中国科学: 信息科学 2023 年 第 53 卷 第 6 期: 1111–1131
SCIENTIA SINICA Informationis
论文

基于层次结构的隐私多维分析查询算法

张啸剑[1*], 周丹[1], 徐雅鑫[1], 林东岱[2], 纪守领[3], 孟小峰[4]

1. 河南财经政法大学计算机与信息工程学院, 郑州 450002
2. 中国科学院信息工程研究所, 北京 100093
3. 浙江大学计算机科学与技术学院, 杭州 310027
4. 中国人民大学信息学院, 北京 100872
* 通信作者. E-mail: xjzhang82@alu.ruc.edu.cn

收稿日期: 2022–08–06; 修回日期: 2022–10–12; 接受日期: 2022–11–30; 网络出版日期: 2023–06–05

国家自然科学基金 (批准号: 62072156, 91746115, 61502146, 91646203) 资助项目

摘要 基于本地化差分隐私的多维分析查询 (multi-dimensional analytical query, MDA) 已得到了研究者的广泛关注. 现有基于最优局部哈希 (optimal local Hashing, OLH) 机制与层次树结构的扰动方法存在泄露根结点隐私的风险. 针对现有结合层次树结构的本地扰动机制不足, 提出了一种有效且满足本地化差分隐私的 MDA 查询算法 H4MDA (hierarchical structure for MDA), 该算法充分利用层次树的横向与纵向结构特征设计了 3 种基于用户分组策略的本地扰动算法 HGRR, LGRR-FD, LGRR. 算法 HGRR 结合层次树横向结构与 GRR 机制本地扰动用户元组数据, 通过摈弃根结点组合来响应 MDA 查询. 不同于 HGRR, LGRR-FD 算法利用层次树的纵向结构与 GRR 机制扰动本地数据, 同时通过添加假数据来避免叶子结点的隐私泄露. LGRR 算法通过摈弃叶子结点层纵向扰动本地数据. 收集者结合 LGRR 的扰动结果利用局部一致性处理技术重构层次树最后两层, 通过添加虚拟叶子结点来响应 MDA 查询, 而虚拟叶子结点计数之和等于其父节点计数. HGRR, LGRR-FD, LGRR 算法与现有扰动算法在 3 种数据集上实验结果表明, 其响应 MDA 查询的精度优于同类算法.

关键词 多维分析查询, 层次结构, 本地化差分隐私, 本地扰动, 随机应答机制

1 引言

人工智能技术的高速发展使得多维数据的收集与分析变得尤为容易, 例如在线购物、医疗、金融等数据的收集与分析. 通过收集与分析用户的个人数据可以改变企业产品与设备的服务质量, 向用户提供个性化服务. 然而, 多维数据通常蕴含着丰富的个人敏感信息, 在提供给收集者或者第三方时, 个人的敏感信息有可能被泄露. 例如, 用户利用某在线网购 APP 购物后, 会产生相应的多维数据, 如

引用格式: 张啸剑, 周丹, 徐雅鑫, 等. 基于层次结构的隐私多维分析查询算法. 中国科学: 信息科学, 2023, 53: 1111–1131, doi: 10.1360/SSI-2022-0310
Zhang X J, Zhou D, Xu Y X, et al. Answering private multidimensional analytical queries with hierarchical structure (in Chinese). Sci Sin Inform, 2023, 53: 1111–1131, doi: 10.1360/SSI-2022-0310

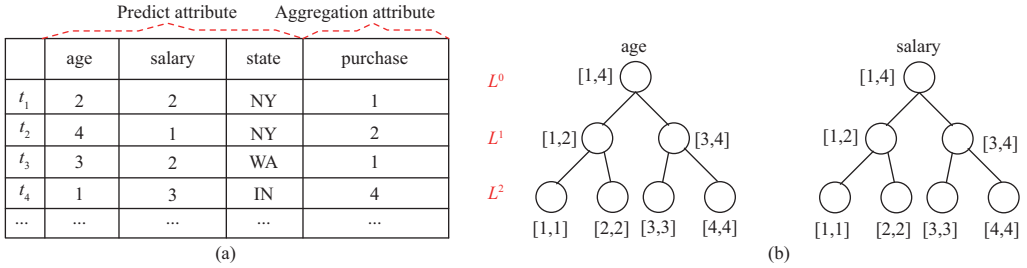

图 1 (网络版彩图) 基于用户属性信息的层次树结构

Figure 1 (Color online) Hierarchical trees of user's attributes. (a) Shopping information of app users, T_{app}; (b) hierarchical trees of age and salary

图 1(a) 关系表 T_{app} 所示. 如果每个用户直接把数据共享给收集者, 则其敏感属性的对应值会被泄露, 如 $t_2[\text{salary}] = 1$. 基于 T_{app} 响应攻击者的 SQL 查询时, 用户的敏感信息也会被泄露. 例如基于 T_{app} 响应 Q_1 查询, COUNT(*) = 2. 如果攻击者以用户 t_1 作为攻击背景知识, 根据 COUNT(*) = 2 可以推理出只有 t_2 满足 WHERE 子句的谓词条件, 进而获得 t_2 的敏感信息 $t_2[\text{salary}] = 1$. 在此情况下, 用户无法掌控自己的多维数据. 本地化差分隐私 (local differential privacy, LDP) [1] 技术能够有效解决此类问题. 该技术允许每个用户采用本地扰动机制处理自身数据后报告给收集者, 而原始数据依旧掌控在自己手中.

Q_1: SELECT COUNT(*) FROM T_{app} WHERE age $\in [2, 4]$ And state = NY.

Q_2: SELECT SUM(purchase) FROM T_{app} WHERE age $\in [2, 4]$ And salary $\in$ [1, 3].

Q_3: SELECT AVG(purchase) FROM T_{app} WHERE age $\in [1, 4]$ And salary $\in$ [1, 2].

基于 LDP 的多维分析查询 (multi-dimensional analytical queries, MDA) 已成为隐私保护领域研究热点, 例如基于图 1(a) 给出的 MDA 查询 Q_1, Q_2 与 Q_3. 该类查询通常是在谓词表达式约束情况下响应聚集函数 (COUNT, SUM 与 AVG 等) 查询. 如基于图 1(a) 的 Q_2 查询结果为 SUM(purchase) = 4, Q_3 查询结果为 AVG(purchase) = $\frac{4}{3}$. 基于 LDP 响应 MDA 查询的算法分为边缘表法与层次树结构法. CALM (consistent adaptive local marginal) [2] 算法与 MargHT (Hadamard transform of a random marginal) [3] 算法是采用边缘表响应 MDA 查询的代表, 通过构建满足 LDP 的边缘表响应 MDA 查询. 然而该类算法的 MDA 查询误差随着维度 d 与值域 m 增加而增加, 通常正比于 m^d [4]. 基于层次树的 MDA 查询算法结合每个属性的值域构建相应层次树 (如图 1(b) 中属性 age 与 salary 对应的层次树), 再利用层次树实现隐私预算分割或者用户分组. 该类算法响应 MDA 查询的误差通常为 $\log^{O(d)} m$ [4].

目前, 结合层次树结构与 LDP 技术的 MDA 查询算法主要包括 HH (hierarchical histograms) [5], HI (hierarchical-interval mechanism) [4], HIO (hierarchical-interval optimized mechanism) [4], AHIO (augment hierarchical-interval optimized mechanism) [6], EHIO (embed hierarchical-interval optimized mechanism) [6], AHEAD (adaptive hierarchical decomposition) [7] 等算法. HH 算法结合层次树与直方图结构响应隐私范围查询. 用户采用 b-ary 树结构编码自身数据, 再利用 OUE (optimized unary encoding) [8] 或者 HRR (Hadamard randomized response) [9] 机制扰动编码值. 收集者利用 b-adic 技术分割每个范围查询, 利用分割后的子查询遍历 b-ary 层次树即可获得结果. 然而, HH 算法只在单表上响应一维范围查询. HI 算法是结合层次树响应 MDA 查询的典型代表, 该算法结合隐私预算分割与 OLH (optimal local Hashing) [8] 机制构建层次树. 然而, 该算法的 MDA 查询误差易受层次树树高的影响, 树越高导致每层分到的隐私预算越少, 使得每个结点计数值越不精确. 不同于 HI 算法, HIO

算法结合用户分组策略与 OLH [8] 机制实现了 MDA 查询. 该算法结合层次树树高进行用户分组, 每一层用户负责响应该层结点的计数查询. 然而, HIO 与 HI 相同, 主要用来响应谓词属性敏感与聚集属性非敏感的 MDA 查询. 为了弥补 HIO 与 HI 无法直接响应谓词属性与聚集属性都敏感情况下的 MDA 查询, AHIO 算法结合两次用户分组策略与聚集属性随机凑整技术实现了该类需求. 而响应 MDA 查询时 AHIO 需要 $d+1$ 棵层次树. 与 AHIO 相比, EHIO 算法只需要 d 棵层次树, 然而该算法需要加倍聚集属性的值域. AHEAD 算法利用树高进行用户分组, 每一层用户按照网格粒度大小进行重新划分. 尽管上述基于用户分组系列算法能够响应 MDA 查询, 然而该类算法存在以下不足.

(1) HIO, AHIO, EHIO 算法结合用户层次分组与 OLH 机制实现本地扰动, 把每层结点所在范围作为 OLH 机制的哈希地址. 由于根结点的范围只有一个, OLH 机制无法把该范围映射到其他哈希地址. 分配到根结点的真实用户个数会被泄露, 进而会造成根结点用户的隐私泄露. 例如, 以图 1(b) 中 salary 层次树为例. 假设共有 15 个用户的 salary 数据需要收集. 结合用户层次分组策略, 每层分到用户数量为 5. 分配到根结点 [1, 4] 的用户数量为 5. 根据式 (4) OLH 机制可知, 每一层的哈希地址数量为 $g=\mathrm{e}^{\epsilon}+1$. 由于根结点所处范围只有一个 [1, 4], 即是 $g=1$, 这与 OLH 机制的 $g=\mathrm{e}^{\epsilon}+1$ 相矛盾. 无论采用 OLH 机制的哪种概率扰动, 根结点 [1, 4] 均无法变成其他的哈希地址, 相当于没有扰动, 破坏了 OLH 机制. 尽管根结点中 5 个人的 salary 值被匿名到区间 [1, 4], 根据文献 [10] 可知, 攻击者可以利用一定背景知识对某个目标用户进行链接攻击, 进而导致敏感属性泄露. 此外, OLH 机制在设计层次树每层的哈希地址时, 均设定哈希地址的数量为定值 $\mathrm{e}^{\epsilon}+1$, 而实际值与层次中结点个数相关.

(2) HI 与 HIO 算法无法直接处理聚集属性为敏感的情况. HH 算法仅处理一维 MDA 查询. AHEAD 算法比较适合属性分布偏斜且稀疏的情况, 该算法构造的层次树中叶子结点数量随着维度增加呈现指数级增长 (例如, 维度为 d 时, 叶子结点数量为 2^d). 随着层次树高度的增加, 会导致每层分配的用户数量减少, 进而造成 MDA 查询精度降低. 结合文献 [4∼7] 可知, LDP 模型下响应 MDA 分析查询存在的挑战包括: 如何在不泄露根结点中用户隐私的情况下高精度地响应 MDA 查询; 如何更合理地应用层次树的索引特性设计本地扰动机制. 总而言之, 目前还没有一个行之有效且满足本地化差分隐私的 MDA 分析查询算法, 能够克服上述算法存在的挑战性问题. 为此, 本文基于 LDP 模型与层次树用户分组策略提出了一种 MDA 查询响应算法 H4MDA (hierarchical structure for MDA) 能够弥补上述算法的不足.

本文主要贡献如下.

(1) 为了解决挑战 1, 在 H4MDA 算法中提出了一种结合用户分组策略与 GRR (generalized randomized response) [11] 机制的横向本地扰动算法 HGRR (horizontal GRR). 不同于 HI, HIO, AHIO, EHIO 算法, HGRR 算法在本地扰动用户元组数据时, 通过摈弃层次树笛卡尔积 (Cartesian product) 结果中的根结点组合, 来保护根结点中用户的隐私.

(2) 为了解决挑战 2, 在 H4MDA 算法中提出了两种结合用户分组策略与 GRR 机制的纵向扰动算法 LGRR-FD (longitudinal GRR with fake data) 与 LGRR. 无需摈弃根结点组合, LGRR-FD 通过纵向扰动叶子 – 根路径上的结点组合实现本地保护. 此外, 利用随机添加固定大小的假数据防止叶子层用户隐私泄露. LGRR 算法通过摈弃叶子结点层防止叶子结点泄露用户隐私, 收集者利用 LGRR 算法的报告值重构虚拟叶子结点层来响应相应 MDA 查询.

(3) 理论分析了 HGRR, LGRR-FD 与 LGRR 算法满足 ϵ – 本地化差分隐私, 以及 MDA 分析查询估计的无偏性与均方误差. 通过合成数据与真实数据实验分析, HGRR, LGRR-FD 与 LGRR 算法具有较高的可用性.

