IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 29, 2019, accepted August 18, 2019, date of publication September 2, 2019, date of current version October 1, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2939015

An Effective Data Privacy Protection Algorithm Based on Differential Privacy in Edge Computing

YI QIAO[1], ZHAOBIN LIU[1], HAOZE LV[1], MINGHUI LI[1], ZHIYI HUANG[2], ZHIYANG LI[1], AND WEIJIANG LIU[1]

[1]School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China
[2]Department of Computer Science, University of Otago, Otago 9010, New Zealand

Corresponding author: Zhaobin Liu (zhbliu@dlmu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61370198, Grant 61370199, Grant 61672379, and Grant 61300187.

ABSTRACT With the rapid development of information science and the Internet of Things (IoT), people have an unprecedented ability to collect and share data, especially the various sensors as the entrance to data collection. At the same time, edge computing has begun to grasp the public's attention because of the difficult challenges of massive equipment access and massive data. Although such large amount of data provides a huge opportunity for information discovery, the privacy leakage has also been concerned. When the data publisher publishes various statistics, the attacker can obtain the statistical rules in the data by simply using the query function without contacting the user or the data publisher. Therefore, how to protect the data privacy of statistical information has become the focus of attention. In this paper, we proposed a partitioned histogram data publishing algorithm based on wavelet transform. Firstly, a partitioning algorithm based on greedy algorithm is used to obtain a better partition structure. Then, we use wavelet transform to add noise. Finally, for the authenticity and usability of histogram, we get the reductive original histogram structure. On the one hand, our algorithm can reduce the complexity of wavelet tree constructed by wavelet transform. On the other hand, the query noise changes from linear growth to multiple logarithm growth. So the accuracy of histogram counting query is improved. Experiments show that our algorithm has the significant improvement in data availability.

INDEX TERMS Internet of Things, edge computing, wavelet transforms, differential privacy.

I. INTRODUCTION

The continuous development of big data and communication technology has driven the applications of IoT becoming more and more global, and mobile intelligent terminals are playing an increasingly important role in people's lives. In big data, it is an inevitable trend for users to query the required information efficiently, quickly and accurately from the complicated data. As the third-party applications have become prevalent, the data stored by users in mobile intelligent terminals has also increasing rapidly. These data have enormous social value, but the leak of sensitive user information is also shocking us. Therefore, protecting users' privacy has become an urgent problem in privacy protection. Edge computing is an open platform that integrates core capabilities of network, computing, storage and application at the edge of the data sources network. The core is to migrate computing tasks from the cloud computing center to the edge devices that generate the source data. The edge computing is divided into the sensing control layer, the network layer, the agile controller and the application layer. In the process of uploading data to the data center, it is easy for an attacker to obtain the data in the edge device through simple counting query, which increases the risk of disclosing the privacy data of the user [1]–[3].

Differential privacy mainly uses non-interactive framework to publish sensitive data, and makes the published data meet the needs of data analysts. Commonly used publishing technologies include histogram, partition and sampling

The associate editor coordinating the review of this manuscript and approving it for publication was Raja Wasim Ahmad.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

filtering [4]. Histogram-based publishing techniques are primarily used for statistical data, such as hospital patient records. The early histogram publishing method is the Laplace mechanism[5], which represents data into the original histogram with equal-width interval, and then combines the Laplace mechanism to protect the datasets. References [6] and [7] proposed two multidimensional partitioning strategies for the histogram. The first partitioning strategy is to publish histogram of equal width grid based on cell partitioning strategy. The second partition strategy is implemented through the first partition strategy and the Kd-tree. Since most studies focus only on simple count values and ignore the histogram structure. So [8] and [9] turn their attention to the interval structure of histogram and propose two methods of publishing histogram, NoiseFirst and StructureFirst. Reference[10] proposed a more accurate method for range count query while satisfying differential privacy. In a histogram, the data is divided into disjointing subsets according to different attribute values, and the width of each subset is a query range, which is the range count query. The shortcomings in [10] are too obvious. Its sensitivity is relatively high.

The disadvantages of this approach are conspicuous. On the one hand, some of these methods divided too much histogram intervals, which violate the ϵ-differential privacy. On the other hand, because too many intervals are divided and too much noise is added, adding noise to each interval makes the data unusable. Based on these practical considerations, we propose a new design scheme, namely, wavelet transform based partition histogram publishing algorithm (PH_WT), to protect the data collected by edge devices. We aggregate similar values of histograms to form partitions, and then using Haar wavelet transform to add noise.

The originality and main contributions of this paper are summarized as follows:

1) In order to improve data security and availability, a PH_WT algorithm based on haar wavelet transform is proposed. Through analysis, we prove that our PH_WT algorithm not only satisfies the availability of published data, but also prevents data from being attacked.
2) We proposed a new partitioning method, the partition histogram algorithm(Partition_Histogram Algorithm), which can partition the histogram optimized through *Sum of Squared Error(SSE)*. This partition method effectively solves the problem that too many partitions will destroy the ϵ-differential privacy.
3) In order to prevent excessive noise from breaking data availability, we proposed the wavelet transform algorithm based on partition(DiffHWT Algorithm). This algorithm adds noise by the wavelet coefficient. In order to ensure the availability of published data, we get the original histogram sequence.
4) We proved the correctness of our algorithm and carried out a lot of experiments to verify our algorithm. The results show that this algorithm is superior to other algorithms in privacy protection and data availability.

II. RELATED WORK

A. DIFFERENTIAL PRIVACY MODEL

In the cloud computing model, all data needs to be uploaded to the data center through the IoT, and the privacy is easily leaked in this process. Privacy leakage may also cause the ownership of intellectual property rights, so protecting privacy is a huge challenge.

Differential privacy is a powerful privacy concept. The core idea of differential privacy is using random noise to interfere with data before it is published. So the released data cannot be traced back to the individual. This model is completely independent of background knowledge, so it has become a hot-spot in recent years research. Differential privacy model is a rigorous statistical model that greatly facilitates the use of mathematical tools as well as quantitative analysis and certification. Due to the advantages of differential privacy, it replaces the previous privacy model quickly and becomes the core of privacy research, and attracts the attention of computer science. Including data mining, machine learning and other fields. In particular, Mcsherry and Mironov provide users some behavior recommendations while satisfying differential privacy[11]. Hay *et al.* proposed an effective differential privacy algorithm for graph degree distribution[12]. Chen *et al.* proposed the differential privacy mechanism to the transaction database[13]. In summary, differential privacy is an effective privacy protection mechanism which preserves enough useful information for data analysis while protecting data privacy.

B. FOURIER TRANSFORMATION

Fourier transform is a powerful tool for analyzing signal frequency components. It can protect the privacy while maintaining the statistical characteristics of the data based on distance. But when the distribution of data sets is unknown or uneven, the threshold required in the algorithm is difficult to set. The other drawback of the Fourier transform is that if people use the Fourier transform over the entire time axis, people will not know when a particular frequency is rising. Although the short-time Fourier transform uses a sliding window to find a frequency in the spectrum graph, there is also a problem: the length of the window limits the resolution of the frequency. To solve these problems, we propose a partition histogram publishing algorithm based on wavelet transform. Our algorithm uses the wavelet transform, the wavelet transform is the development of the Fourier transform, they complement each other. The construction and result analysis of wavelet transform depend on the Fourier transform, but the wavelet transform can solve the problems in Fourier transform very well. Moreover, wavelet transform can optimize the time complexity of the algorithm and improve the execution efficiency of the algorithm.

