第3章 异方差加噪下的差分隐私直方图发布

3.1 引言

现有基于区间树的差分隐私直方图发布方法大多采用同方差的加噪方式。Hay[1]建立了差分隐私区间树并进行了同方差加噪与一致性调节。Xu[2]等人提出了 StructureFirst，优化直方图划分策略，并在划分区间后的构造区间树中进行了同方差加噪。采用同方差加噪方式的发布方法有效提升了发布数据的可用性和算法效率。而实际上，通过研究可以发现，若采用异方差加噪方式，可进一步提高发布精度。文献[3]提出了通过迭代方式，在区间树中进行层次间隐私预算分配的方法，有效降低了查询误差，但其在相同层次的节点中仍使用了相同的隐私预算，因此仍具有进一步优化的空间。文献[4]提出了 DP-tree，通过异方差加噪，能够对多维数据进行发布并提高查询精度，但该方法采用了完全 k 叉树结构，限制了树结构调节的灵活性。

针对以上问题，本章提出一种异方差加噪下面向任意区间树结构的差分隐私直方图发布算法 LUE-DPTree，以期进一步降低区间计数查询误差，提高发布数据可用性。算法 LUE-DPTree 首先根据区间计数查询的分布计算区间树中节点的覆盖概率，并据此分配各节点的隐私预算，从而实现异方差加噪；接着通过分析指出该异方差加噪策略适用于任意区间树结构下的差分隐私直方图发布，且从理论上进一步证明，对于任意区间树结构下基于异方差加噪的差分隐私直方图发布，仍然可在一致性约束下利用最优线性无偏估计进一步降低区间计数查询的误差；最后通过实验对算法 LUE-DPTree 所发布直方图数据的区间计数查询精度及算法效率与同类算法进行了比较分析。

3.2 基础知识与问题提出

在现有差分隐私直方图发布方法的研究中，主要通过重构直方图结构来回答区间计数查询，代表方法是基于差分隐私区间树的发布方法。它利用区间树结构重构原始直方图，可有效提高发布数据的精确度和算法运行效率。

定义 3.1（差分隐私区间树[1]） 对于给定计数直方图 $H=\{C_1, C_2, \cdots, C_n\}$，对 H 建立的差分隐私区间树 T 满足以下特性：

(1) 非叶节点的子节点数大于或等于 2。

(2) 每个节点 x 对应于 H 中的一个区间范围，表示为$[L(x),R(x)]$，根节点所代表的区间为$[1,n]$。

(3) 每个节点 x 的真实计数值为 $h_x=\sum_{i=L(x)}^{R(x)}C_i$，通过对每个节点添加噪声[5-7]（本章采用 Laplace 噪声[5]），得到加噪计数值 $\tilde{h}_x=h_x+\mathrm{Lap}(1/\varepsilon_x)$，其中 ε_x 为节点 x 的隐私预算，使该节点满足 ε_x- 差分隐私。

定义 3.2（同/异方差加噪方式[1,4]） 给定区间树 T，通过噪声机制[1]使每个节点 x 满足 ε_x-差分隐私。若对任意节点 x、y，均有 $\varepsilon_x=\varepsilon_y$，则称作同方差加噪方式；若存在节点 x、y，使得 $\varepsilon_x\neq\varepsilon_y$，则称作异方差加噪方式。

如图 3.1 所示，当 $\varepsilon=1.0$ 时，在图 3.1(a)的同方差加噪方式下，区间树的每个节点的隐私预算 ε_x 均为 0.5。而在图 3.1(b)的异方差加噪方式下，节点 1 隐私预算为 0.33，节点 2、3、4 的隐私预算均约为 0.67。由于通常情况下区间树中的各个节点被查询区间覆盖的频率并不完全相同，例如在查询区间随机分布的情况下，节点 2、3、4 具有高于节点 1 的覆盖概率（计算过程将在后文给出），因此，在图 3.1(b)中，能够对多数高频覆盖节点添加更少的噪声，对少数低频覆盖节点添加更多的噪声，从而降低整体区间计数查询误差。

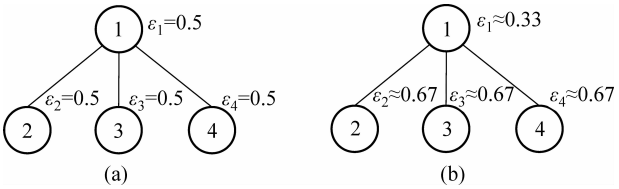

图 3.1　同/异方差加噪下的隐私预算分配

定义 3.3（查询一致性约束[1]） 在发布后的差分隐私区间树 T 中，任意节点 x 的计数值应与其子节点的计数值总和相等，称为查询一致性约束：

$$\bar{h}_x=\sum_{y\in \mathrm{Son}(x)}\bar{h}_y$$

其中，$\bar{h}$ 代表最终发布后节点的计数值，$\mathrm{Son}(x)$为节点 x 的子节点集合。

本章的研究问题及目标是：对于给定的原始直方图 H，建立与其对应的差分隐私区间树 T，并通过异方差方式进行加噪；接着说明该加噪方式适用于任意区间树结构，并利用查询一致性约束条件进一步优化区间计数查询精度；最后提出异方差加噪下面向任意区间树结构的差分隐私直方图发布算法 LUE-DPTree，同时保证算法满足 ε-差分隐私。

3.3　基于区间查询概率的差分隐私直方图发布

3.3.1　问题提出

对于统计直方图，若所有可能的区间计数查询的概率相同，则称区间计数查询符合均匀分布。例如，直方图大小为 2，则用户可能提出的区间计数查询有[1,1]、[1,2]、[2,2] 3 种，

且每一个查询的概率均为 1/3。

根据定义 2.2 中关于查询覆盖的说明，区间树节点被覆盖的概率可通过穷举覆盖区间树节点 x 的查询 Q 计算：

$$\mathrm{Pro}(x)=\sum_{Q\in \mathrm{Cover}(x)} P_Q \tag{3.1}$$

其中，$\mathrm{Cover}(x)$表示覆盖节点 x 的区间查询集合，P_Q 表示某个查询 Q 的概率。若假设所有查询的概率相同，则有 $\mathrm{Pro}(x)=|\mathrm{Cover}(x)|/|Q_{\mathrm{all}}|$，其中 Q_{all} 表示所有可能的区间计数查询。覆盖节点 x 的查询数目越多，则节点被覆盖的概率越大。

对一个区间计数查询 Q，令 $\mathrm{Node}(Q)$表示查询 Q 覆盖节点集合，则有

$$\mathrm{Err}_s(Q)=\sum_{x\in \mathrm{Node}(Q)} \frac{2\Delta^2}{\varepsilon^2}$$

其中，$\mathrm{Err}_s(Q)$表示 Q 查询的方差，Δ 表示差分隐私敏感度，即区间树的高度。在任意区间计数查询概率相等的背景下，其查询的方差的期望为

$$E(\mathrm{Err}_s(Q))=\frac{\sum_{Q\in Q_{\mathrm{all}}}\sum_{x\in \mathrm{Node}(q)} \frac{2\Delta^2}{\varepsilon^2}}{|Q_{\mathrm{all}}|}$$

通过进一步化简，可得

$$E(\mathrm{Err}_s(Q))=\sum_x \frac{2\Delta^2|\mathrm{Cover}(x)|}{|Q_{\mathrm{all}}|\varepsilon^2}=\sum_x \frac{2\Delta^2\mathrm{Pro}(x)}{\varepsilon^2} \tag{3.2}$$

例 3.1 对于大小为 5 的直方图，可构造出图 3.2 所示的 2-区间树。在区间计数查询满足均匀分布，即所有的区间查询等概率被提出的情况下，容易利用式(3.1)计算出每个节点被覆盖的概率，其概率分布如图 3.2 所示。

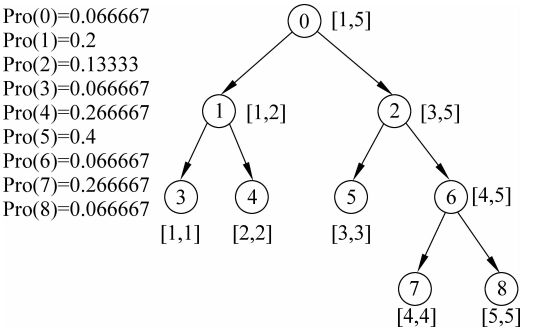

图 3.2 区间计数查询均匀分布下区间树节点被覆盖的概率

从图 3.2 可以看出，在区间计数查询均匀分布背景下，每个区间树节点最终被覆盖的概率不尽相同。节点 5 被覆盖的概率高达 0.4，而节点 0 被返回的概率仅仅为 0.066 667。这意味着对于一个随机的查询，节点 5 被覆盖的可能性远远大于节点 0 被覆盖的可能性，由此可以认为节点 5 在概率意义上更加重要。