2 相关工作

基于中心化差分隐私 (central differential privacy, CDP) 与 LDP 模型下利用层次树结构响应计数查询, 范围查询以及 MDA 查询的研究得到广泛关注. BOOST [12] 算法是 CDP 模型下利用层次树响应范围计数查询的典型代表, 该算法利用层次树索引用户数据, 并利用树高控制每层拉普拉斯噪声 (Laplace noise) [13] 的大小. Privelet (privacy-preserving wavelet) [14] 算法基于 CDP 模型利用离散哈尔 (Haar) 小波变换构建哈尔小波树, 利用拉普拉斯噪声扰动树中的小波系数, 基于噪声系数响应范围计数查询. Hb (hierarchical method with fan-out b) [15] 算法结合 CDP 模型对层次树结构进行较为系统的研究, 该算法利用层次树对用户进行层次分割, 探讨树高度、扇出、数据维度对查询误差的影响, 并利用后置处理技术对范围计数查询进行求精处理. 此外, 基于 CDP 模型, KD-Stand [16] 算法利用指数机制 [17] 构建 KD – 树索引用户数据, 再利用拉普拉斯噪声扰动结点中的计数值. PrivTree [18] 算法结合 CDP 模型利用四分树索引用户数据, 通过叶子结点噪声计数与非叶子结点索引响应范围计数查询. 该算法不通过树高控制噪声大小, 而是采用结点噪声计数的偏移值来减少噪声量.

上述算法均是在 CDP 模型下构建层次树来响应范围计数查询. LDP 技术被谷歌 [19] 与微软 [20] 公司商用后得到较为系统的研究. 基于 LDP 模型与层次树的频率查询、范围查询、MDA 分析查询成为近几年该领域的研究热点. TreeHist [21] 算法借助于计数概要与 Hadamard [9] 转换技术构建满足 LDP 的二叉前缀层次树, 遍历前缀树即可响应计数查询. 不同于 TreeHist 算法, PEM (prefix extending method) [22] 算法把用户按照前缀层次树结构进行分组, 每组用户采用 OLH 机制本地扰动自身数据后报告给收集者. 结合所有用户的报告值, 收集者重构前缀层次树. 不同于 PEM 与 TreeHist 算法, PrivTrie [23] 算法利用动态用户分组策略构建前缀层次树, 以自顶向下的方式分配用户数量, 越往下分配, 用户的数量越多. 然而, 这些基于前缀层次树的算法通常无法直接应用于 MDA 查询. 其主要原因是这些算法通常把用户数据编码成 0/1 字符串, 再利用前缀树对字符串进行索引. 而本文关注单个关系型数据表下的 MDA 查询问题. 边缘表与层次树结构是 LDP 模型下响应 MDA 查询常用的数据结构. CALM [2], LoPub [24], MargRR [3] 算法分别在收集者端构建边缘表来响应 MDA 查询. CALM 算法借鉴 PriView [25] 思想利用 GRR 与 OUE 机制创建 k-way 边缘表, 并利用后置处理技术处理噪声边缘表. 尽管 CALM 算法采用了用户分组策略响应 MDA 查询, 然而, 该算法的查询均方误差会随着属性维度 d 与属性值域 m 增加而增加. LoPub 算法允许每个用户报告所有属性值, 利用期望最大化算法重构噪声属性之间的 k-way 边缘表. 然而, 该算法采用隐私预算分割导致查询误差较大. MargRR 算法采用用户分组策略与 Hadamard 转换实现 k-way 边缘表构建, 利用 RR [26] 机制扰动 Hadamard 转换的结果. 然而, 该算法的不足与 CALM 类似.

为了弥补上述算法的不足, LDP 模型下出现了结合层次树响应 MDA 查询的算法. HH [5] 算法利用 b-ary 层次树进行用户分组, 每个用户结合 b-ary 树副本进行本地编码, 并采用 OUE 机制进行扰动. 收集者利用重构的 b-ary 树进行范围计数查询. HI [4] 算法利用隐私预算分割策略与 OLH 机制构建层次树来响应 MDA 查询, 然而, 该算法的 MDA 查询误差容易受层次树树高的影响, 树越高, 每层分到的隐私预算越少, 进而导致每个结点中计数值越不精确. 为了弥补 HI 算法不足, HIO 算法结合用户分组策略响应 MDA 查询. 然而, 该算法只关注聚集属性为非敏感的情况. 为了解决聚集属性与谓词属性同时敏感情况, AHIO [6] 与 EHIO [6] 算法分别结合两次用户分组与聚集属性凑整操作实现了 MDA 查询. 此外, AHEAD 算法同样按照树高进行用户分组, 每一层的用户再按照网格的粒度大小进行重新划分. 尽管上述算法可以响应 MDA 查询, 然而 HI 算法及其基于 HIO 系列算法均有可能泄露根结点隐私的风险, 其主要原因是根结点的哈希地址是单一的, OLH 机制无法把该地址扰动其他的哈希地址.

此外, AHEAD 算法要求属性分布满足偏斜性与且稀疏性, 然而, 该算法构造的层次树叶子结点数量随着维度增加呈现指数级增长. 基于上述分析, 提出了一种有效的 MDA 响应算法, 该算法利用用户分组策略实现了横向与纵向 GRR 扰动机制, 能够有效弥补上述算法的不足.

3 基础知识与问题描述

3.1 本地化差分隐私

LDP 模型假设用户不信任收集者, 利用随机响应机制本地扰动自身数据, 把扰动值报告给收集者. 收集者结合用户报告值响应查询. 设 T ($T \in \mathcal{D}$) 为分布式关系数据表, $T = \{t_1, t_2, \ldots, t_n\}$ 由 n 条元组构成, $t_i = (a_1, \ldots, a_d)$, d 表示维度大小, $a_1, \ldots, a_d$ 均为敏感属性 (例如位置、薪水、病史等). 本地化差分隐私的形式化定义如下所示.

定义1 (ϵ – 本地化差分隐私) 给定随机算法 $\mathcal{R}$ 及其定义域 $\mathcal{D}$ 和输出值域 $\mathcal{O}$, 若 $\mathcal{R}$ 在 $\mathcal{D}$ 中任意 2 条元组 t 与 t' ($t, t' \in \mathcal{D}$) 上得到相同输出结果 y ($y \in \mathcal{O}$) 的概率满足下列不等式, 则 $\mathcal{R}$ 满足 ϵ – 本地化差分隐私.

$$\Pr[\mathcal{R}(t) = y] \leqslant \mathrm{e}^{\epsilon} \times \Pr[\mathcal{R}(t') = y], \tag{1}$$

其中, ϵ 表示表示隐私预算, 该值越小表示本地保护程度越强.

3.2 随机应答机制

由定义 1 可知, 实现 LDP 需要本地随机扰动机制. 由 Warner [26] 提出的随机应答 RR (randomized response) 机制是实现 LDP 常用技术. RR 机制的原始思想是每个用户在响应敏感的布尔问题时, 以概率 p 真实应答, 以 q 给出相反的应答. GRR [11] 机制与 OLH [8] 机制是 RR 机制的拓展. 给定任意元组 t_i 且 $t_i \in \mathcal{D}$, GRR 机制与 OLH 机制如式 (2) 和 (4) 所示.

定义2 (GRR 本地扰动机制)

$$\forall_{y \in \mathcal{D}} \Pr[\mathrm{GRR}(t_i) = y] = \begin{cases} p = \dfrac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon} + m - 1}, & y = t_i, \\ q = \dfrac{1-p}{m-1}, & y \neq t_i, \end{cases} \tag{2}$$

其中, ϵ 表示表示隐私预算, m 表示 t_i 的值域大小.

由式 (2) 可知, $\frac{p}{q} \leqslant \mathrm{e}^{\epsilon}$. 根据定义 1 可知, GRR 机制满足 ϵ – 本地化差分隐私.

定义3 (OLH 本地扰动机制) 给定 t_i 且 $t_i \in \mathcal{D}$. LH (local Hashing) [8] 利用哈希簇 $\mathcal{H}$ 把 t_i 编码到 $\{1, 2, \ldots, g\}$ ($g \ll m$) 中某个哈希值, 即 $x = H(t_i)$, 且 $x \in \{1, 2, \ldots, g\}$, $H \in \mathcal{H}$. 根据文献 [8] 可知 LH 的表达式如式 (3) 所示, 其中 $y \in \{1, 2, \ldots, g\}$.

$$\forall_{y \in \{1,2,\ldots,g\}} \Pr[\mathrm{LH}(x) = y] = \begin{cases} p = \dfrac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon} + g - 1}, & y = x, \\ q = \dfrac{1}{g}, & y \neq x. \end{cases} \tag{3}$$

根据文献 [8] 可知, 当 $g = \mathrm{e}^{\epsilon} + 1$, $q = \frac{1}{g}$ 时, LH 机制变成了 OLH 机制, 表达式如下:

$$\forall_{y \in \{1,2,\ldots,g\}} \Pr[\mathrm{OLH}(x) = y] = \begin{cases} p = \dfrac{1}{2}, & y = x, \\ q = \dfrac{1}{\mathrm{e}^{\epsilon} + 1}, & y \neq x, \end{cases} \tag{4}$$

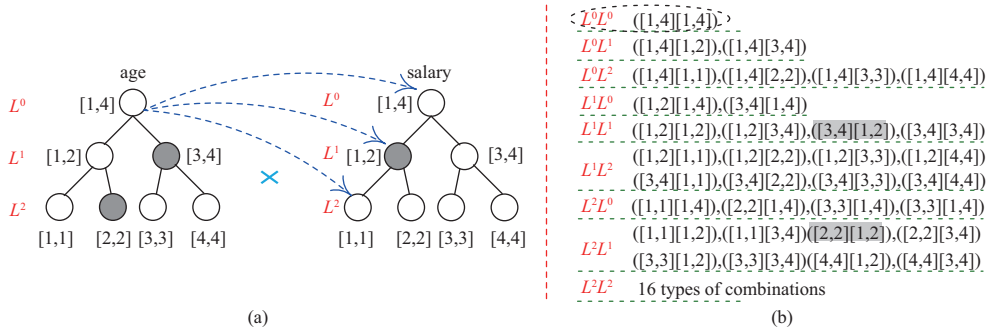

图 2 (网络版彩图) 属性 age 与 salary 对应的层次树及其组合结果

Figure 2 (Color online) Combinations and hierarchical trees of age and salary. (a) Hierarchical trees of age and salary; (b) cartesian product of hierarchical trees of age and salary

其中, ϵ 表示表示隐私预算, g 表示哈希地址值域大小.

3.3 层次树结构

设层次树 $\mathbb{T}$ 的扇出为 b, 则每个非叶子结点对应 b 个孩子结点, 每个孩子结点对应一个区间. 假设某个属性 a 的值域为 $\{v_1, v_2, \ldots, v_m\}$. 根据扇出 b 把该值域分解成不同子区间. $\mathbb{T}_a$ 的非叶子结点表示粗粒度的子区间, 叶子结点表示值域中某个取值. $\mathbb{T}_a$ 的根结点所处层次为 $L^0 = \{[v_1, v_m]\}$, 叶子结点层为 $L^h = \{[v_1, v_1], [v_2, v_2], \ldots, [v_m, v_m]\}$, 树高 $h = \log_b m$. $\mathbb{T}_a$ 每层包含 b^j $(0 \leqslant j \leqslant h)$ 个子区间, 每个子区间覆盖 m/b^j 个值. 根据图 2(a) 可知, $b = 2$, 属性 age 的值域为 [1, 2, 3, 4], $h = 2$. 则 $L^0 = \{[1, 4]\}$, $L^1 = \{[1, 2], [3, 4]\}$, $L^2 = \{[1, 1], [2, 2], [3, 3], [4, 4]\}$.

收集者根据用户的报告值重构每个属性对应的层次树结构, 然后结合重构的层次树响应 MDA 查询. 给定 MDA 查询 $Q(\mathrm{Agg}(a), C)$, $\mathrm{Agg}(a)$ 与 C 分别表示聚集函数与条件谓词, a 为聚集属性. 收集者对 Q 进行查询重写, 把 C 中的属性范围按照对应层次树重新划分. 假设 C 中谓词属性 $a_i \in [l, r]$ 的值域包含 m 个不同的值, 层次树扇出为 b. 按照 a_i 的层次树, $[l, r]$ 被分解成 $2(b-1)\log_b m$ 个子区间. 按照这些子区间, Q 可以重写成不同的子查询, 而 $Q(\mathrm{Agg}(a), C)$ 查询可以转换成层次树结点 (子区间) 中的计数查询.

例如, 给定查询 Q_4: SELECT COUNT(*) FROM T_{app} WHERE age $\in$ [2, 4] AND salary $\in$ [1, 2]. 利用 age 与 salary 的层次树分割 [2,4] 与 [1, 2]. [2, 4] 被分割成 [2, 2] 与 [3, 4], [1, 2] 被分割成 [1, 2], 如图 2(a) 灰色结点所示. age 与 salary 的层次树按照层次进行笛卡尔积, 即 $\mathbb{T}_{\mathrm{age}} \times \mathbb{T}_{\mathrm{salary}}$, 结果如图 2(b) 所示. 遍历 [2, 2] [1, 2] 与 [3, 4] [1, 2] 结点中的用户计数即可获得 Q_4 的查询结果, 如图 2(b) 中 L^1L^1 与 L^2L^1 中的灰色结点.

