数据流处理中一种自适应的直方图维护算法*

韩近强+, 杨冬青, 唐世渭

(北京大学 信息科学技术学院,北京 100871)

An Adaptive Algorithm of Histogram Maintain in Data Stream Processing

HAN Jin-Qiang+, YANG Dong-Qing, TANG Shi-Wei

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

+ Corresponding author: Phn: +86-10-62755440, Fax: +86-10-62765822, E-mail: jqhan@db.pku.edu.cn, http://www.pku.edu.cn

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

Han JQ, Yang DQ, Tang SW. An adaptive algorithm of histogram maintain in data stream processing. *Journal of Software*, 2004,15(Suppl.):90~95.

Abstract: Nowadays data stream processing is becoming the new hot field of database research. Due to the volume and rapidness of data in stream, conventional techniques of query processing won't be suitable any more. In such an environment a query is approximate. Histogram is commonly used to describe the distribution of data. This article presents a new algorithm of maintaining histogram under limited memory and guaranteed error. Experiments show that the algorithm is practical and efficient.

Key words: data stream; histogram; quartile

摘 要: 目前数据流的处理成为数据库领域新的研究方向.由于数据流中的数据量大、速度快,传统的查询处理在这种情况下不再适用,这种环境中的查询只能是一种近似查询.直方图通常被用于描述数据的分布.给出了一种新的直方图维护算法,它适用于有限的内存并能保证一定的误差要求.实验证明了算法的实用性和有效性.

关键词: 数据流;直方图;分位数

随着网络技术的飞速发展,数据流(data stream)处理逐渐成为当前数据库领域新的研究热点[1].数据流是一种新的数据处理模型.这种模型中数据不再是永久的关系形式,而是大量的、连续的、快速的、随时间变化的数据流.数据流出现在许多应用中,比如股市交易、网络检测、电话通信、传感器网络等等.虽然数据流中的数据的基本单位还是关系模型中的元组,但是由于数据流的快速而且数据量大,数据流中的所有数据是不可能存储在传统的关系数据库中的.因此数据流系统中的查询和传统的关系数据库不同,数据流的查询是连续查询Continuous Query,即当新数据到达时所有当前活动的查询被执行.数据流中的查询时静态存储在数据库中的,数据是动态的;而传统关系数据库中数据是静态的,用户使用动态的查询语句去查静态的数据,这两种形式是截

* Supported by the National High-Tech Research and Development Plan of China under Grant No.2002AA4Z3440 (国家高技术研究发展计划(863)); 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: 韩近强(1979-),男,硕士生,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与信息系统;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.

然不同的,因此数据流中的查询处理有两个很重要的性质－近似性和自适应性.近似性是指查询出的结果不是准确值,在很多情况下是一个近似值.自适应就是指查询计划是随着数据流的变化而变化的,不是像传统的关系数据库系统查询计划是固定不变的.

目前出现了许多数据流的原型系统,例如 Stanford 大学的 STREAM 系统[2]、加州大学 Berkeley 分校的 TelegraphCQ 系统[3]和布朗大学 Aurora 原型系统[4].STREAM 原型系统是一个数据流管理系统(data stream manager system),它着重处理在内存有限的情况下计算近似的查询结果的情况.TelegraphCQ 是一个自适应的数据流系统,它用自适应的查询执行引擎来处理各种情况下的数据流.Aurora 系统是一个数据流的监视系统,它允许用户定义多个触发器形成触发器网,并且在运行时会对触发器网进行优化.

由于上述特性,数据流的查询处理只能是一趟扫描数据,数据流中的数据一旦流过就不能再次访问,因此必须使用一种方法来近似的描述已经流过的数据.目前所采用的近似查询的方法主要一下几种:随机抽样、数据写生(sketching)[5]、直方图[6~8]、小波变换、滑动窗口(sliding windows)几种.本文采用是直方图方法.直方图作为一种简单描述总体分布的方法原本只是用于传统关系数据库的查询优化,现在被广泛的用于数据流处理中.目前来说主要存在 3 种不同的直方图方法:

等宽直方图:这种直方图最简单.它把整个值域分成等宽长度的桶.Greenwald 等人[9]给出了一种一趟扫描的确定性算法,它需要空间并且可以保证的精度.N 为数据总数,ε为误差要求.

