差分隐私流数据实时发布中的自适应参数优化

吴英杰 黄 鑫 葛 晨 孙 岚
(福州大学数学与计算机科学学院 福州 350116)

摘 要 当前许多实际应用需要持续地对流数据的区间统计查询做出实时响应,并使用差分隐私保护模型来应对信息发布过程中的敏感数据泄露问题。现有研究采用树状数组作为组织和存储流数据的数据结构,以满足信息发布的实时性要求。然而,现有方法中的相关参数为预先确定的,并不能很好地适应查询的动态变化。为此,文中提出在流数据实时发布的框架上,引入历史查询信息,以实现发布过程中树高参数的动态调整。首先,使用移动平均法分析历史查询记录,并预测后续的查询范围分布;继而针对预测结果,通过理论推导,得出使得期望误差最小的树高;最终实现差分隐私流数据实时发布中树高参数的自适应优化。实验结果表明,该方法在保证了时间效率的同时,有效地提高了发布结果的精度。

关键词 差分隐私,流数据发布,自适应参数优化,历史查询

中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.09.013

Adaptive Parameter Optimization for Real-time Differential Privacy Streaming Data Publication

WU Ying-jie HUANG Xin GE Chen SUN Lan
(College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350116, China)

Abstract Recently, many practical applications need to continuously respond to range queries over streaming data in real-time, and adopt the differential privacy protection to deal with the disclosure of sensitive data in the information publication process. Existing research adopts Fenwick tree as data structure to organize and store data items in the stream to satisfy the real-time requirement in the information publication process. However, parameters in the previous method are predefined, which cannot adapt dynamic changes of the queries well. To solve this problem, based on the framework of real-time differential privacy streaming data publication, this paper proposed a method introducing the historical query records to achieve the adaptive parameter optimization. Firstly, based on moving average method, the historical queries are analyzed to predict the subsequent queries. Then according to the prediction results, the optimistic height of the tree is calculated theoretically, which minimizes the expected error. Finally, the adaptive parameter optimization is achieved in real-time differential privacy streaming data publication. The experimental results show that this method can significantly improve the query accuracy while guaranteeing the time efficiency.

Keywords Differential privacy, Streaming data publication, Adaptive parameter optimization, Historical queries

1 引言

当前,许多实际应用需要持续地响应对流数据的区间统计查询,如导航软件需要统计各道路特定时间范围内的车流总量,搜索引擎需要统计各关键词在特定时段内的搜索次数。这些场景均可抽象为对流数据在特定时间区间内的统计值做实时统计发布。并且,多数场景均需要快速地给出查询结果,即对算法的实时性有较高要求。在提供统计信息的同时,该类发布也可能泄露相关用户的敏感隐私信息[1]。

差分隐私[2]是一种强健的隐私保护模型,其通过添加随机噪声等方式,限制了任意一条数据对最终统计结果的影响,从而模糊了该条数据在数据集中的存在性,从根本上保护了用户隐私。该模型被广泛应用于各种数据类型及应用场景中[3]。对于流数据发布中的隐私泄露问题,现有工作也多使用差分隐私模型来解决。Dwork等[4]通过分层计数的方法实现了事件层的连续统计发布。Chan等[5]为了减少加噪机制对发布结果的影响,提出利用二叉树的分治结构分解及存储流数据。该方法被日后许多关于流数据发布的研究所借鉴。

到稿日期:2018-07-13 返修日期:2018-09-15 本文受国家自然科学基金项目(61300026),福建省自然科学基金项目(2017J01754,2018J01797)资助。

吴英杰(1979-),男,博士,教授,CCF会员,主要研究方向为数据挖掘、数据安全与隐私保护,E-mail:yjwu@fzu.edu.cn;黄 鑫(1993-),男,硕士,主要研究方向为差分隐私;葛 晨(1992-),男,硕士,主要研究方向为差分隐私;孙 岚(1978-),女,硕士,讲师,主要研究方向为数据安全与隐私保护,E-mail:15009466@qq.com(通信作者)。

在此基础上，Ge 等[6]对树结构做出改进，并引入树状数组以存储流数据，满足了实时发布的需求。文献[7]在滑动窗口机制上，基于抽样提出了一种面向流式直方图的差分隐私发布方法。Bolot 等[8]扩展了数据发布的形式，将重点放在离当前最近的数据流中，通过对数据添加衰减权重和滑动窗口的方式解决了带权的流数据发布问题。文献[9]改进了加噪及直方图重组方法，实现了对流数据差分隐私发布中的区间查询、滑动窗口及事件监测等各类任务的同步处理。

然而，在现有的流数据实时发布框架中，树结构的相关参数被事先确定，即其在整个算法流程中是固定的。实际上，不同场景下不同用户类型的查询范围具有动态变化的规律。例如：对于交通路况的车流量查询，用户在工作日更倾向于对早高峰/晚高峰等较短时段进行查询，而在节假日前后，查询的时间跨度就可能延长至出发日/返程日等较长时段；并且查询规律不是骤变的，其具有一定连续性和可预测性。在实际应用中，收集并记录用户的历史查询数据是很容易的。而且用于组织流数据的树模型也可以动态地构建。因此，通过不断收集用户的历史查询记录，分析其变化规律，预测后续查询的范围，继而对流数据发布中的树结构的树高参数做针对性的调整，能有效地提高响应结果的精度。

