计算机研究与发展
Journal of Computer Research and Development
ISSN 1000-1239/CN 11-1777/TP
45(2):287~292, 2008

数据流上的分位数近似算法研究

杨 蓓[1,2] 黄厚宽[1]
[1](北京交通大学计算机与信息技术学院 北京 100044)
[2](郑州大学信息工程学院 郑州 450001)
(yangbei@zzu.edu,cn)

Research on an Algorithm for Approximate Quantile Computation over Data Streams

Yang Bei[1,2] and Huang Houkuan[1]
[1](*School of Computer and Information Technology*, *Beijing Jiaotong University*, *Beijing* 100044)
[2](*School of Information Engineering*, *Zhengzhou University*, *Zhengzhou* 450001)

Abstract Data stream is a new data model that has attracted attentions in numerous applications such as traffic monitoring, telephone records management, sensor networks, stock-market analysis, Web click streams, *etc*. The importance of quantile estimation of data streams has been highlighted by more and more researchers in recent years. Due to the characteristics of continuity and boundlessness of streaming data, it is unfeasible to memorize the entire information of data streams and thus difficult to get the exact answer to the query on streaming data. In this paper, a novel synopsis data structure—Nord_Histogram for storing streaming data summary is designed to get a balance between the memory cost and the query accuracy, and a one-pass online approximate algorithm for quantile computation is presented. The algorithm implements the approximate quantile queries over data stream with the time and space requirements being linear with the number of the buckets, which has no business with the length of data streams, and therefore has great scalability. This method has very good performance on data with uniform distribution. The correlation between the computation accuracy and main memory requirement is also analyzed. Experimental results show that the algorithm brings about quite small relative errors and works well over data streams.

Key words data stream; synopsis data structure; histogram; quantile; approximate algorithm

摘 要 数据流是一种新型数据模型,广泛应用于交通流量监控、通信管理、传感器网络、股票分析、Web点击流等众多领域.近年来越来越多的学者关注于数据流上的分位数计算研究.由于流数据的连续、无界、易失等特性,存储完整的流数据信息并得到精确的查询结果几乎是不可能的.在实施查询计算时追求内存用量与查询精度之间的最佳均衡.设计了规范数直方图的概要数据结构以存储流数据的摘要信息,并在此基础上提出了单遍扫描的、联机的分位数近似算法,其时间和空间复杂度均线性于概要结构中桶的个数,而与数据流的长度无关,因而具有很好的可规模性.该方法在均匀分布的数据上取得了优良性能.分析了算法精度与内存需求的关系.实验结果表明该算法具有较精确的查询结果,具备良好的实用性和有效性.

关键词 数据流;概要数据结构;直方图;分位数;近似算法

中图法分类号 TP311

随着计算机网络和通信技术的迅猛发展,一种称为数据流(data stream)[1]的新型数据模型应运而生.该数据模型广泛应用于交通拥塞监测与管理、网络流量监控、电信数据管理、金融服务、商业交易管理和分析、传感器网络等众多领域.近年来,数据流处理技术引起了国内外许多研究机构和学者的关

收稿日期:2006-12-26;修回日期:2007-08-27
基金项目:国家"九七三"重点基础研究发展规划基金项目(2006CB705500)

注.数据流模型不同于传统的数据库数据模型,具有实时性、连续性、无界性、无序性等特点.在传统的数据库中,数据被静态地存储在介质中,可以被多次访问;查询计划是静态的、单次的,且最终生成确定的查询结果;算法以时间及空间复杂度作为性能评价标准.流数据以大量的、快速的、无法预测的、无限制的流的形式到达,与传统的关系型数据有以下显著区别:

1)数据是在线联机到达的;

2)系统无法控制数据到达的次序,可以是单流或多流的形式;

3)数据流大小可能是无界的;

4)数据流中的元素一经处理则被抛弃,除非特意保存,不可能像传统数据库那样将其完全保存到介质上,再次提取数据代价昂贵甚至不可实现.