III. PRELIMINARIES

A. DIFFRENTIAL PRIVACY

D_1 and D_2 is neighboring datasets and they differ in at most one record, differential privacy ensures that changing a single

record in the input database does not affect the output results, so as to achieve the purpose of protecting privacy. The definition of differential privacy is given below:

Definition 1(ϵ-Differential Privacy): For all adjacent data sets D_1 and D_2 with only one record difference, given privacy algorithm A, if all output O(O∈ Range(A)) satisfies the following inequality, then algorithm A satisfies ϵ-differential privacy:

$$Pr(A(D_1 = O)) \leq e^{\epsilon} \cdot Pr[A(D_2) = O] \tag{1}$$

Definition 2(Sensitivity): For a function $f : D \rightarrow R^d$, the *Sensitivity* of f is defined as:

$$\Delta f = max_{D_1,D_2} \|f(D_1) - f(D_2)\|_1 \tag{2}$$

where $\|f(D_1) - f(D_2)\|_1$ denote the distance between $f(D_1)$ and $f(D_2)$, d identifies the query dimension of f, R represents the mapped real space. *Sensitivity* is only relevant to the query function.

The main implementation mechanism of differential privacy is noise disturbance mechanism. Laplace mechanism and exponential mechanism are the two most commonly used mechanisms. Where the Laplace mechanism deals with numerical data, and the exponential mechanism mainly deals with non-numerical data.

Definition 3 (Laplace Noise): Laplace mechanism achieves ϵ-differential privacy by adding noise into the original query value which is obeyed the Laplace distribution. The Laplace distribution is also known as Double Exponential distribution. The frequency function of the Laplace distribution is:

$$f(x|\mu, b) = \frac{1}{2b} exp(-\frac{|x - \mu|}{b}) \tag{3}$$

Theorem 1 (Laplace Mechanism): For a dataset D and privacy budget ϵ, the sensitivity of function f:$D \rightarrow R^d$ is Δf, the condition for f to satisfy Laplace mechanism is:

$$L(D) = f(D) + Lap(\Delta f/\epsilon) \tag{4}$$

Lap(λ) indicates the positional parameter of Laplace distribution is 0, and the scale parameter is λ, so the scale parameter of Laplace mechanism is $\Delta f/\epsilon$.

Theorem 2 (Exponential Distribution): The input of random algorithm A is dataset D and output is an entity r∈ ξ, $\xi$is the set of output results, *u(D,r)* is the availability function, the sensitivity of it is Δu. If algorithm A select output r from and the probability is proportional to exp($\frac{\epsilon \cdot u(D,r))}{2\Delta u}$), then we can say algorithm A provides ϵ-differential privacy protection.

Where the *sensitivity* of u(D,r)→R is

$$\Delta u = max_{D_1,D_2=nbds(D)} \|u(D_1, r) - u(D_2, r)\| \tag{5}$$

Usually, a practical problem requires the combination of multiple privacy algorithms, which requires the differential privacy sequence composition.

Theorem 3 (Sequence Composition): Suppose there is a set of algorithms $A_1(D), A_2(D), \ldots, A_n(D)$ satisfy differential privacy, the corresponding privacy budget is $\epsilon_1, \epsilon_2, \ldots \epsilon_n$ respectively. For data set D, after the new algorithm combined by the above algorithm $A(A_1D, A_2D, \ldots A_nD)$, algorithm A can provide $\sum_{i=1}^{n} \epsilon_i$ differential privacy. This property indicates a combined algorithm consisting of a sequence of differential privacy protection models whose privacy protection level is the sum of all budgets[14].

B. WAVELET TRANSFORM

Wavelet transform is based on finite-duration wavelet, which can better represent the signal by analyzing the resolution. The time complexity of Fourier transform is *O(NlogN)*, The time complexity of wavelet transform is *O(N)*(where *N* is the size of the data). By divide-and-conquer method, the problem of time complexity of matrix multiplication is optimized. For the Fourier transform, the number of coefficients of each recursive polynomial is *N/2*. That is to say, the first level of recursion requires *N/2* multiplication operations, and it will run *N* times. The second level of recursion requires *N/4* multiplication operations, and it will also run *N* times... By analogy, the recursive depth of halving the recursive scale of each layer is *logN*, so the time complexity of Fourier transform is *O(NlogN)*. Among them, Haar wavelet transform is a relatively simple method of wavelet transformation. Combining with the methods of Dwork *et al.*, it can be known that it protects privacy by adjusting the frequency matrix M of the input data. Dwork's method is to directly inject noise into M. And after combining with Haar wavelet, the frequency matrix M firstly transformed into another matrix M, then inject noise into matrix M, and finally convert M back to the noise-added frequency matrix M^*.

In [15], the overall framework of Haar wavelet transform is proposed. We focus on one-dimensional wavelet transform which can effectively process statistical data. One-dimensional Haar wavelet transform needs to input counting values, suppose its number is k. And then convert them into 2^h counting values by inserting false counting values[16]. Then calculate the wavelet coefficients, which is the basic coefficient c_0 and other coefficients $c_i (i \in [1, l])$, when $t = \lfloor log_2^k \rfloor, l \leq k \leq 2^h$

$$\begin{aligned} c_0 &= \sum_{i=1}^{2^h} v_i \\ c_k &= \sum_{i=l+(k-2^t*2^{h-t})}^{(2k-2^{t+l}+l)*2^{h-t-l}} v_i - \sum_{i=l+(2k-2^{t+1}*2^{h-t-l})}^{(k-2^t+l)*2^{h-t}} v_i \end{aligned} \tag{6}$$

For example, for a histogram count set $M = \{9, 3, 6, 2, 8, 4, 5, 7\}$, the Haar wavelet transform is more intuitively represented by constructing a full binary tree R, where the square represents count value and the circle represents wavelet coefficients. The calculation from bottom non-leaf node to the root node is as follows: $c_4 = (v_1 - v_2)/2$, $c_2 = ((v_1 + v_2) - (v_3 + v_4))/4$, $c_1 = ((v_1 + v_2 + v_3 + v_4) - (v_5 + v_6 + v_7 + v_8))/8$, the basic coefficient $c_0 = (v_1 + v_2 + v_3 + v_4 + v_5 + v_6 + v_7 + v_8)/8$, other wavelet coefficients are shown in Fig 1. M can be reconstructed

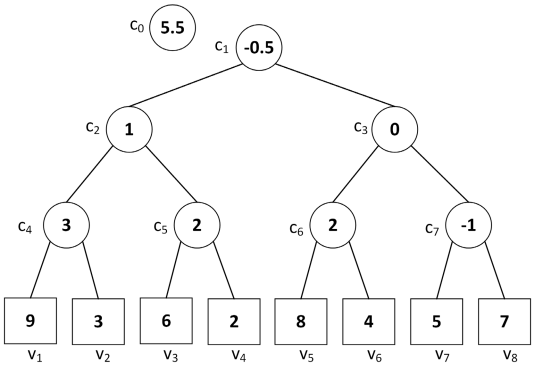


FIGURE 1. One-dimensional haar wavelet transform.

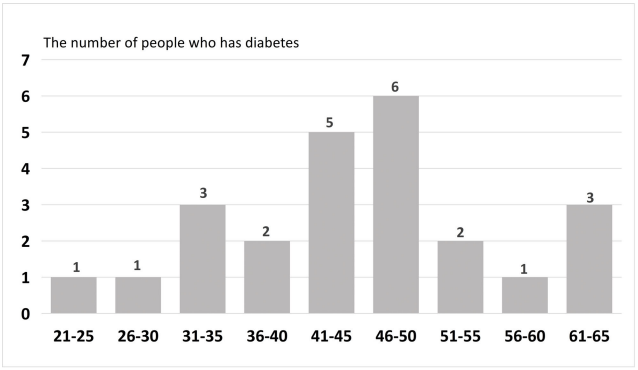


FIGURE 2. Age distribution histogram of diabetic patients.

