到实践当中。新算法包含了原有的 Lossy Counting 算法,具有更大的灵活性。新算法可根据实际情况划分窗口,时间窗口大小灵活可变。Lossy Counting 算法的时间窗口不可变,事实上就是窗口大小为 $\frac{1}{\varepsilon}$、权值为 1 时的 Lossy Weight 算法的特例。通过灵活地选取窗大小,新的 Lossy Weight 算法可以得到更好的内存占用情况。

## 3 Lossy Weight 算法的实验分析

### 3.1 Lossy Weight 算法的特性实验

本文采用国泰君安 CSMAR(China Stock Market Accounting Research)系列数据库中的中国股票交易高频数据库作为实验数据[3]。本实验采用了上海证券交易所 2009 年 12 月 5 日~12 月 7 日三天的股票交易高频数据。日均 20 万条交易记录,总计为 590 233 条交易记录。在流数据频繁项挖掘实验中,将数据按时间排序,并模拟其实时到达的特性,对送达流数据处理引擎进行频繁项挖掘。

对整个交易日所有个股的交易信息采用 LW 算法进行数据处理,对交易量所占比重大于 1%的个股进行频繁项挖掘,然后对内存使用情况进行分析。原有的 LC 算法不能处理带权重的挖掘任务。在实验中,定义了不同窗口大小,并对其进行了分析。

图 1 所示实验是在 $s$=1%、$\varepsilon$=0.1%情况下,截取交易日前 5 000 个数据的内存使用情况进行对比。实验显示,LW 算法的窗口尺寸越小,裁剪次数越频繁,则内存使用效果越好。但过多的裁剪无疑会加大系统的负荷。所以可以根据系统的负载大小来合理地确定窗口宽度。LW 算法中窗口尺寸的可伸缩性使得算法适应能力更强。

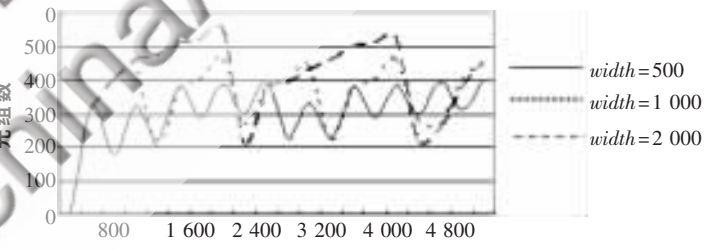


图 1 不同窗口宽度下 LW 算法的内存占用情况

LW 算法的内存占用情况取决于窗口尺寸和错误容许度 $s$ 的大小。容许的错误度越大,内存使用情况就越好。在窗口大小相等的情况下,对不同的错误容许度进行频繁项挖掘。

图 2 显示了在相同窗口大小($width$=1 000)情况下,不同 $\varepsilon$ 的内存占用情况。实验显示,LW 算法对内存空间的需求与误差 $\varepsilon^{-1}$ 近似成正比。因此,在不影响最终决策的前提下,错误容许度 $\varepsilon$ 越大越好。

### 3.2 LW 算法对 LC 算法的对比实验

Lossy Weight 算法是对 Lossy Counting 算法的改进。在应用上有更广的范围,在原有的问题领域,新算法同样占有优势。LC 算法的窗口大小是固定的 $\varepsilon^{-1}$,LW 算法的窗口是动态的,可以应对任意窗口大小。这就可以面对更复杂的应用情况。在数据流量大时,扩大窗口尺寸,

能起到批处理的效能。当系统较空闲时，减少窗口尺寸，以得到更好的内存使用情形。

如图3所示，在实验中，截取交易日前5 000个数据的内存使用情况进行对比。实验设置LW窗口大小为LC大小的一半。在第一个窗口，可以看到LW算法与LC算法的内存占用是相同的。但到窗口边沿时，裁剪后的内存占用得到明显的下降。通过对整个流的处理对比，可以明显地看出LW算法具有更好的内存使用情况。

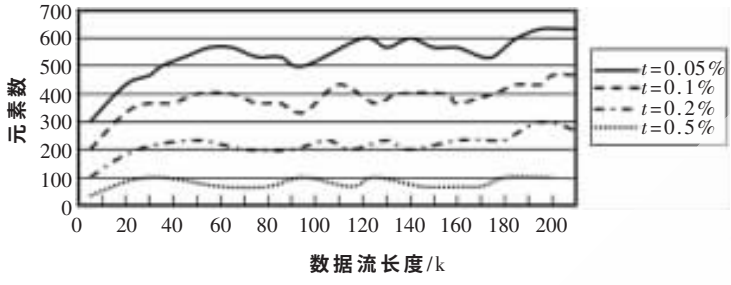


图2 相同窗口不同误差下LW算法的内存占用

本文提出了一种新的基于权重的流数据频繁项挖掘算法。扩展了流数据频繁项的作用域。Lossy Weight算法不仅可用于传统的基于计数的频繁项挖掘，还可以挖掘出在整个流数据中所占权重比重大于门槛值的数据。

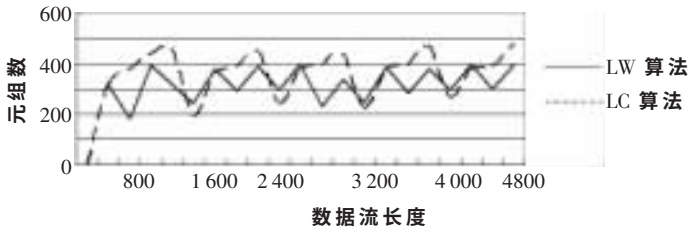


图3 LW算法与LC算法的内存占用对比

## 参考文献

[1] MANKU Q S, MOTWANI R. Approximate frequency counts over data streams[C]. Proc. of the 28th Intl. Conf. on VeD, Large Data Bases. Hongkong: MorganKaufmann, 2002: 346-357.

[2] 潘云鹤，王金龙，徐从富. 数据流频繁模式挖掘研究进展[J]. 自动化学报，2006，32(4)：594-602.

[3] 朱世武，严玉星. 金融数据库[M]. 北京：清华大学出版社，2007：12-14.

(收稿日期：2010-06-27)

作者简介：

杨立，男，1978年生，讲师，硕士，主要研究方向：数据库信息安全。