流数据的特点决定了其分析技术通常只能是一次处理,其算法应是单遍扫描(one-pass).由于存储容量的有限性,不可能完整地保存全部数据流元素.考虑设计一个远小于原数据流规模的结构,保存已流过数据的概要特征,以便数据流的查询处理及分析.由此,概要数据结构(synopsis data structure)[2]的设计成为数据流技术的热点研究问题之一.

1 相关研究

由于数据流的到达是连续的且可能是无界的,而存储器的容量是有限的,因此很难获得数据流上的精确查询结果.在许多实际应用中,用户并不需要获得精确的查询值,仅仅需要一个近似结果即可.设计概要数据结构抽取原数据的总体分布,以获得查询精度与内存的最佳均衡.常用的概要数据结构构建技术包括随机抽样(reservoir sampling)[3]、精确抽样(concise sampling)[4]、Sketching 技术[5]、直方图技术(histogram)[6-9]和小波技术(wavelet)[10]等.

直方图是一种常用的概要数据结构,使用分箱技术近似反映数据的分布特征,可用于多种任务环境,如分位数估计、冰山查询等.直方图划分为等宽直方图[6](equi-width histogram)、V-优化直方图[7](V-optimal histogram)、指数直方图[8](exponential histogram)等.多数直方图算法假定数据集是确定且有限的,查询处理的时间和空间线性(或次线性)于流过的数据量,有些需要多次扫描数据集,不适合于数据流的查询.

Govindaraju 等人[8]研究了基于图形处理器GPU(graphic processor units)计算大规模数据流的分位数和频数近似算法,以指数直方图作为摘要数据结构,需要维护当前滑动窗口上的 $\lceil \log N \rceil$ 个指数直方图,N 为滑动窗口的宽度.Greenwald 等人[11]提出了确定的单遍扫描算法,利用等宽直方图有效地计算数据集的分位数.算法的空间复杂度为 $O(1/\varepsilon \log \varepsilon N)$,其中 ε 为误差精度,N 为数据流的长度.Lin 等人[12]研究了数据流上的最近 N 个元素的分位数近似算法,其最大空间需求及误差上界分别为 $O(\log(\varepsilon^2 N)/\varepsilon + 1/\varepsilon^2)$ 和 εN.Arasu 等人[13]提出的基于滑动窗口的分位数算法需要 $O(1/\varepsilon \operatorname{polylog}(1/\varepsilon, N))$的主存空间.

上述研究在时间或空间上均线性(或次线性)于数据流大小,随着流元素的增加,维护开销不断增大.本文设计了一个基于规范数直方图——Nord(normalized data)_Histogram——的概要结构,并在此基础上实现了数据流的分位数近似查询算法 NORMAL,其时间和空间复杂度均线性于概要结构中桶的个数,而与数据流的长度无关.理论分析及实验均表明该算法时间空间复杂度低,准确性高.

2 问题的提出及相关定义

2.1 问题的提出

分位数计算是大数据集和数据流上经常使用的一种统计方法.设 $x_i(i=1,\cdots,N)$是按递增序排序的数据,使得 x_1是最小的观测值,而 x_N是最大的观测值.每个观测值 x_i与一个 $f_i(0<f_i<1)$配对,指出大约 $100\times f_i\%$的数据小于或等于 x_i,则 x_i是相应于 f_i的分位数.

通过分位数查询能够获得统计信息,以便为决策管理层提供数据支持.例如,将移动电话公司的通话记录视为数据流,每一条通话记录视为一个数据元素(为描述方便,我们仅考虑记录中的通话时长属性).移动公司可能需要获取这样的信息:70%的用户的通话时间长度不超过多少?将数据流元素按通话时长 x_i顺序存储,此问题即是计算与 $f_i=70\%$对应的 x_i.鉴于通话记录的数量极大,不可能全部存储,考虑设计一个满足存储限制的且能够大致捕获原始数据分布的概要数据结构,为问题的解决提供数据支持.

