Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

MBE, 16(5): 4053–4068.
DOI: 10.3934/mbe.2019200
Received: 18 January 2019
Accepted: 09 April 2019
Published: 08 May 2019

Research article

A reversible database watermarking method with low distortion

Yan Li[1,2,]*, Junwei Wang[1], Shuangkui Ge[3], Xiangyang Luo[1] and Bo Wang[4]

[1] State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, Henan, China
[2] Zhengzhou University, Zhengzhou 450001, Henan, China
[3] Beijing Institute of Electronic Technology Application, Beijing 100000, China
[4] State University of New York at Buffalo, USA

* **Correspondence:** Email: ly79@zzu.edu.cn.

Abstract: In this paper, a low distortion reversible database watermarking method based on histogram gap is proposed in view of the large gap in histogram of database integer data. By using the method, the tolerance of the attribute column containing all integer data is firstly calculated and the prediction error is obtained according to the tolerance. Then according to the watermark bits to be embedded, the database tuples will be randomly grouped and the histogram can be constructed by using the prediction error. Finally, the histogram correction rule is used to find the histogram peak bin, the number of consecutive non-zero prediction errors on the left and right sides of the peak is obtained, and the histogram shift is performed on the side with a smaller number of non-zero prediction errors, and then the watermark embedding will be realized. The results of the experiments based on the published dataset of FCTD (Forest Cover Type Dataset) show that compared with the existing GAHSW which also considers distortion, the proposed method significantly reduces the number of histogram column shift while embedding the watermarks, greatly reduces the changes to the carrier data, and effectively reduces the database's data distortion caused by watermark embedding.

Keywords: database watermark; reversible watermark; histogram gap; data distortion; histogram column shift

1. Introduction

As an important technology to protect database security [1], database watermarking technology was firstly introduced and applied to database copyright protection by IBM Almaden Research

Center in 2002 [2]. Since then, a variety of database watermarking techniques emerge, such as a relational database watermark based on a special tag tuple method [3,4], a digital fingerprint watermark based on the idea of multimedia watermark block [5] and a method of watermarking relational database based on optimization techniques [6]. These methods may cause permanent distortion of the data in the original host when watermark is embedded. If the distortion ratio is small, it will not affect the usage of the database, but if the distortion ratio is large, it will become unacceptable in some specific fields (such as financial, medical, legal, military, etc.). Therefore reducing data distortion and recovering data are important aspects the researchers will concern. To solve such problems, the researchers have proposed a reversible database watermarking method [7–10] which is a technique for hiding the watermark information into the original data and recovering the original data without loss after extracting the watermark .With reversible database watermark method, data can be recovered after being attacked by insertion, deletion and modification. But not every user with legal identity has the right to recover data. The lower the distortion ratio of the database data is used, the better will be. Under such circumstance mentioned above, the reversible database watermarking method with low distortion is proposed to effectively reduce the data distortion. Therefore, it is of practical significance to carry out research on reversible database watermarking methods with low distortion.

The reversible database watermarking methods can be divided into method with data distortion and method with no data distortion. In the followings, we mainly introduce the robust watermarking method with data distortion. In [7], it proposes the reversible database watermarking method based on histogram shifting watermarking. The reference [11,12] use a difference expansion based watermarking (DEW) to restore the original database, and realizes the reversible watermarking mechanism of the relational database. The reference [13] groups the tuples and then uses DEW to embed the watermark. Based on GADEW method, the reference [14] combines genetic algorithm (GA) and DEW, and proposes a robust reversible database watermarking solution which improves the DEW capacity while maintaining a certain distortion. The reference [15] proposes a Prediction-Errors Expansion Watermarking (PEEW) reversible database watermarking method which has a better result in the aspect of anti-aggression. The reference [16], for numerical databases, proposes a Robust and Reversible Watermarking (RRW) method which can effectively resist various attacks like insertion, deletion and modification, and can better protect the quality of data. In [17], the authors proposes A New Robust Approach for Reversible Database Watermarking with Distortion Control) which is abbreviated as GAHSW (Genetic Algorithm and Histogram Shifting Watermarking) for numerical relation data. The method combines GA with a newly proposed Histogramfting of prediction error Watermarking (HSW) method to minimize distortion and improve r obustness for database watermarking. The characteristic of GAHSW is to realize the optimized group ing with GA and to embed a watermark bit in each group consequently with HSW by which the robu Shi stness of watermarked database can be ensured. Compared with previous methods, this method can not only significantly improve the capacity of the watermark embedded and reduce the data distortion, but also acquire higher robustness. However, since the histogram [18–25] will be wholly shifted when the watermark is embedded, the data distortion is still serious.