对于大小为 n 的统计直方图，存在多种区间树结构与之对应。然而常使用均方的策略构建 k-区间树。其中 k 是一个事先指定的数字，一般取 2。

例 3.2 考虑大小为 3 的直方图，图 3.3 展示了两种可能的区间树结构。

利用式(3.1)，可得这两种区间树结构中各节点被覆盖的概率，如图 3.4 所示。

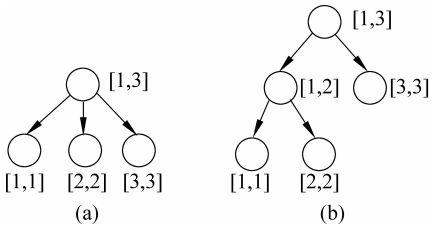

图 3.3　直方图大小为 3 的情况下两种可能的区间树结构

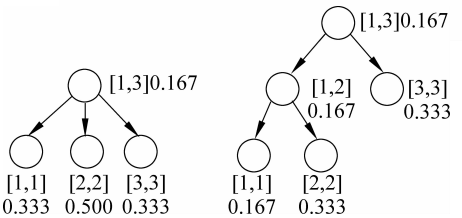

图 3.4　直方图大小为 3 的情况下两种可能的区间树结构节点被覆盖概率

根据式(3.2),图 3.4 中对应的两种可能的区间树结构在区间查询均匀分布前提下,其区间查询误差期望值分别为(令 $\varepsilon=1.0$)

$$E_1(\mathrm{Err}_s(Q)) = 2\times\frac{2^2}{1.0^2}\times(0.167+0.333+0.500+0.333)\approx 10.664$$

$$E_2(\mathrm{Err}_s(Q)) = 2\times\frac{3^2}{1.0^2}\times(0.167+0.167+0.333+0.167+0.333)\approx 21.006$$

可见,在区间计数查询符合均匀分布前提下,图 3.3(a)的区间树结构与图 3.3(b)的区间树结构相比,其区间计数查询误差期望值更小,即查询的精度更高。由于图 3.3(b)的区间树结构对应的敏感度大于图 3.3(a)的区间树结构对应的敏感度,因此,即使图 3.3(b)中节点被覆盖概率之和略小,图 3.3(a)的区间树结构对应的区间计数查询误差期望值也将更小。

由上述例子可以看出,不同的区间树结构对应不同的节点被覆盖概率以及不同的区间查询方差期望。因此,如何根据区间计数查询的分布计算区间树节点的被覆盖概率,进行区间树结构的构建,以进一步减小区间计数查询方差的期望,提高区间计数查询的精度,是本节要解决的主要问题。

3.3.2　基于区间计数查询概率的差分隐私直方图发布算法

1. 相关定义与性质

性质 3.1　在区间计数查询概率相等背景下,具有 n 个叶节点的区间树中节点 x 被覆盖概率为

$$\mathrm{Pro}(x)=\begin{cases}\dfrac{1}{|Q_{\mathrm{all}}|}, & x\text{ 为根节点}\\ P_L(x)+P_R(x)-P_C(x), & \text{否则}\end{cases}$$

其中(令 y 为 x 的父节点)

$$\begin{cases}P_L(x)=(L(x)-L(y))(n-R(x)+1)/|Q_{\mathrm{all}}|\\ P_R(x)=L(x)(R(y)-R(x))/|Q_{\mathrm{all}}|\\ P_C(x)=(L(x)-L(y))(R(y)-R(x))/|Q_{\mathrm{all}}|\end{cases}$$

证明:对于根节点,易知有且仅有一个查询能覆盖它。对于非根节点 x,令其父节点为 y,则根据 2.3.1 节中关于 k-区间树的定义可知

$$L(y)\leqslant L(x)\leqslant R(x)\leqslant R(y)$$

若某个查询 $Q=[L,R]$覆盖 x 节点,则 Q 不能覆盖其父节点 y,因此 Q 必须满足((L(y)+

$1 \leqslant L \leqslant L(x)) \wedge (R(x) \leqslant R \leqslant n)) \vee ((1 \leqslant L \leqslant L(x)) \wedge (R(x) \leqslant R \leqslant R(y)-1))$。其中，满足$(L(y)+1 \leqslant L \leqslant L(x)) \wedge (R(x) \leqslant R \leqslant n)$的可能的 Q 为$(L(x)-L(y))(n-R(x)+1)$个，而由于各区间被提出的概率相同，因此满足此条件的区间被提出的概率为

$$P_L(x) = (L(x)-L(y))(n-R(x)+1)/|Q_{\text{all}}|$$

同理，满足$(1 \leqslant L \leqslant L(x)) \wedge (R(x) \leqslant R \leqslant R(y)-1)$的可能 Q 为 $L(x)(R(y)-R(x))$个，因此，满足此条件的区间被提出的概率为

$$P_R(x) = L(x)(R(y)-R(x))/|Q_{\text{all}}|$$

然而，某些区间将同时满足以上两个条件，因此在概率中将被重复计算。其中，被重复计算的区间个数为$(L(x)-L(y))(R(y)-R(x))$个，因此，被重复计算的概率值为 $P_C(x)=(L(x)-L(y))(R(y)-R(x))/|Q_{\text{all}}|$，根据容斥原理，最终父节点为 y 的 x 节点被覆盖概率为 $P_L(x)+P_R(x)-P_C(x)$。证毕。

例 3.3 令 $n=6$，$k=3$，则其 3-区间树结构如图 3.5 所示。

显然$|Q_{\text{all}}| = \binom{7}{2} = 21$。考虑树中 x 节点，其父节点为 y，则

$$\begin{cases} P_L(x)=(3-1)(6-4+1)/21=2/7 \\ P_R(x)=3(6-4)/21=2/7 \\ P_C(x)=2\times 2/21=4/21 \end{cases}$$

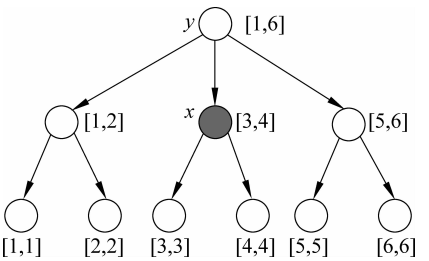

图 3.5 直方图大小为 6 时的 3-区间树

其中 $P_L(x)$为以下区间被提出的概率之和：

$$[2,4],[2,5],[2,6]$$
$$[3,4],[3,5],[3,6]$$

同理，$P_R(x)$为以下区间被提出的概率之和：

$$[1,4],[1,5]$$
$$[2,4],[2,5]$$
$$[3,4],[3,5]$$

$P_C(x)$为以下区间被提出的概率之和：

$$[2,4],[2,5]$$
$$[3,4],[3,5]$$

$P_C(x)$即为 $P_L(x)$与 $P_C(x)$同时考虑的区间被提出的概率之和。因此，在这棵 3-区间树中，x 节点被覆盖的概率为

$$P_L(x)+P_R(x)-P_C(x)=8/21 \approx 0.381$$

定义 3.4(区间子树估价函数值) 令 $H(L(x),R(x),\text{ary})$为节点 x 所代表区间$[L(x),R(x)]$按照 ary-区间树的划分方式进行划分得到的子树的估价函数值，若令其划分后得到的子树为 T，则