2.2 相关定义

在我们的概要数据结构中涉及以下几个概念:

定义 1. 数据跨度 $SPAN_D$ 是数据流 D 中元素的值域,其长度等于 $MAX_D - MIN_D + 1$,其中 MAX_D 和 MIN_D 分别表示数据流 D 中的元素最大值和最小值.

例如,在数据流 D 中,e_1 是最大值,e_2 是最小值,则 $SPAN_D = [e_2, e_1]$,且 $|SPAN_{D_i}| = e_1 - e_2 + 1$.

定义 2. 数据跨度中的任一元素可通过某函数映射为较小范围内的值,概构跨度(或规范跨度) $SPAN_N$ 就是数据流元素经此函数的映射值域,本文中该函数为 f_N(见定义 3). 令 MAX_N 和 MIN_N 分别表示流元素的最大和最小映射值,则 $SPAN_N = [MIN_N, MAX_N]$,$|SPAN_N| = MAX_N - MIN_N + 1$.

概构跨度与内存容量紧密相关,在我们的算法中,其中的数据可完全保留在内存中.

定义 3. 规范函数 $f_N: SPAN_D \rightarrow SPAN_N$ 将数据流元素 e 映射为规范数 $NORMe$:

$$NORMe = f_N(e) = \frac{e - MIN_D}{|SPAN_D - 1|} |SPAN_N - 1| + MIN_N. \quad (1)$$

在任意时刻,当有新的数据元素到达时,利用式(1)可计算出该元素的规范数 $NORMe$.

定理 1. 对于数据跨度 $SPAN_D$ 上的任意数据元素 e,其对应的规范数 $NORMe$ 一定在概构跨度 $SPAN_N$ 中.

证明. 令 e 为 $SPAN_D$ 上的任意元素,则 $e \leqslant MAX_D$.

$$NORMe = f_N(e) = \frac{e - MIN_D}{|SPAN_D - 1|} |SPAN_N - 1| + MIN_N \leqslant \frac{MAX_D - MIN_D}{MAX_D - MN_D}(MAX_N - MIN_N) + MIN_N = MAX_N,$$

即 $NORMe \leqslant MAX_N$.

同理可证:$NORMe \geqslant MIN_N$. 命题得证.

对于数据跨度范围内的任意数据元素,通过规范化映射可转化为概构跨度内的数据,因而可以保存在内存中. 算法使用规范数直方图作为维护数据流摘要信息的概要数据结构.

定义 4. 一个规范数直方图($Nord\_Histogram$)是一系列规范桶($Nord\_Bucket$)的集合. 规范桶是一个二元组($NORMe$, $count$),其中 $count$ 是迄今为止的数据流中被映射为 $NORMe$ 的流元素的个数.

每当一个新的数据元素 e 到达,算法 NORMAL 计算出其相应的 $NORMe$,同时其对应的规范桶被更新. 规范数直方图能及时记录数据元素的摘要信息,因而可以用于实现数据流的分位数近似计算.

3　算法描述

算法 NORMAL 可分为两个过程:1) $Nord\_Histogram$ 的构建与更新;2) $Nord\_Histogram$ 上的分位数计算. 在过程 1)中,令 Sp 表示概构跨度,取决于内存容量的限制,其上下界分别表示为 MAX_N 和 MIN_N. 数据流元素的最大与最小值分别表示为 MAX_D 和 MIN_D. 为 Sp 内的每个值创建一个 $Nord\_Bucket$. 每当一个新的数据元素 e 到达,计算出其相应的 $NORMe$,同时更新其对应的规范桶,从而及时得到更新的 $Nord\_Histogram$. 具体实现步骤如下:

过程 1):$Nord\_Histogram$ 的构建与更新

输入:数据流元素 e、概构跨度 Sp、数据流的最大值 MAX_D、数据流的最小值 MIN_D.

输出:规范直方图 NH、数据流计数 N.