本文的主要贡献如下：

(1)针对现有的差分隐私流数据实时发布框架中存在的参数固定、不能适应动态变化的查询的问题，提出了基于历史查询规律的自适应参数优化方法。通过不断记录近期查询记录，并使用移动平均法实时地计算出后续查询范围的预测值。

(2)通过理论推导得出理论误差与树高及查询范围的关系；进而基于预测结果，得到使理论误差最小的树高，据此动态构建实时发布模型中的树结构。

(3) 针对发布误差及运行时间等指标，通过实验在真实数据集上将本文算法与原始框架进行对比，验证了自适应参数优化算法能有效提高发布结果的精度，同时满足实时性的要求。

2 基础知识与相关概念

2.1 差分隐私

Dwork 等[2]首次提出了差分隐私模型，该模型是一种强健的隐私保护框架，通过限制数据集中单条记录的变化对查询结果的影响，使得攻击者即使知道了除某条记录外的所有记录信息，也无法准确获得该条记录中的敏感信息。

定义 1(ε-差分隐私[2]) 给定两个只差一条记录的数据集 D 和 D'，即

$$|(D'-D)\bigcup(D-D')|=1 \tag{1}$$

则称 D 和 D' 为兄弟数据集。其中 $|D|$ 表示数据集中记录的数量。

若随机算法 A 对于任意一对兄弟数据集 D 和 D'，以及所有可能的输出 $Orange(A)$ 均满足：

$$Pr(A(D)\in O)\leqslant e^{\varepsilon}\times Pr(A(D')\in O) \tag{2}$$

则称算法 A 满足 $\varepsilon\#$-差分隐私。其中 ε 是隐私参数，ε 越小则隐私保护程度越高。

定义 1 只给出了差分隐私模型对于隐私保护的要求，其实现需要基于具体的加噪算法。大部分加噪算法依赖于全局敏感度的定义(见定义 2)。

定义 2(全局敏感度，即敏感度) 对所有可能的兄弟数据集 D 和 D' 及其某个发布算法 Q，用 $\boldsymbol{Q}(\boldsymbol{D})$ 和 $\boldsymbol{Q}(\boldsymbol{D}')$ 分别表示 Q 作用在 D 和 D' 上得到的数值向量形式的统计结果，即 $\boldsymbol{Q}(\boldsymbol{D})=(x_1,x_2,\cdots,x_n)$，$\boldsymbol{Q}(\boldsymbol{D}')=(x_1',x_2',\cdots x_n')$。则算法 Q 的全局敏感度 ΔQ 表示如下：

$$\Delta Q=\max\|\boldsymbol{Q}(\boldsymbol{D})-\boldsymbol{Q}(\boldsymbol{D}')\|_1 \tag{3}$$

对于差分隐私模型下输出结果为数值类型的场景，Laplace 机制[10]是最为常用的加噪方法。

定义 3(Laplace 机制[10]) 对于一个原始的查询函数 Q，可使用 Laplace 机制对其进行处理，从而得到一个满足 ε-差分隐私的随机发布算法 A。Laplace 机制的定义如下：

$$A(x,Q,\varepsilon)=Q(x)+\langle Y_1,Y_2,\cdots,Y_k\rangle \tag{4}$$

其中，Y_i 表示服从参数为 $\Delta Q/\varepsilon$ 的 Laplace 分布的随机变量 $Lap(\Delta Q/\varepsilon)$。该机制在输出向量的各维加入了相互独立且服从 Laplace 分布的噪声。Laplace 分布表示为：

$$Lap(b)=\frac{1}{2b}e^{-\frac{|x|}{b}} \tag{5}$$

2.2 流数据及相关查询操作

定义 4(流数据) 流数据指一组顺序连续到达的数值序列。一般情况下，流数据可被视为一个随时间延续而增长的动态数值集合，其中每一项数值元素称为该流数据的一个数据项。

定义 5(滑动窗口内的区间统计查询) 区间统计查询定义为对流数据中某段连续数据项的累加值的查询操作，即设当前时刻为 t，数据序列为 $S=\{D_1,D_2,\cdots,D_t\}$，用户提出的查询操作的查询范围为 $[l,r]$，则相应的查询结果可由下式表示：

$$result(l,r)=\sum_{i=l}^{r}D_i \tag{6}$$

滑动窗口模型假定不关心流数据序列中时间过于久远的数据项，即若事先确定的滑动窗口大小为 W，则用户的查询范围 $[l,r]$ 满足 $t-W<l\leqslant r\leqslant t$。