3.4 问题定义

设 Q 为关系表 T 上的 MDA 查询, $P(Q, T)$ 和 $P(Q, \mathcal{R}(T))$ 分别为 Q 的真实与噪声响应结果. 基于 OLH 机制的算法在响应 MDA 查询时, 无法保护其根结点里的用户真实计数, 原因是 OLH 机制无法将其扰动成其他哈希地址. 本文要解决的问题是如何设计有效的扰动机制 $\mathcal{R}$ 能够避免根结点泄露用户隐私, 查询结果又具有较高的可用性. 采用均方误差度量 MDA 查询的可用性, 如下所示:

$$\mathrm{MSE}(P(Q, \mathcal{R}(T))) = \mathbb{E}[(P(Q, \mathcal{R}(T)) - P(Q, T))^2]. \tag{5}$$

由式 (5) 可知, 若 P $(Q, \mathcal{R}(T))$ 是一个无偏结果, 则 $\mathrm{MSE}(P(Q, \mathcal{R}(T))) = \mathrm{Var}(P(Q, \mathcal{R}(T)))$.

表 1 符号及其含义

Table 1 Basic notation in this paper

Symbol	Meaning	Symbol	Meaning
n	The number of users	$P(Q,T), P(Q,\mathcal{R}(T))$	The true/noise aggregation for query Q
$t_i \in T, t_i[a]$	The i-th tuple in T, the value of a in t_i	d, d_q	The numbers of attributes in t_i and C
$\mathbb{T}_a, m, b$	The hierarchical tree of attribute a, the domain size of a, and the fan-out	L^j	Level j in hierarchical tree
$Q, \mathrm{Agg}(), C$	Aggregation query, aggregation function, and predicates	$\mathcal{R}, s$	Local perturbation mechanism, the number of fake nodes
$T \in \mathcal{D}$	The distributed table in $\mathcal{D}$	$c(g_i), c'(g_i), \tilde{c}(g_i)$	True count value, observed count value, and estimated count value

为了便于叙述文中内容, 表 1 列出了所使用的符号与它们的含义.

4 基于层次结构的 MDA 查询算法

设计基于 LDP 的 MDA 查询算法时需要考虑 2 个原则: 针对根结点隐私泄露问题, 所设计的扰动算法应避免该问题发生; 针对现有扰动算法没有合理应用层次树索引特性问题, 所设计的扰动算法尽可能利用层次树的横向与纵向特性. 针对上述原则, 提出了一种结合 LDP 与层次树结构的 MDA 查询算法 H4MDA.

4.1 H4MDA 算法

本小节介绍 H4MDA 算法的层次树构建、MDA 查询重写、子查询估计等操作, 具体细节如算法 1 所示.

算法 1 H4MDA 算法

输入: n 个用户的元组数据 t_i ($1 \leqslant i \leqslant n$), 隐私预算 ϵ, MDA 查询 $Q(\mathrm{Agg}(a), C)$, 其中 $C = \{a_1 \in [l_1, r_1] \wedge a_2 \in [l_2, r_2] \wedge \cdots \wedge a_d \in [l_d, r_d]\}$, 每个维度的扇出和值域均为 b 与 m.

输出: 满足 ϵ – 本地化差分隐私的 P(Q).

$M_{j_1,j_2,\ldots,j_d} \leftarrow \emptyset$; //$(j_1, j_2, \ldots, j_d) \in \{0, 1, \ldots, \log_b m\}^d$
收集者结合扇出参数 b 与值域参数 m 构建 d 棵层次树;
收集者把 d 棵层次树共享给 n 个用户;
for each user u_i **do**
    if 收集者收到 u_i 的报告值 $\{(j_1, j_2, \ldots, j_d), \mathcal{R}(t_i)\}$ **then**
        $M_{j_1,j_2,\ldots,j_d} = M_{j_1,j_2,\ldots,j_d} \cup \{t_i\}$; //$t_i$ 表示用户 u_i 拥有的元组
    end if
end for
for each attribute $a_i \in C$ **do**
    收集者结合 $\mathbb{T}_{a_i}$, 把属性 a_i 所在区间 $[l_i, r_i]$ 分解成 k_i 个结点; //对查询 Q 进行重写
end for
for each combination g_i of nodes in $\{1, 2, \ldots, k_1\} \times \cdots \times \{1, 2, \ldots, k_d\}$ **do**
    收集者计算 $\tilde{c}(g_i) = \mathrm{LC} \times \tilde{c}_{M_{j_1,j_2,\ldots,j_d}}(g_i)$; //LC 表示层次组合的个数
end for
return $P(Q) = \sum_{1 \leqslant i_1 \leqslant k_1, \ldots, 1 \leqslant i_d \leqslant k_d} \tilde{c}(g_i)$.

收集者根据设定的扇出 b 与属性值域 m 构建 d 棵层次空树, 并把 d 棵层次树副本共享给 n 个用户 (步骤 2 和 3); 其次收集者结合用户的报告值, 计算 d 棵树的笛卡尔积结点中计数 (步骤 4~8); 结合 d 棵树, 对 Q 进行重写 (步骤 9~11). 结合谓词的重写区间, 统计所有组合中的计数 (步骤 12~14). 由算法 1 可知, 扰动机制 $\mathcal{R}$ 是 H4MDAP 算法的核心步骤. 接下来阐述 3 种 $\mathcal{R}$ 扰动算法的具体实现细节.

4.2 基于层次树横向结构的扰动算法 HGRR

基于 OLH 机制的扰动算法 (如 HIO, AHIO, EHIO 等), 通常是按照树的层次结构横向扰动. 然而该类算法无法保护 $\mathbb{T}_1 \times \cdots \times \mathbb{T}_d$ 中根结点 $L^0L^0\cdots L^0$ 的用户隐私. 基于此, 提出了一种基于层次横向结构的扰动算法 HGRR, 该算法主要思想是摈弃根结点 $L^0L^0\cdots L^0$, 在剩余的组合中分配用户. 具体细节如算法 2 所示.

算法 2 HGRR 算法

输入: 用户 u_i 的元组数据 t_i, 隐私预算 ϵ.

输出: 满足 $\{(j_1, j_2, \ldots, j_d), y_i\}$.

1: u_i 从 d 个属性中随机抽取一个属性 a; //相对于对 n 个用户分成 d 组

2: **if** a 是 Agg(a) 中的聚集属性 **then**

3: u_i 对 $t_i[a]$ 进行凑整 $t_i[a] = \begin{cases} a_{\max}, & \text{w.p } \dfrac{t_i[a]-a_{\min}}{a_{\max}-a_{\min}}, \\ a_{\min}, & \text{w.p } \dfrac{a_{\max}-t_i[a]}{a_{\max}-a_{\min}}; \end{cases}$ //$a_{\min}$ 与 $a_{\max}$ 为 a 的最小值与最大值

4: u_i 对 a 的值域进行加倍操作, 形如 $(t_i[a] \in [2a_{\min} - a_{\max} - 1, a_{\min} - 1]) \vee (t_i[a] \in [a_{\min}, a_{\max}])$;

5: 若 $t_i[a]$ 被凑整成 $a_{\min}$, 则 u_i 把 $t_i[a]$ 映射成 $2a_{\min} - t_i[a] - 1$;

6: **end if**

7: u_i 计算 d 棵层次树的笛卡尔积 $\mathbb{T}_1 \times \cdots \times \mathbb{T}_d$, 并随机选择除根结点组合之外的任意一层次组合 $(j_1, j_2, \ldots, j_d) \in \{0, 1, \ldots, \log_b m\}^d/(0, 0, \ldots, 0)$ in $\mathbb{T}_1 \times \cdots \times \mathbb{T}_d$; //摈弃根结点

8: u_i 计算 $(j_1, j_2, \ldots, j_d)$ 层中所有结点组合形成的集合, 记为 g. 把元组 t_i 映射到某个组合 g_i 中, 且 $g_i \in g$;

9: u_i 扰动 g_i, 形如 $\Pr[\text{HGRR}(g_i) = y_i] = \begin{cases} \dfrac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon} + b^{\sum_{i=1}^{d} j_i} - 1}, & \text{if } y_i = g_i, \\ \dfrac{1}{\mathrm{e}^{\epsilon} + b^{\sum_{i=1}^{d} j_i} - 1}, & \text{if } y_i \neq g; \end{cases}$

10: return $\{(j_1, j_2, \ldots, j_d), y_i\}$.

HGRR 算法首先对 u_i 按照 d 个维度进行随机分组 (步骤 1). 若 u_i 抽取的 a 是聚集属性, 需要凑整 a 的取值 $t_i[a]$, 加倍 a 的值域, 再映射 $t_i[a]$ 值 (步骤 2~6). u_i 计算 d 个层次树副本的笛卡尔积, 基于笛卡尔积随机抽取除根结点之外的任意一层 $(j_1, j_2, \ldots, j_d)$, 计算该层次中结点的所有组合形式, 并把自身元组 t_i 映射到相应的结点中 (步骤 7 和 8). 结合 $(j_1, j_2, \ldots, j_d)$ 中的结点组合, u_i 利用 GRR 扰动自身数据 (步骤 9). 根据步骤 8 可知, HGRR 算法比较适合扇出值 b 较小的 MDA 查询场景.

例如, 根据图 2 设定 $t_i = (\text{age} = 3, \text{salary} = 2)$. u_i 结合 age 与 salary 属性的层次树副本计算层次的笛卡尔积, 如图 2(b) 中 9 种组合结果. u_i 在图 2(b) 中除去 L^0L^0 之外的 8 种层次中任意抽取一个组合 L^1L^1, 即 $j_{\text{age}} = 1$, $j_{\text{salary}} = 1$. L^1L^1 层次内结点组合为 $g = \{([1,2][1,2]), ([1,2][3,4]), ([3,4][1,2]), ([3,4]\ [3,4])\}$. u_i 把自身元组 t_3 映射到结点 $g_3 = ([3,4][1,2])$. g_3 则以概率 $\frac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon}+3}$ 扰动自身, 以概率 $\frac{1}{\mathrm{e}^{\epsilon}+3}$ 扰动成集合 g 中其他结点组合.

定理1 HGRR 算法满足 ϵ – 本地化差分隐私.

证明 设 u_i 的元组 t_i 有 d 个维度, 每个维度的值域大小为 m, 对应层次树扇出为 b. 每个维度对应一棵 b 叉树, 则有 d 棵. d 棵树的层次笛卡尔积共出现 $(\log_b m + 1)^{d-1}(\log_b 2m + 1)$ 种层次

组合, 其中 $\log_b 2m+1$ 表示聚集属性值域加倍所对应的层次数. 摈弃根结点组合后, 共有 $(\log_b m+1)^{d-1}(\log_b 2m+1)-1$ 种组合. u_i 随机从组合中抽取一层 $(j_1, j_2, \ldots, j_d)$ 且该层中结点组合集合为 g. 若元组 t_i 映射到的结点为 g_i ($g_i \in g$), 则 g_i 本地扰动成自身的概率为 $p_{g_i}=\frac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon}+b^{\sum_{i=1}^{d} j_i}-1}$, 扰动成 g 中其他结点的概率为 $q_{g_i}=\frac{1}{\mathrm{e}^{\epsilon}+b^{\sum_{i=1}^{d} j_i}-1}$. 进而 $\frac{p_{g_i}}{q_{g_i}} \leqslant \mathrm{e}^{\epsilon}$ 成立, 则 HGRR 算法满足 ϵ – 本地化差分隐私.

定理2 收集者结合所有用户的 HGRR 算法报告值重构每个结点组合的计数值. 令 $c(g_i)$ 表示结点组合 g_i 中用户个数的真实计数, $\tilde{c}(g_i)$ 表示 $c(g_i)$ 的估计值, 则 $\mathbb{E}[\tilde{c}(g_i)]=c(g_i)$ 成立.

证明 从一维与多维两个角度给出证明.

(1) 设每个元组只有一个维度, 即 $d=1$, 值域为 m, 层次树扇出为 b. 令 $c'(g_i)$ 表示 g_i 结点中计数的观测值, 则以下等式成立:

$$c'(g_i)=c(g_i)p_j+\left(\frac{n}{\log_b m}-c(g_i)\right)q_j, \tag{6}$$

其中, $p_j=\frac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon}+b^j-1}$, $q_j=\frac{1}{\mathrm{e}^{\epsilon}+b^j-1}$.

由极大似然定理可知 $c(g_i)$ 的估计值为 $\tilde{c}(g_i)=\frac{c'(g_i)-\frac{n}{\log_b m}q_j}{p_j-q_j}$.

则 $\mathbb{E}[\tilde{c}(g_i)]=\mathbb{E}\big[\frac{c'(g_i)-\frac{n}{\log_b m}q_j}{p_j-q_j}\big]=\frac{\mathbb{E}[c'(g_i)]-\frac{n}{\log_b m}q_j}{p_j-q_j}=\frac{c(g_i)p_j+(\frac{n}{\log_b m}-c(g_i))q_j-\frac{n}{\log_b m}q_j}{p_j-q_j}=c(g_i)$ 成立.