$$H(L(x),R(x),\text{ary}) = \sum_{y \in T} \text{Pro}(y)$$

例 3.4 在图 3.2 所示区间树结构中，选取节点 2 作为区间子树的根节点，令 ary=2，如图 3.6 所示。

根据定义 3.1 可得

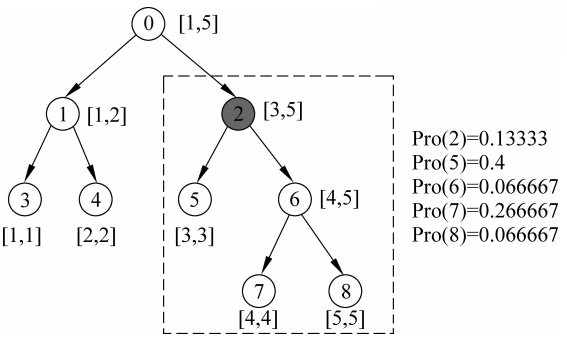

图 3.6　节点 2 按照 ary=2 划分后的区间树结构

$$H(3,5,2)=(0.13333+0.4+0.066667+0.266667+0.066667)\approx 0.933$$

若令 ary=2，即节点 2 所代表区间按照 3-区间树的划分规则来划分，则划分后区间树结构将与图 3.6 中的区间树结构有细微不同，如图 3.7 所示。

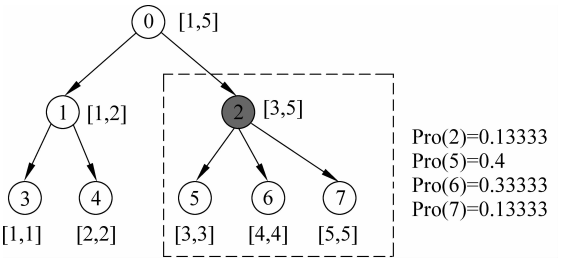

图 3.7　节点 2 按照 ary=3 划分后的区间树结构

此时

$$H(3,5,3)=(0.13333+0.4+0.33333+0.13333)\approx 1.0$$

2. 算法描述

设直方图大小为 n，且属性值离散化后的值从 1 开始。算法首先通过在适当规模内选取叉数(本节中选取 2～20 的数字)$k=\underset{2\leqslant \text{ary}\leqslant 20}{\operatorname{argmin}} \; E(\text{Err}_s(Q))$，使得其区间查询误差期望最小。在此情况下，递归地对区间树非根节点进行调整：对于左端点不为 1 的区间树节点进行选择性划分，并按照区间子树估价函数值大小贪心选择最优划分叉数。在本算法中，规定在对区间树节点 x 进行划分时，若对于 $y_0, y_1 \in \text{Son}(x)$ 有 $L(y_0)<L(y_1)$，则 $|\text{Seg}(y_0)|\leqslant|\text{Seg}(y_1)|$。

算法 3.1　Struct-Construct(SC)算法

输入：区间树节点 x，区间划分叉数 k，划分叉数上界 U

输出：根据贪心策略调整后的区间树结构

1. 若 $L(x)=1$ 则按照 k-区间树规则划分子节点，并继续步骤 3，否则继续步骤 2
2. (选择性划分)在$[k,U]$内选取一个数 w，使得 $H(L(x),R(x),w)$最小，并按照 w-区间树规则划分子节点
3. 对于 $y\in\text{Son}(x)$，运行 Struct-Construct(y,k,U)

通过 k 的选取，可先粗略地构造出一棵 k-区间树。在此基础上，通过 SC 算法进一步构

造区间树结构，使得区间计数查询误差期望值进一步减小。

步骤 1 的判定是为了不对左端点为 1 的区间树节点进行步骤 2 的选择性划分。步骤 2 则是选择性划分，在叉数范围$[k,U]$内选择一个合适的叉数 w 划分节点 x，使得按照 w 划分 x 以及其子树后其节点被覆盖概率之和最小。步骤 3 则是对于 x 节点的每一个子节点 y 递归地调用 SC 算法。

3. 算法分析

结论 3.1 SC 算法运行后，未被选择性划分的节点构成了一条从根节点到某个叶节点的路径(令这些节点集合为 Path)。

证明：根据 SC 算法步骤 1 可知根节点$[1,n]$未被划分，其子节点$[1,\lceil n/k \rceil]$也未被选择性划分，以此类推，可从根节点开始，每次走向左端点为 1 的子节点，直至走到某一叶节点。证毕。

结论 3.2 SC 算法构造出的区间树，其高度为结论 3.1 中的路径长度，即$|\mathrm{Path}|$。

证明：假设根节点的子节点分别为 $y_0, y_1, \cdots, y_{k-1}$，若对于 $y_i, y_j \in \mathrm{Son}(x)$有 $L(y_i) < L(y_j)$，则$|\mathrm{Seg}(y_i)| \geqslant |\mathrm{Seg}(y_j)|$，因此 SC 算法运行后对于根节点的子节点有$|\mathrm{Seg}(y_0)| \geqslant |\mathrm{Seg}(y_1)| \geqslant \cdots \geqslant |\mathrm{Seg}(y_{k-1})|$。对于 $y_1, y_2, \cdots, y_{k-1}$，由于 $L(y_i) <> 1$，因此 SC 算法会对它们进行选择性划分，且划分参数大于或等于 k。由于划分参数越大，区间树高度越小，因此 y_i 对应的子树高度 $\mathrm{Height}(y_i)+1 \leqslant \mathrm{Height}(\mathrm{root})$。因此树高

$$\mathrm{Height}(\mathrm{root}) = \mathrm{Height}(y_0) + 1 \tag{3.3}$$

将式(3.3)不断展开，易知区间树高度恰好为根节点至叶节点$[1,1]$的路径中的节点数，即 $\mathrm{Height}(\mathrm{root}) = |\mathrm{Path}|$。证毕。

根据结论 3.2 可知，该差分隐私区间树对应敏感度为$|\mathrm{path}|$。化简式(3.2)可得

$$E(\mathrm{Err}_s(Q)) = \frac{2\Delta^2}{\varepsilon^2}\sum_x \mathrm{Pro}(x) = \frac{2\,|\,\mathrm{Path}\,|^2}{\varepsilon^2}\sum_x \mathrm{Pro}(x)$$

在 SC 算法的选择性划分阶段，一次划分将会通过贪心选择将节点 x 按照划分参数 w 划分为 w-区间子树。图 3.8 给出了 w 为 3 的情况下对于 x 进行区间子树划分的例子。

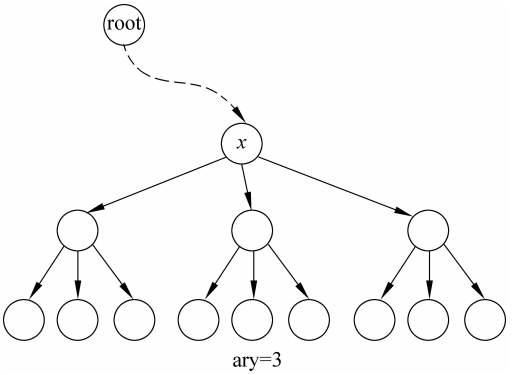

图 3.8 ary=3 划分区间子树后的区间树结构

由于 $w=3$ 对应的 $H(L(x),R(x),w)$ 最小，因此有 $\sum_{x\in T_{w=3}} \mathrm{Pro}(x) \leqslant \sum_{x\in T_{w<>3}} \mathrm{Pro}(x)$，其中 T_w 表示区间树节点 x 按照 $\mathrm{ary} = w$ 进行划分，故

$$E_{w=E}(\mathrm{Err}_s(Q)) \leqslant E_{w<>3}(\mathrm{Err}_e(Q))$$

因此，在每一次选择性划分中，区间计数查询误差的期望值非增。

结论 3.3　SC 算法的时间复杂度为 $O(Un\log_2 n)$。

证明：由于划分参数不小于 k，因此构建出的区间树高度为 $O(\log_2 n)$。根据区间树划分定义易得每一层的节点区间长度不超过 n，即 $|\mathrm{Seg}(y_i)| \leqslant n$。

设区间树中某层节点如图 3.9 所示，若对于节点 y_i 进行选择性划分，需要选择 $O(U)$ 次划分参数，且每次需要通过构建完全 w- 区间子树并遍历子树以计算 $H(L(y_i), R(y_i), w)$，由于遍历的时间复杂度为 $O(|\mathrm{Seg}(y_i)|)$，因此对于节点 y_i，其选择性划分时间复杂度为 $O(U|\mathrm{Seg}(y_i)|)$。而对于某层的所有叶节点，其选择性划分的时间复杂度之和为 $O\left(U\sum|\mathrm{Seg}(y_i)|\right)=O(Un)$。由于区间树高度为 $O(\log_2 n)$，故选择性划分的总时间复杂度为 $O(Un\log_2 n)$。而无论对于节点 x 是否进行选择性划分，最终均需要按照一个划分参数进行子节点的划分，因此划分步骤的时间复杂度为 $O(U)$。由于区间树的节点数目为 $O(n)$，因此划分步骤的总时间复杂度为 $O(Un)$。故 SC 算法的总时间复杂度为 $O(Un\log_2 n+Un)=O(Un\log_2 n)$。证毕。