3 流数据实时发布中的自适应参数优化

3.1 流数据实时发布框架及其理论误差

本节简要介绍原始的流数据差分隐私实时发布框架，并说明树高参数如何影响其理论误差。

对于差分隐私保护下的流数据范围查询，Chan 等[5]提出利用二叉树的分治结构存储流数据，其中二叉树的叶子节点依次对应流数据中的原始数据项，其余节点的值为其所有后继叶子节点的值之和。相比于直接套用 Laplace 机制，利用二叉树存储数据的方法由于发掘了数据间的关联性，减少了加噪规模，提升了发布精度。

在此基础上，Ge 等[6]对树结构做出改进。如图 1 所示，二叉树中右子节点的值可通过其父节点及兄弟节点获取，因

此可以去除这些冗余节点，得到的数据结构被称为树状数组[11]。该方法精简了二叉树结构，实现了新节点的快速插入，满足了流数据发布中的实时性要求。

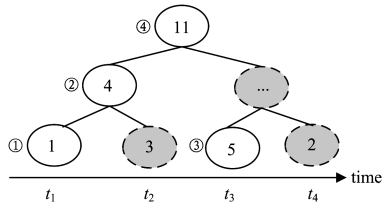

图 1　以树状数组结构组织的流数据

Fig. 1　Streaming data organized by Fenwick tree

在树状数组中，为求某段区间统计值，需要组合树状数组中某些实节点的值。例如，区间[1,3]的统计值可表示为节点②＋节点③。文献[6]给出了具体的求和方法。对于单次查询，其所涉及的节点数量不会超过树的高度。因此，该方法的全局敏感度等于树高 h。根据 Laplace 机制，该方法对每个实节点添加规模为 h/ε 的 Laplace 噪声。因此，对于某次查询 q，该方法返回结果的期望方差为：

$$error(q)=\frac{2h^2}{\varepsilon^2}|related\_nodes(q)| \tag{7}$$

其中，$|related\_nodes(q)|$ 表示该次查询涉及的实节点的数目。容易发现，该值会受到树高的影响。

由于流数据是连续到来的，如果只使用一棵树来表示整个流数据，误差将随着树高的增加而无限增长。因此，文献[6]使用多棵高度固定的树，分段并动态地存储流数据。如图 2 所示，在 t 时刻(当前时刻)，前一棵树表示的数据范围已被用尽，对于新到来的数据项 Dt，开辟了一棵新树进行存储。分段存储的方式也使动态调整树高具有可行性。

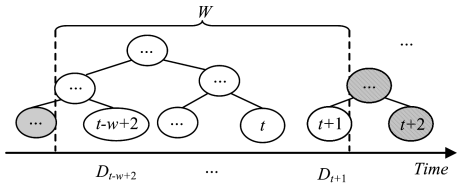

图 2　建立新树以存储新节点

Fig. 2　Building new tree to store new data items

算法 1 给出了分段动态构建树状数组的流程。

算法 1　树状数组的动态构建

输入：预设树高 H，流数据 D，隐私预算 ε

输出：更新后的树状数组结构

1. 对于当前时刻 t，及新到来的流数据项 Dt，若当前一棵树已使用完毕，则转到步骤 2；否则转到步骤 3。
2. 根据预设树高，建立一棵新树，转到步骤 3。
3. 将树状数组中 t 时刻对应节点的值加上 Dt，同时将 Dt 加至其父节点。
4. 将 Laplace 噪声 Lap(H/ε)加至 t 时刻对应节点。
5. 若有新到来的流数据项，则转至步骤 1；否则算法结束。

在多棵树分段存储的设定下，树高越小，则根据式(7)，加到每个节点上的噪声规模就越小，但由于树的规模变小，查询所涉及的节点数目可能增加，因此树高对误差的影响是较为复杂的。下面举例说明在其他参数均一致的情况下，不同的树高对结果误差产生的影响。

由于隐私预算参数 ε 在误差计算中是一个独立的变量，不妨设其等于 1。对比图 3 和图 4，由于图 3 中的树高为 3，因此每个节点上添加的噪声为 $Lap(3)$(根据式(5)，$Lap(n)$表示噪声规模为 n 的 Laplace 噪声，其方差为 $2n^2$)；而对于图 4 而言，其树高为 4，每个节点上添加的噪声为 $Lap(4)$。对于查询区间[1,8]，按照图 3 的树结构设置，$result[1,8]=④+⑧$；对于图 4 而言，$result[1,8]=⑧$。由此可计算出其相应的误差 $error_1=2*32+2*32=36$ 和 $error_2=2*42=32$，得出 $error_2<error_1$。