(2) 设用户元组有 $d>1$ 个维度, 每个维度的值域与扇出分别为 m 与 b. g 表示被抽取的 $(j_1, j_2, \ldots, j_d)$ 层次中结点组合的集合. 给定任意一个 g_i ($g_i \in g$), 令 $c'(g_i)$ 表示 g_i 中计数的观测值如下所示:

$$c'(g_i)=c(g_i)p_{g_i}+\left(\frac{n}{d\left((\log_b m+1)^{d-1}(\log_b 2m+1)-1\right)}-c(g_i)\right)q_{g_i}, \tag{7}$$

其中, $p_{g_i}=\frac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon}+b^{\sum_{i=1}^{d} j_i}-1}$, $q_{g_i}=\frac{1}{\mathrm{e}^{\epsilon}+b^{\sum_{i=1}^{d} j_i}-1}$.

同样结合极大似然定理可知 $c(g_i)$ 的估计值 $\tilde{c}(g_i)$ 如下所示:

$$\tilde{c}(g_i)=\frac{c'(g_i)-\frac{nq_{g_i}}{d((\log_b m+1)^{d-1}(\log_b 2m+1)-1)}}{p_{g_i}-q_{g_i}}. \tag{8}$$

与一个维度的无偏性证明类似, $\mathbb{E}[\tilde{c}(g_i)]=c(g_i)$.

定理3 设 $c(g_i)$ 表示结点组合 g_i 中用户个数的真实计数, $\tilde{c}(g_i)$ 表示 $c(g_i)$ 的估计值. 则 g_i 中的计数值由 HGRR 算法引起的方差为 $\mathrm{Var}[\tilde{c}(g_i)]=\frac{n(b^{\sum_{i=1}^{d} j_i}-2+\mathrm{e}^{\epsilon})}{d((\log_b m+1)^{d-1}(\log_b 2m+1)-1)(\mathrm{e}^{\epsilon}-1)^2}$.

证明 (1) 一维情况下, 即 $d=1$, 由式 (6) 可知 g_i 中估计值 $\tilde{c}(g_i)$ 带来的方差为

$$\mathrm{Var}[\tilde{c}(g_i)]=\mathrm{Var}\left[\frac{c'(g_i)-\frac{n}{\log_b m}q_j}{p_j-q_j}\right]=\frac{\mathrm{Var}[c'(g_i)]}{(p_j-q_j)^2}. \tag{9}$$

把式 (6) 带入式 (9) 得 $\mathrm{Var}[\tilde{c}(g_i)]=\frac{c(g_i)p_j(1-p_j)+(\frac{n}{\log_b m}-c(g_i))q_j(1-q_j)}{(p_j-q_j)^2}=\frac{\frac{n}{\log_b m}q_j(1-q_j)}{(p_j-q_j)^2}+\frac{c(g_i)(1-p_j-q_j)}{p_j-q_j}$.

把 $p_j=\frac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon}+b^j-1}$, $q_j=\frac{1}{\mathrm{e}^{\epsilon}+b^j-1}$ 带入式 (9), 则 $\mathrm{Var}[\tilde{c}(g_i)]=\frac{nq_j(1-q_j)}{\log_b m(p_j-q_j)^2}=\frac{n(b^j-2+\mathrm{e}^{\epsilon})}{\log_b m(\mathrm{e}^{\epsilon}-1)^2}$.

(2) 多维情况下, d, b 与 m 分别为维度个数、扇出与属性值域. 根据式 (7) 和 (8) 可知

$$\mathrm{Var}[\tilde{c}(g_i)]=\mathrm{Var}\left[\frac{c'(g_i)-\frac{nq_{g_i}}{d((\log_b m+1)^{d-1}(\log_b m+1)-1)}}{p_{g_i}-q_{g_i}}\right]. \tag{10}$$

把式 (7) 带入式 (10) 可知, $\mathrm{Var}[\tilde{c}(g_i)]=\frac{\mathrm{Var}[c'(g_i)]}{(p_{g_i}-q_{g_i})^2}=\frac{n(b^{\sum_{i=1}^{d} j_i}-2+\mathrm{e}^{\epsilon})}{d((\log_b m+1)^{d-1}(\log_b 2m+1)-1)(\mathrm{e}^{\epsilon}-1)^2}$.

定理 1∼3 从无偏性与方差的角度证明了 HGRR 的隐私性与可用性. 以下定理是从响应 MDA 查询的角度度量 HGRR 算法的可用性. 本文只用 COUNT 与 SUM 查询度量 HGRR 算法的误差, 利用真实数据集上的实验结果度量 HGRR 算法响应 AVG 函数查询时的优劣性.

定理4 HGRR 算法响应 COUNT 查询时, 最坏误差为

$$\mathrm{Error(COUNT)}=\frac{n(2(b-1))^{d_q}(\log_b m)^{d_q-1}(\log_b 2m)(2m^d-2+\mathrm{e}^{\epsilon})}{(\mathrm{e}^{\epsilon}-1)^2}, \tag{11}$$

其中, d_q 表示 COUNT 查询谓词条件中属性的个数.

证明 响应 COUNT 查询的最坏误差同样是从单个维度与多维度给予证明.

(1) 当 $d=1$ 且 $d_q=1$ 时, 响应 COUNT 误差为

$$\begin{aligned}\mathrm{Error(COUNT)}&=2(b-1)\log_b 2m\times\frac{n(b^j-2+\mathrm{e}^{\epsilon})}{\log_b 2m(\mathrm{e}^{\epsilon}-1)^2}\times\log_b 2m\\&=\frac{n(2(b-1)\log_b 2m)(b^j-2+\mathrm{e}^{\epsilon})}{(\mathrm{e}^{\epsilon}-1)^2}.\end{aligned} \tag{12}$$

当 $j=\log_b 2m$ 时, COUNT 查询误差最大, 即 $\mathrm{Error(COUNT)}=\frac{n(2(b-1)\log_b 2m)(2m-2+\mathrm{e}^{\epsilon})}{(\mathrm{e}^{\epsilon}-1)^2}$.

(2) 当 $d>1$ 且 $d_q>1$ 时, 当且仅当 COUNT 查询的响应结果均来自于 d 棵树叶子结点层的笛卡尔积时, 查询误差最大. 其原因是叶子结点层笛卡尔积导致 HGRR 扰动值域最大, 进而产生的误差也最大.

$$\begin{aligned}\mathrm{Error(COUNT)}=&(2(b-1)\log_b m)^{d_q-1}(2(b-1)\log_b 2m)\times\frac{n}{d((\log_b m+1)^{d-1}(\log_b 2m+1)-1)}\\&\times\frac{b^{(\sum_{i=1}^{d-1}\log_b m)\log_b 2m}-2+\mathrm{e}^{\epsilon}}{(\mathrm{e}^{\epsilon}-1)^2}\times((\log_b m+1)^{d-1}(\log_b 2m+1)-1)\times d\\=&\frac{n(2(b-1))^{d_q}(\log_b m)^{d_q-1}(\log_b 2m)(2m^d-2+\mathrm{e}^{\epsilon})}{(\mathrm{e}^{\epsilon}-1)^2}.\end{aligned} \tag{13}$$

定理5 HGRR 算法响应 SUM 查询, 最坏情况下的误差为

$$\mathrm{Error(SUM)}=\frac{n(2(b-1))^{d_q}(\log_b m)^{d_q-1}(\log_b 2m)(2m^d-2+\mathrm{e}^{\epsilon})(a_{\min}^2+a_{\max}^2)}{(\mathrm{e}^{\epsilon}-1)^2}, \tag{14}$$

其中, $a_{\min}$ 与 $a_{\max}$ 分别表示聚集属性 a 的最小值与最大值.

证明 SUM 函数是对符合谓词条件的聚合属性数值进行求和. 响应 SUM 查询的误差可由 COUNT 查询误差来估算, 该误差可分为两种极端情况.

(1) 假设满足 COUNT 查询的计数及其对应聚集属性的值均为 $a_{\min}$. 结合式 (13), $\mathrm{SUM}(a_{\min})$ 的误差为

$$\mathrm{Error(SUM}(a_{\min}))=na_{\min}^2\times\frac{(2(b-1))^{d_q}(\log_b m)^{d_q-1}(\log_b 2m)(2m^d-2+\mathrm{e}^{\epsilon})}{(\mathrm{e}^{\epsilon}-1)^2}. \tag{15}$$

(2) 假设满足 COUNT 查询的计数及其对应聚集属性的值均为 $a_{\max}$. 结合式 (13), $\mathrm{SUM}(a_{\min})$ 的误差为

$$\mathrm{Error(SUM}(a_{\max}))=na_{\max}^2\times\frac{(2(b-1))^{d_q}(\log_b m)^{d_q-1}(\log_b 2m)(2m^d-2+\mathrm{e}^{\epsilon})}{(\mathrm{e}^{\epsilon}-1)^2}. \tag{16}$$

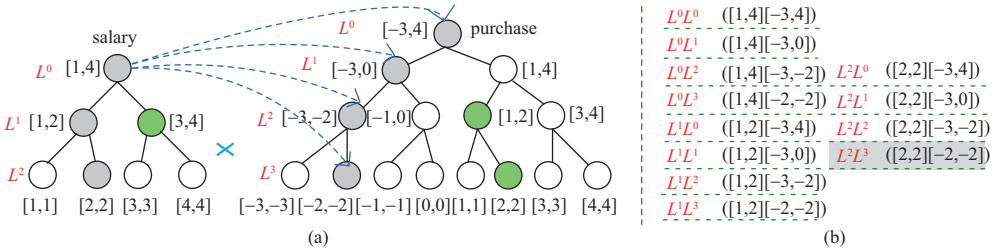

图 3　谓词属性 salary 与聚集属性 purchase 对应的层次树及其组合情况

Figure 3　Combinations and hierarchical trees of salary and purchase. (a) Hierarchical trees of salary and purchase; (b) Cartesian product of nodes of two paths

结合式 (15) 和 (16) 的平均值可知定理 5 成立.

HGRR 是结合摈弃根结点的思路设计的算法, 而大多数实际的 MDA 查询通常遍历根结点即可获得响应结果. 如果我们想利用根结点响应查询, 则需要设计新的本地扰动算法. 目前现有的 MDA 查询通常只利用了层次树的横向结构特征, 而忽视了纵向结构特征. 基于此, 本文提出了两种纵向扰动算法.

4.3　基于层次树纵向结构的扰动算法 LGRR-FD

LGRR-FD 算法细节如算法 3 所示. 类似于 HGRR 算法, LGRR-FD 的用户分组策略是把用户分配到树的不同层次. 该算法无需去掉根结点组合, 如图 3(b) 中保留了 L^0L^0 组合. LGRR-FD 算法的步骤 1∼5 与 HGRR 算法相同. 用户 u_i 把自身元组 t_i 的每个属性取值直接映射到 d 棵树的不同路径上 (步骤 2). 利用 d 个叶子节点 – 根结点的路径生成所有的结点组合. u_i 随机选择一个组合 g_i, 连同 g_i 所在的层次索引 $(j_1, j_2, \ldots, j_d)$ 一起扰动 (步骤 3∼5). 为避免叶子结点中用户隐私泄露, 在 $\mathbb{T}_1 \times \cdots \times \mathbb{T}_d$ 中随机选择 s 个假结点, 协同 g_i 的扰动结果一起发送给收集者 (步骤 6 和 7). 由步骤 5 可知, LGRR-FD 算法比较适合属性值域 m 较小的 MDA 查询场景.

算法 3 LGRR-FD 算法

输入: 用户 u_i 的元组数据 t_i, 隐私预算 ϵ, 假数据个数 s.

输出: 满足 $\{(j_1, j_2, \ldots, j_d), y_i\}$ 与 s 个假数据.

1: 与 HGRR 算法的步骤 1∼5 相同;

2: u_i 把元组 t_i 的所有属性映射到 d 条叶子结点 – 根结点的纵向路径上;

3: u_i 计算 d 条纵向路径上所有结点组合形成的集合, 记为 g;

4: u_i 随机选择一个组合 g_i ($g_i \in g$), g_i 所在的层次为 $(j_1, j_2, \ldots, j_d) \in \{0, 1, \ldots, \log_b m\}^d$; //保留根结点

5: u_i 把 g_i 与 $(j_1, j_2, \ldots, j_d)$ 扰动成 $y_i (y_i \in g)$, 形如

$$\Pr[\text{LGRR}-\text{FD}(g_i) = y_i] = \begin{cases} \dfrac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon} + |g| - 1}, & \text{if } y_i = g_i, \\ \dfrac{1}{\mathrm{e}^{\epsilon} + |g| - 1}, & \text{if } y_i \neq g_i, \end{cases}$$

where $|g| = (\log_b m + 1)^{d-1}(\log_b 2m + 1)$;

6: u_i 在 $\{\mathbb{T}_1 \times \cdots \times \mathbb{T}_d\}$ 中随机抽取 s 个假结点 $\{\{(j_1, j_2, \ldots, j_d), y_1\}, \{(j_1, j_2, \ldots, j_d), y_s\}\}$;

7: return $\{(j'_1, j'_2, \ldots, j'_d), y_i\} \cup \{\{(j_1, j_2, \ldots, j_d), y_1\}, \ldots, \{(j_1, j_2, \ldots, j_d), y_s\}\}$.