① 对于 Sp 内的每一个值 v_i,构建一个桶,并对其初始化;

② 若元素 e 不在数据流的最大与最小值范围内(极其个别的情况),将其强制转换为最大值或最小值;

③ 计算元素 e 对应的规范数 $NORMe$;

④ 更新 $NORMe$ 对应的规范桶 $Nord\_Bucket$,所有 $Nord\_Bucket$ 的集合构成 NH;

⑤ 更新数据流计数 N.

在过程 1)中,构建和维护 NH 仅需 $O(B)$ 的时间和空间,其中 B 是桶的个数;当新的数据元素到来时,计算及更新规范桶仅需 $O(1)$ 的时间和空间,这些均与数据流元素个数 N 无关,这正是 NORMAL 的特点所在.

过程 2):实现分位数的计算.

具体的算法步骤如下:

输入:规范直方图 NH、分位数 q、当前数据流计数 N.

输出:对应于 q 的数据流元素值 Vq.

① 依据 N 和 q 计算出对应于 q 的元素个数 Cq;

② 从 NH 中的第 1 个桶开始,累计每个桶 $Nord\_Bucket$ 的计数项,直到达到或超过 Cq 为止,令此时有 i 个桶被累计,累计和为 sum;

③ 若 sum 刚好达到 Cq 值,则依据 $Nord\_Bucket_i$ 对应的 $NORMe_i$,反向使用式(1)计算出 e_i 的

值,即为所求的 Vq,即 $Vq = NORMe_i \times (MAX_D - MIN_D + 1)/|Sp|$;

④ 若 sum 超过 Cq 值,则在计算 Vq 时需按比例减去超出的部分,即

$$Vq = (NORMe_i - (sum - Cq)/Nord\_Bucket_i.count) \times (MAX_D - MIN_D + 1)/|Sp|;$$

⑤ 返回 Vq.

过程 2)计算分位数值 Vq 的时间复杂度为 $O(B)$(步骤②),而空间复杂度为 $O(1)$. 综合过程 1)和过程 2),基于规范数直方图 $Nord\_Histogram$ 的分位数近似算法的时间和空间复杂度均为 $O(B)$,与数据流的长度 N 无关,因而可以实现数据流的及时查询计算.

此外,由上述算法过程,对分位数近似计算的误差精度进行分析,可以得到如下定理.

定理 2. 设概构跨度为 $SPAN_N$,则由算法 NORMAL 计算所得的分位数误差为 $\varepsilon \leqslant \frac{1}{|SPAN_N|}$.

证明. 令 NORMAL 的数据跨度为 $SPAN_D$,由过程 2)和式(1)知,$SPAN_D$ 中的元素被均匀地映射到概构跨度 $SPAN_N$ 中,且有 $t = \frac{|SPAN_D|}{|SPAN_N|}$ 个数据元素被映射为同一个规范数 $NORMe$. 因此在过程 2)中由规范数计算分位数时,至多会产生 t 个元素的误差,因而误差精度为 $\varepsilon \leqslant \frac{t}{|SPAN_D|} = \frac{1}{|SPAN_N|}$.

命题得证.

定理 2 揭示了算法误差的上界反比于概构跨度,但下面的实验结果显示,我们的实验误差远远小于该上界. 只有当数据分布极端倾斜时才会接近上界.

算法 NORMAL 的内存需求与概构跨度密切相关. 依定理 2 可知,概构跨度愈大误差愈小,由此可以得到以下推论.

推论 1. 对于给定 ε $(0 < \varepsilon < 1)$,当算法 NORMAL 的概构跨度不小于 $\frac{1}{\varepsilon}$ 时,计算误差不大于 ε.

证明. (略).

4 实验及分析

我们在不同的随机数据集上对 NORMAL 加以验证,将其与基于 reservoir sampling(简称 Reservoir)和 concise sampling(简称 Concise)概要数据结构上进行的分位数查询在误差和时间复杂度方面进行了比较,并且与文献[11]中的分位数查询算法在误差精度与内存需求方面进行对比分析. 实验证明我们的算法能够在与数据流大小无关的时间和空间取得较小的分位数查询误差.