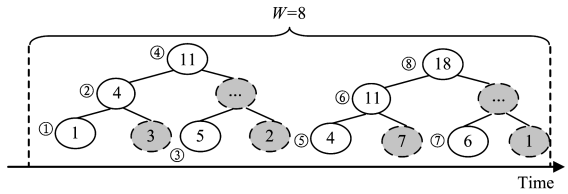

图 3　树高为 3 的树结构示例

Fig. 3　Example of tree structure with tree height of 3

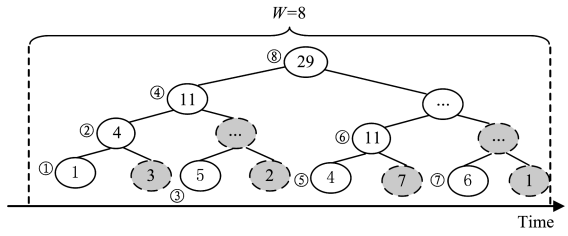

图 4　树高为 4 的树结构示例

Fig. 4　Example of tree structure with tree height of 4

而对于另一个查询区间[1,4]，按照图 3 的树结构设置，$result[1,4]=④$，按照图 4，同样有 $result[1,4]=④$，即均只需要通过一个节点值即可获取对应的查询结果。但是由于树高的差异，每个节点上添加的噪声是有差异的。此时，$error_1=2*32=18$，$error_2=2*42=32$，$error_2>error_1$，得出了与树高为 3 时不同的结果。

以上例子对树高与误差的关系进行了简单说明。后文说明查询范围的预测方法后，将得出误差关于查询范围和树高的具体表达式，进而求出理论误差最小时对应的树高。

3.2　基于历史查询记录的查询区间预测

为确定使得误差最小的最优树高，首先需要根据历史查询记录求出后续的查询范围长度的预测值。在统计学中，移动平均法是预测数据点的一种计算方法，它通过实时地维护数据集中较新数据项的平均值来预测后续的数据点。

定义 6(移动平均法)　给定一系列数和一个固定子集的大小，通过取数列的初始固定子集的平均值，得到第一个平均值。然后通过“向前移动”来修改子集，即舍弃子集序列中的第一个元素，并将子集序列之后的下一个元素纳入子集。其计算形式为：

$$\bar{q}_{SM}=\frac{q_M+q_{M-1}+\cdots+q_{M-n+1}}{n}=\frac{1}{n}\sum_{i=0}^{n-1}q_{M-1} \tag{8}$$

其中，n 表示移动平均法求解过程中的子集大小；M 表示当前

序列的编号，即当前的查询编号；q_i 表示第 i 次查询的查询范围长度。

如果直接使用原始公式(8)，那么单次平均值计算的时间复杂度为线性的，效率较低。将式(8)变形为式(9)后，即能在常数时间内计算出下一个平均值：

$$\bar{q}_M=\bar{q}_{M,prev}+\frac{q_M}{n}-\frac{q_{M-n}}{n} \tag{9}$$

移动平均法虽然形式简单，但其不需要先验知识，适合本文通用算法的构建；其常数规模的时间复杂度也能满足实时性要求。并且，它减小了单个数据项的波动对预测结果的影响，因此能较好地过滤历史记录中出现的离群点。称其变形形式为加权移动平均法，如定义 7 所示。

定义 7(加权移动平均法) 加权平均同样是计算一个子集中所有数的平均数，但是子集中的每个数均乘以对应的权重因子，给样本窗口中不同位置的数据赋予了不同的权重。

$$WMA_M=\frac{N(0)q_M+N(1)q_{M-1}+\cdots+N(n-1)q_{M-n+1}}{N(0)+N(1)+\cdots+N(n-1)} \tag{10}$$

其中，WMA_M 表示加权平均数，$N(i)$表示权重因子函数。加权移动平均法同样可以在“向前移动”的过程中计算，如当权重因子函数为指数衰减模式，即 $N(i)=pi$ 时，更新公式为：

$$WMA_{M+1}=\frac{(WMA_M-q_{M-n+1})\times p+q_{M+1}}{N(0)+N(1)+\cdots+N(n-1)} \tag{11}$$

加权移动平均法适用于如文献[8]提出的指数衰减模型等数据带有时效性的应用环境。不论是一般形式的移动平均还是加权移动平均，其只影响得出的预测结果，并不影响预测结果确定后的分析步骤。因此本文基于一般形式的移动平均法展开分析。

上述动态计算预测值的过程结合了 3.1 节中多棵树分段存储的方法，使其可以在不断接收流数据及查询的同时动态调整树高。在一颗树即将完全移入滑动窗口时，根据得出的预测结果，选择合适的树高预构建树，如图 5 所示，滑动窗口内各包含两棵二叉树的一部分。其中，左侧灰色节点已移出滑动窗口，之后的过程将不再涉及这些节点，而右侧条纹节点为即将使用的节点，在树中第一个节点进入滑动窗口时，后续两棵二叉树中的树高即已确定。这与 3.1 节中不含自适应参数优化的构建过程的差异在于：前后两组树高不一定相同，而是根据历史查询预测结果，构建出不同高度的树状数组。

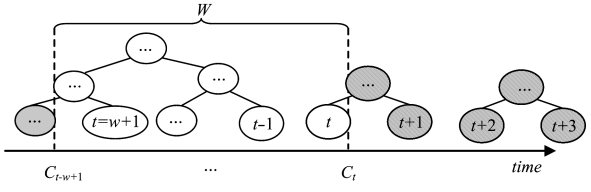

图 5 滑动窗口下的区间树构建过程

Fig. 5 Building process of interval tree under sliding window

3.3 基于预测结果的最优树高自适应优化

3.2 节通过移动平均法得出后续查询范围的预测长度(以下用 Len 表示)。本节通过理论推导，基于得到的查询范围长度 Len 得出使得理论误差最小的树高取值。

由式(7)可知，查询误差与查询过程中涉及的树中节点的个数及树高的平方成正比。为了将查询误差表示为以树高为变量的形式，需先求出第一项，即单次查询所涉及的树中节点个数的平均值与树高的关系。