例如, 设 $t_i = (\text{salary} = 2, \text{purchase} = 3)$, salary 与 purchase 的值域均为 [1, 4]. 利用 HGRR 的步骤 1∼5 对 $t_i[\text{purchase}] = 3$ 进行凑整映射, 以及加倍 purchase 的值域成为 $[-3, 0] \vee [1, 4]$. 若 $t_i[\text{purchase}] = 3$ 被凑整为最小值 1, LGRR-FD 把 $t_i[\text{purchase}] = 3$ 转换成 $t_i[\text{purchase}] = -2$, 并把该值映射

到 $[-2,-2]$ 结点所在的路径上, 如图 3(a) 中 purchase 层次树灰色路径. $t_i[\text{salary}]=2$ 直接映射到 $[2,2]$ 叶子结点所在的路径上, 如图 3(a) 中 salary 层次树灰色路径. 求解所有纵向路径上结点的笛卡尔积, 图 3(b) 是两条灰色路径中结点的 $12(|g|=12)$ 种组合. 随机选择组合 $L^2L^3([2,2],[-2,-2])$, 如图 3(b) 所示, 该组合以 $\frac{\mathrm{e}^\epsilon}{\mathrm{e}^\epsilon+11}$ 的概率扰动成自身, 以 $\frac{1}{\mathrm{e}^\epsilon+11}$ 扰动成 g 中其他组合.

由图 3(a) 可知, 属性的具体值全部分布在叶子节点. 当所有包含叶子结点组合被 LGRR-FD 扰动后, 还是有可能泄露用户的元组数据. 例如, 尽管 $L^2L^3([2,2],[-2,-2])$ 以概率 $\frac{1}{\mathrm{e}^\epsilon+11}$ 扰动成 g 中其他值, 而收集者依然根据 $[2,2]$ 可以推测出 $t_i[\text{salary}]=2$. 为了避免此类情况, 用户随机从 $\mathbb{T}_1\times\cdots\times\mathbb{T}_d$ 结点组合中选择 s 个假结点, 与扰动值一起发送给收集者. 例如, 从 $\mathbb{T}_{\text{salary}}\times\mathbb{T}_{\text{purchase}}$ 中随机选择 2 个假结点组合, 如选择 $L^1L^2([3,4],[1,2])$ 与 $L^1L^3([3,4],[2,2])$ (如图 2(a) 中浅绿色结点组合). $\{L^2L^3([2,2],[-2,-2])$, $L^1L^2([3,4],[1,2])$, $L^1L^3([3,4],[2,2])\}$ 是收集者得到的数据. 由于收集结果中附带着假数据, 收集者无法推测出哪个是元组属性的真实值.

定理6 LGRR-FD 算法满足 ϵ – 本地化差分隐私.

证明 假设用户 u_i 的元组 t_i 所有的维度值域为 m, 层次树扇出为 b. 在 d 个维度情况下, 每一维对应一棵 b 叉树, 进而构成 d 棵 b 叉树. t_i 的所有属性值被映射到 d 个单路径中 (如图 3(a) 所示). 按 d 个路径中结点进行笛卡尔积, 则存在 $(\log_b m+1)^{d-1}(\log_b 2m+1)$ 种结点组合. 用户 u_i 随机选择一个结点组合 g_i, 则以概率 $p_{g_i}=\frac{\mathrm{e}^\epsilon}{\mathrm{e}^\epsilon+(\log_b m+1)^{d-1}(\log_b 2m+1)-1}$, $q_{g_i}=\frac{1}{\mathrm{e}^\epsilon+(\log_b m+1)^{d-1}(\log_b 2m+1)-1}$ 对 g_i 进行扰动. 进而可知, LGRR-FD 算法满足 ϵ – 本地化差分隐私. 为了避免所有包含叶子结点的组合泄露隐私, 以 $\frac{1}{\Gamma}$ 的概率生成 s 个假结点, 其中 $\Gamma=\left(\frac{b^{\log_b m+1}-1}{b-1}\right)^{d-1}\left(\frac{b^{\log_b 2m+1}-1}{b-1}\right)$.

定理7 设 $c(g_i)$ 为结点组合 g_i 中用户计数的真实值, $\tilde{c}(g_i)$ 为 $c(g_i)$ 的估计值, 则 $\mathbb{E}[\tilde{c}(g_i)]=c(g_i)$ 成立.

证明 (1) 在 $d=1$ 情况下, 当 n 个用户均使用 b 叉树映射自身元组后, 整棵树上的用户总计数为 $n+ns$, 每一层内的结点中计数之和为 $\frac{n+ns}{\log_b m+1}$. 第 j 层的结点组合 g_i 中的用户个数观察值 $c'(g_i)$ 可以表示为 $c'(g_i)=c(g_i)p_{g_i}+(\frac{n}{b^j}-c(g_i))q_{g_i}+\frac{ns}{\Gamma}$, 其中 $p_{g_i}=\frac{\mathrm{e}^\epsilon}{\mathrm{e}^\epsilon+\log_b m}$, $q_{g_i}=\frac{1}{\mathrm{e}^\epsilon+\log_b m}$, $\Gamma=\frac{b^{\log_b m+1}-1}{b-1}$. 由极大似然定理可知 $c(g_i)$ 的估计量 $\tilde{c}(g_i)$ 可表示为

$$\tilde{c}(g_i)=\frac{c'(g_i)-\frac{n}{b^j}q_{g_i}-\frac{ns}{\Gamma}}{p_{g_i}-q_{g_i}}, \tag{17}$$

结合 $c'(g_i)$ 的表达式, 由以下等式可知 $\mathbb{E}[\tilde{c}(g_i)]=c(g_i)$ 成立:

$$\begin{aligned}\mathbb{E}[\tilde{c}(g_i)]&=\mathbb{E}\left[\frac{c'(g_i)-\frac{n}{b^j}q_{g_i}-\frac{ns}{\Gamma}}{p_{g_i}-q_{g_i}}\right]=\frac{\mathbb{E}[c'(g_i)]-\frac{n}{b^j}q_{g_i}-\frac{ns}{\Gamma}}{p_{g_i}-q_{g_i}}\\&=\frac{c(g_i)p_{g_i}+(\frac{n}{b^j}-c(g_i))q_{g_i}+\frac{ns}{\Gamma}-\frac{n}{b^j}q_{g_i}-\frac{ns}{\Gamma}}{p_{g_i}-q_{g_i}}=c(g_i).\end{aligned} \tag{18}$$

(2) 在 $d>1$ 维情况下, 第 $(j_1,j_2,\ldots,j_d)$ 层的结点组合 g_i 中用户个数的观察值 $c'(g_i)$ 可以表示为

$$c'(g_i)=c(g_i)p_{g_i}+\left(\frac{n}{db^{\sum_{i=1}^d j_i}}-c(g_i)\right)q_{g_i}+\frac{ns}{d\Gamma}, \tag{19}$$

其中, $\Gamma=\left(\frac{b^{\log_b m+1}-1}{b-1}\right)^{d-1}\left(\frac{b^{\log_b 2m+1}-1}{b-1}\right)$.

结合极大似然定理和一维情况分析可得 $c(g_i)$ 的估计量 $\tilde{c}(g_i)$ 可表示为

$$\tilde{c}(g_i)=\frac{c'(g_i)-\frac{nq_{g_i}}{db^{\sum_{i=1}^d j_i}}-\frac{ns}{d\Gamma}}{p_{g_i}-q_{g_i}}, \tag{20}$$

结合式 (19) 可知 $\mathbb{E}[\tilde{c}(g_i)] = c(g_i)$ 成立.

定理8 设 $c(g_i)$ 为结点组合 g_i 中用户计数的真实值, $\tilde{c}(g_i)$ 为 $c(g_i)$ 的估计值, 则该结点中计数的估计值由 LGRR-FD 算法引起的方差为

$$\begin{aligned}\mathrm{Var}[\tilde{c}(g_i)] = &\frac{n((\log_b m+1)^{d-1}(\log_b 2m+1)-2+\mathrm{e}^{\epsilon})}{(db^{\sum_{i=1}^{d} j_i})(\mathrm{e}^{\epsilon}-1)^2} + \frac{ns((\log_b m+1)^{d-1}(\log_b 2m+1)-1+\mathrm{e}^{\epsilon})^2}{d(\mathrm{e}^{\epsilon}-1)^2}\\ &\times\left(\frac{(b^{\log_b m+1}-1)^{d-1}(b^{\log_b 2m+1}-1)(b-1)^d-(b-1)^{2d}}{(b^{\log_b m+1}-1)^{2d-2}(b^{\log_b 2m+1}-1)^2}\right).\end{aligned}$$

证明 (1) 在 $d=1$ 情况下, 根据式 (17) 可知第 j 层结点组合 g_i 中由估计值 $\tilde{c}(g_i)$ 产生的方差为

$$\begin{aligned}\mathrm{Var}[\tilde{c}(g_i)] &= \mathrm{Var}\left[\frac{c'(g_i)-\frac{n}{b^j}q_{g_i}-\frac{ns}{\Gamma}}{p_{g_i}-q_{g_i}}\right] = \frac{\mathrm{Var}[c'(g_i)]+\mathrm{Var}\left[\frac{ns}{\Gamma}\right]}{(p_{g_i}-q_{g_i})^2}\\ &= \frac{\frac{n}{b^j}q_{g_i}(1-q_{g_i})}{(p_{g_i}-q_{g_i})^2} + \frac{c(g_i)(p_{g_i}-q_{g_i})(1-p_{g_i}-q_{g_i})}{(p_{g_i}-q_{g_i})^2} + \frac{ns\frac{1}{\Gamma}(1-\frac{1}{\Gamma})}{(p_{g_i}-q_{g_i})^2}.\end{aligned} \tag{21}$$

然而, 通常情况下结点组合 g_i 中真实的用户计数值 $c(g_i)$ 比较小, 在计算 $\tilde{c}(g_i)$ 的误差时 $c(g_i)$ 可忽略不计. 结合 $p_{g_i}=\frac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon}+\log_b m}$, $q_{g_i}=\frac{1}{\mathrm{e}^{\epsilon}+\log_b m}$, $\Gamma=\frac{b^{\log_b m+1}-1}{b-1}$, 则等式 (21) 可以表示为

$$\begin{aligned}\mathrm{Var}[\tilde{c}(g_i)] &\approx \frac{n}{b^j}\times\frac{\log_b m-1+\mathrm{e}^{\epsilon}}{(\mathrm{e}^{\epsilon}-1)^2} + \frac{ns(b-1)(b^{\log_b m+1}-b)}{(b^{\log_b m+1}-1)^2}\times\frac{(\mathrm{e}^{\epsilon}+\log_b m)^2}{(\mathrm{e}^{\epsilon}-1)^2}\\ &= \frac{n(\log_b m-1+\mathrm{e}^{\epsilon})}{b^j(\mathrm{e}^{\epsilon}-1)^2} + \frac{ns(b-1)(b^{\log_b m+1}-b)(\mathrm{e}^{\epsilon}+\log_b m)^2}{(b^{\log_b m+1}-1)^2(\mathrm{e}^{\epsilon}-1)^2}.\end{aligned} \tag{22}$$

(2) 在 $d>1$ 维情况下, 根据式 (19) 和 (20) 可知估计 $\tilde{c}(g_i)$ 产生的方差为

$$\begin{aligned}\mathrm{Var}[\tilde{c}(g_i)] &= \mathrm{Var}\left[\frac{c'(g_i)-\frac{nq_{g_i}}{db^{\sum_{i=1}^{d} j_i}}-\frac{ns}{d\Gamma}}{p_{g_i}-q_{g_i}}\right] = \frac{\mathrm{Var}[c'(g_i)]+\mathrm{Var}[\frac{ns}{d\Gamma}]}{(p_{g_i}-q_{g_i})^2}\\ &= \frac{\frac{n}{db^{\sum_{i=1}^{d} j_i}}q_{g_i}(1-q_{g_i})}{(p_{g_i}-q_{g_i})^2} + \frac{c(g_i)(p_{g_i}-q_{g_i})(1-p_{g_i}-q_{g_i})}{(p_{g_i}-q_{g_i})^2} + \frac{(\frac{ns}{d})^2\Gamma(1-\frac{1}{\Gamma})}{(p_{g_i}-q_{g_i})^2}.\end{aligned} \tag{23}$$

由于 $c(g_i)$ 的实际值比较小, 在式 (23) 中忽略其影响. 根据定理 6 中 p_{g_i}, p_{g_i}, Γ 的取值, $\mathrm{Var}[\tilde{c}(g_i)]$ 可以近似表示为

$$\begin{aligned}\mathrm{Var}[\tilde{c}(g_i)] \approx &\frac{n((\log_b m+1)^{d-1}(\log_b 2m+1)-2+\mathrm{e}^{\epsilon})}{(db^{\sum_{i=1}^{d} j_i})(\mathrm{e}^{\epsilon}-1)^2} + \frac{((\log_b m+1)^{d-1}(\log_b 2m+1)-1+\mathrm{e}^{\epsilon})^2}{d(\mathrm{e}^{\epsilon}-1)^2}\\ &\times\left(\frac{(b^{\log_b m+1}-1)^{d-1}(b^{\log_b 2m+1}-1)(b-1)^d-(b-1)^{2d}}{(b^{\log_b m+1}-1)^{2d-2}(b^{\log_b 2m+1}-1)^2}\right).\end{aligned} \tag{24}$$

与 HGRR 算法类似, 上述定理 7 和 8 阐述了由 LGRR-FD 算法产生计数的无偏性与方差. 以下从响应 MDA 查询 (COUNT, SUM) 的角度度量 LGRR-FD 算法的可用性.