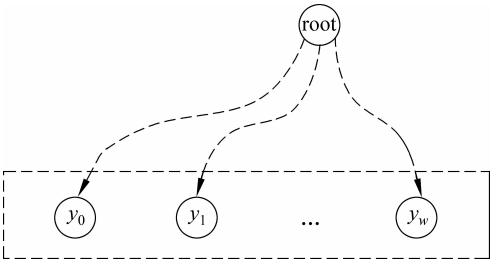

图 3.9　区间树中某层节点

3.3.3　实验结果与分析

本节将在区间计数查询符合均匀分布的背景下对直方图发布算法中区间查询的精度进行实验研究。其中，在同一数据集上，区间计数查询精度由随机区间查询结果与真实查询值之间的差异表示；在同一数据集上，比较并分析区间计数查询符合均匀分布前提下区间计数查询误差期望值的差异。SC 算法的比对对象是传统二叉区间树(regular 2-range tree)发布算法以及直接对直方图添加噪声的 Dwork 算法。对于现实数据集分别运行 SC 算法、传统二叉区间树发布算法以及 Dwork 算法，对各个算法运行前后的直方图随机区间查询精度进行衡量以及对比。

1. 实验数据与环境

实验数据分别来自 Amazon(亚马逊)网站于 2005 年 3 月 1 日 0 时至 2010 年 8 月 31 日晚 23 时期间被访问的采样记录(称为 Amazon 数据集)以及从 AOL 导出用户的点击网址为 http://www.ebay.com 的数据(称为 AOL 数据集)。在本实验中区间查询长度为 2^i($i\geqslant 0$)；对于每一个区间查询长度，随机生成 500 个查询区间；采样 100 次加噪声后的数据；误差通过区间查询的均方误差平均估值来衡量。

实验环境为：1.8GHz Intel Core i5；4GB 内存；Mac OS X 10.8.3 操作系统；算法用 C++ 语言实现。

表 3.1 是两个数据集的统计信息。其中，时间顺序划分表示将要进行发布的直方图中的属性按照日期（精确到天或分钟）递增顺序划分。

表 3.1　数据集统计信息

数据集	数据规模	划分单位	划分后的数据规模
Amazon	716 064	d	2010
AOL	36 389 577	min	48 130

2. 区间计数查询误差期望值

对 AOL 以及 Amazon 数据集运行 SC 算法与传统二叉区间树发布算法，并通过式(3.2)计算得到区间计数查询均匀分布下查询误差的期望值，如图 3.10 所示。

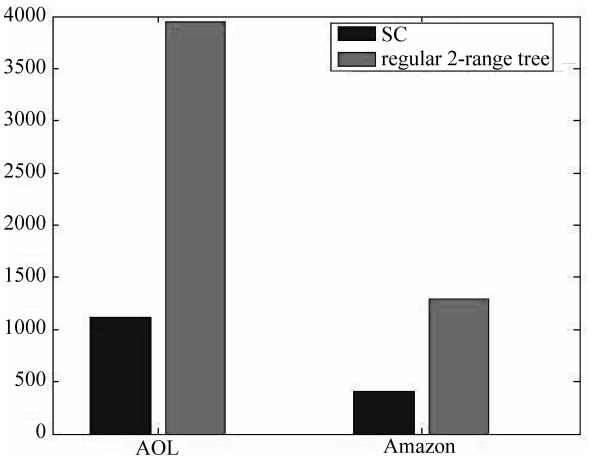

图 3.10　AOL/Amazon 数据集下随机区间查询误差期望值

从实验结果可以看出，在 AOL 以及 Amazon 数据集中随机计数区间查询下，SC 算法对应的区间查询误差期望值远小于传统二叉区间树发布算法。SC 算法中参数 k 是通过式(3.2)选择的，因此其误差期望值应小于传统二叉区间树发布，而在 SC 算法的每一次选择性划分中，区间计数查询误差期望值非增，因此划分完后其误差期望值小于原先 k-区间树的误差期望值。实验结果符合理论预期。

3. 区间计数查询下的精度

分别对 AOL 以及 Amazon 数据集运行 SC 算法、传统二叉区间树发布算法以及 Dwork 算法，对于随机区间查询计算其均方差，并对此误差取对数。实验结果分别如图 3.11 和图 3.12 所示。

从实验结果可以看出，在 AOL 以及 Amazon 数据集中随机计数区间查询下，SC 算法的误差曲线完全在传统二叉区间树发布算法下方。而在区间长度较小的情况下，Dwork 算法的查询误差较小；在区间长度较大的情况下，Dwork 算法的误差大于传统二叉区间树发布算法以及 SC 算法的误差。由于 Dword 算法中的误差为线性累加，因此其误差和查询区间大

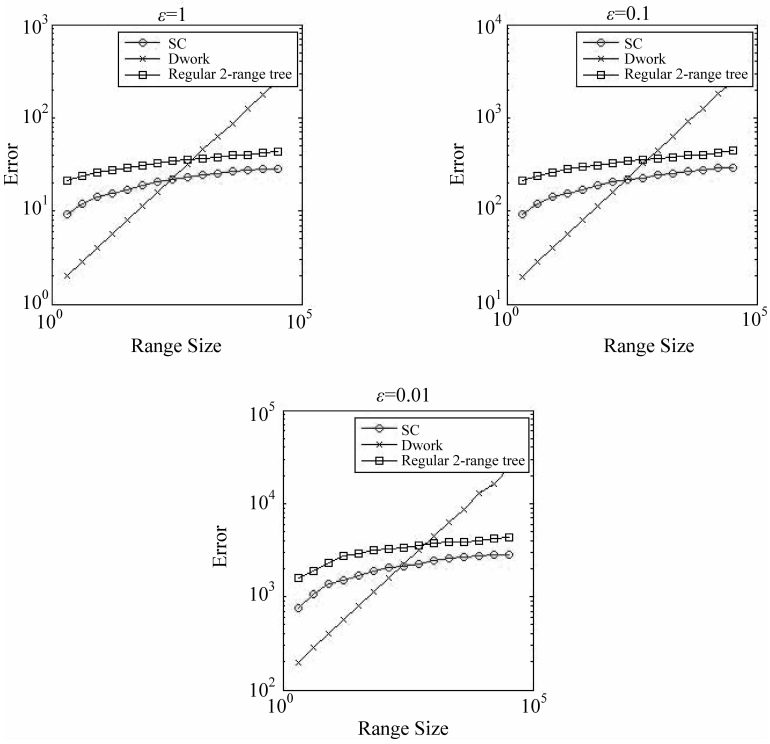

图 3.11　AOL 数据集下随机区间查询误差

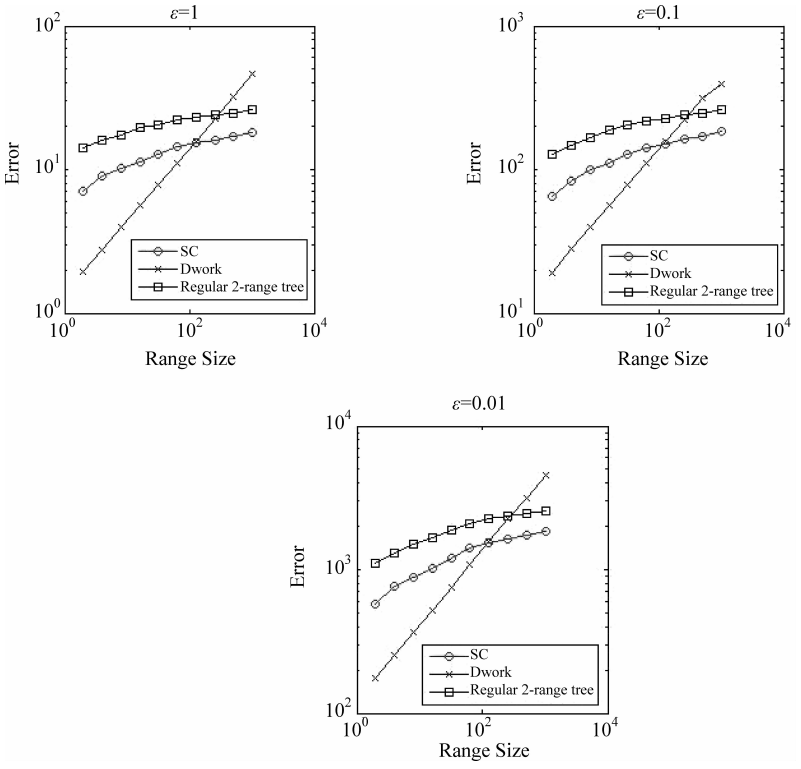

图 3.12　Amazon 数据集下随机区间查询误差

小呈线性关系；传统二叉区间树发布算法以及 SC 算法均通过区间树发布数据，因此其区间计数查询误差和区间大小呈类对数关系。本实验中，传统二叉区间树发布算法以及 SC 算法均运行于相同二叉区间树的树结构下，根据算法分析，SC 算法优势明显，符合理论预期。