TABLE 1. Medical records of diabetic patients.

Name	Age	Have diabetes or not
Alice	23	Yes
Bob	37	Yes
Chris	34	Yes
Carol	47	Yes
Daniel	34	No
Darnell	52	No

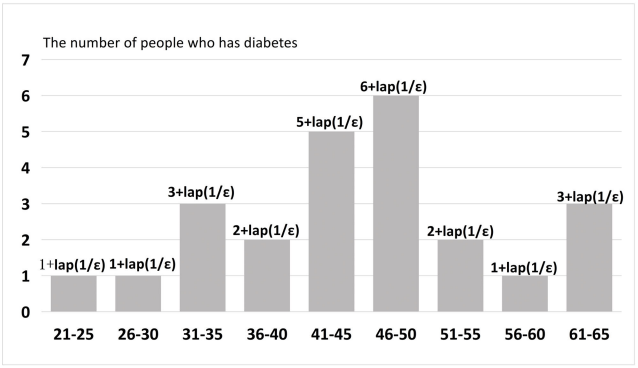


FIGURE 3. Age distribution histogram of diabetic patients.

through nodes in the hierarchy tree, $v_i = c_0 + \sum_{i=1}^{l}(g_i \cdot c_i)$, v is each element in M, where $c_i (i \in [1, l])$ is the ancestor of v in i-th layer. If v is the left child of c_i, so g_i is equal to 1, if v is the right child of c_i, so g_i is equal to -1.For example, leaf nodes v_2 has three ancestor nodes $c_1 = -0.5$, $c_2 = 1$, $c_4 = 3$, and v_2 is the right child of v_4, the left child of c_1 and c_2, $c_0 = 5.5$, so $v_2 = c_0 + c_1 + c_2 - c_4 = 3$. The other count values are equally available. Therefore, the original count value can be obtained by wavelet coefficients.

C. DATA RELEASE BASED ON HISTOGRAM

Histogram is mainly used to describe statistical data. We consider the scene of hospital patient data collected by edge devices. The distribution of diabetes is given in TABLE 1. Fig. 2 generates disjoint equal-width histogram corresponding to TABLE 1. The values in each histogram interval correspond to the number of people with diabetes in the corresponding age group in TABLE 1. After publication, it can be easily used for linear query and research.

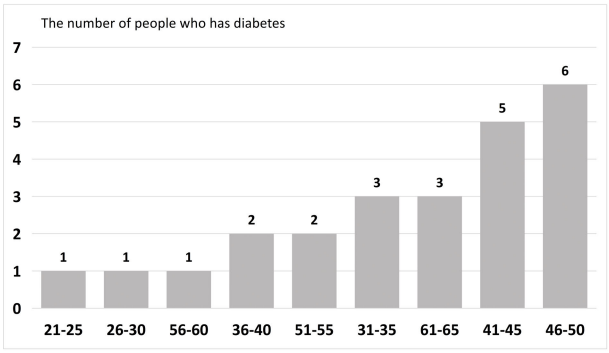
The histogram of Fig. 2 gives statistical data on whether people have diabetes or not, but it may leak the disease information of patients. For example, the attacker already knew that Chris was 34 years old and wanted to know if he had diabetes. Suppose the attacker already knows the condition of the two other people in interval [31], [35] except Chris, then through [31], [35] with the count of 3, it can be inferred that Chris has diabetes and the patient's privacy issues are compromised. Dwork added noise to each histogram's interval which is obeyed the Laplace distribution[9]. Its idea is shown in Fig. 3.

Taking Fig.2 as an example, when the ages between [20], [25] and [26]–[30] are combined, the query range is expanded by twice than before. And that is long-range count query. For the long-range count queries, the above publishing method accumulates too much noise, making the range count query results unavailable. The availability of query results is poor due to excessive noise accumulation. So we need to balance data privacy and data availability to make the final results are available. Fig. 4 shows the algorithm steps proposed in [15]. Although this method improves the availability of each interval, the noise amount of the long-range count query still increases linearly. In addition, [17] use Haar wavelet transform to reduce the noise amount of long-range count query. Although this method can reduce the noise level, since this method need to construct the wavelet tree structure for the original partition, and usually the number of histogram intervals is large. So the number of intermediate nodes correspond to the wavelet tree may be exponential, which turns out the query results are poor[18].

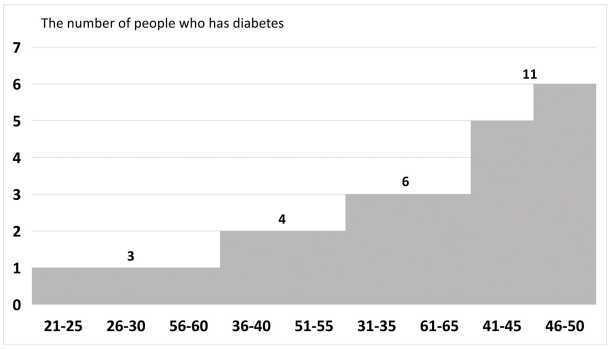
IV. PARTITION HISTOGRAM PRIVACY PROTECTION

A. SYSTEM ARCHITECTURE

Fig. 5 is the model of privacy protection architecture under edge computing. Smart mobile devices such as sensors and smartphones are the producers and consumers of data that is sent to **edge computing platforms**. **Edge computing**



(a) First Step: Partitioning



(b) Second Step: Add Noise

FIGURE 4. Example of partition scheme for histogram publication.

platform is not only the data transfer center, but also the data encryption center. It can add noise to the data, so that it can keep data encrypted between the edge device platform and the cloud server. This improves the ability of the **edge device platform** to protect private. In the data transmission process, we obtain histograms from statistical data, then we use our proposed algorithm to protect privacy. Our algorithm solves the noise accumulation problem. And the complexity of the wavelet tree structure after wavelet transform of original histogram is also solved.

Our algorithm has two aspects: on the one hand, it reduces the number of histogram intervals by partitioning algorithm; on the other hand, the wavelet transform is used to reduce the noise added in each interval. The specific steps of the algorithm are as follows: first of all, we partitioned the original histogram by aggregating similar count values. Then the wavelet tree is constructed according to the partitioned histogram and noise is added to the wavelet coefficient. After that we use the wavelet tree to get the reductive partition. Finally, adjust the order of each partition and revert to the original histogram.

In the whole process, the privacy budget needs to be divided into two parts, one part is added to the original histogram, and the other part is added to the wavelet coefficient of the wavelet tree. The reason why the original histogram interval needs to add noise is that in the limit case, sorting the original values in the histogram would destroy the differential privacy mechanism[19]. Therefore, a small amount of noise should be added to the original histogram. Our privacy budget

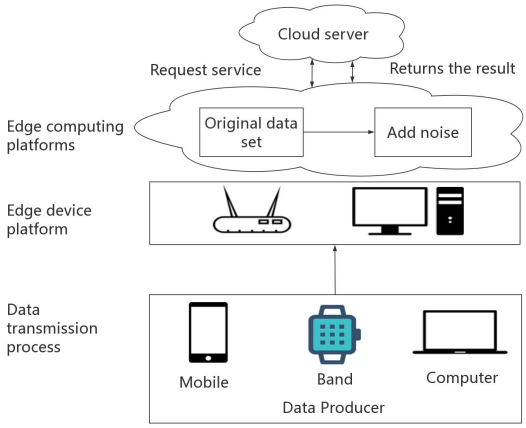


FIGURE 5. Privacy protection architecture under edge computing.

is mainly used in adding noise to the wavelet tree. And it is the second part. So we only need to consider the first part of the privacy budget in the algorithm and experiment part.

