中国科学: 信息科学 2017 年 第 47 卷 第 11 期: 1493–1509
SCIENTIA SINICA Informationis
《中国科学》杂志社
SCIENCE CHINA PRESS
机器学习专题 · 论文

指数衰减模式下基于矩阵机制的差分隐私流数据发布算法

吴英杰*, 葛晨, 张立群, 孙岚

福州大学数学与计算机科学学院, 福州 350116
* 通信作者. E-mail: yjwu@fzu.edu.cn

收稿日期: 2017–05–17; 接受日期: 2017–06–13; 网络出版日期: 2017–11–14
国家自然科学基金 (批准号: 61300026) 和福建省自然科学基金 (批准号: 2017J01754) 资助项目

摘要 当前许多实际应用需要持续地对流数据进行统计发布, 并且对当前的数据关注度高于对历史数据的关注度. 现有关于该问题的解决方案是使数据项带有权重, 并提出指数衰减下的差分隐私流数据发布方法. 然而, 现有的方法仅考虑单次查询, 未能有效利用连续统计发布背景下查询间的关联性, 以进一步提高查询的精度. 为此, 本文利用矩阵在处理关联性查询方面的优势, 提出一种指数衰减模式下基于矩阵机制的差分隐私流数据发布算法 DMFDA. 算法首先使用构造法生成满足流数据实时发布要求的矩阵分解策略; 其次, 利用对角矩阵对构造的策略矩阵进行调整, 以提高发布精度; 最后, 根据所构造策略矩阵的子结构特性, 提出快速求解对角矩阵的方法. 实验对算法 DMFDA 发布的查询结果与同类指数衰减流数据发布算法进行比较分析. 实验结果表明, 算法 DMFDA 是有效可行的.

关键词 差分隐私, 流数据发布, 指数衰减, 矩阵机制, 对角矩阵

1 引言

当前, 许多实际应用需要持续地对流数据进行统计发布, 如购物网站需要实时统计物品的销售额以向用户推荐热销产品, 搜索引擎需要统计搜索频率较高的词组以根据用户的部分输入列出可能要搜索的词组. 这些应用均需统计发布流数据在某种意义下的实时计数值. 对该类统计值的发布在提供科学决策依据的同时, 还可能泄露有关用户的敏感隐私信息 [1].

为此, 近年来一些研究人员基于差分隐私 [2~5] 保护模型对该类流数据统计发布问题进行了研究 [6~10]. Dwork 等 [6] 提出了利用分段计数的发布方法, 实现对单条流数据从时刻 1 到当前时刻 t 的计数值总和进行连续发布. Chan 等 [7] 提出了利用二叉树结构的发布方法, 实现查询精度和算法效率的进一步提升. Cao 等 [8] 在系统运行前, 对预先定义的查询集合进行统计分析, 以实现对特定用户批

引用格式: 吴英杰, 葛晨, 张立群, 等. 指数衰减模式下基于矩阵机制的差分隐私流数据发布算法. 中国科学: 信息科学, 2017, 47: 1493–1509, doi: 10.1360/N112017-00111
Wu Y J, Ge C, Zhang L Q, et al. An algorithm for differential privacy streaming data publication based on matrix mechanism under exponential decay mode (in Chinese). Sci Sin Inform, 2017, 47: 1493–1509, doi: 10.1360/N112017-00111

量范围查询进行回答并优化查询精度. 文献 [9] 则采用滑动窗机制和划分等方法, 提供了滑动窗口内计数值总和发布和从时刻 1 开始的计数值总和发布等. 在以上的问题背景中, 数据项均不带有权重. 然而, 一些实际部分应用往往更注重对近期数据的统计发布, 而对历史数据的关注度较低, 这是由于近期的事件的统计监测与其目的相关性更强. 为解决该问题, 一种直接的方法就是使得数据项带有权重, 距离当前时刻越近, 则权重越大. Bolot 等 [10] 提出了权重衰减下的差分隐私流数据统计发布方法, 利用区间树结构对指数衰减模式下的滑动窗口内统计值加权累和进行统计发布. 然而, 文献 [10] 中使用区间树的发布方法未能充分利用连续统计发布中查询间的关联性来进一步提高数据的发布精度. 本文使用矩阵机制来处理连续统计发布中的关联性查询, 构造相应的负载矩阵, 设计出高效的指数衰减模式下差分隐私流数据发布算法, 在提高数据发布质量的同时, 可有效满足流数据发布的时空复杂度要求.

本文的主要工作如下:

(1) 针对指数衰减模式下的流数据连续统计发布, 利用树状数组, 设计出有效的策略矩阵与还原矩阵的构造方法;

(2) 分析策略矩阵与还原矩阵的特性, 在保持策略矩阵 1– 范数不变的前提下通过对角矩阵优化还原矩阵与策略矩阵, 提高发布精度;

(3) 针对流数据发布的实时性要求, 利用对角矩阵的结构特性, 提出快速求解对角矩阵中对角元素的方法;

(4) 设计出指数衰减模式下的流数据连续统计发布算法, 并通过实验验证该算法的有效性.

2 相关概念

2.1 差分隐私定义

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, 通过减少修改数据集中一条记录对查询结果的影响, 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.

在差分隐私保护模型中, 对兄弟数据集概念定义如下.

定义1 (兄弟数据集) 给定数据集 $\boldsymbol{D}$, $\boldsymbol{D}'$, 当 2 个数据集之间只相差 1 条记录时, 即

$$||\boldsymbol{D}| - |\boldsymbol{D}'|| = 1, \tag{1}$$

则称 $\boldsymbol{D}$, $\boldsymbol{D}'$ 为兄弟数据集, 其中 $|\boldsymbol{D}|$, $|\boldsymbol{D}'|$ 表示数据集中记录的数量.

在兄弟数据集的定义基础上, Dwork 给出了 ϵ– 差分隐私的定义.

定义2 (ϵ– 差分隐私 [2]) 对于任意给定的两个兄弟数据集 $\boldsymbol{D}$, $\boldsymbol{D}'$, 若发布算法 A 对该对兄弟数据集的所有可能输出 $O \subseteq \mathrm{range}(A)$ 均满足

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

则称算法 A 满足 ϵ– 差分隐私.

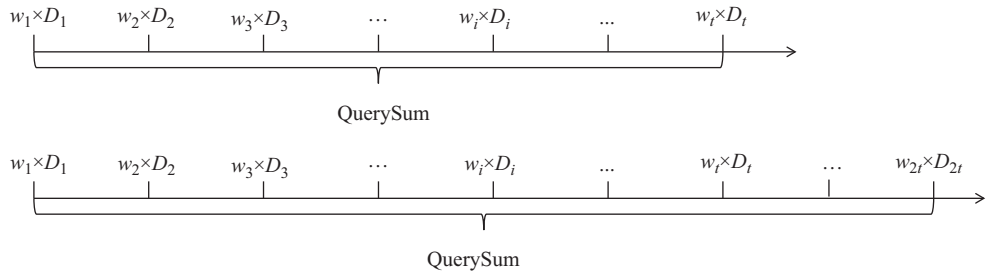

图 1 噪声累加问题

Figure 1 Noise accumulation

2.2 矩阵机制

矩阵机制 [11,12] 是一种在线性计数查询模型上进行差分隐私数据统计发布的方法, 并且具有成熟的理论框架. 给定一个负载矩阵 $\boldsymbol{W}$, 其中包含多条线性计数查询, 通过寻找最优策略矩阵 $\boldsymbol{L}$ 来提高负载矩阵查询结果的精度.

矩阵机制将负载矩阵 $\boldsymbol{W}$ 表示成两个矩阵 $\boldsymbol{B}$, $\boldsymbol{L}$ 相乘的形式, 即 $\boldsymbol{W} = \boldsymbol{BL}$. 其中 $\boldsymbol{L}$ 为最优策略矩阵, 先通过矩阵 $\boldsymbol{L}$ 将原始数据进行基变换, 并在变换的结果上添加独立噪声, 之后再通过还原矩阵 $\boldsymbol{B}$ 转换为最终的查询结果. 公式表示如下:

$$A(\boldsymbol{W}, \boldsymbol{X}) = \boldsymbol{B}\left(\boldsymbol{LX} + \mathrm{Lap}\left(\frac{\Delta \boldsymbol{L}}{\epsilon}\right)\right). \tag{3}$$

矩阵机制的均方误差由下式 [12] 表示:

$$\mathrm{error}(\boldsymbol{W}) = \frac{2}{\epsilon^2}\mathrm{trace}\left(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}\right)\Delta L^2. \tag{4}$$

矩阵机制中添加的 Laplace 噪声规模为策略矩阵的 1– 范数, 由文献 [11] 中的结论可得其满足 ϵ– 差分隐私. 本文将采用与矩阵机制相同的加噪方法.