以上实验结果表明，在 Amazon 及 AOL 数据集下，SC 算法在随机区间计数查询下误差较小，总体与理论预期相符，具有一定的实用性。

3.4 异方差加噪下面向任意树结构的差分隐私直方图发布算法

3.4.1 节点覆盖概率计算

当对区间树进行区间计数查询时，其计数值为查询区间$[Q_L, Q_R]$所覆盖的多个节点计数值之和[1]。查询区间所覆盖的节点互不相交，且并集等于查询区间。因此，被覆盖节点 x 需满足 $Q_L \leqslant L(x) \leqslant R(x) \leqslant Q_R$。同时，对任意一次查询，若其父节点 f_x 能被查询区间覆盖，节点 x 将被忽略。因此，节点 x 被查询区间覆盖的条件为

$$(Q_L \leqslant L(x) \leqslant R(x) \leqslant Q_R) \wedge \neg (Q_L \leqslant L(f_x) \leqslant R(f_x) \leqslant Q_R) \tag{3.4}$$

当 x 为根节点时，因其父节点 f_x 不存在，令$(Q_L \leqslant L(f_x) \leqslant R(f_x) \leqslant Q_R)$为假。

本节假定所有查询区间的出现概率相等。由式(3.1)可得出节点被覆盖概率 p_x 的计算公式：

$$p_x = \begin{cases} \dfrac{1}{n(n+1)/2}, & x \text{ 为根节点} \\ \dfrac{L(x)(n-R(x)+1) - L(f_x)(n-R(f_x)+1)}{n(n+1)/2}, & \text{否则} \end{cases} \tag{3.5}$$

根据式(3.5)，可由算法 3.2 计算节点被覆盖概率。

算法 3.2 CNCP(Calculate Node Coverage Probability)

输入：待计算节点 x 及其父节点 f_x

输出：区间树中所有节点的被覆盖概率 p_x

1. 若 x 为根节点：

$$p_x \leftarrow \frac{1}{n(n+1)/2}$$

2. 若 x 为其他节点：

$$p_x \leftarrow \frac{L(x)(n-R(x)+1) - L(f_x)(n-R(f_x)+1)}{n(n+1)/2}$$

3. 对所有 $y \in \mathrm{Son}(x)$，执行 CNCP(y, x)

实际上，当查询区间满足其他分布特性时，通过对式(3.5)的调整，其节点被覆盖概率也可通过本算法计算。由于算法 3.2 仅要求树结构满足区间树定义，因此适用于任意树结构的差分隐私区间树。

3.4.2 节点系数计算及隐私预算分配

在计算出节点被覆盖概率后，即可据此调整隐私预算，通过异方差加噪方式降低整体的

查询误差。而在此之前，需分析如何保证异方差加噪后的差分隐私区间树满足 ε-差分隐私。

结论 3.4　给定区间树 T，Leaf(x) 表示以 x 为根节点的子树的叶节点集合，Path(x，y) 表示节点 x 到其子节点 y 路径上的节点集合。若通过添加 Laplace 噪声使每个节点 x 满足 ε_x- 差分隐私，则整体发布过程满足 $\max\limits_{x\in \text{Leaf(root)}}\left(\sum\limits_{y\in \text{Path(root},x)}\varepsilon_y\right)$- 差分隐私。

证明：对任意节点 x，设 Sub(x)表示以节点 x 为根节点的子树，$A(x)$表示对子树 Sub(x) 进行发布的算法，则：

(1) 对节点 x 的任意两个节点 y_i，$y_j\in$Son(x)，由于 Sub(y_i)$\cap$Sub(y_j)$=\varnothing$，根据差分隐私并行组合性[8]，算法 $A(y)_{y\in \text{Son}(x)}$ 组合后满足 $\max\limits_{y\in \text{Son}(x)}A(y)$-差分隐私。

(2) 对于节点 $y\in$Son(x)，由于节点 x 的隐私预算为 ε_x，且 Sub(y)$\cap$Sub(x)=Sub(y)，由差分隐私序列组合性[8]可得，$A(x)$满足$(\varepsilon_x+\max\limits_{y\in \text{Son}(x)}A(y))$-差分隐私。

由(1)、(2) 可得，A(root) 满足 $\max\limits_{x\in \text{Leaf(root)}}\left(\sum\limits_{y\in \text{Path(root},x)}\varepsilon_y\right)$- 差分隐私，证明完毕。

根据结论 3.4，若要保证异方差加噪后的区间树满足 ε-差分隐私，需满足以下条件：

$$\max_{x\in \text{Leaf(root)}}\left(\sum_{y\in \text{Path(root},x)}\varepsilon_y\right)=\varepsilon$$

定义 $H(z)=\sum\limits_{y\in \text{Path(root},z)}\varepsilon_y$，$z$ 为任意叶节点，为最小化区间计数查询误差期望，将以上条件转换为

$$H(z)_{z\in \text{Leaf(root)}}=\sum_{y\in \text{Path(root},z)}\varepsilon_y=\varepsilon$$

同时，令 Node(Q)表示区间计数查询 Q 所覆盖的节点集合，由于 Lap($1/\varepsilon_x$)噪声引起的标准方差[9]为 $2/\varepsilon_x^2$，因此，查询 Q 的误差方差为

$$\text{Err}(Q)=\sum_{x\in \text{Node}(Q)}\frac{2}{\varepsilon_x^2}$$

则区间计数查询误差期望为

$$E(\text{Err}(Q))=\frac{\sum\limits_{Q\in Q_{\text{all}}}\sum\limits_{x\in \text{Node}(Q)}\frac{2}{\varepsilon_x^2}}{|Q_{\text{all}}|}=\frac{p_x|Q_{\text{all}}|\sum\limits_x\frac{2}{\varepsilon_x^2}}{|Q_{\text{all}}|}=2\sum_x\frac{p_x}{\varepsilon_x^2}$$

因此，在满足 ε-差分隐私的前提下，求解区间树上各节点的差分隐私预算，从而最小化区间计数查询误差期望的问题可转化为以下最优化问题：

$$\begin{aligned}&\min\quad f(\boldsymbol{\varepsilon})=\frac{1}{2}E(\text{Err}(Q))=\sum_x\frac{p_x}{\varepsilon_x^2}\\&\text{s.t.}\quad H(z)_{z\in \text{Leaf(root)}}=\varepsilon\end{aligned}\tag{3.6}$$

其中，$\boldsymbol{\varepsilon}$ 表示区间树中节点的差分隐私预算向量。

下面通过定义 3.5 与结论 3.5 进行求解。

定义 3.5(路径隐私预算和)

$$\text{psum}(x)=\begin{cases}\varepsilon_x, & x\in \text{Leaf(root)}\\ \text{psum}(\text{SonBound}(x))+\varepsilon_x, & \text{否则}\end{cases}$$

其中，SonBound(x)$=\{y\mid y\in \text{Son}(x)\wedge L(y)=L(x)\}$，由式(3.6)中的约束条件可得 psum(root)$=\varepsilon$。

结论 3.5 为最小化区间计数查询误差期望(即求解式(3.6)),区间树中差分隐私预算分配方案需满足

$$\varepsilon_x = \frac{a_x}{b_x}\mathrm{psum}(x)$$

其中 a_x、b_x 称为节点 x 的节点系数,有

$$a_x = \begin{cases} 1, & x \in \mathrm{Leaf}(\mathrm{root}) \\ \left(\dfrac{p_x}{\sum\limits_{y \in \mathrm{Son}(x)} \dfrac{p_y b_y^3}{a_y^3}} \right)^{\frac{1}{3}}, & \text{否则} \end{cases}$$

$$b_x = \begin{cases} 1, & x \in \mathrm{Leaf}(\mathrm{root}) \\ a_x + 1, & \text{否则} \end{cases}$$

证明:

(1) 若 x 为叶节点,则结论 3.5 显然成立。

(2) 若 x 为非叶节点,利用拉格朗日乘数法,可构造如下函数:

$$F(\boldsymbol{\varepsilon}, \lambda) = \sum_x \frac{p_x}{\varepsilon_x^2} + \sum_{z \in \mathrm{Leaf}(\mathrm{root})} \lambda_z \ (H(z) - \varepsilon)$$

其中,λ 为引入的未知标量。因目标函数 $f(\varepsilon)$为凸函数,同时,求解式(3.5)与求解函数 $F(\boldsymbol{\varepsilon}, \lambda)$ 的全局最优解等价,因此,将 $F(\boldsymbol{\varepsilon}, \lambda)$对 $\boldsymbol{\varepsilon}$ 求导,得

$$\frac{\partial F(\boldsymbol{\varepsilon}, \lambda)}{\partial \varepsilon_x} = \frac{-2p_x}{\varepsilon_x^3} + \sum_{z \in \mathrm{Leaf}(x)} \lambda_z = 0 \tag{3.7}$$