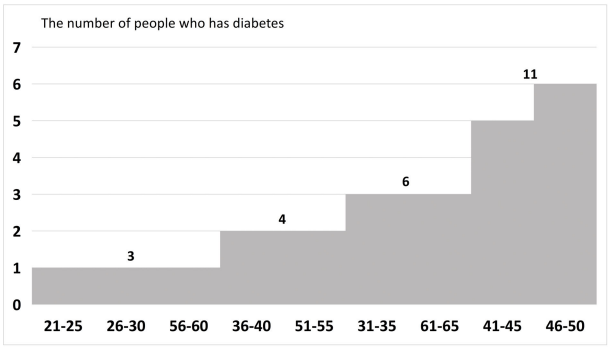
Fig. 6 shows the steps of the wavelet transform partitioned histogram algorithm. We sort the histograms value to form each partition. In order to do that, we use the clustering method to merge the histograms with similar values. Afterwards we construct the wavelet tree and calculate the wavelet coefficients. Then we add noise to the wavelet coefficients and restore it to obtain the noise value. Finally we disperse the noise evenly and to get the reductive histogram partition. We will give specific instructions for each step in the next section.

B. PARTITION HISTOGRAM ALGORITHM BASED ON WAVELET TRANSFORM

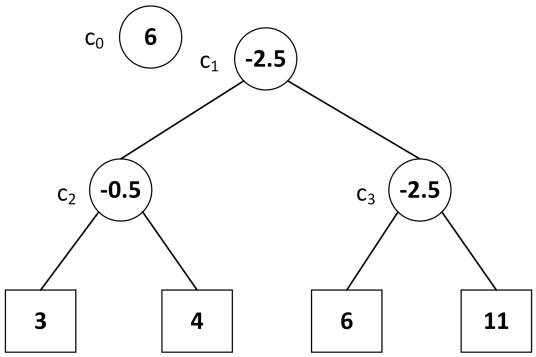
This section introduces the overview of the partition histogram algorithm and the specific implementation details of the algorithm.

1) PH_WT ALGORITHM

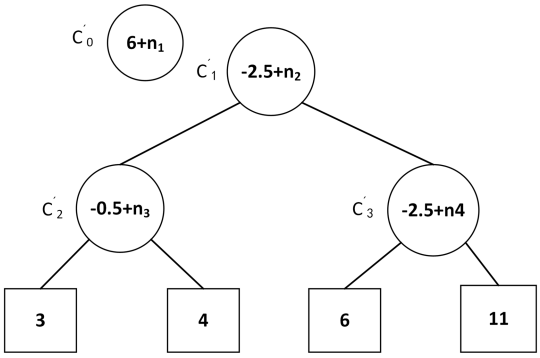
For dataset $D = \{x_1, x_2, \ldots x_n\}$, the original histogram is $H = \{h_1, h_2, \ldots h_n\}$, $h_i (1 \leq i \leq n)$ is a certain interval of the histogram, and its corresponding count value is x_i, in which $|h_i| = x_i$. The first step of the algorithm is to add Laplace noise to the original histogram H and sort it to get the histogra $\tilde{H} = \{\tilde{h_1}, \tilde{h_2}, \ldots \tilde{h_n}\}$ with added noise, where the noise added in the first part is ϵ_1. Then, cluster partition operation is performed on $\tilde{H}$ to obtain the new partition histogram $H' = \{p_1, p_2, \ldots p_k\}$. Among them $p_i (1 \leq i \leq k)$ represents one of the partitions and represents it as a triple $p_i(l_i, r_i, v_i)$. l_i is the left boundary of partition p_i, r_i is the right boundary of partition p_i, and $v_i = \frac{1}{r_i - l_i + 1} \cdot \sum_{i=l_i}^{r_i} |h_i| (1 \leq l_i, r_i \leq n)$ is the average value of the partition. The valid criterion for partition H' is that all intervals of $\tilde{H}$ are contained in H' without partition coverage, which means that p_i and $p_j (i \neq j \quad and \quad 1 \leq i, j \leq k)$ must not coincide, and $l_i = 1, l_k = n$, $l_j = r_{j-1} + 1 (2 \neq j \neq k)$. After that is to do Haar wavelet transform on $H' = \{h_1', h_2', \ldots h_n'\}$, add noise to the wavelet coefficient by constructing the wavelet tree, and then restore



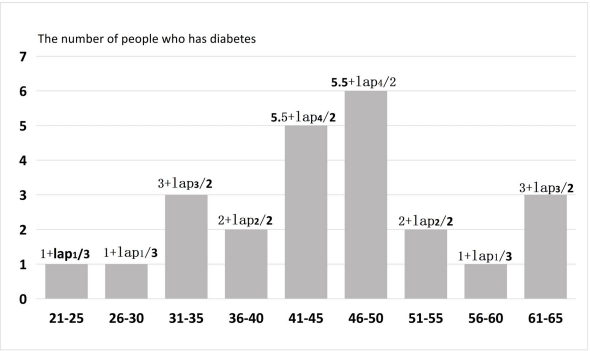
(a) First Step: Sorting and Partitioning



(b) Second Step:Build a Wavelet tree



(c) Third Step:Add Noise to Wavelet tree



(d) Final Step:Equalize Noise & Restore Partition

FIGURE 6. Partition histogram algorithm based on wavelet transform.

the noise-added histogram $H^*\{p_1^*, p_2^* \dots p_k^*\}$.And finally is to restore each partitions back to the original histogram order to get $\bar{H}$ for final release.

Before introducing the algorithm of distributing histogram of partition based on wavelet transform, we first introduce a standard of quantization partitioning: *Sum of Squared Error*(*SSE*). *SSE* is used to describe the similarity between $\tilde{H} = \{\tilde{h_1}, \tilde{h_2}, \dots \tilde{h_n}\}$ and $H' = \{p_1, p_2, \dots p_k\}$, and it is calculated as follows:

$$SSE(H', \tilde{H}) = \sum_{i=1}^{k} \sum_{l_i \le j \le r_i} (v_i - |h_j|)^2 \tag{7}$$

The smaller the $SSE(H', \tilde{H})$ is, the smaller the error generated by the partition, and the better the partitioning effect. Combining $SSE(H', \tilde{H})$ with each parameter of histogram p_i, *SSE* for each interval is calculated as follow:

$$\begin{aligned} SSE(p_i) &= SSE(\sum_{j=l_i}^{r_i}(\tilde{h_j} - v_i - \frac{Lap(1/\epsilon_2)}{(r_j - l_j + 1)})^2) \\ &= \sum_{j=l_i}^{r_i}(\tilde{h_j} - v_i)^2 - 2 \cdot (r_i - l_i + 1) \cdot \frac{Lap(1/\epsilon_2)}{r_i - l_i + 1} \cdot \\ &\sum_{j=l_i}^{r_i}(\tilde{h_j} - v_i) + (r_i - l_i + 1) \cdot (\frac{Lap(1/\epsilon_2)}{r_i - l_i + 1})^2 \\ &= \sum_{j=l_i}^{r_i}(\tilde{h_j} - v_i)^2 - 2 \cdot Lap(1/\epsilon_2) \cdot \\ &\sum_{j=l_i}^{r_i}(\tilde{h_j} - v_i) + (\frac{(Lap(1/\epsilon_2))^2}{r_i - l_i + 1}) \\ &= \sum_{j=l_i}^{r_i}(\tilde{h_j} - v_i)^2 + \frac{2}{(r_i - l_i + 1) \cdot \epsilon^2} \end{aligned} \tag{8}$$