显然存在唯一的 i，使得 $2i-1\leqslant Len<2i$。首先说明，最优的树高取值不会大于 i：若树高取值大于 i，则根据树状数组的性质可知，第 $i+1$ 层及以上的节点所表示的范围已超过 Len，即它们不可能在查询中被涉及；而根据树状数组的自相似性质[11]可知，其第 i 层及以下的部分的结构与高为 i 的树的结构是一样的，即对于同一个查询，涉及的树中节点的个数不会改变，而树高的增加将导致每个点噪音的增加。因此，树高 $k>i$ 时的误差值一定大于 $k=i$ 时的误差，即最优树高不会超过 i。

在此基础上，树高 k 的取值实际上决定了将长度为 $2i-1$ 的流数据片段等分为若干份的份数，其中每份用一棵树表示。因此，可假定长度为 Len 的查询在长度为 $2i-1$ 的流数据片段中各个位置等可能出现。进一步，不妨设其在每个位置均出现一次，即长度为 Len 的查询出现的总次数(下文以 n 表示)为 $2i-1$，以便于后续计算。

对于某个节点 x 而言，涉及 x 的查询范围的个数为：

$$count(x)=\begin{cases}Qsum(x), & \text{if } x=root\\ Qsum(x)-count(fx), & \text{otherwise}\end{cases} \tag{12}$$

其中，fx 表示节点 x 的父节点。$Qsum(x)$指查询范围包含了节点 x 所对应范围的查询个数，可由下式得出：

$$Qsum(x)=\begin{cases}Len-r_x+l_x, & \text{if } (r_x-l_x)<len\\ 0, & \text{otherwise}\end{cases} \tag{13}$$

其中，$[l_x,r_x]$为节点 x 的对应范围，在树状数组中，其长度必定为 2 的幂次。根据树状数组的性质可知，除了少数的顶层节点，在这一组待构建树中，对应范围长度为 $2j$ 的节点个数为 $2i-1-j$。则该范围长度所有节点被覆盖的次数可用下式表示：

$$Qsum_{2^j}=2^{i-1-j}\times(Len-2^j+1) \tag{14}$$

将式(14)代入式(12)中，并对式(12)枚举的所有节点求和。利用容斥原理，可得到树高为 k 时，对于总计 n 个查询，所有节点被覆盖的总次数为：

$$\begin{aligned}Sum(k)&=\sum_{j\in S_k}count(j)\\&=\sum_{j=0}^{k-1}(-1)^j\times 2^{i-1-j}\times(Len-2^j+1)\end{aligned} \tag{15}$$

将上式代入式(7)，最终得到：当查询区间长度 Len 确定时，以树高 k 为变量的平均期望误差为：

$$\begin{aligned}error&=\frac{(\sum_{j=0}^{k-1}(-1)^j*2^{i-1-j}*(Len-2^j+1))}{2^{i-1}}*\frac{k^2}{2\varepsilon^2}\\&=(\sum_{j=0}^{k-1}(-1)^j*2^{-j}*(Len-2^j+1))*\frac{k^2}{2\varepsilon^2}\end{aligned} \tag{16}$$

下面分析引入自适应参数优化对算法时间复杂度的影响。由于 $k\leqslant i$，即 k 可能的取值有限，可在 $O(i)$时间内直接计算所有的 k 值各自对应的结果。当滑动窗口大小为 W 时，由于只有在之前构造的一组树状数组完全进入滑动窗口后，才需要计算下一组的高度，因此树结构选择过程的时间复杂

度为 $O((W/2i)*i)$，加上建树操作的复杂度 $O(W)$ 后，可得算法的总体复杂度为 $O(W+(W/2i)*i)$，即 $O(W)$。平摊至单个节点后，其建树复杂度依旧为 $O(1)$。即引入自适应参数优化可在提高算法精度的同时，不影响时间复杂度的渐近阶。

综合以上分析，在算法 1 的基础上，给出引入自适应树高优化的算法，如算法 2 所示。

算法 2　自适应树高优化

输入：初始树高 H，流数据 D，隐私预算 ε

输出：更新后的树状数组结构

1. 初始化：使用初始树高 H 建立第一棵树，并针对前 n 个查询记录，用式(8)计算出初始均值 q。
2. 对于当前时刻 t，以及新到来的流数据项 Dt，若当前一棵树已使用完毕，则转到步骤 3；否则转到步骤 4。
3. 根据 q 及式(16)计算出最优树高 H′，并以 H′建立新树，转到步骤 4。
4. 将树状数组中 t 时刻对应节点的值加上 Dt，同时将 Dt 加至其父节点。
5. 将 Laplace 噪声 Lap(H′/ε)加至 t 时刻对应节点。
6. 对于新到来的查询，使用式(9)更新 q。
7. 若有新到来的流数据项，转至步骤 2；否则算法结束。