优化的直方图:这种方法中每个桶的宽度可以是不相等的.算法的目的就是为了寻找一种最优分配方案来使误差最小.Jagadish 等人[10]给出一种动态规划的算法来寻找最优的桶的分配方法.他们的算法需要 $O(N)$的空间和 $O(N^2B)$的时间代价.这种方法需要多次扫描数据,这对于数据流来说这种算法显然是不适用的.最近 Gilbert 等人[6]给出一种算法,算法的思想就是把每一个数据元素看成是对一个使用直方图进行近似的向量的更新操作,它的时间和空间的复杂性都是桶数 B,Log$N$ 和 $1/\varepsilon$的多项式.

有目的偏向的直方图(end-biased):这种方法只对出现频率大于一定阈值的数据使用直方图.这种方法主要用于数据挖掘中的冰山查询(iceberg query).一般来说它需要多趟扫描,不适合数据流这种一趟操作的数据模型.Manku 和 Motwani 给出了两种关于频率计数的一趟扫描的近似算法[11].他们的算法需要空间代价并且保证错误个数小于εN.

我们的算法和上面的 3 种算法有相似之处也有不同之处.我们认为等宽的直方图在数据流环境下是不适用的.由于数据分布的不确定性,数据比较集中的部分我们应该使用更多的桶来描述数据的分布,这样得到的查询结果才能更加精确.最优化的方法在数据流系统中也是不适用,一是因为最优化算法时间复杂性比较高,而且要求数据是已知.这对于数据流处理显然是不适用.有偏直方图只是适用于特殊情况并不适用于一般情况.我们要给出了一种简单的快速的直方图维护算法,它满足内存的限制和误差要求,而且还适用于数据不连续分布的情况.我们在前人的工作中没有找到的能满足所有这些特性的算法,我们给出的这个算法将是第一个.

本文第 1 节我们给出问题的定义.第 2 节我们描述了具体的算法.第 3 节我们给出我们算法正确性的证明.第 4 节我们对我们的算法进行了试验.最后的部分我们给出结论.

1 问题定义

不失一般性的,我们考虑关系 R 中的一个数值属性 A.属性 A 的值域为 $V=\{v_i|1\le i\le\infty\}$,并且 $v_i<v_j$ 如果 $i<j$.每个值 v_i 出现的频率为 f_i.记 A 的频率向量 $F=\{f_1,f_2,\ldots\}$.

本文的主要工作就是要维护最大桶数为 B(内存限制)的直方图 H 来近似 F,并且分位数(quartile)总体误差小于ε.

定义 1(桶). 桶 *Bucket* 为一个三元组(max,min,*frequency*):max 为桶中的最大取值,min 为最小取值,*frequency* 为在此数值范围内总频率.

定义 2(线密度). 单位取值范围上的频率,也就是桶的总频率除以桶中数据的取值范围,即 *frequency*/(max−min).

定义 3(分位数). 百分比为 p 的数值小于 $x(x\in V)$,则称 x 为对于 p 的分位数.这种查询也是会经常遇到的,例

如要查询 75%的人年龄小于多少?

定义 4(分位数误差). 实际的分位数 x_p 和近似出来的分位数 x 的差别.即(x_p-x).

定义 5(分位数总体误差). 分位数误差和总体取值范围的比.即$(x_p-x)/(\max-\min)$.

需要说明的是我们使用的误差要求是要满足一定的分位数误差ε,这一点与前人的误差要求不同.前人一般都是使用的εN 的误差要求.N 为流过的数据总数,ε为相对总数误差要求.可以看出我们的误差要求不是相对数据的总数的而是相对数据的取值范围 Max-Min 的误差要求.在回答一些查询的时候,比如范围查询或者分位数查询,用户要求的更多的是分位数的误差要求.而且由于一般的数据流中数据的取值范围变化比较小,因此我们的误差要求随数据量的变化要小的多,这就是我们误差要求的优势所在.