假设对于 $y \in \mathrm{Son}(x)$,结论均成立,则

$$\varepsilon_y = \frac{a_y}{b_y}\mathrm{psum}(y) \tag{3.8}$$

对于节点 $\mathrm{SonBound}(x)$和任意 $y \in \mathrm{Son}(x)$,由于式(3.6)的约束条件下,任意叶节点到根节点路径上的隐私预算和等于 ε,因此:

$$\varepsilon\text{-}\mathrm{psum}(y) = \sum_{u \in \mathrm{Path}(\mathrm{root}, x)} \varepsilon_u = \varepsilon\text{-}\mathrm{psum}(\mathrm{SonBound}(x)) \tag{3.9}$$

由式(3.8)和式(3.9),得

$$\varepsilon_y = \frac{a_y}{b_y}\mathrm{psum}(\mathrm{SonBound}(x)) \tag{3.10}$$

为满足式(3.7),必有

$$\begin{aligned} \frac{2p_y}{\varepsilon_y^3} &= \sum_{z \in \mathrm{Leaf}(y)} \lambda_z \\ \frac{2p_x}{\varepsilon_x^3} &= \sum_{z \in \mathrm{Leaf}(x)} \lambda_z = \sum_{y \in \mathrm{Leaf}(x)} \ \sum_{z \in \mathrm{Leaf}(y)} \lambda_z = \sum_{y \in \mathrm{Leaf}(x)} \frac{2p_y}{\varepsilon_y^3} \end{aligned} \tag{3.11}$$

将式(3.10)代入式(3.11)可得

$$\frac{p_x}{\varepsilon_x^3} = \frac{\sum\limits_{y \in \mathrm{Leaf}(x)} \dfrac{p_y b_y^3}{a_y^3}}{\mathrm{psum}(\mathrm{SonBound}(x))^3}$$

因此

$$\mathrm{psum}(\mathrm{SonBound}(x)) = \frac{\varepsilon_x}{\left(\dfrac{p_x}{\sum\limits_{y\in \mathrm{Leaf}(x)} \dfrac{p_y b_y^3}{a_y^3}}\right)^{\frac{1}{3}}}$$

令

$$a_x = \left(\frac{p_x}{\sum\limits_{y\in \mathrm{Leaf}(x)} \frac{p_y b_y^3}{a_y^3}}\right)^{\frac{1}{3}}$$

则

$$\mathrm{psum}(x) = \mathrm{psum}(\mathrm{SonBound}(x)) + \varepsilon_x = \frac{1+a_x}{a_x}\varepsilon_x = \frac{b_x}{a_x}\varepsilon_x$$

综合(1)、(2),结论得证。

至此,可通过算法 3.3 计算节点系数 a_x、b_x。

算法 3.3　CNP (Calculate Node Parameter)

输入:待计算节点 x

输出:区间树中所有节点的节点系数 a_x、b_x

1. 若 x 为叶节点:

$$a_x \leftarrow 1, \quad b_x \leftarrow 1$$

2. 若 x 为其他节点:

$$a_x \leftarrow \left(\frac{p_x}{\sum\limits_{y\in \mathrm{Son}(x)} \frac{p_y b_y^3}{a_y^3}}\right)^{\frac{1}{3}}$$

$$b_x \leftarrow a_x + 1$$

3. 对于 $y\in \mathrm{Son}(x)$,运行 CNP(y)

计算出节点系数 a_x、b_x 后,可通过算法 3.4 分配每个节点的差分隐私预算 ε_x 并进行异方差加噪。

算法 3.4　NPBD(Non-Uniform Private Budget Distribute)

输入:待加噪节点 x、路径隐私预算和 psum(x)

输出:区间树所有节点的加噪计数值 $\tilde{h}_x$

1. $\varepsilon_x \leftarrow \frac{a_x}{b_x}\mathrm{psum}(x)$　　//分配隐私预算

2. $\tilde{h}_x \leftarrow h_x + \mathrm{Lap}(1/\varepsilon_x)$　　//添加噪声

3. 对所有 $y\in \mathrm{Son}(x)$,执行 NPBD(y,psum(x)$-\varepsilon_x$)

下面以图 3.1 所示差分隐私区间树为例,分析异方差加噪对区间计数查询精度的影响。

将图 3.1 所示区间树通过算法 3.2 及算法 3.3 进行节

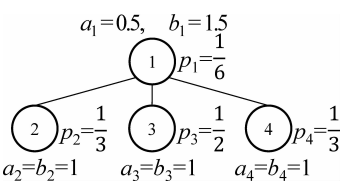

图 3.13　异方差加噪示例

点被覆盖概率和系数计算，结果如图 3.13 所示。再通过算法 3.4 进行隐私预算分配，得到各节点的隐私预算：

$$\varepsilon_1 \approx 0.33,\quad \varepsilon_2 \approx 0.67,\quad \varepsilon_3 \approx 0.67,\quad \varepsilon_4 \approx 0.67$$

该方案与图 3.1(b)所示一致。分别对图 3.1(a)、图 3.1(b)中的隐私预算分配方案计算区间计数查询误差期望：

$$E_a = 2\left(\frac{1/6}{0.5^2}+\frac{1/3}{0.5^2}+\frac{1/2}{0.5^2}+\frac{1/3}{0.5^2}\right)\approx 10.67$$

$$E_b = 2\left(\frac{1/6}{0.33^2}+\frac{1/3}{0.67^2}+\frac{1/2}{0.67^2}+\frac{1/3}{0.67^2}\right)\approx 8.25 < E_a$$

查询误差期望由原先同方差加噪的 10.67 降低至异方差加噪的 8.25，查询精度得到提高。

3.4.3 算法描述与分析

上述步骤实现了对差分隐私区间树进行的异方差加噪。由于仅要求树结构满足差分隐私区间树定义，并无其他限定，因此，该异方差加噪策略不仅适用于完全 k 叉树，还能够运用在任意区间树结构上。

不同树结构的差分隐私区间树示例如图 3.14 所示。

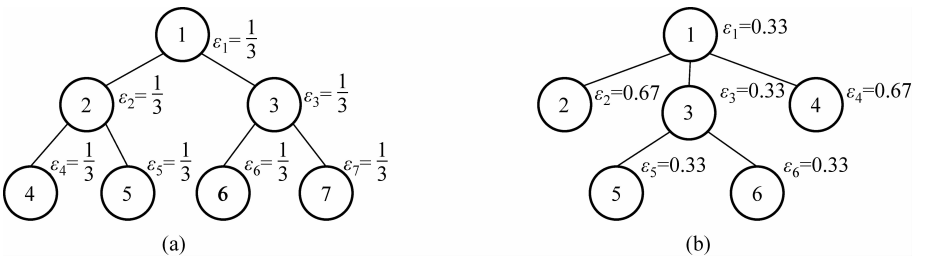

图 3.14　不同树结构下的区间树

如图 3.14 所示，对于长度为 2^2 的直方图数据集而言，可采用如图 3.14(a)所示的完全二叉树进行统计表示，也可用类似于图 3.14(b)所示的任意树结构差分隐私区间树进行表示。对其分别计算区间计数查询误差期望：

$$E_a = 2\left(3\frac{1/10}{1/3^2}+2\frac{2/10}{1/3^2}+2\frac{3/10}{1/3^2}\right)\approx 23.4$$

$$E_b = 2\left(\frac{1/10}{0.33^2}+2\frac{3/10}{0.67^2}+\frac{3/10}{0.33^2}+2\frac{2/10}{0.33^2}\right)\approx 17.1 < E_a$$

可以看出，采用任意区间树结构之后，可以对树结构进行更灵活的调整，从而有效降低查询误差。结合异方差加噪方式，能够进一步提升查询精度。

任意树结构差分隐私区间树的构建算法如下。

算法 3.5　TSC(Tree Structure Construct)

输入：待构建子树节点 x，当前区间$[l,r]$

输出：差分隐私区间树 T

1. $L[x]=l, R[x]=r$;　　//设置节点表示的区间

```
2. k=choose(x,l,r);                    //选择分支数 k
3. for(i=0;i<k;i++){
     分配新节点编号 child 和子区间[cl,cr];
     TSC(child,cl,cr);                 //继续构建子树
   }
```

在算法 3.5 中，对分支数 k 的选择和子区间长度的分配方式进行改变，均会导致树结构的变化。树结构构建完成并进行节点覆盖概率计算(CNCP)、节点系数计算(CNP)和异方差加噪(NPBD)后，即可得到异方差加噪下的任意树结构差分隐私区间树。