4　实验及分析

本节将从查询效率和查询精度两个方面对以下两组加入自适应优化前后的算法做比较。其中 RTP 算法为文献[6]提出的基于树状数组的原始发布算法，RTP_MM 算法[6] 是在 RTP 的基础上加入了矩阵机制[12] 进行精度优化，但同样未考虑自适应树高调整。History_Query_RTP(HQ_RTP)算法和 History_Query_RTPMM(HQ_RTPMM)算法则为本文提出的在前两种基础算法上添加自适应树高调整后的算法。为了排除差分隐私加噪过程中固有的随机性对实验结果的影响，每组实验运行 30 次取平均值。

4.1　实验数据与环境

本文采用文献[13]中的 Search Logs 和 NetTrace 数据集，以及文献[14]中的 WorldCup98 数据集。Search Logs 为 2004 年 1 月至 2009 年 8 月期间，某网站对关键词“Obama”的搜索次数的统计数据。NetTrace 数据集包含了一些机构在特定时间段内对特定 IP 段的数据包请求次数。WorldCup98 为 1998 年 4 月至 1998 年 7 月期间，世界杯官网的访问量统计记录。各数据集的数据规模如表 1 所列。

表 1　各数据集的数据规模

Table 1　Data size of data sets

Data Set	Size
Search Logs	32768
Nettrace	65536
WorldCup98	7518579

在实验中，采用均方误差衡量算法发布数据的查询精度，误差公式如下：

$$Error(Q)=\frac{\sum_{q\in Q}(q(D)-q(D'))^2}{|Q|} \tag{17}$$

其中，$|Q|$ 为查询的数量，D 为原始数据集，D' 为添加噪声后的发布结果，q 表示一次查询。

实验环境为：Intel Core i5 4570 3.2GHz 处理器，8GB 内存，Windows 7 操作系统；算法用 C++语言实现。

4.2　查询精度的对比分析

4.2.1　不同查询规律下的查询误差对比分析

本节中设置不同的查询区间规律来验证算法在特定查询规律下的效果。3 种不同的查询规律分别为：1)small，查询区间集中于 1～1024；2)middle，查询区间集中于 1025～8192；3)large，查询区间集中于 8193～W。其中 W 为设置的滑动窗口大小。设置 $\varepsilon=1.0$。实验结果如图 6-图 8 所示。

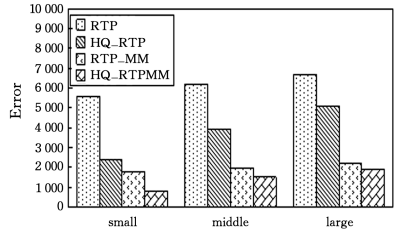

图 6　不同查询规律下的查询误差对比(Search Logs)

Fig. 6　Accuracy under different query pattern (Search Logs)

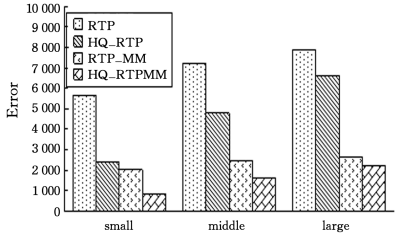

图 7　不同查询规律下的查询误差对比(Nettrace)

Fig. 7　Accuracy under different query pattern (Nettrace)

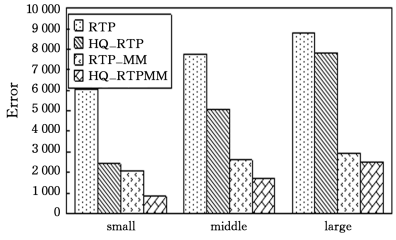

图 8　不同查询规律下的查询误差对比(WorldCup98)

Fig. 8　Accuracy under different query pattern (WorldCup98)

从图 6－图 8 可以看出，随着查询区间的增大，RTP 与 HQ_RTP、RTP_MM 与 HQ_RTPMM 的查询误差趋于接近。这是因为对于 RTP 和 RTP_MM 算法而言，其树高只与滑动窗口的大小相关，当查询区间大小越接近滑动窗口大小 W 时，越能带来较好的结果。而当查询区间大小与滑动窗口差异较大时，由于每个节点添加了不必要的噪声，使得误差增加。HQ_RTP 和 HQ_RTPMM 算法根据历史查询调节树高，在不同的查询规律下都能带来较好的精度优化，当查询区间越小时，其带来的优化效果越明显。

4.2.2　不同查询区间大小及隐私预算下的查询误差对比

本节通过随机生成特定大小的查询区间来比较查询误差。区间大小分别取 $2^0, 2^1, \cdots, 2^{16}$，每时刻生成一条查询。同时，每组实验又分别取隐私预算参数 $\varepsilon=1, 0.1, 0.01$。实验结果如图 9－图 11 所示。