2 算法描述

算法的主要思想就是对于数据流中的每个数值 v,如果它属于某个已经存在的桶 b,则该桶 b 的频率加 1;如果不属于则新建一个桶以该值为中心,宽度为最大分位数误差要求.如果和相邻的桶 $b1$ 和 $b2$ 出现重叠时则收缩到最小,如图 1 所示.

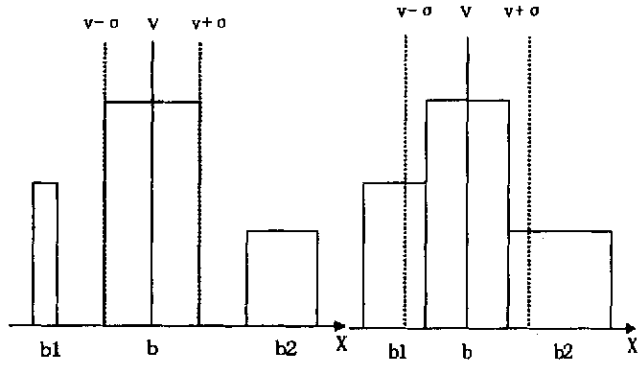

图 1 桶的建立

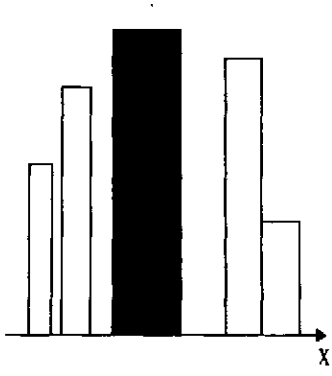

图 2 桶的合并

如果桶的总数超过限制 B,我们合并相邻的桶.合并算法的主要思想就是寻找距离最近的桶中、线密度最接近的两个桶进行合并.我们只合并最"接近"的两个桶,我们认为最大的利用内存空间才能使近似计算更加准确,所以在内存空间达到最大值时,只合并最优的两个桶,为新桶的建立准备出内存空间.图 2 描述了合并桶的情况,对于图中的 3 组相邻桶来说,最优的合并桶为阴影部分的两个桶,因为它们距离最近,并且线密度最接近.

下面给出我们的算法:

算法 1.

记录新数据 v 到当前的直方图中.

输入:桶数 B(内存限制)和最大的分位数总体误差ε,数据流数据 v.

输出:新的自适应的直方图.

如果 max<v 则 max=v;

如果 min>v 则 min=v;

总体计数加 1

计算最大允许分位数误差$\sigma=(\max-\min)*\varepsilon$;

对于所有的桶

如果 v 属于某个已经存在的桶

则该桶频率加 1

否则

$b=[v-\sigma,v+\sigma]-b1-b2$($b1$ 和 $b2$ 为和 v 相邻最近的两个桶)

如果还有内存可以作为桶

则加入 b 桶;

否则合并两个桶,然后加入 b 桶.

算法 2.

合并两个桶

输入:直方图的所有桶.

输出:合并后的所有桶.

最小距离 min_distant=∞

合并桶 $B1$ 和 $B2$ 为空

对于所有的相邻的桶 $b1$ 和 $b2$

如果两个的桶的距离 *distant* 小于最小距离 min_*distant*

min_*distant*=*distant*;

$B1=b1,B2=b2$

如果 *distant* 等于 min_*distant* 并且 $b1$ 和 $b2$ 的线密度的差距小于 $B1$ 和 $B2$

则 $B1=b1,B2=b2$

(下边合并 $B1$ 和 $B2$,假设 $B1$ 在 $B2$ 前)

$B1$ 的频率=$B1$ 的频率和 $B2$ 的频率之和

$B1$.min=min($B1$.min,$B2$.min)

$B1$.max=max($B1$.max,$B2$.max)

删除桶 $B2$

有了近似直方图后,计算近似的分位数计算和近似的范围计算都是很容易的.下面我们给出一个近似范围计算的算法.