定理9 LGRR-FD 算法响应 COUNT 查询时, 最坏情况下的误差为

$$\begin{aligned}\mathrm{Error}(\mathrm{COUNT}) = &\left[\frac{n((\log_b m+1)^{d-1}(\log_b 2m+1)-2+\mathrm{e}^{\epsilon})}{(\mathrm{e}^{\epsilon}-1)^2 b^{\sum_{i=d_q+1}^{d} j_i}} + \frac{ns((\log_b m+1)^{d-1}(\log_b 2m+1)-1+\mathrm{e}^{\epsilon})^2}{(\mathrm{e}^{\epsilon}-1)^2}\right.\\ &\left.\times\left(\frac{(b^{\log_b m+1}-1)^{d-1}(b^{\log_b 2m+1}-1)(b-1)^d-(b-1)^{2d}}{(b^{\log_b m+1}-1)^{2d-2}(b^{\log_b 2m+1}-1)^2}\right)\right]\\ &\times(\log_b m+1)^{d-1}(\log_b 2m+1),\end{aligned}$$

其中, d_q 表示 COUNT 查询谓词条件中属性的个数.

证明 类似于 HGRR 的无偏性与方差, LGRR-FD 算法响应 COUNT 查询的最坏误差同样是从单个维度与多维度给予证明.

(1) 当 $d=1$ 且 $d_q=1$ 时, 当且仅当 $j=0$ 时, 响应 COUNT 查询的误差最大, 即

$$\begin{aligned}\mathrm{Error(COUNT)}=&\left(\frac{n(\log_b 2m-1+\mathrm{e}^{\epsilon})}{b^j(\mathrm{e}^{\epsilon}-1)^2}+\frac{ns(b-1)(b^{\log_b 2m+1}-b)(\mathrm{e}^{\epsilon}+\log_b 2m)^2}{(b^{\log_b 2m+1}-1)^2(\mathrm{e}^{\epsilon}-1)^2}\right)\\&\times(\log_b 2m+1).\end{aligned}\tag{25}$$

(2) 当 $d>1$ 且 $d_q>1$ 时, 响应 COUNT 查询的最坏误差来自于 LGRR-FD 算法的扰动误差与随机值误差之和. 当 $(j_1,j_2,\ldots,j_d)=\{0\}^d$ 时, 在所有层次组合的情况中, d_q 个属性均选择第 L^0 层进行响应 COUNT 查询时误差最大. 其中, 扰动误差如下所示:

$$\frac{n((\log_b m+1)^{d-1}(\log_b 2m+1)-2+\mathrm{e}^{\epsilon})}{(db^{\sum_{i=1}^{d_q}0}b^{\sum_{d_q+1}^{d}j_i})(\mathrm{e}^{\epsilon}-1)^2}\times(\log_b m+1)^{d-1}(\log_b 2m+1)\times d.\tag{26}$$

随机值误差如下所示:

$$\begin{aligned}&\left[\frac{ns((\log_b m+1)^{d-1}(\log_b 2m+1)-1+\mathrm{e}^{\epsilon})}{d(\mathrm{e}^{\epsilon}-1)^2}\times\frac{(b^{\log_b m+1}-1)^{d-1}(b^{\log_b 2m+1}-1)(b-1)^d-(b-1)^{2d}}{(b^{\log_b m+1}-1)^{2d-2}(b^{\log_b 2m+1}-1)^2}\right]\\&\times(\log_b m+1)^{d-1}(\log_b 2m+1)\times d.\end{aligned}\tag{27}$$

由式 (26) 与 (27) 之和可知定理 9 成立

定理10 结合定理 9 可知 LGRR-FD 算法响应 SUM 查询的最坏误差为

$$\mathrm{Error(SUM)}=\mathrm{Error(COUNT)}\times\left(\frac{a_{\min}^2+a_{\max}^2}{2}\right),\tag{28}$$

其中, $a_{\min}$ 与 $a_{\max}$ 分别表示聚集属性 a 的最小值与最大值.

证明 与 HGRR 算法证明方式类似, 此处省去证明细节.

LGRR-FD 是结合层次树的纵向结构与添加随机假结点设计的算法. 添加随机假数据主要是为了防止叶子结点泄露某些属性的真实值. 而添加随机噪声不可避免地会产生误差. 为了减少添加假数据带来的误差, 提高 MDA 查询的响应精度同时又能避免叶子结点泄露隐私, 提出了一种有效本地扰动算法 LGRR. 该算法主要思想是在计算路径结点笛卡尔积时, 摈弃掉每个属性对应树的叶子结点层.

4.4 基于层次树纵向结构的扰动算法 LGRR

LGRR 算法细节如算法 4 所示. LGRR 算法同样利用层次树的纵向结构扰动用户元组属性. 与 LGRR-FD 不同的是 LGRR 在映射用户元组时, 摈弃了路径上的叶子结点 (步骤 2∼4). 纵向扰动方式与 LGRR-FD 算法思路一致 (步骤 5).

例如, 设 u_i 的元组 $t_i=(\mathrm{salary}=2,\mathrm{purchase}=3)$. 与 LGRR-FD 相同, 若 $t_i[\mathrm{purchase}]=3$ 被凑整为最小值 1, 则 LGRR 把 $t_i[\mathrm{purchase}]=3$ 映射到 $[-3,4]\to[-3,0]\to[-3,-2]$ 所在的路径上, 如图 4(a) 中 purchase 层次树灰色路径. $t_i[\mathrm{salary}]=2$ 映射到 $[1,4]\to[1,2]$ 所在的路径上, 如图 4(a) 中 salary 层次树灰色路径. 灰色路径上的结点进行笛卡尔积, 结果如图 4(b) 所示, 共 $|g|=6$ 种组合. 随机选择一个组合 $L^1L^0([1,2],[-3,4])$, 如图 4(b) 中灰色部分. 则该组合以 $\frac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon}+5}$ 的概率扰动成自身, 以 $\frac{1}{\mathrm{e}^{\epsilon}+5}$ 的概率扰动成其他 5 种组合中的任意一种.

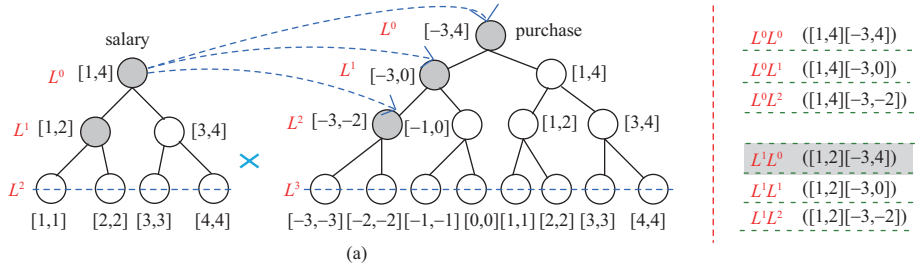

图 4 (网络版彩图) 谓词属性 salary 与聚集属性 purchase 对应的无叶结点层次树及其组合情况

Figure 4 (Color online) Combinations and hierarchical trees of salary and purchase. (a) Hierarchical trees of salary and purchase (without leaf node); (b) Cartesian product of nodes of two paths

算法 4 LGRR 算法

输入: 用户 u_i 的元组数据 t_i, 隐私预算 ϵ.

输出: 满足 $\{(j_1, j_2, \ldots, j_d), y_i\}$.

1: 与 HGRR 算法的步骤 1∼5 相同;

2: u_i 映射 t_i 属性值到 d 条无叶结点的纵向路径上;

3: u_i 计算 d 条纵向路径上所有结点的组合, 记为 g;

4: u_i 随机选择一个组合 g_i ($g_i \in g$), g_i 所在的层次为 $(j_1, j_2, \ldots, j_d) \in \{0, 1, \ldots, \log_b m - 1\}^d$; //摈弃叶结点

5: u_i 把 g_i 与 $(j_1, j_2, \ldots, j_d)$ 扰动成 y_i ($y_i \in g$), 形如

$$\Pr[\mathrm{LGRR}(g_i) = y_i] = \begin{cases} \dfrac{\mathrm{e}^{\epsilon}}{\mathrm{e}^{\epsilon} + |g| - 1}, & \text{if } y_i = g_i, \\ \dfrac{1}{\mathrm{e}^{\epsilon} + |g| - 1}, & \text{if } y_i \neq g_i, \end{cases}$$

where $|g| = (\log_b m + 1)^{d-1}(\log_b 2m)$;

6: return $\{(j_1, j_2, \ldots, j_d), y_i\}$.

为了提升 H4MDA 算法第 10 行的计算精度, 收集者获得 LGRR 算法的报告值后, 采用局部一致性处理策略来重构虚拟叶子结点. 其目的是能够响应那些需要遍历到叶子结点的 MDA 查询. 具体思路是利用父结点计数重构虚拟叶子结点计数, 设 $\tilde{c}(\cdot)$ 是某父结点噪声计数, 则每个叶子结点的计数为 $\frac{\tilde{c}(\cdot)}{b}$, b 为扇出. LGRR 算法的隐私性, 层次树中结点组合计数的无偏性以及产生的方差与 LGRR-FD 算法类似, 因此, 此部分只给出定理, 省去证明过程.

定理11 LGRR 算法满足 ϵ – 本地化差分隐私.

定理12 经过 LGRR 算法扰动之后产生的组合结点 g_i, 设 $c(g_i)$ 表示结点组合 g_i 中用户个数的真实计数, $\tilde{c}(g_i)$ 表示 $c(g_i)$ 的估计值, 则 $\mathbb{E}[\tilde{c}(g_i)] = c(g_i)$ 成立.

定理13 设 $c(g_i)$ 为结点组合 g_i 中用户个数的真实计数, $\tilde{c}(g_i)$ 为 $c(g_i)$ 的估计值, 则该结点由 LGRR 算法引起的方差为

$$\mathrm{Var}[\tilde{c}(g_i)] = \frac{n}{db^{\sum_{i=1}^{d} j_i}} \frac{(\log_b m)^{d-1}(\log_b 2m) - 2 + \mathrm{e}^{\epsilon}}{(\mathrm{e}^{\epsilon} - 1)^2}. \tag{29}$$

定理14 LGRR 算法响应 COUNT 查询时, 最坏情况下的误差为

$$\mathrm{Error}(\mathrm{COUNT}) = \frac{n((\log_b m)^{d-1}(\log_b 2m) - 2 + \mathrm{e}^{\epsilon}) \times d((\log_b m)^{d-1}(\log_b 2m))}{db^{\sum_{i=d_q+1}^{d} j_i}(\mathrm{e}^{\epsilon} - 1)^2}. \tag{30}$$

表 2　响应 SUM 查询误差描述

Table 2　Error description for sum query

Algorithm	Query error for SUM (setting $d_q = d$ in the worst case)
AHIO	$O\left(\frac{n2d(\log m)^{2d}(a_{\min}^2+a_{\max}^2)}{\epsilon^2}\right)$
EHIO	$O\left(\frac{nd(\log m)^{2d-2}(\log 2m)^2(a_{\min}^2+a_{\max}^2)}{\epsilon^2}\right)$
HGRR	$O\left(\frac{n(\log m)^{d-1}(\log 2m)2m^d(a_{\min}^2+a_{\max}^2)}{\epsilon^2}\right)$
LGRR-FD	$O\left(\left(\frac{n(\log m)^{2d-2}(\log 2m)^2}{\epsilon^2}+\frac{ns(\log m)^{3d-3}(\log 2m)^3}{\epsilon^2 m^d}\right)(a_{\min}^2+a_{\max}^2)\right)$
LGRR	$O\left(\frac{n(\log m)^{2d-2}(\log 2m)^2(a_{\min}^2+a_{\max}^2)}{\epsilon^2}\right)$

定理15　LGRR 算法响应 SUM 查询时, 最坏情况下的误差为

$$\mathrm{Error(SUM)}=\frac{n((\log_b m)^{d-1}(\log_b 2m)-2+\mathrm{e}^{\epsilon})\times d((\log_b m)^{d-1}(\log_b 2m))}{db^{\sum_{i=d_q+1}^{d}j_i}(\mathrm{e}^{\epsilon}-1)^2}\left(\frac{a_{\min}^2+a_{\max}^2}{2}\right). \tag{31}$$