2.3 加权连续统计发布

加权连续统计发布是指设当前用户的查询范围为 $[1, t]$, 随着时间的增长, 每个数据项在查询结果中的权重会产生变化, 进而对结果产生影响. 查询结果可用如下公式进行表示:

$$\mathrm{result}(t) = \sum_{i=1}^{t}(w_i \times D_i). \tag{5}$$

在式 (5) 中, w_i 表示编号为 i 的节点的查询权重, t 表示当前时刻. 如图 1 所示, 通过分别将各个时间节点上的统计值加上独立 Laplace 噪声, 并与权重相乘, 而后进行数据发布. 然而, 由于原始值与加噪值存在噪声误差, 随着时间 t 增长, 大范围的连续计数查询会累积大量噪声, 降低数据发布精度.

为解决上述问题, 本文首先将加权连续统计发布查询转化为负载矩阵 $\boldsymbol{W}$, 进而利用矩阵机制处理连续统计发布查询间的关联性以提高其发布精度. 在指数衰减模式中其权重变换可以表示为如下公式:

$$w_i = p^{t-i}, \tag{6}$$

其中 t 表示当前时刻, w_i 表示当前时刻的权重, 随着发布时刻距离当前时刻越远, 则其权重越小. $p \in (0,1)$ 为衰减因子, 影响着权重衰减的速度.

基于权重衰减公式 (6), 可得到相应的负载矩阵 $\boldsymbol{W}$ 如下所示:

$$\boldsymbol{W}_p = \begin{pmatrix} 1 & 0 & 0 & \cdots \\ p & 1 & 0 & \cdots \\ p^2 & p & 1 & \cdots \\ \cdots & \cdots & \cdots & \ddots \end{pmatrix}. \tag{7}$$

3 指数衰减模式下的差分隐私流数据发布

在流数据连续统计发布中, 每两次发布之间都具有相关性, 若将其转换为矩阵表示, 则可以通过矩阵机制利用查询的相关性来提升发布精度. 为实现流数据的有效发布, 首先应结合流数据特性构造相应的策略矩阵.

3.1 策略矩阵构造

在指数衰减模式下的差分隐私流数据发布中, 数据是在发布过程中动态产生的, 未来的数据无法预先得知, 且两次发布之间存在相关性, 因此只能通过当前数据和历史数据来优化查询结果. 这一特征反映到矩阵机制时, 就要求矩阵机制所构造的策略矩阵 $\boldsymbol{L}$ 为下三角矩阵, 且对角线以上元素均为 0, 从而保证与当前发布时刻相关的数据只有历史数据及当前的数据. 同时, 为使发布数据更具可用性, 应将误差期望控制在一定范围内, 可借鉴 Boost [13] 的均方误差期望 $O(\log_2^3 N)$, 构造的策略矩阵应使得噪声带来的均方误差不高于 $O(\log_2^3 N)$. 经研究发现, 若利用树状数组进行策略矩阵构造可使所构造的策略矩阵满足下三角满秩特性, 同时满足要求的误差期望.

树状数组是一个查询和修改复杂度都为 $O(\log_2 N)$ 级别的数据结构, 对于给定的 r, 可以快速求得区间 $[1,r]$ 的和值. 设区间 $[1,r]$ 的和为 $\mathrm{Sum}(r)$, 即 $\mathrm{Sum}(r) = \sum_{j=1}^{r} D_j$.

树状数组在计算过程中, 生成了中间统计量 $S_i (i \in [1,r])$, 如下:

$$S_i = \sum_{j=i-\mathrm{lowbit}(i)+1}^{i} D_j \quad (i = 1, 2, \ldots, r), \tag{8}$$

其中 D_j 表示第 j 个数的值, $\mathrm{lowbit}(x)$ 为用二进制表示 x 后, 只保留其最低位的 1. 以 $x = 12$ 为例, 其二进制数表示为 $(1100)_2$, 最低位的 1 为右数第三位, 则 $\mathrm{lowbit}(x) = (0100)_2 = (4)_{10}$. 通过补码性质可知, $\mathrm{lowbit}(x) = x\&(-x)$. 而后, 树状数组通过中间统计量 S_i, 得到区间和值如下:

$$\mathrm{Sum}(r) = \sum_{i=\lfloor \log_2(\mathrm{lowbit}(r)) \rfloor}^{\lfloor \log_2(r) \rfloor} S_{r-r \bmod i}. \tag{9}$$

树状数组生成中间变量的过程可以通过矩阵与向量相乘的形式表示, 当 $r=7$ 时, 其表示形式如下所示:

$$\boldsymbol{S}=\boldsymbol{L}\times\boldsymbol{D}=\begin{pmatrix}1&0&0&0&0&0&0\\1&1&0&0&0&0&0\\0&0&1&0&0&0&0\\1&1&1&1&0&0&0\\0&0&0&0&1&0&0\\0&0&0&0&1&1&0\\0&0&0&0&0&0&1\end{pmatrix}\times\begin{pmatrix}1\\3\\5\\2\\4\\7\\6\end{pmatrix}=\begin{pmatrix}1\\4\\5\\11\\4\\11\\6\end{pmatrix},\tag{10}$$

其中 $\boldsymbol{L}$ 表示策略矩阵, $\boldsymbol{D}$ 表示原始数据集, $\boldsymbol{S}$ 表示中间变量向量, 即通过策略矩阵将数据表示为中间变量的形式, 添加噪声后再利用矩阵将其还原为查询结果.

由于 $\boldsymbol{W}=\boldsymbol{BL}$, $\boldsymbol{L}$ 和 $\boldsymbol{W}$ 均已知且为可逆矩阵, 因此还原矩阵 $\boldsymbol{B}$ 可以表示为 $\boldsymbol{B}=\boldsymbol{WL}^{-1}$. 但矩阵求逆运算的时间复杂度为 $O(N^3)$, 无法满足流数据发布的实时性要求, 因此矩阵 $\boldsymbol{B}$ 不能直接计算得到, 而需通过构造得到. 为此, 可利用式 (9) 对矩阵 $\boldsymbol{B}$ 进行构造, 当 $r=7$ 时, 其形式如下:

$$\boldsymbol{WD}=\boldsymbol{B}\times\boldsymbol{S}=\begin{pmatrix}1&0&0&0&0&0&0\\0&1&0&0&0&0&0\\0&1&1&0&0&0&0\\0&0&0&1&0&0&0\\0&0&0&1&1&0&0\\0&0&0&1&0&1&0\\0&0&0&1&0&1&1\end{pmatrix}\times\begin{pmatrix}1\\4\\5\\11\\4\\11\\6\end{pmatrix}=\begin{pmatrix}1\\4\\9\\11\\15\\22\\28\end{pmatrix}.\tag{11}$$

在指数衰减模式下, 由于数据项带有权重, 因此, 需要对式 (8) 与 (9) 进行调整, 从而使得查询结果满足指数衰减的要求. 调整后的公式如下:

$$S_i=\sum_{j=i-\mathrm{lowbit}(i)+1}^{i}p^{i-j}D_j\quad(i=1,2,\ldots,r),\tag{12}$$

$$\mathrm{Sum}(r)=\sum_{i=\lfloor\log_2(\mathrm{lowbit}(r))\rfloor}^{\lfloor\log_2(r)\rfloor}p^{r\bmod i}S_{r-r\bmod i},\tag{13}$$

其中 p 表示预设的衰减因子.