The description of PH_WT algorithm is as follows. The input of the algorithm is dataset D and the privacy budget ϵ, and the output is transformed histogram $\bar{H}$. We first divided the privacy budget ϵ into two parts. ϵ_1 is used to add a small amount of noise to the original histogram to prevent the original histogram from leaking privacy. ϵ_2 is used in the wavelet transform process. Then generate a histogram and add noise to the original histogram $\tilde{H}$. Sort the histogram $\tilde{H}$ after adding noise. The partition algorithm *Partition_Histogram*$(\tilde{H}, \epsilon_2)$ executed to complete histogram partitioning. The input ϵ_2 is only used to estimate the *SSE* and is not actually used in the histogram. We then execute Haar wavelet transform algorithm *DiffHWT*(H', k, ϵ_2) to add noise. Then adjust the histogram to its original order and return to the final histogram $\bar{H}$ to publish. After using the *Partition_Histogram*$(\tilde{H}, \epsilon_2)$, the noise variance of range query changes from $O(k/\epsilon_2)$ to $O((log_2^m)^3/\epsilon^2)$, and the optimization effect is obvious in the long-range count query.

2) PARTITION_HISTOGRAM ALGORITHM

The Partition_Histogram Algorithm represents the histogram-based partitioning Algorithm. As mentioned above, when

Algorithm 1 PH_WT Algorithm

Input: dataset D, privacy budget ϵ
Output: sanitized histogram $\bar{H}$

split ϵ into ϵ_1 and ϵ_2 with $\epsilon_1 + \epsilon_2 = \epsilon$
generate histogram H according to D
add $Lap(1/\epsilon_1)$ in each bin of H, generate $\tilde{H}$ //add Laplace noise to every bin in H
sort bins in $\tilde{H}$ by their counts
H' and k equal to the return value of *Partition_Histogram*($\tilde{H}, \epsilon_2$)
$H^* = DiffHWT(H', k, \epsilon_2)$
reorder bins order to get $\bar{H}$
return $\bar{H}$

a long-range count query is performed, the query scope is doubled, and the amount of noise added to the region is reduced to Lap(1/2ϵ). That is, if the range is merged by n ranges, the query sensitivity of the range becomes 1/n, and the noise amount becomes Lap(1/nϵ). Therefore, the purpose of partitioning is to reduce noise errors caused by long-range count queries. The main idea of the algorithm is to optimize the partition by calculating the *SSE*. Suppose a partition $p_i = (l_i, r_i, v_i)$is already built, then there are two cases in the next histogram interval: the first is to merge into the current partition p_i, and the second is to merge into the next partition p_{i+1}. The specific criterion is the *Sum of Squared Error*: $SSE(p_i)$.The corresponding formula is calculated as: in the first case the *SSE* is $SSE(p_i \cup h_{r_i+1})$, in the second case, the *SSE* is $SSE(p_i) + SSE(h_{r_i+1})$. Where the value of $SSE(h_{r_i+1})$ cannot be determined, so we use the quantitative method proposed in [20] to consider two extreme cases:the first one is that let h_{r_i+1} becomes a single interval, and the second one is that let $h_{r_i+1}, h_{r_i+2}, \ldots, h_n$ merge into the same partition p_{i+1},and calculate the corresponding $SSE^*(h_{r_i+1}) \leq \frac{2}{(n-j+1)^2 \cdot \epsilon^2}$.So the *SSE* calculation formula for the second case is $SSE(p_i) + \frac{2}{(n-j+1)^2 \cdot \epsilon^2}$.After all the partition are divided, we store the new partition into the set P, and then partition the histogram to get H'.

3) DIFFHWT ALGORITHM

DiffHWT Algorithm represents a partition-based wavelet transform algorithm.The algorithm first converts the counting values into the frequency vector v which had k partitions in $H' = \{h'_1, h'_2, \ldots h'_k\}$, and k count values are converted into 2^h count values. Then calculate the wavelet coefficients, which are the basic coefficient c_0 and the other coefficients $c_i (i \in [1, l])$. Next, add noise to the wavelet coefficients, the scale parameters of the Laplace noise added to c_i are $\frac{h+1}{\epsilon_2 \cdot W_{Haar}(c_i)}$, where the sensitivity of the wavelet transform is $\Delta f = h + 1$, $W_{Haar}$is the weight function. Which is defined as follows. For c_0, $W_{Haar}(c_0) = k$. For the coefficient of the i-th layer $c_i (i \in [1, l])$, $W_{Haar}(c_i) = 2^{h-i+1}$. Using the noise-added wavelet coefficient can get the reductive frequency

Algorithm 2 Partition_Histogram($\tilde{H}, \epsilon_2$)

Input: histogram H with n bins, privacy budget ϵ_2
Output: histogram H' with k partitions

P = $\varnothing$
i = 1
add h_1 to p_1
for j from 2 to n **do**
  $V_1 = SSE(p_i \cup h_j)$
  $V_2 = SSE(p_i) + \frac{2}{(n-j+1)^2 \cdot \epsilon^2}$
  if $V_1 < V_2$ **then**
    $p_i = p_i \cup h_j$
  else
    add p_i to P
    i = i + 1
    add h_j to p_i
  end if
end for
add p_i to P
merge bins in H according to P
return H' with the number of partition of i

vector $\tilde{v_i}$:

$$\tilde{v_i} = \frac{1}{2^h}\tilde{c_0} + \sum_{k=0}^{h-1} \frac{(-1)^{\lfloor (i-1)/2^{h-k-1} \rfloor mod 2}}{2^{h-k}} \tilde{c_{2^k}} + \left\lfloor (i-1)/2^{h-k} \right\rfloor \tag{9}$$

Finally, we get the reductive noise-added histogram H^*. We need to save the boundary of each interval in advance. The reason why we need these intervals is that it can help us to get the original histogram order.

Algorithm 3 DiffHWT(H',k,ϵ_2)

Input: histogram H' with k bins, privacy budget ϵ_2
Output: histogram H^*

map H' to its frequency vector v
let the number k of v equals 2^h by inserting dummy counts
for i from 0 to 2^h **do**
  compute the HWT wavelet coefficients c_i of v
  add Laplace noise to coefficient c_i with magnitude $\frac{1+log_2^k}{\epsilon_2 \cdot W_{Haar}(c_i)}$
end for
for i from 0 to 2^h **do**
  convert the noisy coefficient back to a noisy count $\tilde{v_i}$
  add $\tilde{v_i}$ to noisy frequency vector $\tilde{v}$
end for
adjust bins count to get H^* by $\tilde{v}$
return H^*

4) PRIVACY ANALYSIS

Using Theorem 3, we divided the privacy budget into two parts, ϵ_1 and ϵ_2. Where ϵ_1 is used to add Laplace noise

to the original histogram, so that this step satisfies the ϵ_1-differential.

In the process of wavelet transformation, the sensitivity of the weight function is $1 + log_2^k$, where k is the number of partitions obtained by *Partition_Histogram*($\tilde{H}$,ϵ_2), and the scale parameter of the Laplace distribution is $\frac{1+log_2^k}{\epsilon_2 \cdot W_{Haar}(c_i)}$. This step satisfies the ϵ_2-differential privacy.

In summary, the partition histogram publishing algorithm based on wavelet transform satisfies ϵ- differential privacy.

V. EXPERIMENT

In this section, we implement PH_WT algorithm through a large number of experiments, and evaluate its performance.

A. EXPERIMENT SETUP

1) EXPERIMENT PLATFORM

Table 2 shows the relevant configuration information of the experimental platform.

TABLE 2. Configuration of Experiment.