在建立任意区间树并进行异方差加噪后，可有效提高区间计数查询精度。然而，通过图 3.15 的示例可以发现，加噪后的区间树并不满足一致性约束。

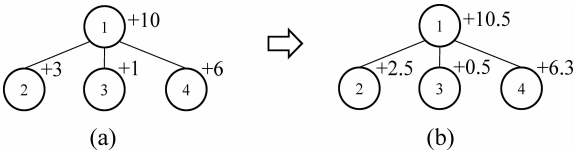

图 3.15　加噪后造成的一致性约束问题示例

图 3.15(a)所示的未加噪区间树经加噪后变为图 3.15(b)所示的加噪区间树，其父节点 1 的计数值 10.5 与子节点计数值之和 9.3 不同，不满足一致性约束。以下将针对此问题，通过理论分析，证明异方差加噪下的任意区间树仍可利用一致性约束进行最优线性无偏估计优化。

结论 3.6　在一致性约束下，求解差分隐私区间树节点加噪统计值 $\tilde{h}$ 的线性无偏估计值 $\bar{h}$，等同于求解如下的加权最小二乘问题[1]：

$$\begin{aligned} \min \quad & \sum_{x} \varepsilon_x^2 (\tilde{h}_x - \bar{h}_x)^2 \\ \text{s.t.} \quad & \sum_{y \in \text{Leaf}(x)} \bar{h}_y = \bar{h}_x \end{aligned} \tag{3.12}$$

其中，$\tilde{h}_x$ 为节点 x 的加噪统计值，$\bar{h}_x$ 为节点 x 的线性无偏估计值。

结论 3.7　差分隐私区间树从叶节点 w 到根节点路径上的节点线性无偏估计值 $\bar{h}$，满足下式：

$$\sum_{x \in \text{Path}(\text{root},w)} \varepsilon_x^2 \bar{h}_x = \sum_{x \in \text{Path}(\text{root},w)} \varepsilon_x^2 \tilde{h}_x \tag{3.13}$$

证明：式(3.12)可转换为求解下式：

$$\min \quad \sum_{x} \varepsilon_x^2 \Big(\sum_{y \in \text{Leaf}(x)} \bar{h}_y - \tilde{h}_x \Big)^2$$

对于任意叶节点 w，对 $\bar{h}_w$ 求偏导：

$$\frac{\partial f}{\partial \bar{h}_w} = 2 \sum_{x \in \text{Path}(\text{root},w)} \varepsilon_x^2 \Big(\sum_{y \in \text{Leaf}(x)} \bar{h}_y - \tilde{h}_x \Big) = 2 \sum_{x \in \text{Path}(\text{root},w)} \varepsilon_x^2 (\bar{h}_x - \tilde{h}_x)$$

令偏导数 $\dfrac{\partial f}{\partial \bar{h}_w} = 0$，则

$$\sum_{x \in \text{Path}(\text{root},w)} \varepsilon_x^2 \bar{h}_x = \sum_{x \in \text{Path}(\text{root},w)} \varepsilon_x^2 \tilde{h}_x$$

证明完毕。

结论 3.8 以节点 x 为根节点的子树中，节点 x 的估计值 $\bar{h}_x$、叶节点到节点 x 的估计值加权和 $\bar{g}_x$ 均是关于叶节点的线性方程：

$$\bar{g}_x = \sum_{y\in \mathrm{Path}(x,\mathrm{Bound}(x))} \varepsilon_y^2 \bar{h}_y = \alpha_x \bar{h}_{\mathrm{Bound}(x)} + c_x \tag{3.14}$$

$$\bar{h}_x = \sum_{y\in \mathrm{Leaf}(x)} \bar{h}_y = \beta_x \bar{h}_{\mathrm{Bound}(x)} + d_x$$

其中：

$$\alpha_x = \begin{cases} \varepsilon_x^2, & x \in \mathrm{Leaf}(\mathrm{root}) \\ \alpha_w + \varepsilon_x^2 \beta_x, & \text{否则} \end{cases}$$

$$\beta_x = \begin{cases} 1, & x \in \mathrm{Leaf}(\mathrm{root}) \\ \sum\limits_{y\in \mathrm{Son}(x)} \dfrac{\beta_y \alpha_w}{\alpha_y}, & \text{否则} \end{cases}$$

$$c_x = \begin{cases} 0, & x \in \mathrm{Leaf}(\mathrm{root}) \\ c_w + \varepsilon_x^2 d_x, & \text{否则} \end{cases}$$

$$d_x = \begin{cases} 0, & x \in \mathrm{Leaf}(\mathrm{root}) \\ \sum\limits_{y\in \mathrm{Son}(x)} \left(\dfrac{\beta_y}{\alpha_y} (\tilde{g}_y - \tilde{g}_w - c_y + c_w) + d_y \right), & \text{否则} \end{cases}$$

$$\tilde{g}_y = \sum_{u\in \mathrm{Path}(y,\mathrm{Bound}(y))} \varepsilon_u^2 \tilde{h}_u$$

$\mathrm{Bound}(x)$是以 x 为根节点的子树中的第一个叶节点，$w=\mathrm{SonBound}(x)$。

证明：定义节点高度

$$\mathrm{Height}(x) = \begin{cases} 0, & x \in \mathrm{Leaf}(\mathrm{root}) \\ \max\limits_{y\in \mathrm{Son}(x)} \{\mathrm{Height}(y)\} + 1, & \text{否则} \end{cases} \tag{3.15}$$

(1) 若节点 $x\in\{y|\mathrm{Height}(y)=0\}$，即 $x\in\mathrm{Leaf}(\mathrm{root})$，结论 3.8 显然成立。

(2) 假设对任意节点 $y\in\{y|\mathrm{Height}(y)\leqslant n\}$，式(3.14)均成立。当 $\mathrm{Height}(x)=n+1$ 时，

令 $\mathrm{hsum}(x)$表示从叶节点 x 到根节点路径上节点集合的加噪值加权和，由结论 3.7 可知：

$$\mathop{\mathrm{hsum}(x)}_{x\in \mathrm{Leaf}(\mathrm{root})} = \sum_{y\in \mathrm{Path}(\mathrm{root},x)} \varepsilon_y^2 \tilde{h}_y = \sum_{y\in \mathrm{Path}(\mathrm{root},x)} \varepsilon_y^2 \bar{h}_y$$

令 $w=\mathrm{SonBound}(x)$，由式(3.15)可知，对于节点 $y\in\mathrm{Son}(x)$，有 $\mathrm{Height}(y)\leqslant n$，$\mathrm{Height}(w)\leqslant n$，且根据结论 3.7，有

$$\begin{aligned} &\tilde{g}_y - \tilde{g}_w \\ &= \mathrm{hsum}(\mathrm{Bound}(y)) - \mathrm{hsum}(\mathrm{Bound}(w)) \\ &= \bar{g}_y - \bar{g}_w \\ &= \alpha_y \bar{h}_{\mathrm{Bound}(y)} + c_y - \alpha_w \bar{h}_{\mathrm{Bound}(w)} - c_w \end{aligned}$$

因此：

$$\bar{h}_{\mathrm{Bound}(y)} = \frac{\tilde{g}_y - \tilde{g}_w - c_y + \alpha_w \bar{h}_{\mathrm{Bound}(w)} + c_w}{\alpha_y}$$

代入式(3.14)可得

$$\bar{h}_y = \beta_y \frac{\tilde{g}_y - \tilde{g}_w - c_y + \alpha_w \bar{h}_{\mathrm{Bound}(w)} + c_w}{\alpha_y} + d_y$$
$$= \left(\frac{\beta_y \alpha_w}{\alpha_y}\right)\bar{h}_{\mathrm{Bound}(w)} + \frac{\beta_y}{\alpha_y}(\tilde{g}_y - \tilde{g}_w - c_y + c_w) + d_y$$

由一致性约束可得

$$\bar{h}_x = \sum_{y \in \mathrm{Son}(x)} \bar{h}_y = \left(\sum_{y \in \mathrm{Son}(x)} \frac{\beta_y \alpha_w}{\alpha_y}\right)\bar{h}_{\mathrm{Bound}(y)} + \sum_{y \in \mathrm{Son}(x)} \left(\frac{\beta_y}{\alpha_y}(\tilde{g}_y - \tilde{g}_w - c_y + c_w) + d_y\right)$$

令

$$\beta_x = \left(\sum_{y \in \mathrm{Son}(x)} \frac{\beta_y \alpha_w}{\alpha_y}\right)$$
$$d_x = \sum_{y \in \mathrm{Son}(x)} \left(\frac{\beta_y}{\alpha_y}(\tilde{g}_y - \tilde{g}_w - c_y + c_w) + d_y\right)$$