当 $r=7$ 时, 根据式 (12), 衰减因子为 p 时的策略矩阵如下:

$$\boldsymbol{L}=\begin{pmatrix}1&0&0&0&0&0&0\\p&1&0&0&0&0&0\\0&0&1&0&0&0&0\\p^3&p^2&p&1&0&0&0\\0&0&0&0&1&0&0\\0&0&0&0&p&1&0\\0&0&0&0&0&0&1\end{pmatrix}.\tag{14}$$

当 $r=7$ 时, 根据式 (13), 衰减因子为 p 时的还原矩阵如下所示:

$$\boldsymbol{B}=\begin{pmatrix}1&0&0&0&0&0&0\\0&1&0&0&0&0&0\\0&p&1&0&0&0&0\\0&0&0&1&0&0&0\\0&0&0&p&1&0&0\\0&0&0&p^2&0&1&0\\0&0&0&p^3&0&p&1\end{pmatrix}. \tag{15}$$

通过式 (12) 和 (13) 的计算过程, 和对矩阵 (14) 和 (15) 的观察可以发现, $\boldsymbol{L}$ 矩阵与 $\boldsymbol{B}$ 矩阵均为稀疏矩阵, 因此在计算矩阵乘法的过程中, 无需直接进行矩阵运算, 只要计算非零元素即可.

基于以上分析, 设计策略矩阵 $\boldsymbol{L}$ 和还原矩阵 $\boldsymbol{B}$ 的一般性构造方法如算法 1 和 2 所示.

算法 1 策略矩阵构造算法 BuildL

Require: 数据规模 N, 衰减因子 p;
Ensure: 策略矩阵 $\boldsymbol{L}$;
1: 初始化策略矩阵 $\boldsymbol{L}$, 将所有元素置为 0;
2: **for** $j=1$ to N **do**
3: /* 按顺序遍历每一列 */
4: $i=j$; /* 第 j 列从 j 行开始构造 */
5: **while** $i<N$ **do**
6: 更新矩阵元素 $\boldsymbol{L}[i][j]=p^{i-j}$;
7: $i\leftarrow i+\text{lowbit}(i)$;
8: **end while**
9: **end for**
10: 返回矩阵 $\boldsymbol{L}$.

算法 2 策略矩阵构造算法 BuildB

Require: 数据规模 N, 衰减因子 p;
Ensure: 还原矩阵 $\boldsymbol{B}$;
1: 初始化策略矩阵 $\boldsymbol{B}$, 将所有元素置为 0;
2: **for** $i=1$ to N **do**
3: /* 按顺序遍历每一行 */
4: $j=i$; /* 第 i 行从第 i 列开始构造 */
5: **while** $j>0$ **do**
6: 更新矩阵元素 $\boldsymbol{B}[i][j]=p^{i-j}$;
7: $j\leftarrow j-\text{lowbit}(j)$;
8: **end while**
9: **end for**
10: 返回矩阵 $\boldsymbol{B}$.

设流数据规模为 N, 在矩阵 $\boldsymbol{B}$ 中的非零元素个数为 $O(N\log_2 N)$, 且均小于等于 1, 因此$\text{trace}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B})$的大小为 $O(N\log_2 N)$. 在矩阵 $\boldsymbol{L}$ 中, 每一列非零元素个数为 $O(\log_2 N)$, 且均小于等于 1, 因此其 1– 范数为 $O(\log_2 N)$, 根据式 (4) 可得总体均方误差为 $O(N\log_2^3 N)$, 对于每条查询的均方误差为 $O(\log_2^3 N)$. 因此, 利用树状数组构造策略矩阵是符合均方误差复杂度要求的. 至此, 完成了完整的策略矩阵构造过程. 然而, 通过预先构造 $\boldsymbol{L}$ 和 $\boldsymbol{B}$ 来进行发布, 无法满足流数据的实时性要求. 因此, 在实际数据发

布过程中, 结合树状数组中间变量与和值的计算方法, 可以在 $O(\log_2 N)$ 的时间复杂度下发布一次数据. 据此, 设计出满足流数据发布实时性要求的指数衰减模式下的流数据发布算法 DM, 具体过程如算法 3 所示.

算法 3 指数衰减模式下差分隐私流数据发布 DM

Require: 预设时刻上限 T, 衰减因子 p;
Ensure: 每一时刻的发布结果 s_t;
1: **for** $t = 1$ to T **do**
2: 更新实际统计量 $\Phi_{\text{lowbit}(t)} \leftarrow D_t + \sum_{j=0}^{\text{lowbit}(t)-1} \Phi_j$;
3: 添加噪声 $\tilde{\Phi}_{\text{lowbit}(t)} \leftarrow \Phi_{\text{lowbit}(t)} + \text{Lap}(\frac{\Delta L}{\epsilon})$;
4: $k \leftarrow t, s_t \leftarrow 0$; /* 初始化发布值 */
5: **while** $k > 0$ **do**
6: $s_t \leftarrow s_t + (\tilde{\Phi}_{\text{lowbit}(k)} \times p^{i-k})$;
7: $k \leftarrow k - \text{lowbit}(k)$;
8: **end while**
9: 发布隐私数据 s_t.
10: **end for**

3.2 利用对角矩阵优化发布精度

通过对策略矩阵的进一步分析可以发现, 该矩阵每一列的和值并未全部达到 1– 范数, 因此, 在敏感度不变的前提下, 仍可通过对该矩阵调整, 使得 $\text{trace}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B})$ 的值降低, 以进一步提升数据发布的精度. 如下所示为一个衰减因子为 0.3 的策略矩阵优化过程:

$$\boldsymbol{L}_7 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.3 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0.027 & 0.09 & 0.3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.3 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \Rightarrow \boldsymbol{L}_7' = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.3 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0.027 & 0.09 & 0.3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.3 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.3 \end{pmatrix}. \tag{16}$$

通过计算可得 $\|\boldsymbol{L}_7\|_1 = 1.327$. 若将第 7 行乘以 1.3, 得到 $\boldsymbol{L}_7'$, $\boldsymbol{L}_7'$ 的列范数仍然为 1.327, 在不改变策略矩阵敏感度的同时, 可将相应的还原矩阵 $\boldsymbol{B}_7$ 改变为 $\boldsymbol{B}_7'$,

$$\boldsymbol{B}_7 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.3 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.3 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0.09 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.027 & 0 & 0.3 & 1 \end{pmatrix} \Rightarrow \boldsymbol{B}_7' = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.3 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.3 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0.09 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.027 & 0 & 0.3 & 0.769 \end{pmatrix}. \tag{17}$$

由式 (4) 得转换前的查询误差为 $\text{error}(\boldsymbol{W}) = \frac{2}{\epsilon^2}\text{trace}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B})\Delta L^2 = \frac{25.635}{\epsilon^2}$. 在对第 7 行乘以 1.3 后, 查询误差变为 $\text{error}(\boldsymbol{W}) = \frac{2}{\epsilon^2}\text{trace}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B})\Delta L^2 = \frac{24.196}{\epsilon^2}$. 由此可发现, 若通过对策略矩阵进行行变换, 即对策略矩阵左乘一个对角矩阵, 将能够进一步提升 DM 算法的数据发布精度. 如下将给出对角矩阵的求解过程.

首先, 设对角矩阵

$$\boldsymbol{\Lambda}_N = \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_N \end{pmatrix},$$

表示一个 $N \times N$ 的对角矩阵, 在引入该对角阵后, 算法框架由式 (3) 修改为

$$A(\boldsymbol{W}, \boldsymbol{D}) = \boldsymbol{B}\boldsymbol{\Lambda}^{-1}\left(\boldsymbol{\Lambda}\boldsymbol{L}\boldsymbol{D} + \mathrm{Lap}\left(\frac{\Delta_{\boldsymbol{\Lambda}\boldsymbol{L}}}{\epsilon}\right)\right), \tag{18}$$

当 $\boldsymbol{\Lambda} = \boldsymbol{E}$ 时, 式 (18) 即退化为式 (3).