Hardware and software	Configuration
Processor	Intel @ Xeon @ CPU E5-2670 2.27 GHz
Hardware	1TB Mechanical hard disk
Network card	Intel 82551 10M/100M Adaptive network card
OS	Ubuntu Server 16.04.1 LTS

2) EXPERIMENT DATABASE

Since our two data release algorithms are applied to different datasets, we select three numerical datasets to experiment. In order to verify our proposed algorithm, we selected three datasets to verify it.

a: SEARCH LOGS

This dataset is obtained by Google Trend data and AOL search log. 32,768 records were included, and each record was searched for the keyword "Obama" every 90 minutes between January 1, 2004, and August 9, 2009 (ranging from 1 to 496). Differential privacy ensures that an attacker cannot know whether a user searches the keyword "Obama" at a certain time.

b: AGE

This dataset extract Brazilian census data from the Population Database IPUMS (Integrated Public Use Microdata Series), which contains 100,078,675 records, each of which stores the age of a citizen (ranging from 0 to 101). Differential privacy ensures that an attacker cannot infer the real age of any citizen from the published statistics. It needs to be specially noted that these data set has no "statistical" quality in the single record, so we use the 5-year-old histogram interval to count the number of citizens in the corresponding age group.

c: LOCATION

This dataset is a compilation of the New Zealand demographic data (New Zealand, 2006). It is divided into 7,725 blocks by the location of the 186,471 people living in the Waitakere area. Differential privacy ensures attackers cannot infer the address of any resident from the published statistics data.

We chose these three data sets because they were received by edge devices. And they were the result of the IoT projects. These collected data can be used for data analysis, data aggregation and data filtering. *Search Logs* can provide users with what they want to know. And push the data by priority after data analysis. Through *Age* we can obtain the distribution of the population, and then the analysts can provide different services based on the needs of different age groups. *Location* can be used for transport planning and other smart city construction. Then people can make better use of local resources. Besides, the sparsity of these three datasets is different, and the functions of these datasets are different.

B. EVALUATION METRICS

Differential privacy is mainly achieved by adding noise to the original calculation value. This mechanism serves two purposes, one is to protect the privacy, and the other is to publish results that are still available. Therefore, we quantify these two aspects through various indicators and parameters, hoping to achieve a balance.

1) MEAN-SQUARED ERROR

For histogram statistical data publishing algorithm, we measure the accuracy of published data by calculating the *Mean-squared error*(*MSE*) of range query. The specific method is to perform a fixed window size range query on the released histogram, and obtain the value x_1'. On the original histogram, the range query of the same window is obtained by x_1, and then the window size is not changed. Based on changing the starting point of the range query, we can get two other values x_2' and x_2. For example, we do a range query on the *Age* with a window size of 4, so that we can get the number of citizens aged 1-5, the number of citizens aged 2-6, etc., that is, we can get 97 pairs (x_i', x_i). For the range query with the window size *len*, the *MSE* of it is:

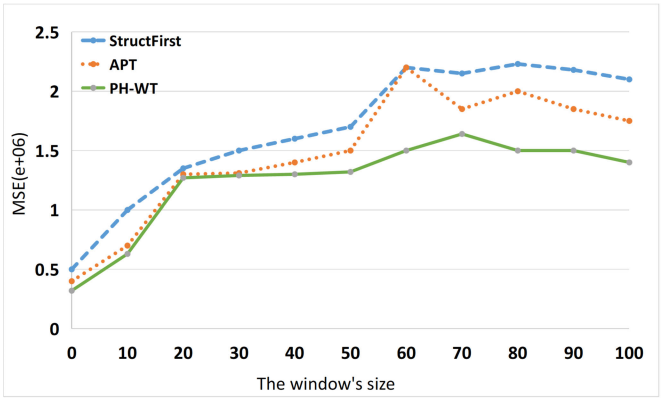
$$MSE_{len} = \sum_{i=1}^{n} (x_i', x_i)^2 \tag{10}$$

where n is the number of the pairs (x_i', x_i) when the window size of range query is *len*. The smaller the value of *MSE*, the closer the published data gets to the original data.

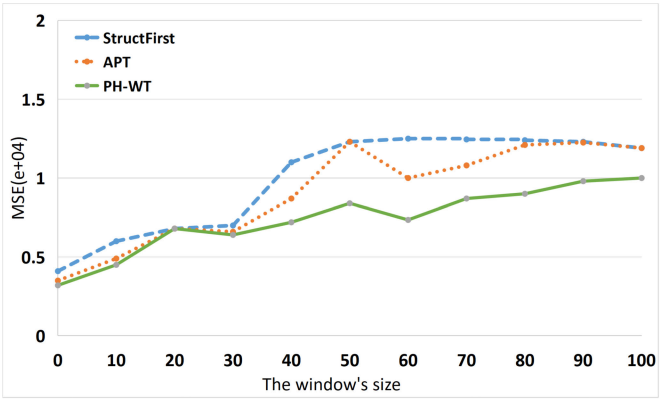
2) KL DIVERGENCE

KL Divergence(*KLD*):KL Divergence is also called relative entropy. The calculation formula of *KLD* is:

$$D_{KLD}(H||\bar{H}) = \sum_{i=1}^{n} H_i ln(\frac{H_i}{\bar{H}_i}) \tag{11}$$

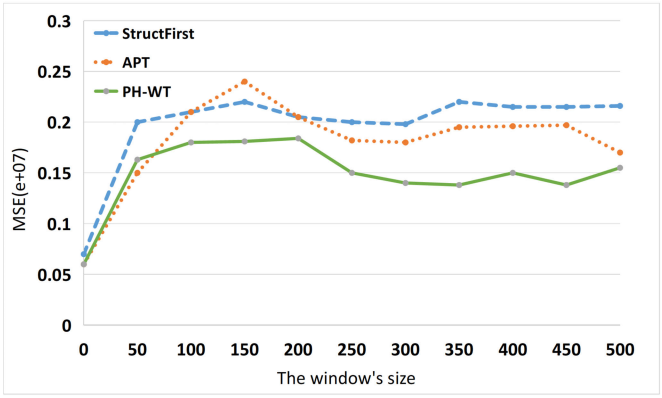


(a) Experiment Result on Dataset *Age* with ϵ=0.01

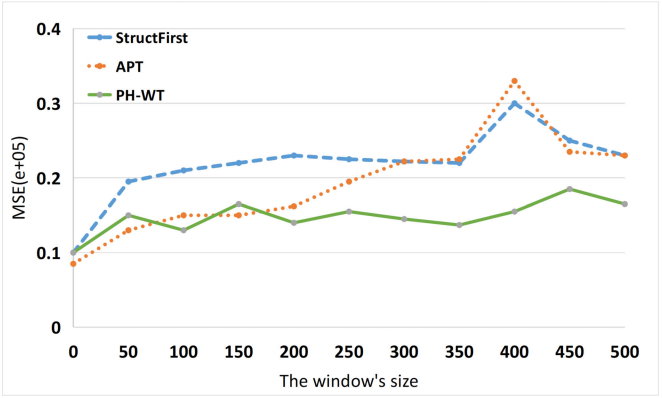


(b) Experiment Result on Dataset *Age* with ϵ=0.1

FIGURE 7. **Experimental results were compared on *Age* with different ϵ.**



(a) Experiment Result on Dataset *Search logs* with ϵ=0.01



(b) Experiment Result on Dataset *Search logs* with ϵ=0.1

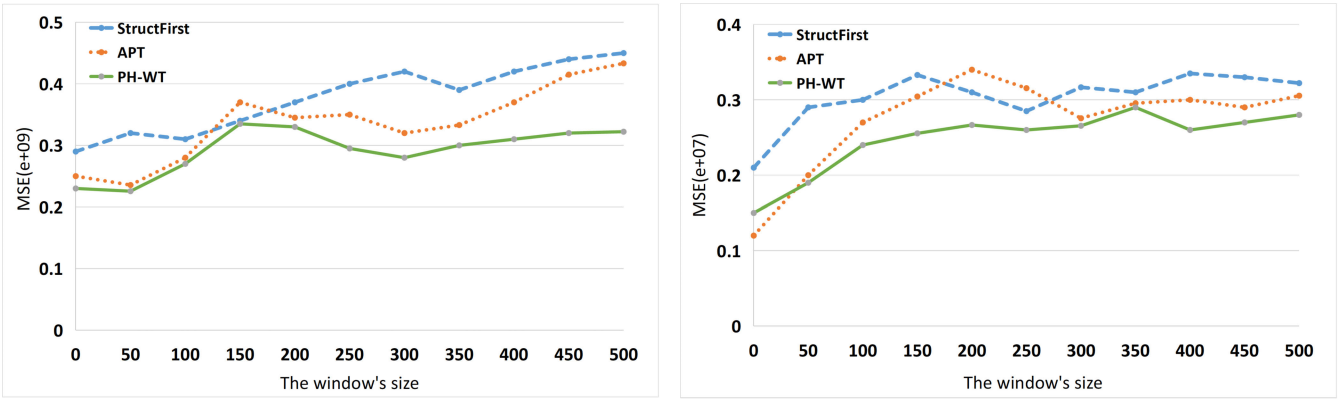
FIGURE 8. **Experimental results were compared on *Search logs* with different ϵ.**

where H is the original histogram, $\bar{H}$ is the published histogram. n is the window size of the range query. *KLD* represents the difference between two functions or probability distribution: the greater the difference, the greater the relative entropy, and vice versa. In particular, if $H = \bar{H}$ then entropy is zero. *KLD* is used to describe the expectation of the probability distribution between the original histogram and the published histogram on the logarithmic difference. Which means, the smaller the *KLD* is, the similar the probability distribution of the original histogram and the released histogram is, and the higher the accuracy of data publication is.

C. EXPERIMENTAL RESULTS AND ANALYSIS

We propose partition histogram publishing algorithm based on wavelet transform, which is mainly used differential privacy to publish numerical data. In the following experiment, StructureFirst represents the algorithm proposed in [21], where the number of partitions is set as k = n/10, and n is the number of intervals of the histogram. APT is an algorithm proposed in [8], PH_WT is our proposed algorithm. The privacy budget of our approach will be divided into two parts, one for noise adding and the other for protecting the partition structure. Based on a large number of experimental, it is confirmed that when the ratio of two parts is 2:1, it can get better privacy protection. The current value of ϵ usually is 0.01, 0.1 and 1, representing three privacy protection degrees. Since attackers are easy to attack the statistical data, we choose ϵ as 0.01 and 0.1 to increase the protection degree of statistical data, and the experiment is performed on the three data sets respectively to obtain the comparison results shown in Fig. 7 to Fig. 9.

Fig. 7(a) and Fig. 7(b) are the experimental results of *Age* when $\epsilon = 0.01$ and $\epsilon = 0.1$, respectively. Fig. 8(a) and Fig. 8(b) are experimental results of the *Search Logs* when $\epsilon = 0.01$ and $\epsilon = 0.1$. Fig. 9(a) and Fig. 9(b) are experimental results of the *Location* when $\epsilon = 0.01$ and $\epsilon = 0.1$. In general, when the window of the PH_WT algorithm is small, the result is close to APT algorithm. However, with the increase of the window, the *MSE* of the PH_WT algorithm is significantly smaller than the other two algorithms. The reason is that the window size determines the number of wavelet coefficients for constructing the wavelet tree. As the window increases, the number of histogram interval



(a) Experiment Result on Dataset *Location* with ϵ=0.01

(b) Experiment Result on Dataset *Location* with ϵ=0.1

FIGURE 9. Experimental results were compared on ***Location*** with different ϵ.

TABLE 3. KLD applied on three types of data sets($\epsilon = 0.01$).

Dataset	StructureFirst	APT	PH_WT
Age	2.821	2.134	0.635
Search logs	6.201	3.289	0.879
Location	5.105	2.625	0.621

TABLE 4. KLD applied on three types of data sets($\epsilon = 0.01$).

Dataset	StructureFirst	APT	PH_WT
Age	2.107	1.972	0.498
Search logs	3.522	2.671	0.224
Location	2.915	1.623	0.324

decreases, and the number of wavelet coefficients decreases. When the histogram is restored, the accumulated noise is reduced, so the experimental effect is better when the window is larger. And in general, the *MSE* of Fig. 7(b), Fig. 8(b) and Fig. 9(b) is smaller than that of Fig. 7(a), Fig. 8(a) and Fig. 9(a) respectively. The reason is that the increase of ϵ reduces the protection degree, and then the whole noise is reduced, and finally the *MSE* is smaller.

After that, we prove the accuracy of our algorithm through *KLD*. Table 3 and Table 4 show the *KLD* of three data sets under StructureFirst, APT and PH_WT with different privacy budget($\epsilon = 0.01, 0.1$). It can be observed from the table that under the same privacy protection level, the proposed PH_WT algorithm has smaller *KLD* than existing StructureFirst and APT algorithms. In other words, the PH_WT algorithm improves the accuracy of data release.

VI. CONCLUSION

In this paper, we first study the problem of designing a partitioned histogram. Focusing on data privacy protection, we present a partition histogram algorithm based on wavelet transform called PH_WT, which consists of a construction of the wavelet tree structure phase and a noise-adding phase.Through the privacy analysis, we prove that our PH_WT algorithm satisfies ϵ-differential privacy. And the extensive experimental results verify that our algorithm can achieve higher accuracy while providing rigorous privacy guarantee. We believe that the design rationale and the solutions developed in this paper will motivate more research endeavors on privacypreserving edge computing.