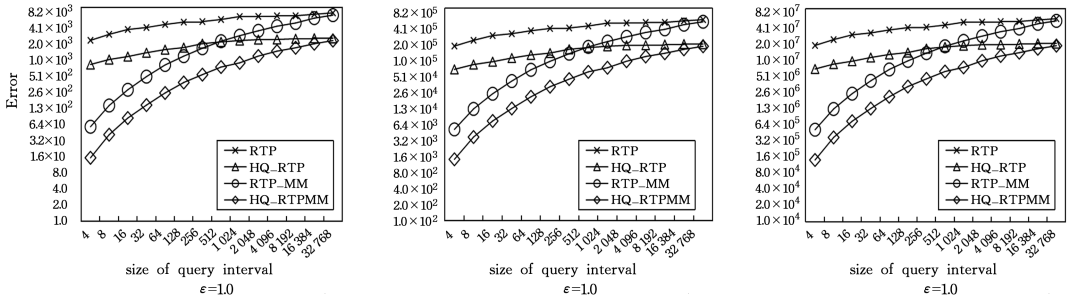

图 9 不同查询区间大小下的查询误差对比(Search Logs)

Fig. 9 Accuracy under different size of query range (Search Logs)

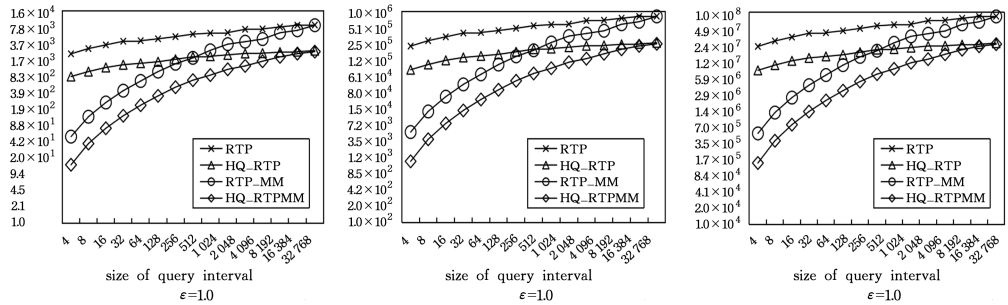

图 10 不同查询区间大小下的查询误差对比(Nettrace)

Fig. 10 Accuracy under different size of query range (Nettrace)

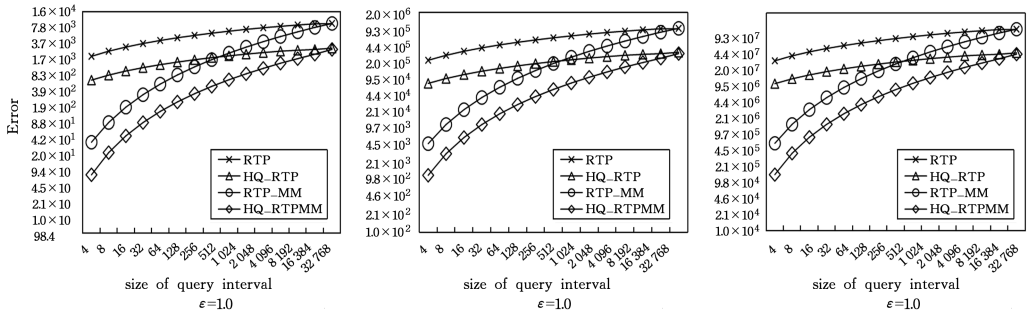

图 11 不同查询区间大小下的查询误差对比(WorldCup98)

Fig. 11 Accuracy under different size of query range (WorldCup98)

从图 9—图 11 可以发现,在区间较小时,HQ_RTP 和 HQ_RTPMM 算法能够有效降低算法的查询误差,随着查询区间的增加,其优化效果有所降低,这是因为 RTP 和 HQ_RTP 的树结构只与滑动窗口大小相关,而随着查询区间长度的增加,基于历史查询统计而得出的树高逐渐接近 $\log_2 W$,当查询区间与滑动窗口大小完全相同时,RTP 与 HQ_RTP、RTP_MM 与 HQ_RTPMM 将会完全相同。

对于不同的隐私预算参数,从图中可以看出,其只是简单地影响了误差的量级,而对于整个误差曲线的变化趋势没有影响。即可以将其看作一个独立的与误差成反比的系数。这是由 Laplace 加噪机制所决定的。

4.2.3 算法运行效率的对比分析

固定滑动窗口大小 W 为 32768,隐私预算 ε 为 1.0,通过将原始算法与结合了自适应树高调整的改进算法(RTP 与 HQ_RTP、RTP_MM 与 HQ_RTPMM)进行比较,来说明增加自适应参数优化是否会对算法的运行时间产生影响,其结果如图 12 所示(由于 world_cup_98 数据集的规模较大,因此使用了单独的坐标刻度)。