引入对角阵后, 误差期望公式转换为

$$\mathrm{error}(\boldsymbol{W}) = \frac{2}{\epsilon^2}\mathrm{trace}\left(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{\Lambda}^{-2}\right)\Delta^2_{\boldsymbol{\Lambda}\boldsymbol{L}}. \tag{19}$$

根据文献 [12] 中的结论, 令 $\boldsymbol{B}' = \alpha\boldsymbol{B}, \boldsymbol{L}' = \alpha^{-1}\boldsymbol{L}$ 有

$$\frac{2}{\epsilon^2}\mathrm{trace}\left(\boldsymbol{B}'^{\mathrm{T}}\boldsymbol{B}'\boldsymbol{\Lambda}^{-2}\right)\Delta^2_{\boldsymbol{\Lambda}\boldsymbol{L}} = \frac{2}{\epsilon^2}\mathrm{trace}\left(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{\Lambda}^{-2}\right)\Delta^2_{\boldsymbol{\Lambda}\boldsymbol{L}},$$

因此, 可约束 $|\boldsymbol{\Lambda}\boldsymbol{L}|_1 \leqslant 1$. 综上, 为使误差期望最小化, 该优化问题将表示为如下形式:

$$\mathrm{opt}: \min_{\boldsymbol{\Lambda}_N} f(\boldsymbol{\Lambda}_N) = \frac{2}{\epsilon^2}\mathrm{trace}\left(\boldsymbol{B}_N^{\mathrm{T}}\boldsymbol{B}_N\boldsymbol{\Lambda}_N^{-2}\right) \quad \text{s.t.} \quad |\boldsymbol{\Lambda}_N\boldsymbol{L}_N|_1 \leqslant 1, \tag{20}$$

其中 $\boldsymbol{B}_N$ 表示大小为 $N \times N$ 的 $\boldsymbol{B}$ 矩阵, $\boldsymbol{L}_N$ 表示大小为 $N \times N$ 的 $\boldsymbol{L}$ 矩阵.

设列向量 $\boldsymbol{H}_N$ 表示对角阵 $\boldsymbol{\Lambda}_N$ 的对角线元素构成, 即 $\boldsymbol{H}_N = (\lambda_1, \ldots, \lambda_N)^{\mathrm{T}}$. 将式 (20) 转化为如下形式表示:

$$\mathrm{opt}: \min_{\boldsymbol{H}_N} f(\boldsymbol{H}_N) = \sum_{j=1}^{N} \frac{\boldsymbol{B}(:,j)^{\mathrm{T}}\boldsymbol{B}(:,j)}{\lambda_j^2} \quad \text{s.t.} \quad \boldsymbol{L}_N^{\mathrm{T}}\boldsymbol{H}_N \leqslant \boldsymbol{E}_{N\times 1},\ \lambda_i > 0 \ \ (i = 1, 2, \ldots, N), \tag{21}$$

其中 $\boldsymbol{E}_{N\times 1}$ 表示全为 1 的列向量. 通过求解表达式 (21), 即可根据求出的 $\lambda_1, \lambda_2, \ldots, \lambda_N$ 得到相应的对角矩阵. (为书写简便, 文中对所涉及的向量间比较运算进行了简写, 如用 “小于” 表示向量中的每一元素均小于另一向量, $\boldsymbol{B}(:,j)$ 表示还原矩阵的第 j 列向量.)

以下将通过 3 个部分详细介绍优化表达式 (21) 的求解过程. 首先, 将矩阵 $\boldsymbol{L}$ 展开表示, 发现其具有如下递推关系:

$$\boldsymbol{L}_{2^m-1} = \begin{pmatrix} \boldsymbol{L}_{2^{m-1}-1} & \boldsymbol{O}_{(2^{m-1}-1)\times 1} & \boldsymbol{O}_{(2^{m-1}-1)\times(2^{m-1}-1)} \\ \boldsymbol{P}_{1\times(2^{m-1}-1)} & 1 & \boldsymbol{O}_{1\times(2^{m-1}-1)} \\ \boldsymbol{O}_{(2^{m-1}-1)\times(2^{m-1}-1)} & \boldsymbol{O}_{(2^{m-1}-1)\times 1} & \boldsymbol{L}_{2^{m-1}-1} \end{pmatrix}, \tag{22}$$

其中 $\boldsymbol{P}_{1\times(2^{m-1}-1)} = (p, p^2, \ldots, p^{2^{m-1}-1})$.

当 $N = 2^m - 1$ 时, 可将 $\boldsymbol{H}_N$ 分解成 3 个部分, 即

$$\boldsymbol{H}_{2^m-1} = \left(\boldsymbol{H}^{(1)\mathrm{T}}_{2^m-1}, \lambda_{2^{m-1}}, \boldsymbol{H}^{(2)\mathrm{T}}_{2^m-1}\right)^{\mathrm{T}}, \tag{23}$$

其中, $\boldsymbol{H}_{2^m-1}^{(1)\mathrm{T}}=(\lambda_1,\ldots,\lambda_{2^{m-1}-1})$, $\boldsymbol{H}_{2^m-1}^{(2)\mathrm{T}}=(\lambda_{2^{m-1}+1},\ldots,\lambda_{2^m-1})$.

因此, 误差期望公式 $f(\boldsymbol{H}_{2^m-1})$ 可分解为 3 部分:

$$f(\boldsymbol{H}_{2^m-1})=f^{(1)}\left(\boldsymbol{H}_{2^m-1}^{(1)}\right)+f^{(2)}(\lambda_{2^{m-1}})+f^{(3)}\left(\boldsymbol{H}_{2^m-1}^{(2)}\right). \tag{24}$$

根据式 (21) 的约束条件, 结合式 (22) 展开得

$$\begin{pmatrix}\boldsymbol{L}_{2^{m-1}-1}^{\mathrm{T}}\boldsymbol{H}_{2^m-1}^{(1)}+\lambda_{2^{m-1}}\boldsymbol{P}_{1\times(2^{m-1}-1)}^{\mathrm{T}}\\ \lambda_{2^{m-1}}\\ \boldsymbol{L}_{2^{m-1}-1}^{\mathrm{T}}\boldsymbol{H}_{2^m-1}^{(2)}\end{pmatrix}\leqslant\boldsymbol{E}_{(2^m-1)\times1}. \tag{25}$$

为使问题可以进行快速求解, 需对解空间进行更加严格的限制, 用次优解代替最优解. 由于对角阵是在保持矩阵 $\boldsymbol{L}$ 敏感度不变的前提下减小矩阵 $\boldsymbol{B}$ 带来的误差, 由此产生的可行解一定会使得总体均方误差不超过 $O(N\log_2^3N)$. 因此, 可利用

$$\boldsymbol{P}_{1\times(2^{m-1}-1)}^{\mathrm{T}}\leqslant\begin{pmatrix}p\\ \vdots\\ p\end{pmatrix},$$

将式 (25) 转化为

$$\begin{cases}\boldsymbol{L}_{2^{m-1}-1}^{\mathrm{T}}\boldsymbol{H}_{2^m-1}^{(1)}\leqslant(1-\lambda_{2^{m-1}}\times p)\boldsymbol{E}_{(2^{m-1}-1)\times1},\\ \lambda_{2^{m-1}}\leqslant1,\\ \boldsymbol{L}_{2^{m-1}-1}^{\mathrm{T}}\boldsymbol{H}_{2^m-1}^{(2)}\leqslant\boldsymbol{E}_{(2^{m-1}-1)\times1}.\end{cases} \tag{26}$$

至此, 将式 (21) 中的约束条件转换为式 (26) 中的 3 个约束条件. 由式 (26) 知 $\lambda_{2^{m-1}}$ 取值影响着 $\lambda_1\sim\lambda_{2^{m-1}-1}$ 的取值. 令 $\lambda_{2^{m-1}}$ 为待定系数, 设 $\lambda_{2^{m-1}}=\frac{1-\delta}{p}(1-p\leqslant\delta\leqslant1)$, 代入式 (21) 得

$$\boldsymbol{L}_{2^{m-1}-1}^{\mathrm{T}}\boldsymbol{H}_{2^m-1}^{(1)}\leqslant\delta\boldsymbol{E}_{(2^{m-1}-1)\times1}\Leftrightarrow\boldsymbol{L}_{2^{m-1}-1}^{\mathrm{T}}\left(\frac{1}{\delta}\boldsymbol{H}_{2^m-1}^{(1)}\right)\leqslant\boldsymbol{E}_{(2^{m-1}-1)\times1}.$$