得

$$\bar{h}_x = \beta_x \bar{h}_{\mathrm{Bound}(x)} + d_x$$

因为

$$\bar{g}_x = \sum_{y \in \mathrm{Path}(x, \mathrm{Bound}(x))} \varepsilon_y^2 \bar{h}_y = \bar{g}_w + \varepsilon_x^2 \bar{h}_x$$
$$= \alpha_w \bar{h}_{\mathrm{Bound}(w)} + c_w + \varepsilon_x^2(\beta_x \bar{h}_{\mathrm{Bound}(x)} + d_x)$$
$$= (\alpha_w + \varepsilon_x^2 \beta_x)\bar{h}_{\mathrm{Bound}(x)} + (c_w + \varepsilon_x^2 d_x)$$

令

$$\alpha_x = (\alpha_w + \varepsilon_x^2 \beta_x)$$
$$c_x = (c_w + \varepsilon_x^2 d_x)$$

得

$$\bar{g}_x = \alpha_x \bar{h}_{\mathrm{Bound}(x)} + c_x$$

综合(1)、(2)，证明式(3.14)成立。

经由以上结论及证明，通过计算参数(α, β, c, d)的值，并利用式(3.14)进行估计值计算，可设计出对差分隐私区间树加噪后，在任意区间树结构下利用最优线性无偏估计进行调整的优化算法。

算法 3.6　PA_BLUE (Parameter Adjust using Best Linear Unbiased Estimate)

输入：待计算发布计数值的节点 x

输出：区间树所有节点的参数(α, β, c, d)

1. 若 x 为叶节点，更新：

$$\alpha_x \leftarrow \varepsilon_x^2, \beta_x \leftarrow 1, c_x \leftarrow 0, d_x \leftarrow 0$$

并结束算法

2. for each $y \in \mathrm{Son}(x)$

　　PA_BLUE(y)；

　　if Bound(y)=Bound(w) then $w=y$；

　end for.

3. 更新 $\beta_x \leftarrow \left(\sum_{y\in \mathrm{Son}(x)} \frac{\beta_y \alpha_w}{\alpha_y}\right)$

4. 更新 $\alpha_x \leftarrow (\alpha_w + \varepsilon_x^2 \beta_x)$

5. 更新 $d_x \leftarrow \sum_{y\in \mathrm{Son}(x)} \left(\frac{\beta_y}{\alpha_y}(\tilde{g}_y - \tilde{g}_w - c_y + c_w) + d_y\right)$

6. 更新 $c_x \leftarrow (c_w + \varepsilon_x^2 d_x)$

计算参数(α,β,c,d)后，通过算法 3.7 计算优化后的最终发布值。

算法 3.7 CRC (Calculate Range Count)

输入：待计算发布计数值的节点 x

令 tot 为 $\sum_{w\in \mathrm{Path}(x,\mathrm{root})\setminus x} \varepsilon_w^2 \bar{h}_w$

输出：区间树所有节点的发布计数值 $\bar{h}_x$

1. $\bar{h}_x \leftarrow \beta_x \frac{\mathrm{hsum}(\mathrm{Bound}(x)) - \mathrm{tot} - c_x}{\alpha_x} + d_x$

2. 对所有 $y\in \mathrm{Son}(x)$，执行 $\mathrm{CRC}(y, \mathrm{tot} + \varepsilon_x^2 \bar{h}_x)$

通过以上步骤，本节提出了异方差加噪下面向任意区间树结构进行最优线性无偏估计优化的差分隐私直方图发布算法。

算法 3.8 LUE-DPTree(Linear Unbiased Estimator for Differential Private Tree)

输入：原始直方图 H，差分隐私参数 ε

输出：异方差加噪，任意树结构的 ε-差分隐私区间树 T_Publish

```
1. T=TSC(root,1,n);                    //构建区间树
2. CNCP(root,0);                       //节点被覆盖概率计算
3. CNP(root);                          //节点系数计算
4. NPBD(root,ε);                       //异方差加噪
5. PA_BLUE(root);                      //最优线性无偏估计优化
6. T_Public=CRC(root,0);               //计算发布值
```

下面对算法 3.8 的差分隐私保护效果和算法复杂度进行分析证明。

结论 3.9 算法 3.8 所生成的差分隐私区间树 T_Publish 满足 ε-差分隐私。

证明：对于区间树 T_Publish，由式(3.6)中的约束条件可知，在计算各节点隐私预算 ε_x 时，始终在该约束下进行：

$$\underset{x\in \mathrm{Leaf}(\mathrm{root})}{H(x)} = \varepsilon$$

即

$$\max_{x\in \mathrm{Leaf}(\mathrm{root})} \{H(x)\} = \max_{x\in \mathrm{Leaf}(\mathrm{root})} \left(\sum_{y\in \mathrm{Path}(\mathrm{root},x)} \varepsilon_y\right) = \varepsilon$$

由结论 3.2 可知，整体发布过程满足 ε-差分隐私。

结论 3.10 算法 3.8 的算法时间复杂度、空间复杂度均为 $O(n)$。

证明：在算法 3.8 中，各步骤均为对区间树进行一次扫描，时间复杂度为 $O(n)$。在 CNCP、CNP 和 PA_BLUE 算法中，分别需存储各节点的节点被覆盖概率、节点系数等值，空间复杂度为 $O(n)$。在 TSC、NPBD、CRC 算法中，分别对树节点的发布值进行计算及改变，

空间复杂度同样为 $O(n)$。因此，算法 3.8 的时间复杂度和空间复杂度均为 $O(n)$，为线性复杂度。

下面从区间计数查询精度和算法运行效率两方面与同类代表算法 Boost[1] 进行对比分析。在文献[3]中，提出了在区间树的不同层次间进行异方差分配的迭代方法，本节将其应用于 Boost 算法中，并标识为 Boost-UN。同时，为了更好地体现本节算法的有效性，实验也与基于小波变换的 Privelet 算法[10]进行了比较分析。由于同样采用了异方差加噪方式的算法 DP-tree[4]并未给出具体算法描述，因此未将本节算法与其进行实验对比。在实验中，隐私参数 ε 取值分别为{1.0，0.1，0.01}。采用如式(3.16)所示的平均方差进行误差衡量。为使实验结果更具一般性，取算法执行 50 次的平均值作为最终结果。

$$\text{Error} = \frac{\sum_{q \in Q} (q(T) - q(T'))^2}{|Q|} \tag{3.16}$$

其中，$q(T)$为区间计数查询的真实结果，$q(T')$为区间计数查询的加噪计数值，$|Q|$为查询集大小。

为便于对比分析，本节采用了与文献[1]相同的实验数据集 Social Network、Search Logs、Nettrace 进行实验。其数据规模如表 3.2 所示。

表 3.2　数据集规模

数据集	Social Network	Search Logs	Nettrace
数据规模	11 342	32 768	65 536

实验硬件环境为：Intel Core i7 930 2.8GHz 处理器，4GB 内存，Windows 7 操作系统。算法实现采用 C++ 语言，由 Matlab 生成实验图表。

3.4.4　实验结果与分析

本节通过与 Boost、Privelet 等算法的对比，分析 LUE-DPTree 在区间计数查询精度上的表现，并通过对树结构的调整，分析不同树结构对查询精度的影响。

1. 与 Boost、Privelet 等算法的对比

本节分别采用随机任意长度区间和随机固定长度区间两种方式对算法查询精度进行检验。其中，任意长度区间随机生成 1000 条，区间的起点 L 和终点 R 随机生成，且 $L \leqslant R$。随机固定长度区间的大小分别取 $2^0, 2^1, \cdots, 2^{13}, \cdots$，每种长度随机生成 1000 条查询区间。考虑到 Boost 和 Privelet 算法适用于 2 的整数幂的数据规模，在这部分实验中，仅选取 Search Logs 和 Nettrace 为实验数据。

在图 3.16 和图 3.17 的实验对比结果中，随着隐私参数 ε 的减小，平均误差约按 10^2 的量级增长。在 4 种算法中，LUE-DPTree 算法的查询精度较优。

在图 3.18 和图 3.19 中，对于固定区间大小的随机查询，误差随区间大小增加而增大。在各区间长度下，LUE-DPTree 算法的查询精度均优于 Boost 和 Privelet 算法。而 Boost-UN 算法采用的异方差加噪方式仅在不同层次中进行了隐私预算分配，而在同一层次的节点中仍采用了相同的隐私预算，因此查询精度介于 Boost 和 LUE-DPTree 算法之间。

上述实验分析表明，与其他两种算法相比，LUE-DPTree 算法具有更高的数据发布