4.5　基于层次树扰动算法对比分析

EHIO 与 AHIO 算法与本文解决的问题最为贴近, 本小节以响应 SUM 查询产生的误差为基础来分析 EHIO, AHIO, HGRR, LGRR-FD 以及 LGRR 算法优劣性. 表 2 是对上述算法进行 O 放缩的结果. 根据表 2 可知, HGRR 算法响应 SUM 查询的误差是 AHIO 算法的 $\frac{m^d\log 2m}{d(\log m)^{d+1}}$ 倍, 是 EHIO 算法的 $\frac{2m^d}{d(\log m)^{d-1}\log 2m}$ 倍. 由于在参数 m 与 d 的值域改变取值, $\frac{m^d\log 2m}{d(\log m)^{d+1}}$ 与 $\frac{2m^d}{d(\log m)^{d-1}\log 2m}$ 均大于 1, 则 HGRR 算法的 SUM 查询误差大于 AHIO 与 EHIO 产生的 SUM 查询误差. LGRR-FD 算法的 SUM 查询误差是 AHIO 算法的 $\frac{(\log 2m)^2}{2d(\log m)^2}+\frac{s(\log m)^{3d-3}(\log 2m)^3}{2d(\log m)^{2d}m^d}$ 倍, 是 EHIO 算法的 $\frac{(m^d+s(\log m)^{d-1})}{dm^d}$ 倍. 由于这两种倍数关系小于 1, 则 AHIO 与 EHIO 算法的 SUM 查询误差高于 LGRR-FD 算法. LGRR 的 SUM 查询误差是 AHIO 的 $\frac{(\log 2m)^2}{2d(\log m)^2}$ 倍, 是 EHIO 的 $\frac{1}{d}$ 倍. 由于这两种倍数关系均小于 1, LGRR 响应 SUM 查询的误差低于 AHIO 与 EHIO. HGRR 响应 SUM 查询的误差是 LGRR-FD 的 $\frac{2m^{2d}}{(\log m)^{d-1}(\log 2m)(m^d+s(\log m)^{d-1}(\log 2m))}$ 倍, 是 LGRR 的 $\frac{2m^d}{(\log m)^{d-1}(\log 2m)}$. LGRR-FD 响应 SUM 查询误差是 LGRR 的 $\frac{(m^d+(\log m)^{d-1}(\log 2m))}{m^d}$ 倍. 由于 3 种倍数关系均大于 1, 进而可知 LGRR 的响应 SUM 查询误差均低于 HGRR 与 LGRR-FD 的查询误差.

5　实验结果与分析

实验平台是 4 核 Intel CPU(4 GHz), 8G 内存, Win7 系统. 代码采用 Python 实现. 采用 SYN, IPUMS-P 和 Adult 作为实验数据集. SYN 是合成数据集, 包含 4 个顺序属性和 1 个非敏感属性. Adult 来源于 UCL ML 库, 共 45222 条信息, 包含 Education-Num, Hours-Per-Wee, Age 三种顺序属性. IPUMS-P 是美国人口普查数据集, 约 330 万条数据, 包含 Age, Uhrswork 和 Wkswork1 三种顺序属性. 信息如表 3 所示.

结合上述数据集, 采用归一化均方误差 (normalized mean squared error, NMSE) 来度量 HIO, AHIO, EHIO, AHEAD, HGRR, LGRR-FD, LGRR 算法响应包含 COUNT 与 SUM 两种聚集函数的 MDA 查询精度. 均方误差 NMSE 如下所示:

$$\mathrm{NMSE}(P(\mathcal{Q}))=\frac{1}{|\mathcal{Q}|}\sum_{Q\in\mathcal{Q}}\left(\frac{P(Q,\mathcal{R}(T))-P(Q,T)}{\sum_T}\right), \tag{32}$$

表 3 6 种数据集信息描述

Table 3 Description of six datasets

Name	Distribution	Number of user	Dimension	Size of domain
SYN	Normal	2000000	5	25∼625
Adult	–	45222	15	125
IPUMS-P	–	3239553	16	125

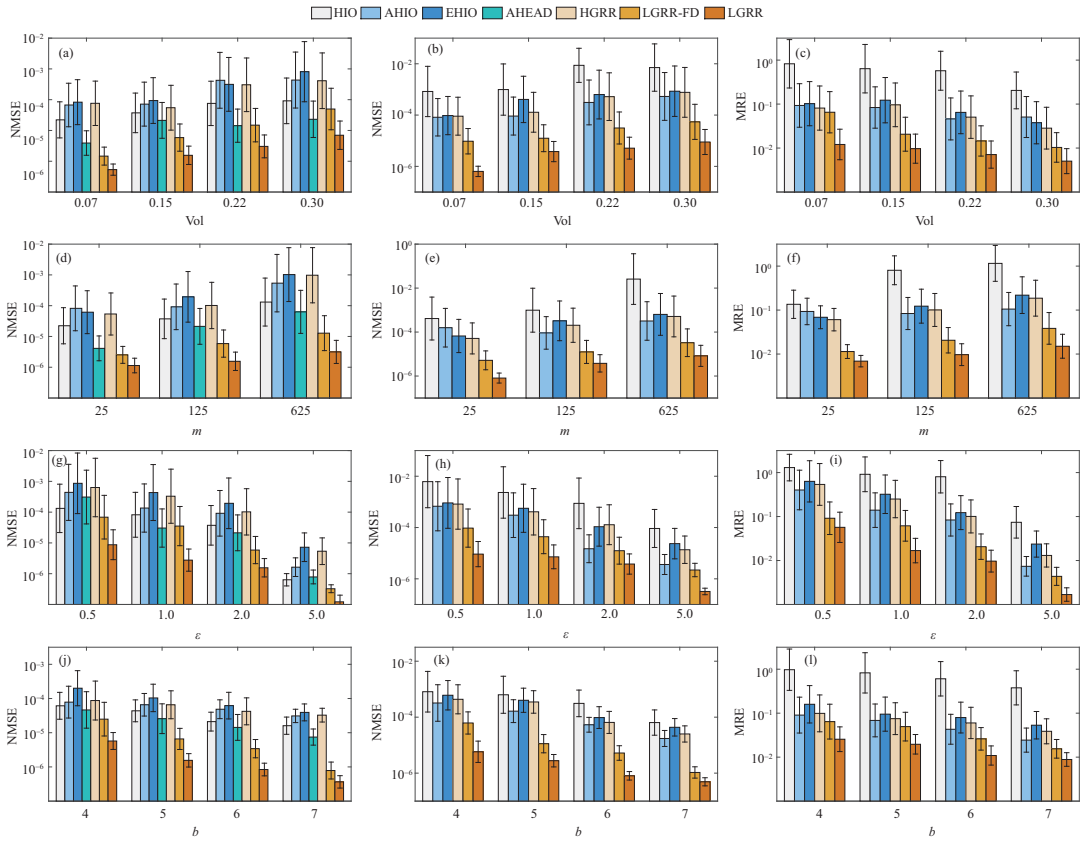

图 5 基于 SYN 数据集算法比较

Figure 5 Comparison of algorithms based on SYN. (a, d, g, j) COUNT; (b, e, h, k) SUM; (c, f, i, l) AVG

其中, $\mathcal{Q}$ 表示所有基于表 T 的 MDA 查询的集合, 对于 COUNT 查询, $\sum_T = |T|$, 对于 SUM 查询, $\sum_T = \sum |t[a]|$.

采用平均相对误差 (mean relative error, MRE) 度量 HIO, AHIO, EHIO, HGRR, LGRR, LGRR-FD 算法响应聚集函数为 AVG 的 MDA 查询精度. 平均相对误差 MRE 如下所示:

$$\mathrm{MRE}(P(\mathcal{Q})) = \frac{1}{|\mathcal{Q}|}\sum_{Q\in\mathcal{Q}}\left|\frac{P(Q,\mathcal{R}(T)) - P(Q,T)}{P(Q,T)}\right|, \tag{33}$$

其中, $P(Q,T)$ 和 $P(Q,\mathcal{R}(T))$ 分别为 Q 查询的真实与噪声响应结果.

设置 LGRR-FD 算法中随机添加假数据的数量 $s = 1$. 隐私预算 ϵ 的选取分别为 0.5, 1.0, 2.0, 5.0. 聚集属性值域的选取分别为 25, 125, 625. 谓词查询范围在整个值域的占比参数 vol 的选取分别

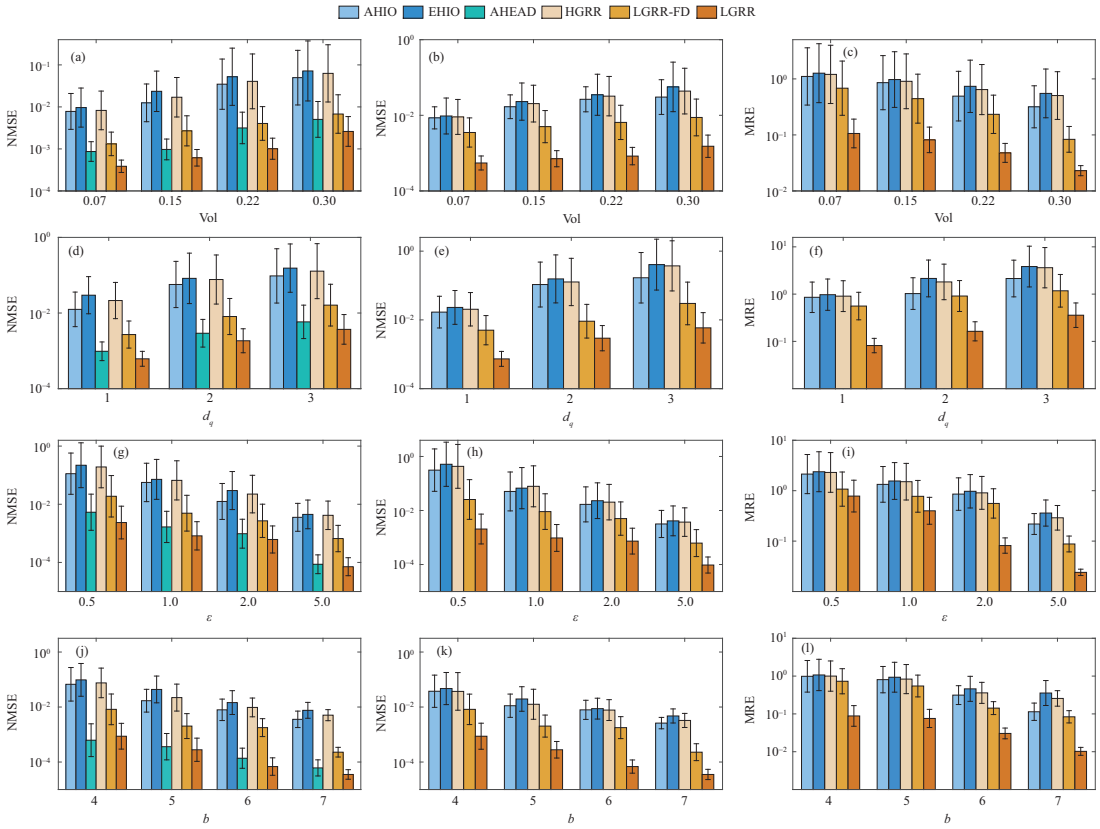

图 6　基于 Adult 数据集算法比较

Figure 6　Comparison of algorithms based on Adult. (a, d, g, j) COUNT; (b, e, h, k) SUM; (c, f, i, l) AVG

为 0.07, 0.15, 0.22, 0.3. 层次树的扇出参数 b 选值分别为 4, 5, 6, 7. 谓词属性的个数参数 d_q 的选值分别为 1, 2, 3.