令 $\mu_i=\frac{1}{\delta}\lambda_i$, $\boldsymbol{G}_N=\frac{1}{\delta}\boldsymbol{H}_N=(\mu_1,\ldots,\mu_N)$, 并将其代入式 (21) 后有

$$f^{(1)}\left(\boldsymbol{H}_{2^m-1}^{(1)}\right)=\frac{1}{\delta^2}\sum_{i=1}^{2^{m-1}-1}\frac{\boldsymbol{B}_{2^m-1}(:,i)^{\mathrm{T}}\boldsymbol{B}_{2^m-1}(:,i)}{\mu_i^2}=\frac{1}{\delta^2}f^{(1)}\left(\boldsymbol{G}_{2^m-1}^{(1)}\right),$$

从而得到新的优化表达式

$$\mathrm{opt}:\ \min\frac{1}{\delta^2}f^{(1)}\left(\boldsymbol{G}_{2^m-1}^{(1)}\right)\Leftrightarrow\min f^{(1)}\left(\boldsymbol{G}_{2^m-1}^{(1)}\right)\quad\text{s.t. }\boldsymbol{L}_{2^{m-1}-1}^{\mathrm{T}}\boldsymbol{G}_{2^m-1}^{(1)}\leqslant\boldsymbol{E}_{(2^{m-1}-1)\times1}. \tag{27}$$

将 $\boldsymbol{H}_{2^{m-1}-1}$ 代入 $\boldsymbol{G}_{2^m-1}^{(1)}$, 则问题等价于求解 $\boldsymbol{H}_{2^{m-1}-1}^{*}$, 即 $\boldsymbol{\Lambda}_{2^{m-1}-1}^{*}$. 由此可得 $\boldsymbol{G}_{2^m-1}^{*(1)}=\boldsymbol{H}_{2^{m-1}-1}^{*}=\frac{1}{\delta}\boldsymbol{H}_{2^m-1}^{*(1)}$, 即 $\boldsymbol{H}_{2^m-1}^{*(1)}=\delta\boldsymbol{H}_{2^{m-1}-1}^{*}$.

因此, 最优对角阵可以表示如下:

$$\boldsymbol{\Lambda}_{2^m-1}^{*}=\begin{pmatrix}\delta_m\boldsymbol{\Lambda}_{2^{m-1}-1}^{*} & & \\ & \frac{(1-\delta_m)}{p} & \\ & & \boldsymbol{\Lambda}_{2^{m-1}-1}^{*}\end{pmatrix}. \tag{28}$$

通过对式 (27) 的转换得到对角矩阵中的子结构性质, 从而使用分治思想对其求解. 假设当 $N = 2^{m-1}-1$ 时, 发布数据的期望最小均方误差为 $\mathrm{err}_{m-1} = \min_{\mathbf{\Lambda}_{2^{m-1}-1}} f(\mathbf{\Lambda}_{2^{m-1}-1})$, 可将上述问题转化为关于 δ 的最优化问题:

$$\mathrm{opt}:\ h(\delta) = \min\left(\frac{\mathrm{err}_{m-1}}{\delta^2} + \frac{2^{m-1}\times p^2}{(1-\delta)^2}\right) + \mathrm{err}_{m-1}, \quad \text{s.t. } 1-p \leqslant \delta \leqslant 1. \tag{29}$$

通过求导可得, 当 $\delta = \frac{\sqrt[3]{\mathrm{err}_{m-1}}}{\sqrt[3]{\mathrm{err}_{m-1}}+\sqrt[3]{2^{m-1}\times p^2}}$ 时, $h(\delta)$ 取得最小值, 最小值为$(\sqrt[3]{\mathrm{err}_{m-1}}+\sqrt[3]{2^{m-1}\times p^2})^3$ $+\mathrm{err}_{m-1}$. 因此将得到如下递归式:

$$\mathrm{err}_m = \begin{cases} 1, & m = 1, \\ \left(\sqrt[3]{\mathrm{err}_{m-1}} + \sqrt[3]{2^{m-1}\times p^2}\right)^3 + \mathrm{err}_{m-1}, & m > 1. \end{cases} \tag{30}$$

通过以上步骤的分析和推导, 以误差最小为目标求出待定系数的解, 从而得到相应的对角矩阵. 利用对角矩阵的子结构性质可以在 $O(\log_2 N)$ 的时间内求出任意一个对角阵系数. 从而, 形成高效对角阵系数的求解算法 (见算法 4).

算法 4 对角阵系数求解算法 getLambda

Require: 时刻上限 T, 下标 k, 衰减因子 p;
Ensure: 对角阵系数 λ_k;
1: 初始化 λ_k 为 1, 根据式 (30) 计算出所需的系数 $\delta_1 \sim \delta_{\log_2(T)+1}$;
2: $kt \leftarrow k,\ m \leftarrow \log_2(T)+1,\ \mathrm{div} \leftarrow 2^{m-1}$;
3: **while** div $\neq kt$ **do**
4: **if** $kt <$ div **then**
5: $\lambda_k \leftarrow \lambda_k \times \delta_m$;
6: **else**
7: $kt \leftarrow kt - \mathrm{div}$;
8: **end if**
9: $\mathrm{div} \leftarrow \frac{\mathrm{div}}{2},\ m \leftarrow m-1$;
10: **end while**
11: $\lambda_k \leftarrow \frac{\lambda_k \times (1-\delta_m)}{p}$;
12: 返回对角阵系数 λ_k.

至此, 提出完整的指数衰减模式下基于对角矩阵优化的差分隐私流数据发布算法 DMFDA (见算法 5).

当流数据规模为 N 时, 通过算法 4 的步骤 9 可知, 每计算一个对角阵元素, 数据规模都会除以 2, 因此对角阵优化算法的时间复杂度为 $O(N\log_2 N)$. 构造负载矩阵分解算法的时间复杂度为 $O(N\log_2 N)$, 因此, DMFDA 算法整体时间复杂度为 $O(N\log_2 N)$ 与 DM 算法同阶.

4 实验分析

4.1 实验数据与环境

为方便对比与分析, 本文采用了文献 [13] 中的数据集 Search Logs 和 NetTrace, 以及文献 [14] 中使用的 WorldCup98 数据集进行对比实验. Search Logs 对在 2004.01 ~ 2009.08 期间, 某网站对关键词

表 1 数据集

Table 1 Data sets

Data set	Search Logs	NetTrace	WorldCup98
Size	32768	65536	7518579

算法 5 指数衰减模式下的基于对角矩阵优化差分隐私流数据发布算法 DMFDA

Require: 时刻上限 T, 衰减因子 p;
Ensure: 每一时刻的发布结果 s_t;
1: **for** $t = 1$ to T **do**
2: 更新实际统计量 $\Phi_{\text{lowbit}(t)} \leftarrow D_t + \sum_{j=0}^{\text{lowbit}(t)-1} \Phi_j$;
3: $\lambda_t \leftarrow \text{getLambda}(T, t, p)$;
4: 添加噪声 $\tilde{\Phi}_{\text{lowbit}(t)} \leftarrow \lambda_t \times \Phi_{\text{lowbit}(t)} + \text{Lap}(\frac{1}{\epsilon})$;
5: $k \leftarrow t$, $s_t \leftarrow 0$; /* 初始化发布值 */
6: **while** $k > 0$ **do**
7: $s_t \leftarrow s_t + \frac{(\tilde{\Phi}_{\text{lowbit}(k)} \times p^{i-k})}{\lambda_k}$;
8: $k \leftarrow k - \text{lowbit}(k)$;
9: **end while**
10: 发布隐私数据 s_t.
11: **end for**

"Obama" 的搜索次数进行了统计. NetTrace 数据集包含了某单位在特定时间段内对特定 IP 段的数据包请求次数. WorldCup98 为 1998.04 ∼ 1998.07 期间, 世界杯官网的访问量的统计记录. 其数据规模如表 1 所示.

由于算法针对连续统计发布, 误差大小与数据集大小相关, 在实验中, 采用该数据集中所有可能出现的连续统计查询的均方误差衡量算法发布数据的查询误差,

$$\text{error}(Q) = \sum_{q \in Q} (q(\boldsymbol{D}) - q(\boldsymbol{D}'))^2, \tag{31}$$

其中, Q 为查询集合, $q(\boldsymbol{D})$ 为连续统计查询的真实计数值, $q(\boldsymbol{D}')$ 为连续统计查询的加噪发布计数值.

实验环境为 Intel Core i5 4570 3.2 GHz 处理器, 8 GB 内存, Windows 7 操作系统; 算法用 C++ 语言实现; 由 EXCEL 生成实验图表.