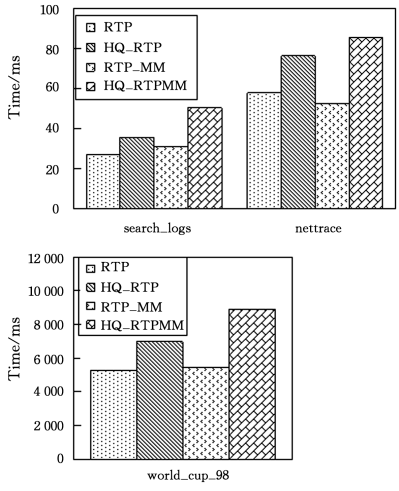

图 12 自适应优化前后运行效率对比

Fig. 12 Comparison of operating efficiency before and after adaptive optimization

从图 12 可知，与 RTP，RTP_MM 算法相比，HQ_RTP，HQ_RTPMM 的运行时间有一定增加，但增加量不大。这是因为自适应优化相关计算部分的时间复杂度为 O(1)，将其加入算法主体后，算法的整体运行效率依旧是 O(1)，只有常数上的改变。这与理论分析是相符的。

综合以上实验可以看出：对于两种不同的流数据发布算法，在原始的树状数组实时发布模型中加入自适应参数优化后，其发布数据精度均有了明显提升；同时，自适应参数优化操作并不会对算法的时间效率产生很大的影响。

结束语　本文在差分隐私流数据实时发布的框架上，引入历史查询，提出了应用于差分隐私流数据实时发布的自适应参数优化算法。通过移动平均法建立统计与分析模型，并根据历史查询的预测结果，结合节点覆盖情况的理论分析，计算出最优树高，以实现理论误差的最小化。通过与不含自适应参数优化的算法进行实验对比，验证了引入历史查询能够显著提升差分隐私流数据实时发布算法的精度。

在日后的工作中，将考虑改进预测方法来提高预测准确率，以及将历史查询优化与其他数据发布形式相结合。

参 考 文 献

[1] FUNG B C M，WANG K，CHEN R，et al. Privacy-preserving data publishing：A survey of recent developments[J]. Acm Computing Surveys，2010，42(4)：14.

[2] DWORK C. Differential Privacy：A Survey of Results[C]// International Conference on Theory and Applications of MODELS of Computation. Springer-Verlag，2008：1-19.

[3] ZHANG X J，MENG X F. Differential privacy in data publication and analysis[J]. Chinese Journal of Computers，2014，37(4)：927-949.(in Chinese)
张啸剑，孟小峰. 面向数据发布和分析的差分隐私保护[J]. 计算机学报，2014，37(4)：927-949.

[4] DWORK C，NAOR M，PITASSI T，et al. Differential privacy under continual observation[C]//Proc. of the 42nd ACM Symposium on Theory of Computing(STOC). New York：ACM，2010：715-724.

[5] CHAN T H，SHI E，SONG D. Private and Continual Release of Statistics[J]. ACM Trans. on Information and System Security，2011，14(3)：1-24.

[6] GE C，WU Y J，SUN L. A Real-time Publishing Method of Differential Privacy Streaming Data[OL]. http://kns.cnki.net/kcms/detail/11.5602.TP.20171016.1629.004.html.(in Chinese)
葛晨，吴英杰，孙岚. 一种差分隐私流数据实时发布方法[OL]. http://kns.cnki.net/kcms/detail/11.5602.TP.20171016.1629.004.html.

[7] ZHANG X J，MENG X F. Stream histogram publication method with differential privacy[J]. Journal of Software，2016，27(2)：381-393.(in Chinese)
张啸剑，孟小峰. 基于差分隐私的流式直方图发布方法[J]. 软件学报，2016，27(2)：381-393.

[8] BOLOT J，FAWAZ N，MUTHUKRISHNAN S，et al. Private decayed predicate sums on streams[C]// Proceedings of the 16th International Conference on Extending Database Technology. New York：ACM，2013：284-295.

[9] CHEN Y，MACHANAVAJJHALA A，HAY M，et al. PeGaSus：Data-Adaptive Differentially Private Stream Processing[C]// ACM Sigsac Conference. ACM，2017：1375-1388.

[10] DWORK C，MCSHERRY F，NISSIM K，et al. Calibrating Noise to Sensitivity in Private Data Analysis[J]. Lecture Notes in Computer Science，2012，3876(8)：265-284.

[11] FENWICK P M. A new data structure for cumulative frequency tables[J]. Software Practice & Experience，1994，24(3)：327-336.

[12] LI C，HAY M，RASTOGI V，et al. Optimizing linear counting queries under differential privacy[C]// Twenty-Ninth ACM Sigmod-Sigact-Sigart Symposium on Principles of Database Systems(PODS 2010). Indianapolis，Indiana，USA，DBLP，2010：123-134.

[13] HAY M，RASTOGI V，MIKLAU G，et al. Boosting the accuracy of differentially private histograms through consistency[J]. Proceedings of the Vldb Endowment，2010，3(1-2)：1021-1032.

[14] KELLARIS G，PAPADOPOULOS S，XIAO X，et al. Differentially private event sequences over infinite streams[J]. Proceedings of the Vldb Endowment，2014，7(12)：1155-1166.