5.1　基于 SYN 数据集的算法对比分析

图 5 描述了上述 7 种算法在 SYN 合成数据集上 NMSE 值与 MRE 值的比较结果. 图 5(a)∼(c) 描述了参数 vol 对 MDA 查询结果的影响. 设定 $\epsilon = 2.0$, $d = 2$, $m = 125$, $d_q = 1$. 为了与文献 [4,6] 中的层次树扇出设置一致, 设置 $b = 5$. 当 vol 从 0.07 变化到 0.3 时, 所有算法响应 COUNT 与 SUM 查询的 NMSE 值均呈现增强趋势, 而响应 AVG 查询的 MSE 呈现下降趋势, 其原因是 vol 增加导致覆盖的结点数增加, 进而导致噪声增加. 相比于 HIO, AHIO, EHIO 算法, HGRR 算法的精度略低, 其原因是 HGRR 算法没有采用哈希技术转换属性值域. 然而, HGRR 却能够避免泄露根结点隐私. 由于假数据原因, LGRR-FD 在 vol = 0.22 时误差略大于 AHEAD 算法. 其他情况下 LGRR-FD 与 LGRR 算法均优于其他算法. 图 5(d)∼(f) 描述了参数 m 变化对 MDA 查询的影响. 当固定 $\epsilon = 2.0$, $d = 2$, $b = 5$, vol = 0.15, $d_q = 1$, m 从 25 变化到 625 时, 所有算法响应 COUNT, SUM 和 AVG 查询的 NMSE 与 MSE 呈现增加趋势, 其原因是值域过大导致层次树高度增加, 分配到每层的用户个数减少, 进而导致响应精度降低. 相比于其他 5 种本地扰动方法, LGRR-FD 和 LGRR 算法的 NMSE 与 MSE 值较小, 其原因是这两种算法利用纵向路径的组合作为扰动值域. 图 5(g)∼(i) 描述了 ϵ 变化对 MDA 查询的影响. 当 $d = 2$, vol = 0.15, $b = 5$, $m = 125$, $d_q = 1$, ϵ 从 0.5 变化到 5.0 时, 所有算法的 NMSE 和 MSE

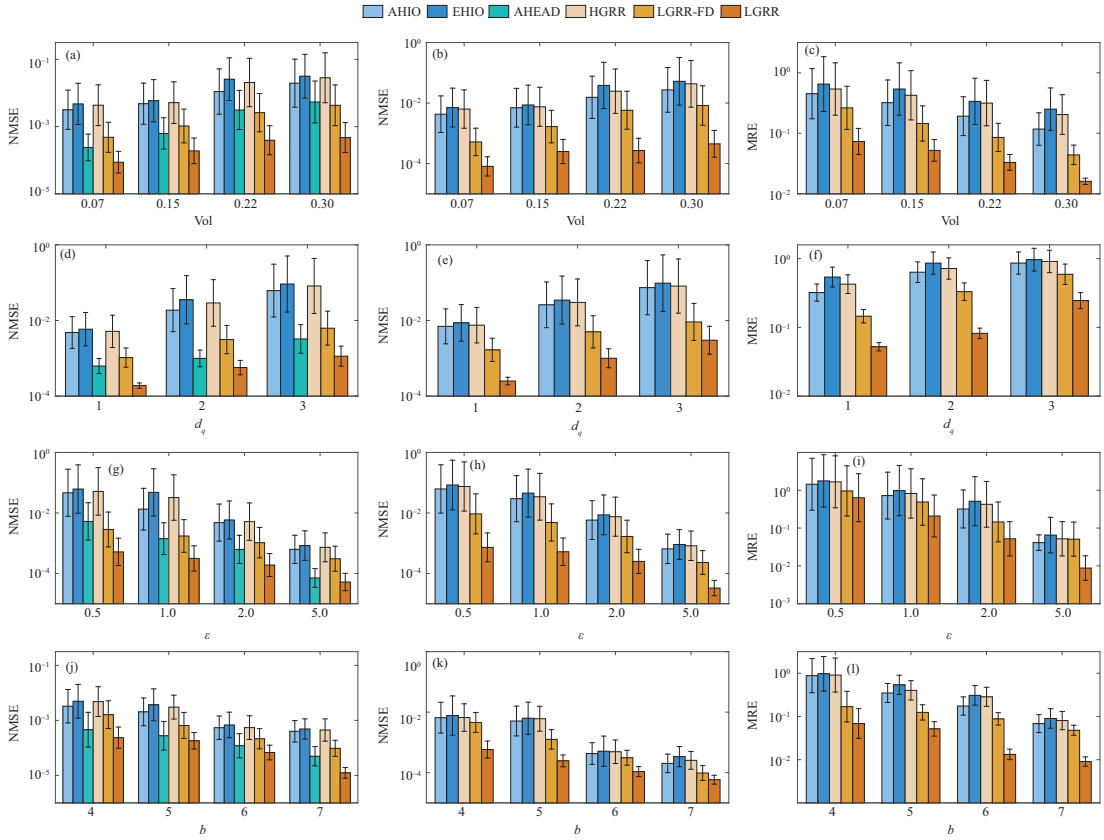

图 7 基于 IPUMS-P 数据集算法比较

Figure 7 Comparison of algorithms based on IPUMS-P. (a, d, g, j) COUNT; (b, e, h, k) SUM; (c, f, i, l) AVG

均呈下降趋势, 其原因是噪声的多少与 ϵ 成反比, ϵ 越大, NMSE 和 MSE 越小. 此种情况下 LGRR-FD 和 LGRR 算法优于其他几种算法. 图 5(j)∼(l) 描述了扇出 b 对 MDA 查询的影响. 当 $d = 2$, vol $= 0.15$, $m = 125$, $d_q = 1$, b 从 4 变化到 7 时, 所有算法响应 COUNT, SUM 和 AVG 查询的误差呈现下降趋势. 其主要原因是随着扇出的增加, 层次树高度降低, 进而导致分配到每个层次的用户数量增加. 每层用户数量增加会直接减少 NMSE 和 MSE.

5.2 基于 Adult 与 IPUMS-P 数据集算法对比分析

图 6 和 7 描述了上述算法在真实数据集 Adult 与 IPUMS-P 上的 NMSE 值和 MRE 值的比较结果. 从实验结果可知, 所有算法在 IPUMS-P 上的 MDA 查询精度均高于 Adult, 其原因是 IPUMS-P 的元组数量远大于 Adult 数据量, 进而分配到层次树每层的用户数量远高于 Adult. 从图 6(a), (d), (g), (j) 与图 7(a), (d), (g), (j) 中的 COUNT 查询可知, AHEAD 算法的 NMSE 值低于 HGRR 和 LGRR-FD 算法, 其原因是 Adult 和 IPUMS-P 比较稀疏, AHEAD 算法比较适应于此类情况. 而 LGRR 算法优于 AHEAD 算法, 其原因是 LGRR 算法摈弃了叶子结点, 并利用局部一致性处理虚构了叶子结点. 从图 6(b), (e), (h), (k), (c), (f), (i), (l) 与图 7(b), (e), (h), (k), (c), (f), (i), (l) 中的 SUM 和 AVG 查询可知, 在 vol, d_q, ϵ, b 变化下, LGRR-FD 与 LGRR 均优于 AHIO 与 EHIO. 当 vol $= 0.07$, $d_q = 2$, $b = 5$, $\epsilon = 5.0$ 时, LGRR-FD 与 LGRR 算法在 Adult 上的 SUM 和 AVG 查询精度比 AHIO 与 EHIO 算法平均高 2 个量级, 在 IPUMS-P 上的查询精度比 AHIO 与 EHIO 平均高 1 个量级, 其原因是 LGRR-FD

与 LGRR 算法合理地应用了层次树的纵向结构以及局部一致性处理技术.

6 总结

本文针对本地化差分隐私下多维分析查询问题, 结合现有本地扰动机制的不足, 提出了 3 种结合用户分组策略的本地扰动算法 HGRR, LGRR-FD 与 LGRR. HGRR 算法利用层次树结构的横向特征与 GRR 机制本地扰动用户元组, LGRR-FD 与 LGRR 算法利用层次树的纵向特征扰动用户元组. 为了避免层次树的叶子结点泄露用户隐私, LGRR-FD 通过添加假数据的方式避免此类情况发生, 而 LGRR 通过摈弃叶子结点层实现这一目的. 通过 3 种数据与现有 4 种算法进行均方差与相对误差对比分析, 实验结果表明, HGRR, LGRR-FD 与 LGRR 算法具有较好的可用性. 今后的工作考虑如下两个方面: (1) 如何在纵向扰动框架实现多关系表的连接查询; (2) 如何在纵向扰动框架实现多关系表的 OLAP 分析.

参考文献

1 Duchi J C, Jordan M I, Wainwright M J. Local privacy and statistical minimax rates. In: Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 2013. 429–438

2 Zhang Z K, Wang T H, Li N H, et al. CALM: consistent adaptive local marginal for marginal release under local differential privacy. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS), 2018. 212–229

3 Cormode G, Kulkarni T, Srivastava D. Marginal release under local differential privacy. In: Proceedings of the International Conference on Management of Data (SIGMOD), 2018. 131–146

4 Wang T H, Ding B L, Zhou J R, et al. Answering multi-dimensional analytical queries under local differential privacy. In: Proceedings of the International Conference on Management of Data (SIGMOD), 2019. 159–176

5 Cormode G, Kulkarni T, Srivastava D. Answering range queries under local differential privacy. Proc VLDB Endow, 2019, 12: 1126–1138

6 Xu M, Ding B, Wang T, et al. Collecting and analyzing data jointly from multiple services under local differential privacy. Proc VLDB Endow, 2020, 13: 2760–2772

7 Du L K, Zhang Z K, Bai S J, et al. AHEAD: adaptive hierarchical decomposition for range query under local differential privacy. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS), 2021. 1266–1288

8 Wang T H, Blocki J, Li N H, et al. Locally differentially private protocols for frequency estimation. In: Proceedings of the 26th USENIX Security Symposium (USENIX), 2017. 729–745

9 Acharya J, Sun Z T, Zhang H Y. Hadamard response: estimating distributions privately, efficiently, and with little communication. In: Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS), 2019. 1120–1129

10 Machanavajjhala A, Gehrke J, Kifer D, et al. L-diversity: privacy beyond k-anonymity. In: Proceedings of the 22nd International Conference on Data Engineering (ICDE), 2006

11 Kairouz P, Bonawitz K A, Ramage D. Discrete distribution estimation under local privacy. In: Proceedings of the 33rd International Conference on Machine Learning (ICML), 2016. 2436–2444

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

13 Dwork C, McSherry F, Nissim K, et al. Calibrating noise to sensitivity in private data analysis. J Priv Confidentiality, 2016, 7: 17–51

14 Xiao X, Wang G, Gehrke J. Differential privacy via wavelet transforms. IEEE Trans Knowl Data Eng, 2011, 23: 1200–1214

15 Qardaji W, Yang W, Li N. Understanding hierarchical methods for differentially private histograms. Proc VLDB Endow, 2013, 6: 1954–1965

16 Cormode G, Procopiuc C M, Srivastava D, et al. Differentially private spatial decompositions. In: Proceedings of the 28th International Conference on Data Engineering (ICDE), 2012. 20–31

17 McSherry F, Talwar K. Mechanism design via differential privacy. In: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 2007. 94–103
18 Zhang J, Xiao X K, Xie X. PrivTree: a differentially private algorithm for hierarchical decompositions. In: Proceedings of the International Conference on Management of Data (SIGMOD), 2016. 155–170
19 Erlingsson U, Pihur V, Korolova A. RAPPOR: randomized aggregatable privacy-preserving ordinal response. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS), 2014. 1054–1067
20 Ding B L, Kulkarni J, Yekhanin S. Collecting telemetry data privately. In: Proceedings of the Annual Conference on Neural Information Processing Systems 2017 (NIPS), 2017. 3571–3580
21 Bassily R, Nissim K, Stemmer U, et al. Practical locally private heavy hitters. In: Proceedings of the Annual Conference on Neural Information Processing Systems 2017 (NIPS), 2017. 2288–2296
22 Wang T, Li N, Jha S. Locally differentially private heavy hitter identification. IEEE Trans Depend Secure Comput, 2021, 18: 982–993
23 Wang N, Xiao X K, Yang Y, et al. PrivTrie: effective frequent term discovery under local differential privacy. In: Proceedings of the 34th International Conference on Data Engineering (ICDE), 2018. 821–832
24 Ren X B, Yu C M, Yu W R, et al. LoPub: high-dimensional crowdsourced data publication with local differential privacy. IEEE Trans Inform Forensic Secur, 2018, 13: 2151–2166
25 Qardaji W H, Yang W N, Li N H. PriView: practical differentially private release of marginal contingency tables. In: Proceedings of the International Conference on Management of Data (SIGMOD), 2014. 1435–1446
26 Warner S L. Randomized response: a survey technique for eliminating evasive answer bias. J Am Stat Assoc, 1965, 60: 63–69

Answering private multidimensional analytical queries with hierarchical structure

Xiaojian ZHANG[1*], Dan ZHOU[1], Yaxin XU[1], Dongdai LIN[2], Shouling JI[3] & Xiaofeng MENG[4]

1. *School of Computer & Information Engineering, Henan University of Economics and Law, Zhengzhou* 450002*, China*;
2. *Institute of Information Engineering, Chinese Academy of Sciences, Beijing* 100093*, China*;
3. *College of Computer Science and Technology, Zhejiang University, Hangzhou* 310027*, China*;
4. *School of Information, Renmin University of China, Beijing* 100872*, China*
* Corresponding author. E-mail: xjzhang82@alu.ruc.edu.cn

Abstract Given a relational table T whose tuples are distributed among n users, we study differentially private algorithms for answering multidimensional analytical (MDA) queries over T. Existing solutions to this problem mostly employ hierarchical trees and optimal local hashing (OLH) mechanisms to index and perturb users' tuples, respectively. However, OLH with hierarchical trees, cannot prevent privacy leakage of root node combinations. To remedy the shortcomings of the existing solutions, this paper proposes a locally differentially private algorithm called hierarchical structure for MDA (H4MDA) to answer MDA queries. With H4MDA, we first propose a horizontal Generalized Random Response (HGRR) mechanism to perturb users' tuples, which refrains from combining root nodes to prevent privacy leaks. To take full advantage of the vertical structure of the tree, we further propose, in contrast to HGRR, a longitudinal GRR mechanism with fake data (LGRR-FD) to perturb users' tuples. The LGRR-FD reports the noise result with fake data to the collector to prevent the privacy disclosure of the leaf nodes. In addition, we propose another longitudinal GRR (LGRR) mechanism that discards the leaf node level to report the noise tuples. To improve the utility of MDA queries, based on LGRR reports, the collector uses the local consistency technique to reconstruct the leaf level visited by some MDA queries. A theoretical analysis and extensive experiments show that our algorithms can effectively improve the utility of MDA queries and outperform existing solutions.

Keywords multidimensional analytical queries, hierarchical structure, local differential privacy, local perturbation, random response mechanism