4.2 查询误差的对比分析

实验在 Search Logs, Nettrace 及 WorldCup98 上进行对比实验, 着重关注在单条流数据下的连续统计发布问题. 通过 3 个算法的比较来说明算法的有效性, 其中 DM 算法为本文所提仅构造策略矩阵, 未加入对角阵优化算法的方法, DMFDA 为本文加入对角阵优化策略后的算法, EX (exponential sum) 算法为文献 [9] 所提出的利用区间树进行指数衰减模式下的流数据发布算法, 其总体误差也为 $O(N\log_2^3 N)$, 因此将通过实验对比的方式来证明利用查询间的关联性可以有效降低查询误差. 本文将通过 DMFDA 与 DM 算法的对比, 说明对角阵优化方法对于降低误差的有效性; 通过 DNFDA 与 EX 算法的对比, 说明相比于区间树方法, DMFDA 能够有效利用查询间的关联性降低误差. 实验设置了不同的隐私预算参数 ϵ, 分别为 1.0, 0.1, 0.01, 为了排除随机参数对实验的影响, 将每组实验运行 100 次的结果取平均值, 作为最终实验对比数据.

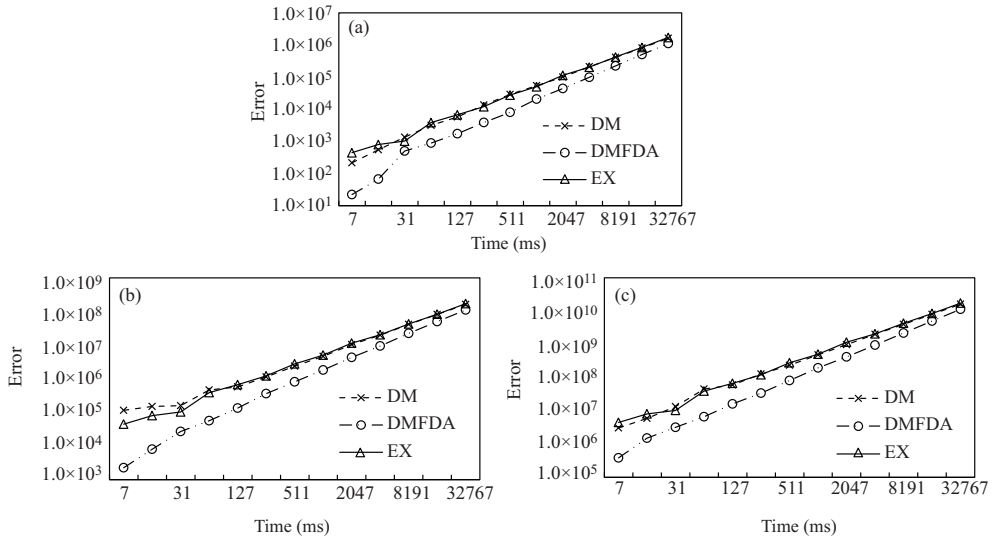

图 2　不同时刻下的查询误差对比 (Search Logs)

Figure 2　Comparison of query distortion with different moments (Search Logs). (a) $\epsilon = 1.0$; (b) $\epsilon = 0.1$; (c) $\epsilon = 0.01$

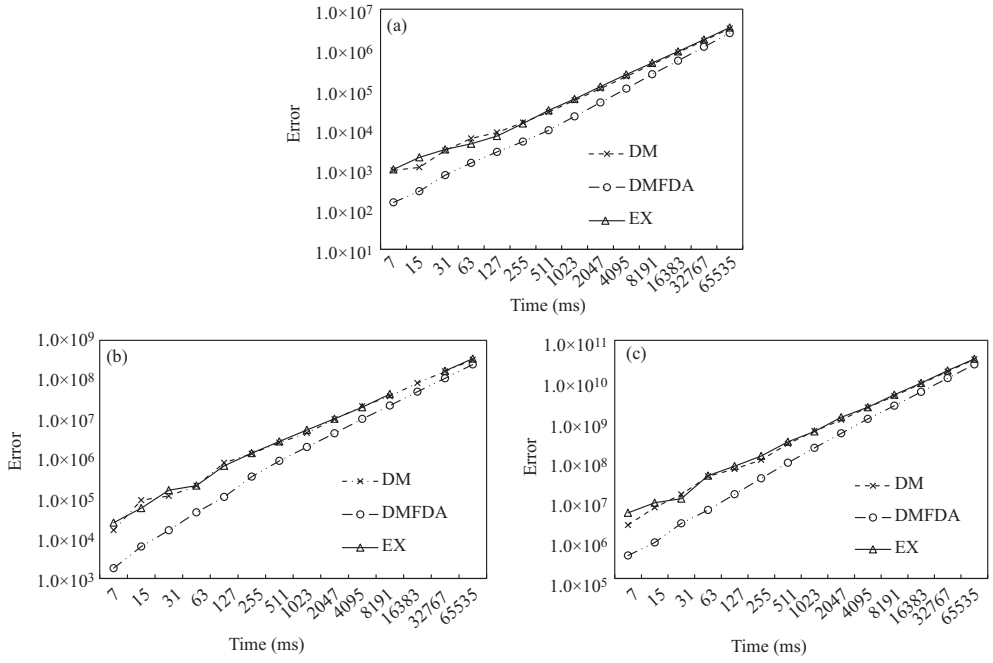

图 3　不同时刻下的查询误差对比 (Nettrace)

Figure 3　Comparison of query distortion with different moments (Nettrace). (a) $\epsilon = 1.0$; (b) $\epsilon = 0.1$; (c) $\epsilon = 0.01$

(1) 不同时刻下的查询误差对比. 本小节分组实验通过选取特定位置的时刻观测点, 对比分析连续统计查询的发布误差. 观测点位置分别选取 $2^3 - 1$, $2^4 - 1, \ldots, 2^{13} - 1, \ldots$. 查询使用的衰减因子 p 设置为 0.3. 横坐标表示时间, 纵坐标表示连续统计发布的均方误差, 并以 10 为底取对数. 实验结果如图 2 ∼ 4 所示.

在图 2 ∼ 4 的实验结果对比中可以发现, 随着时间增长, 均方误差以 10^2 的数量级增长, 同时相比

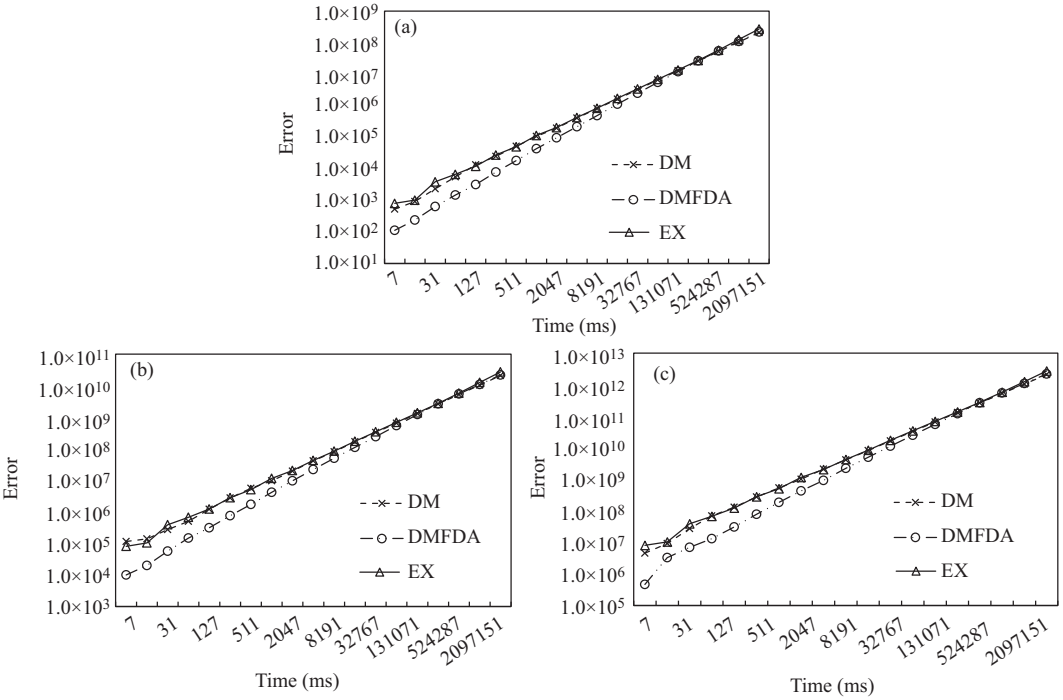

图 4 不同时刻下的查询误差对比 (WorldCup98)

Figure 4 Comparison of query distortion with different moments (WorldCup98). (a) $\epsilon = 1.0$; (b) $\epsilon = 0.1$; (c) $\epsilon = 0.01$

于 DM 和 EX, DMFDA 拥有更高的数据发布精度, 这是由于 DMFDA 使用了快速对角优化算法, 通过在不改变策略矩阵的 1– 范数的前提下, 使得策略矩阵变得更密集, 还原矩阵更稀疏, 从而减少了还原矩阵对误差造成的影响, 降低了发布误差. 在同一算法中, 随着 ϵ 的减小, 连续统计发布误差增加, 这是因为添加的 Laplace 噪声的规模随着隐私预算的减小而增加.