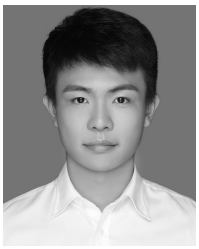
REFERENCES

[1] X. He, J. Liu, R. Jin, and H. Dai, "Privacy-aware offloading in mobile-edge computing," in *Proc. GLOBECOM*, Dec. 2017, pp. 1–6.
[2] J. Zhang, B. Chen, Y. Zhao, X. Cheng, and F. Hu, "Data security and privacy-preserving in edge computing paradigm: Survey and open issues," *IEEE Access*, vol. 6, pp. 18209–18237, 2018.
[3] M. Du, K. Wang, Y. Chen, X. Wang, and Y. Sun, "Big data privacy preserving in multi-access edge computing for heterogeneous Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 62–67, Aug. 2018.
[4] M. Hay, V. Rastogi, G. Miklau, and D. Suciu, "Boosting the accuracy of differentially private histograms through consistency," in *Proc. 36th Conf. Very Large Databases (VLDB)*, Istanbul, Turkey, 2010, pp. 1021–1032.
[5] C. Dwork, *Differential Privacy* (Lecture Notes in Computer Science), vol. 26, no. 2. 2006, pp. 1–12.
[6] Y. Xiao *et al.*, "DPCube: Differentially private histogram release through multidimensional partitioning," *Trans. Data Privacy*, vol. 7, no. 3, pp. 195–222, 2014.
[7] Y. Xiao, L. Xiong, and C. Yuan, "Differentially private data release through multidimensional partitioning," in *Proc. 7th VLDB Conf. Secure Data Manage. (SDM)*, Singapore, 2011, pp. 150–168.
[8] J. Xu, Z. Zhang, and X. Xiao, "Differentially private histogram publication," in *Proc. IEEE 28th Int. Conf. Data Eng.*, Washington, DC, USA, Apr. 2012, pp. 32–43.
[9] J. Xu, Z. Zhang, X. Xiao, Y. Yang, G. Yu, and M. Winslett, "Differentially private histogram publication," *Int. J. Very Large Data Bases*, vol. 22, no. 6, pp. 797–822, Dec. 2013.
[10] X. Xiao, G. Wang, and J. Gehrke, "Differential privacy via wavelet transforms," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 8, pp. 1200–1214, Aug. 2011.
[11] F. Mcsherry and I. Mironov, "Differentially private recommender systems: Building privacy into the netflix prize contenders," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jun. 2009, pp. 627–636.
[12] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in *Proc. 9th IEEE Int. Conf. Data Mining*, Dec. 2009, pp. 169–178.
[13] R. Chen, N. Mohammed, B. C. Fung, B. C. Desai, L. Xiong, "Publishing set-valued data via differential privacy," *Proc. VLDB Endowment*, vol. 4, no. 11, pp. 1087–1098, Aug. 2011.
[14] P. Kairouz, S. Oh, and P. Viswanath, "The composition theorem for differential privacy," *IEEE Trans. Inf. Theory*, vol. 63, no. 6, pp. 4037–4049, Jun. 2015.
[15] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler, "GUPT: Privacy preserving data analysis made easy," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, May 2012, pp. 349–360.

[16] E. J. Stollnitz, T. D. Derose, and D. H. Salesin, *Wavelets for Computer Graphics: Theory and Applications*. San Mateo, CA, USA: Morgan Kaufmann, 1996, pp. 50–60.
[17] I. Roy, S. T. V. Setty, A. Kilzer, V. Shmatikov, and E. Witchel, "Airavat: Security and privacy for MapReduce," in *Proc. 7th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2010, pp. 297–312.
[18] C. Dwork, "Differential privacy: A survey of results," in *Proc. Int. Conf. Theory Appl. MODELS Comput.*, 2008, pp. 1–19.
[19] G. Kellaris and S. Papadopoulos, "Practical differential privacy via grouping and smoothing," *Proc. VLDB Endowment*, vol. 6, no. 5, pp. 301–312, Mar. 2013.
[20] X. J. Zhang, C. Shao, and X. F. Meng, "Accurate histogram release under differential privacy," *J. Comput. Res. Develop.*, vol. 53, no. 5, pp. 1106–1117, 2016.
[21] *You + Big Data = Not Anonymous; Microsoft Develops Differential Privacy for Everyone*. Accessed: Nov. 7, 2012. [Online]. Available: https://www.csoonline.com/article/2223463/you—big-data—not-anonymous–microsoft-develops-differential-privacy-for-everyon.html
[22] B. X. Dong and W. H. Wang, "Frequency-hiding dependency-preserving encryption for outsourced databases,"İin *Proc. IEEE 33rd Int. Conf. Data Eng. (ICDE)*, Apr. 2017, pp. 721–732.
[23] P. Xiong, T. Q. Zhu, and X. F. Wang, "A survey on differential privacy and applications," *Chin. J. Comput.*, vol. 37, no. 1, pp. 101–120, Jan. 2014.
[24] C. Xu, J. Ren, D. Zhang, and Y. Zhang, "Distilling at the edge: A local differential privacy obfuscation framework for IoT data analytics," *IEEE Commun. Mag.* vol. 56, no. 8, pp. 20–25, Aug. 2018.
[25] J. Zhou, Z. Cao, X. Dong, and A. V. Vasilakos, "Security and privacy for cloud-based IoT: Challenges," *IEEE Commun. Mag.*, vol. 55, no. 1, pp. 26–33, Jan. 2017.
[26] E. Bertino, "Security and privacy in the IoT," in *Proc. ICISSP*, 2017, p. 5.
[27] J. Li, F. Palmieri, and Q. Yan, "Special issue on advances in security and privacy in IoT," *J. Netw. Comput. Appl.* vol. 137, pp. 91–92, 2019.
[28] S. Chen, H. Wen, J. Wu, W. Lei, W. Hou, W. Liu, A. Xu, and Y. Jiang, "Internet of Things based smart grids supported by intelligent edge computing," *IEEE Access* vol. 7, pp. 74089–74102, 2019.
[29] W. Yu, F. Liang, X. He, W. G. Hatcher, C. Lu, J. Lin, and X. Yang, "A survey on the edge computing for the Internet of Things," *IEEE Access* vol. 6, pp. 6900–6919, 2018.
[30] R. Bassily, "Linear queries estimation with local differential privacy" in *Proc. AISTATS* 2019, pp. 721–729.
[31] R. Gao and X. Ma, "Dynamic data histogram publishing based on differential privacy" in *Proc. ISPA/IUCC/BDCloud/SocialCom/SustainCom*, Dec. 2018, pp. 737–743.
[32] P. Zhang, M. Durresi, and A. Durresi, "Mobile privacy protection enhanced with multi-access edge computing," in *Proc. AINA*, May 2018, pp. 724–731.

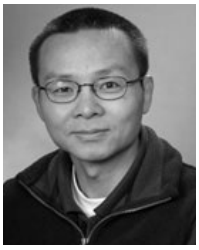
YI QIAO received the B.E. degree in software engineering from Taishan University, China, in 2018. She is currently pursuing the master's degree with the Department of Computer Science, Dalian Maritime University. Her current research interests include big data and cloud computing.

ZHAOBIN LIU received the Ph.D. degree in computer science from the Huazhong University of Science and Technology, China, in 2004. He has been a Senior Visiting Scientist with The University of Auckland, New Zealand, in 2017, and a Visiting Scholar with the University of Central Florida, USA, in 2014, and the University of Otago, New Zealand, in 2008, respectively. He is currently a Professor with the School of Information Science and Technology, Dalian Maritime University, China. His current research interests include big data, cloud computing, and data privacy.



HAOZE LV is currently with the Department of Computer Science, Dalian Maritime University. His current research interests include big data and privacy protection.

MINGHUI LI is currently pursuing the master's degree with the Department of Computer Science, Dalian Maritime University. Her current research interests include big data and privacy protection.



ZHIYI HUANG received the B.Sc. and Ph.D. degrees in computer science from the National University of Defense Technology (NUDT), China, in 1986 and 1992, respectively. He was a Visiting Professor with EPFL (Swiss Federal Institute of Technology Lausanne) and Tsinghua University, in 2005, and a Visiting Scientist with MIT CSAIL, in 2009. He is currently an Associate Professor with the Department of Computer Science, University of Otago, New Zealand. He has published over 150 articles in well-known conferences and journals. His current research interests include parallel/distributed computing, multicore architectures, operating systems, green computing, cluster/grid/cloud computing, high-performance computing, signal processing, and computer architectures and networks.

ZHIYANG LI received the bachelor's and Ph.D. degrees from the Dalian University of Technology, China, in 2004 and 2011, respectively. He is currently an Associate Professor with the School of Information Science and Technology, Dalian Maritime University, China. His current research interests include mobile computing, information retrieval, and computer vision.

WEIJIANG LIU received the Ph.D. degree in computational mathematics from Jilin University, in 1998. From 2004 to 2006, he was a Postdoctoral Fellow of the Post Doctoral Station for Computer Science and Technology, Southeast University, China. He is currently a Professor with the School of Information Technology, Dalian Maritime University, China. He has published more than 80 articles. His current research interests include network measurement, network security, and mobile computing.