首先,生成不同大小的随机数据集. 假定概构跨度为 1000,亦即桶的个数为 1000. 之所以选取这样一个较小的数据,是因为我们认为算法能够在大的概构跨度上工作得更好. 依定理 2 知,概构跨度越大(即桶个数越多)查询精度越高. 表 1 显示了当数据总量为 200000,数据跨度从 20000 变化到 100000(变化增量为 20000)时分位数的计算误差. 表 2 记录了当数据跨度固定为 10000、数据总量从 100000 变化到 180000(变化增量为 20000)时的分位数计算误差. 从表 1 和表 2 可以观察到,误差并没有随着数据跨度或数据总量的增大而增加,基本稳定保持在 10^{-5} 数量级(个别较明显的变化可能是由于随机数据的倾斜分布引起的),且远小于误差上限(0.001).

Table 1 Errors of *Nord_Histogram* Based Approximate Quantile Computation Algorithm with $N = 200000$

表 1 基于 *Nord_Histogram* 的分位数计算误差(数据总量为 200000)

q_i(%)	$SPAN_D$				
	20000	40000	60000	80000	100000
5	0.0000364502	0.0000674316	0.0000350016	0.0000171967	0.0000369434
15	0.0000421021	0.0000141357	0.0000071452	0.0000098145	0.0000076563
25	0.0000166748	0.0000051270	0.0000551270	0.0000112061	0.0000091797
35	0.0000012207	0.0000159180	0.0000284505	0.0000138428	0.0000157813
45	0.0000714355	0.0000223145	0.0000114583	0.0000046387	0.0000305469
55	0.0000162598	0.0000569824	0.0000386068	0.0000155762	0.0000171094
65	0.0000359375	0.0000180176	0.0000508464	0.0000114258	0.0000491797
75	0.0000238770	0.0000693359	0.0000199219	0.0000432617	0.0000222656
85	0.0000300781	0.0000009766	0.0000146484	0.0000211914	0.0000275781
95	0.0000453125	0.0000193359	0.0000167318	0.0000259766	0.0000337500
Average Errors	0.0000319348	0.0000289575	0.0000277938	0.0000174130	0.0000249990

Table 2　Errors of *Nord_Histogram* Based Approximate Quantile Computation Algorithm with $SPAN_D$ = 10000

表 2　基于 *Nord_Histogram* 的分位数计算误差(数据跨度为 10000)

q_i(%)	N				
	100000	120000	140000	160000	180000
5	0.0000017548	0.0000136353	0.0000303040	0.0000425293	0.0000340668
15	0.0000795410	0.0000348511	0.0001044189	0.0000359375	0.0000885498
25	0.0001410645	0.0000348877	0.0000060303	0.0000615234	0.0000212158
35	0.0000429932	0.0000840088	0.0000255859	0.0000337402	0.0000830566
45	0.0000844727	0.0000065430	0.0000472168	0.0000720703	0.0000885742
55	0.0000032227	0.0000379395	0.0000551758	0.0000675781	0.0001133789
65	0.0000713867	0.0000571289	0.0000012695	0.0000829102	0.0001011230
75	0.0000494141	0.0000129395	0.0000889160	0.0001000000	0.0000390625
85	0.0001027344	0.0000957031	0.0000340820	0.0000218750	0.0000833008
95	0.0000316406	0.0000636719	0.0001032227	0.0000568359	0.0000010742
Average Errors	0.0000608224	0.0000441309	0.0000496222	0.0000575000	0.0000653403

表 3 将 NORMAL 与基于 Reservoir 和 Concise 的分位数查询误差进行比较. 显示了当数据总量为 100000、数据跨度为 10000、概构跨度为 1000 时 3 种算法的分位数计算误差结果:

Table 3　Comparison with Sampling_Based Algorithms on Errors and Time Complexity

表 3　与基于抽样算法的误差及时间复杂度的比较

q_i(%)	Algorithm		
	NORMAL	Reservoir	Concise
5	0.0000017548	0.0037	0.0066
15	0.0000795410	0.0021	0.0021
25	0.0001410645	0.0010	0.0144
35	0.0000429932	0.0005	0.0031
45	0.0000844727	0.0023	0.0058
55	0.0000032227	0.0049	0.0145
65	0.0000713867	0.0099	0.0120
75	0.0000494141	0.0165	0.0099
85	0.0001027344	0.0235	0.0176
95	0.0000316406	0.0133	0.0044
Average Errors	0.0000608224	0.0078	0.0090
Time Complexity	$O(B)$	$O(B\log B)$	$O(B\log B)$

Note: With the Number of Data is 100000, $SPAN_D$ = 10000, $SPAN_N$ = 1000)

经观察可知,NORMAL 的误差比 Reservoir 和 Concise 的误差小很多,约相差两个数量级. 在时间复杂度方面,为了进行分位数的计算,Reservoir 和 Concise 的样池中的数据应该是有序的,这就要求每当新的数据元素到来时必须更新样池,并将新元素顺序插入样池,此过程涉及到查找和排序问题,因而至少需要$O(B\log B)$的平均时间.

此外,表 4 报告了算法 NORMAL 与文献[11]研究的两个算法(Preallocated 和 Adaptive)的实验结果. 设定数据总量为 1000000,算法 Preallocated 和算法 Adaptive 的误差上界限定为 0.001;NORMAL 的数据跨度为 50000,概构跨度为 1000. 表中 Space 表示算法中桶的平均个数. 实验表明,与 NORMAL 相比,Preallocated 有相近的误差,但需要更多的存储空间(约 5 倍于算法 1)以保证算法精度;Adaptive 虽在空间上略优于 NORMAL,但查询精度约劣一至两个数量级.

Table 4　Comparison with Equi-Width Histogram_ Based Algorithms on Errors and Memory Cost

表 4　与基于等宽直方图算法的计算误差及内存空间的比较

$q_i(i/16)$	Algorithm		
	NORMAL	Preallocated	Adaptive
1	0.00001815	0.0000541	0.0003173
2	0.00001024	0.0000579	0.0003259
3	0.00000236	0.0000573	0.0003172
4	0.00002664	0.0000557	0.0003546
5	0.00002764	0.0000545	0.0002907
6	0.00000363	0.0000589	0.0002972
7	0.00000516	0.0000503	0.0002951
8	0.00001391	0.0000455	0.0002892
9	0.00000473	0.0000588	0.0003015
10	0.00000309	0.0000714	0.0002924
11	0.00000484	0.0000581	0.0002989
12	0.00000555	0.0000486	0.0003378
13	0.00000000	0.0000530	0.0003128
14	0.00000094	0.0000565	0.0003146
15	0.00001922	0.0000545	0.0002797
Average Errors	0.00000913	0.0000557	0.0003083
Space	1000	5052	920

5 结 语

数据流的出现向数据挖掘与处理研究领域的众多学者提出了新的挑战.鉴于流数据的种种特性,在实施查询计算时追求内存用量与查询精度之间的最佳均衡.本文提出了基于规范数直方图的概要结构构建方法,在此基础上设计了数据流的分位数查询近似算法,分析了算法误差与内存需求之间的相关性.理论分析及实验结果表明,该算法能够在与数据流长度无关的内存空间和时间内取得较小的查询误差,具备良好的实用性和有效性.