(2) 不同衰减因子下的误差对比. 在本小节分组实验中, 以衰减因子作为自变量, 对比分析查询误差. Search Logs 选取的观测点位置为 $2^{15}-1$, Nettrace 选取的观测点位置为 $2^{16}-1$, WorldCup98 选取的观测点位置为 $2^{21}-1$, 衰减因子大小分别取 0.1, 0.2, . . . , 0.9. 对比结果如图 5 ∼ 7 所示.

在图 5 ∼ 7 的对比实验中, 均方误差随着衰减因子增加而增加, 这是因为衰减因子越大, 还原矩阵就越密集, 产生的均方误差也更大. EX 算法通过预先设置的衰减因子计算出相应噪声规模的极限值, 因此在衰减因子接近 1 时, EX 算法造成的均方误差较大.

综合以上实验结果可以得出, 算法 DMFDA 能够有效适应各种衰减因子与隐私预算的应用场景, 使得查询误差更小.

4.3 算法运行效率的对比分析

本小节设置隐私参数为 1.0, 对衰减因子取 [0.1, 0.9], 对每个衰减因子运行 20 次后取平均值, 得到的运行时间取平均值, 对比各个算法的运行效率, 其实验结果如图 8 所示.

从图 8 可以得出, DM 算法具有相比 DMFDA 算法更优的算法执行效率, 这是因为 DMFDA 算法是在 DM 算法的基础上, 增加了对角阵优化, 而使算法时间复杂度的系数变大. 图 9 中给出对角阵优化过程 DA (diagonal matrix algorithm) 在不同数据规模下的运行时间. 与算法整体运行时间相比, 其

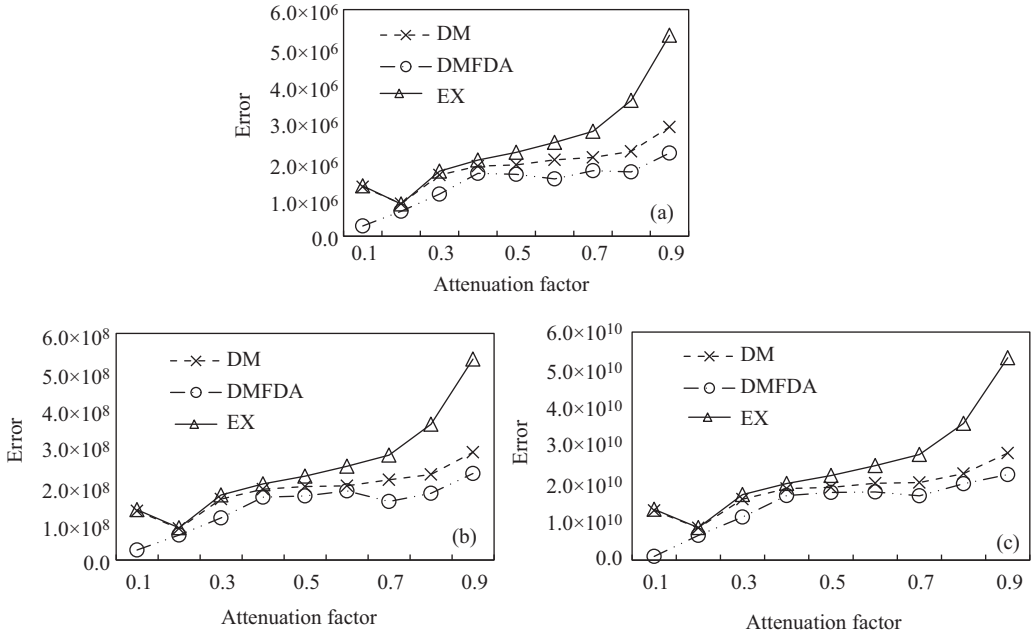

图 5　不同衰减因子下的查询误差对比 (Search Logs)

Figure 5　Comparison of query distortion with different decay factors (Search Logs). (a) $\epsilon = 1.0$; (b) $\epsilon = 0.1$; (c) $\epsilon = 0.01$

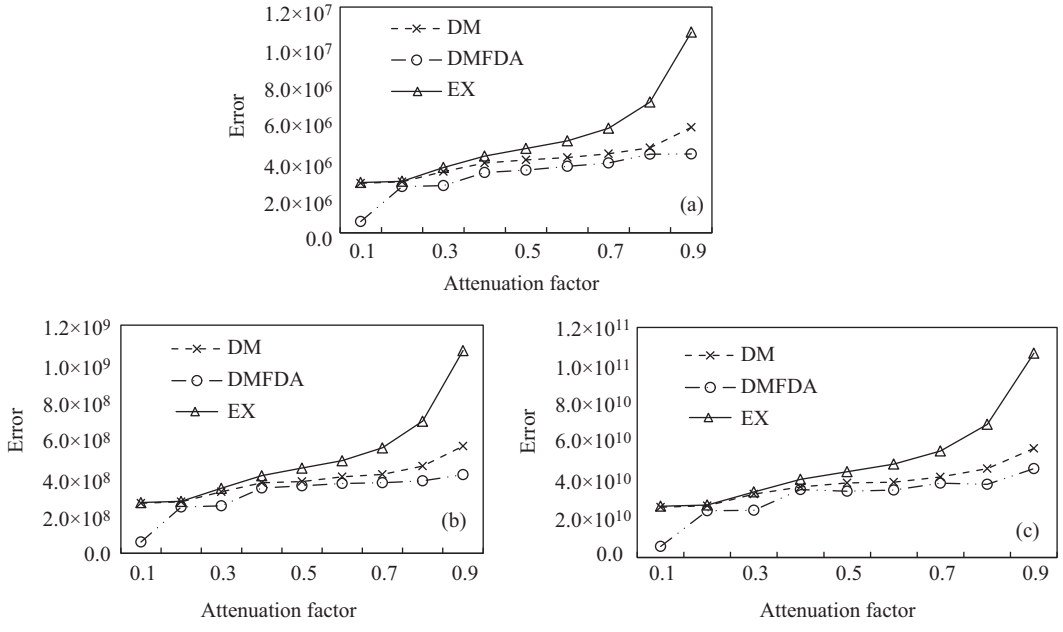

图 6　不同衰减因子下的查询误差对比 (Nettrace)

Figure 6　Comparison of query distortion with different decay factors (Nettrace). (a) $\epsilon = 1.0$; (b) $\epsilon = 0.1$; (c) $\epsilon = 0.01$

所占时间比例较小, 且其时间复杂度仍为 $O(N\log_2 N)$, 因此带来的算法效率降低仍在可接受范围内.

综合以上对比实验可以得出, DMFDA 算法具有更高的数据发布精度, 同时时间复杂度也能适应流数据高效发布的需求.