算法 3. 近似的范围计算.

输入:直方图的所有桶,最大值 MAX 和最小值 MIN.

输出:所有数中介于 MAX 和 MIN 之间的近似个数.

计数器 *frequency* 置 0

对于所有桶 *current*

I=*current*∩(MIN,MAX)

如果 I 为空则继续循环

如果 I 等于 *current*

则 *frequency*+=*current* 的频率

否则 *frequency*+=*current* 的频率*(I.max−I.min)/(*current*.max−*current*.min);

返回 frequency;

3 算法正确性分析

定理 1. 当 $B>1/\varepsilon$时,合并的两个桶必然相邻.

证明:由于建立时每个桶的宽度都等于 ε*(max−min),那么总宽度 ε*(max−min)*$B>\varepsilon$*(max−min)*$1/\varepsilon$ > (max−min).根据抽屉原理,必然有两个桶相邻.

定理 2. 当 $B>1/\varepsilon$时,如果两个桶都满足分位数总体误差<ε,合并之后还满足分位数总体误差<ε.

证明:首先当 $B>1/\varepsilon$时,由定理 1 我们合并的肯定是相邻的桶.我们合并桶的线密度是近似相同的,因此实际

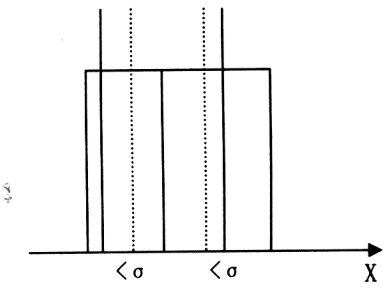

图 3 桶合并后的总体误差

分位数在前面桶的估计值还在前面的桶的范围内,误差不会超过桶宽,也就是不会超过分位数误差σ.如图 3 所示,实线为实际的分位数,虚线为近似的分位数.换句话说就是由于线密度近似没有变化,原来得近似值和合并后的近似值也近似没有变化,因此分位数误差还小于σ.同理,实际分位数在后边的桶的估计值还在后边的桶的范围内,因此分位数误差也不会超过σ.

定理 3. 当 $B>1/\varepsilon$时,该算法可以保证总体的分位数误差小于ε.

证明:在初始化的时候,我们建立的每个桶保证了绝对误差小于等于ε*(max−min),而由于 min 单调减小,而 max 单调增加,所以 max−min 单调增加,分位数误差ε*(max−min)单调增加,原来的建立的桶依然满足给定分位数误差要求.当桶合并时,由定理 2 可知合并后的桶依然满足误差要求.

4 实验分析

我们采用使用平均分布的数据作为测试用例,和 Greenwald 等人的算法进行了比较.平均分布的数据有 3 种数据规模:1 万、10 万、100 万.我们测试环境为:CPUP4 2G,内存 256M,硬盘 40G,操作系统为 Windows XP.