参 考 文 献

[1] B Babcock, S Babu, M Datar, *et al*. Models and issues in data streams [C]. In: L Popa ed. Proc of the 21st ACM SIGACT-SIGMOD-SIGART Symp on Principles of Database Systems. New York: ACM Press, 2002. 1-16

[2] P B Gibbons, Y Matias. Synopsis data structures for massive data sets [C]. The 10th Annual ACM-SIAM Symp on Discrete Algorithms, Baltimore, 1999

[3] J S Vitter . Random sampling with a reservoir [J]. ACM Trans on Mathematical Software, 1985, 11(1): 37-57

[4] P B Gibbons, Y Matias. New sampling-based summary statistics for improving approximate query answers [C]. In: L M Haas, A Tiwary, eds. Proc of the ACM SIGMOD Int'l Conf on Management of Data (SIGMOD1998). New York: ACM Press, 1998. 331-342

[5] M Charikar, K Chen, M Farach-Colton. Finding frequent items in data streams [J]. Theoretical Computer Science, 2004, 312(1): 3-15

[6] P B Gibbons, Y Matias, V Poosala. Fast incremental maintenance of approximate histograms [C]. In: M Jarke, M J Carey, K R Dittrich, eds. Proc of the 23rd Int'l Conf on Very Large Data Bases VLDB '97. San Francisco: Morgan Kaufmann, 1997. 466-475

[7] Y Ioannidis, V Poosala. Balancing histogram optimality and practicality for query result size estimation [J]. SIGMOD Record, 1995, 24(2): 233-244

[8] N K Govindaraju, N Raghuvanshi, D Manocha. Fast and approximate stream mining of quantiles and frequencies using graphics processors [C]. The 2005 ACM SIGMOD Int'l Conference, Baltimore, 2005

[9] Zhang Longbo, Li Zhanhuai, Yan Jianfeng. Histogram data streams with fast incremental maintenance [J]. Computer Engineering, 2005, 31(14): 83-84 (in Chinese)
(张龙波,李战怀,闫剑锋.一种面向数据流处理的直方图增量维护算法[J].计算机工程,2005,31(14):83-84)

[10] Y Matias, J Vitter, M Wang. Wavelet-based histograms for selectivity estimation [C]. The 1998 ACM SIGMOD Int'l Conf on Management of Data, Seattle, 1998

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

[12] X Lin, H Lu, J Xu, *et al*. Continuously maintaining quantile summaries of the most recent N elements over a data stream [C]. The 20th Int'l Conf on Data Engineering, Boston, 2004

[13] A Arasu, G S Manku. Approximate counts and quantiles over sliding windows [C]. In: Proc of the 23rd ACM PODS 2004 Conference. New York: ACM Press, 2004. 286-296

Yang Bei, born in 1967. Ph. D. candidate in computer science from Beijing Jiaotong University, Beijing, China. Student member of China Computer Federation. Her main research interests include data mining, data stream management and artificial intelligence.

杨 蓓,1967年生,博士研究生,中国计算机学会学生会员,主要研究方向为数据挖掘、数据流管理、人工智能.

Huang Houkuan, born in 1940. Professor and Ph. D. supervisor. Senior member of China Computer Federation. His main research interests include artificial intelligence, data mining and machine learning.

黄厚宽,1940年生,教授,博士生导师,中国计算机学会高级会员,主要研究方向为人工智能、数据挖掘、机器学习等(hkhuang@bjtu.edu.cn).

Reseach Background

Statistics over streaming data elements are often required in applications such as traffic monitoring, network intrusion detection, telephone records management, stock price prediction in financial markets, Web log mining for access prediction, and user click stream mining. Among various statistics, computing quantile summary is probably most challenging because of its complexity. In this paper, we study the problem of continuously maintaining quantile summary over a data stream so that quantile queries can be answered in time. We develop a synopsis data structure—Nord_Histogram for storing streaming data summary to get a balance between the memory cost and the query accuracy, and a one-pass approximate algorithm NORMAL for quantile computation. The algorithm implements the approximate quantile queries over data stream with the time and space requirements being linear with the number of the buckets, which has no business with the length of data streams. The correlation between computation errors and main memory required is also analyzed. Our work is supported by the National Grand Fundamental Research 973 Program of China under grant No. 2006CB705500.