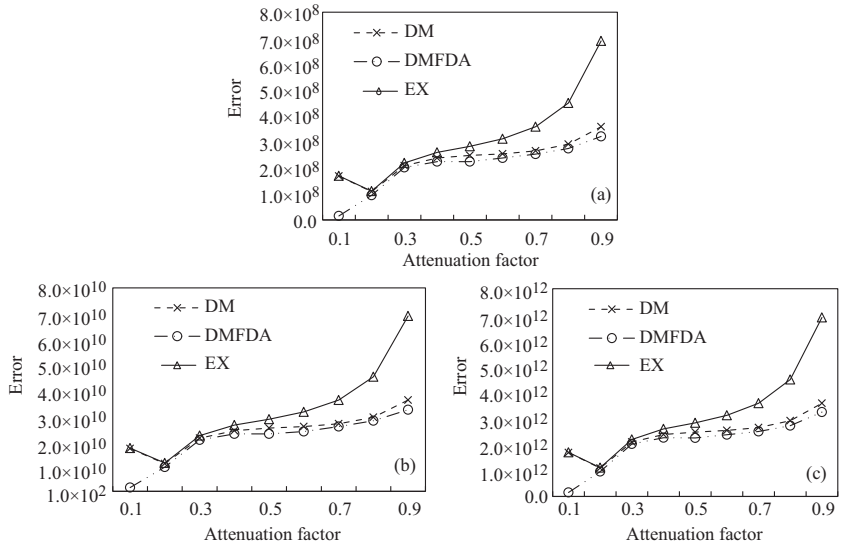

图 7 不同衰减因子下的查询误差对比 (WorldCup98)

Figure 7 Comparison of query distortion with different decay factors (WorldCup98). (a) $\epsilon = 1.0$; (b) $\epsilon = 0.1$; (c) $\epsilon = 0.01$

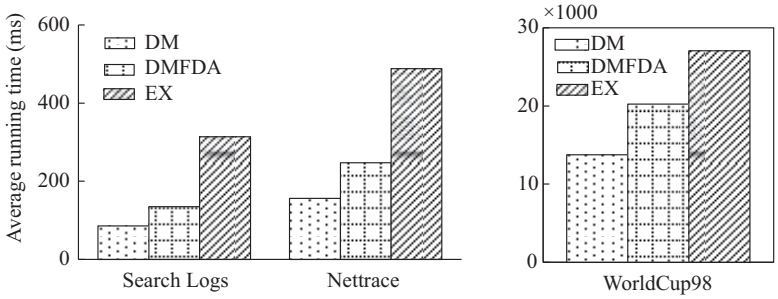

图 8 不同算法运行效率对比

Figure 8 Comparison of different algorithm efficiency

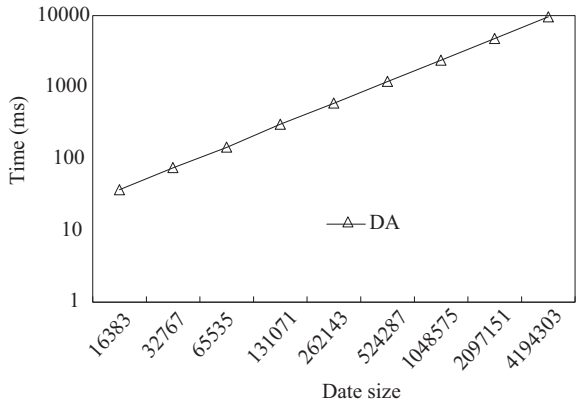

图 9 对角优化运行时间

Figure 9 Run time of diagonal optimization algorithm

5 结束语

本文针对指数衰减下差分隐私流数据发布问题, 提出了基于矩阵机制的差分隐私流数据发布算法, 并提出快速对角阵优化算法, 以有效应对大规模流数据发布问题. 实验对算法 DMFDA 所发布数据精度与同类流数据指数衰减算法进行比较分析, 实验结果表明, 算法 DMFDA 是有效可行的. 在今后的研究应用中, 将考虑把矩阵机制用于其他衰减模式下的流数据发布, 以提升数据的发布质量.

参考文献

1 Fung B, Wang K, Chen R, et al. Privacy-preserving data publishing: a survey of recent developments. ACM Comput Surv, 2010, 42: 2623–2627

2 Dwork C. Differential privacy. In: Proceedings of the 33rd International Conference on Automata, Languages and Programming, Venice, 2006. 1–12

3 Zhou S G, Li F, Tao Y F, et al. Privacy preservation in database applications: a survey. Chin J Comput, 2009, 32: 847–861 [周水庚, 李丰, 陶宇飞, 等. 面向数据库应用的隐私保护研究综述. 计算机学报, 2009, 32: 847–861]

4 Xiong P, Zhu T Q, Wang X F. A survey on differential privacy and applications. Chin J Comput, 2014, 37: 101–122 [熊平, 朱天清, 王晓峰. 差分隐私保护及其应用. 计算机学报, 2014, 37: 101–122]

5 Zhang X J, Meng X F. Differential privacy in data publication and analysis. Chin J Comput, 2014, 37: 927–949 [张啸剑, 孟小峰. 面向数据发布和分析的差分隐私保护. 计算机学报, 2014, 37: 927–949]

6 Dwork C, Naor M, Pitassi T, et al. Differential privacy under continual observation. In: Proceedings of the 42nd ACM Symposium on Theory of Computing, Cambridge, 2010. 715–724

7 Chan T H H, Shi E, Song D. Private and continual release of statistics. ACM Trans Inf Syst Secur, 2011, 14: 405–417

8 Cao J, Xiao Q, Ghinita G, et al. Efficient and accurate strategies for differentially-private sliding window queries. In: Proceedings of the 16th International Conference on Extending Database Technology, Genoa, 2013. 191–202

9 Zhang X J, Meng X F. Stream histogram publication method with differential privacy. J Softw, 2016, 27: 381–393 [张啸剑, 孟小峰. 基于差分隐私的流式直方图发布方法. 软件学报, 2016, 27: 381–393]

10 Bolot J, Fawaz N, Muthukrishnan S, et al. Private decayed predicate sums on streams. In: Proceedings of the 16th International Conference on Database Theory, Genoa, 2013. 284–295

11 Li C, Hay M, Rastogi V, et al. Optimizing linear counting queries under differential privacy. In: Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, Indianapolis, 2010. 123–134

12 Yuan G, Zhang Z, Winslett M, et al. Low-rank mechanism: optimizing batch queries under differential privacy. Proc VLDB Endowment, 2012, 5: 1352–1363

13 Hay M, Rastogi V, Miklau G, et al. Boosting the accuracy of differentially private histograms through consistency. Proc VLDB Endowment, 2010, 3: 1021–1032

14 Kellaris G, Papadopoulos S, Xiao X, et al. Differentially private event sequences over infinite streams. Proc VLDB Endowment, 2014, 7: 1155–1166

An algorithm for differential privacy streaming data publication based on matrix mechanism under exponential decay mode

Yingjie WU*, Chen GE, Liqun ZHANG & Lan SUN

College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350116*, China*
* Corresponding author. E-mail: yjwu@fzu.edu.cn

Abstract At present, many practical applications require the continuous release of statistical streaming data, and the importance of current data is higher than historical data. The solution to this problem is to assign weights to the data and propose a differential privacy data release method under exponential decay. However, existing methods only consider a single query, and cannot effectively use the correlation between queries in the continuous statistical publishing background to further improve the accuracy of the query. In this paper, we present a differential privacy data release algorithm (DMFDA) in exponential decay mode based on a matrix mechanism, which uses the advantages of the matrix to deal with relevant queries. Firstly, we use the construction method to generate the matrix decomposition strategy to meet the real-time requirements of streaming data. Secondly, the diagonal matrix is used to adjust the structure of the constructed strategy matrix so as to improve the release accuracy. Finally, according to the substructure of the constructed strategy matrix, a fast method of solving the diagonal matrix is proposed. The experiment is designed to compare DMFDA and similar algorithms for streaming data release in exponential decay. Experimental results show that the DMFDA algorithm is effective and feasible.

Keywords differential privacy, streaming data publication, exponential decay, matrix mechanism, diagonal matrix

Yingjie WU was born in 1979. He is a professor, Ph.D. holder, and senior member of the China Computer Federation. His main research interests include data mining and data privacy protection.

Chen GE was born in 1992. He is a Master's degree candidate at Fuzhou University. His research interests include data mining and differential privacy.

Liqun ZHANG was born in 1991. He is a Master's degree candidate at Fuzhou University. His research interests include data mining and differential privacy.

Lan SUN was born in 1978. She received her M.S. degree from Xi'an Jiaotong University in 2003. She is currently a lecturer at Fuzhou University. Her research interests include data security and privacy protection.