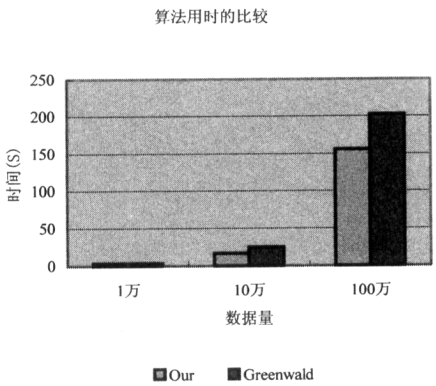

图 4 算法用时

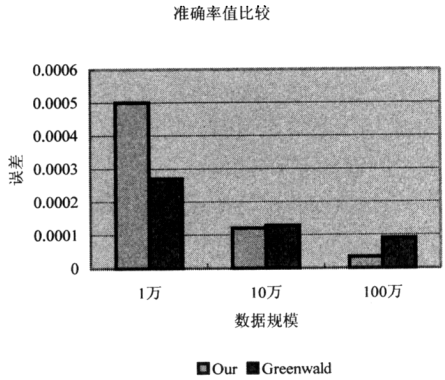

图 5 算法准确率

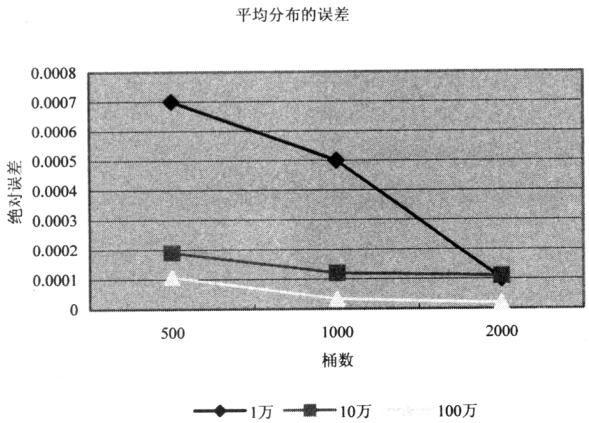

图 6 误差随桶数变化的变化规律

从实验结果中我们可以得出下面的结论:

(1) 我们的算法速度要比 Greenwald 略快,在大数据量上要比 Greenwald 的算法更加准确.

(2) 桶的个数越多算法准确率越高.

(3) 一般来说 B 的取值略大于 $1/\varepsilon$为最好.这样算法用时可以接受,准确率即能得到保证.

5 结论和将来工作

本文给出了一种维护数据流中数据直方图的自适应算法.我们给出的算法有几个优点:算法简单快速、适合数值不连续分布的情况,可以在给定的内存限制下运行并且可以满足一定的误差要求.实验证明它有很好的实用性.

下一步我们会把算法集成到数据流系统的查询引擎中,并使用采样的方法来加快算法的执行速度.

References:

[1] Babcock B, Babu S, Datar M, Motwani R, Widom J. Models and issues in data stream systems. In: Proc. of the 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. New York: ACM Press, 2002. 1~16.

[2] Motwani R, Widom J, Arasu A, Babcock B. Query processing, resource management, and approximation in a data stream management system. In: Proc. of the 1st Biennial Conference on Innovative Data Systems Research (CIDR 2003). 2003. 245~256.

[3] Chandrasekaran S, Cooper O, Desppande A, Franklin MJ. TelegraphCQ: Continuous dataflow processing for an uncertain world. In: Proc. of the 1st Biennial Conference on Innovative Data Systems Research (CIDR 2003). 2003. 269~280.

[4] 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 (VLDB 02). 2002. 215~226.

[5] Alon N, Matias Y, Szegedy M. The space complexity of approximating the frequency moments. In: Proc. of the 1996 Annual ACM Symp. on Theory of Computing. 1996. 20~29.

[6] Gilbert A, Guha S, Indyk P, Kotidis Y, Muthukrishnan S, Strauss M. Fast, small-space algorithms for approximate histogram maintenance. STOC, In: Proc. of the 2002 Annual ACM Symp. on Theory of Computing. 2002. 389~398.

[7] Gehrke J, Korn F, Srivastava D. On computing correlated aggregates over continual data streams. In: Proc. of the 2001 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2001. 13~24.

[8] Yang J, Widom J. Incremental computation and maintenance of temporal aggregates. In: Proc. of the 17th Int'l Conf. on Data Engineering table of Contents. Washington: IEEE Computer Society, 2001. 51~60.

[9] Greenwald M, Khanna S. Space-Efficient online computation of Quantile summaries. In: Proc. of the 2001 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2001. 58~66.

[10] Jagadish H, Koudas N, Muthukrishnan S, Poosala V, Sevcik K, Suel T. Optimal histograms with quality guarantees. In: Proc. of the 24th Int'l Conf. on Very Large Data Bases. San Francisco: Morgan Kaufmann Publishers, 1998. 275~286.

[11] Manku G, Motwani R. Approximate frequency count over data streams. In: Proc. of 28th Int'l Conf. on Very Large Data Bases. Hong Kong: Morgan Kaufmann Publishers, 2002. 346~357.