To solve above problems, we propose a low distortion reversible database watermarking method using histogram gap, which is abbreviated as HGW (Histogram-Gaps based Watermarking). The traditional histogram shift method is improved as followings: finding a vacant position that can introduce minimum distortion since the histogram contains gaps to determine the shift direction and the shift distance, and then moving necessary parts of the columns in the histogram to reduce the number of columns to be shifted when the watermark is embedded, and therefore the modification of the carrier database data will be lowered, which also meets the need of reducing data distortion.

This paper concludes altogether four parts. Section 2 elaborates the basic ideas and main steps of the proposed method. Section 3 performs experimental verification and analysis of results. Section 4 gives conclusions and proposes future research directions.

2. The proposed method

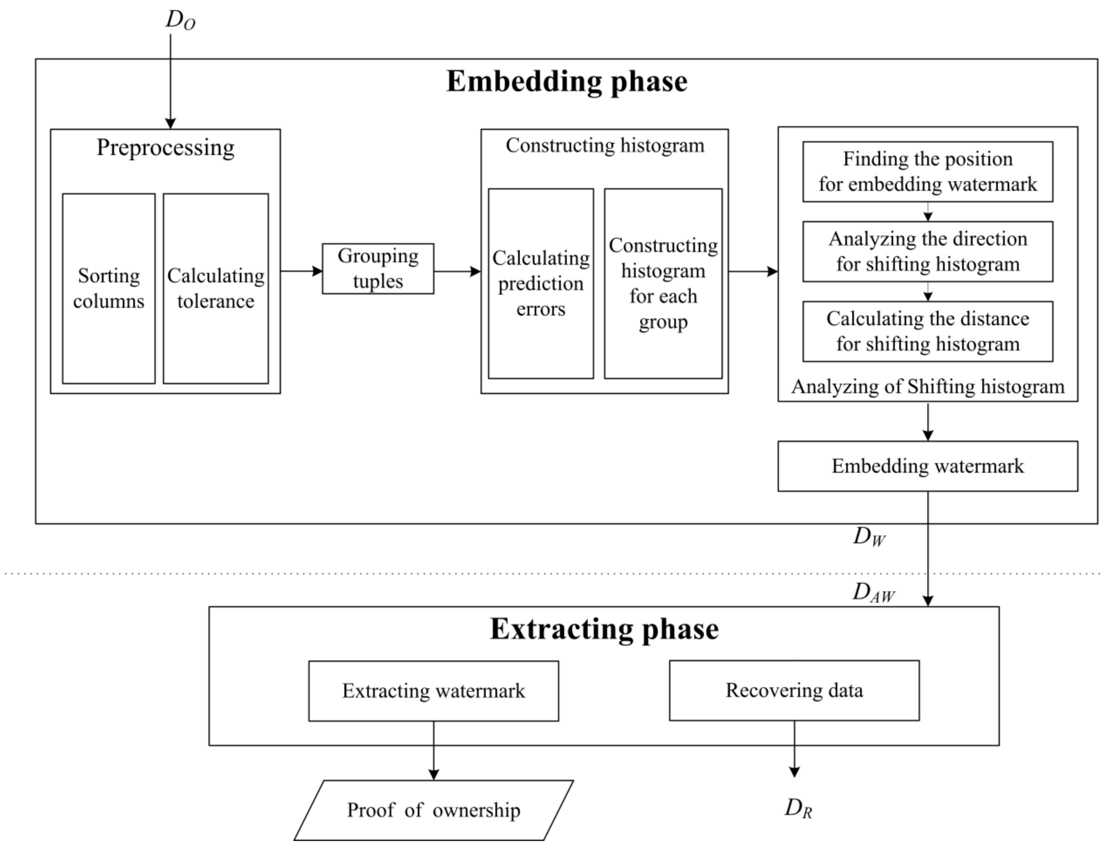
D_O
Embedding phase
Preprocessing
Sorting columns
Calculating tolerance
Grouping tuples
Constructing histogram
Calculating prediction errors
Constructing histogram for each group
Finding the position for embedding watermark
Analyzing the direction for shifting histogram
Calculating the distance for shifting histogram
Analyzing of Shifting histogram
Embedding watermark
D_W
D_{AW}
Extracting phase
Extracting watermark
Recovering data
Proof of ownership
D_R

Figure 1. Schematic diagram of the HGW.

The HGW method consists of two parts which concludes watermark embedding and watermark extraction. As showed in the figure1, D_O is an original database, D_W is the database embedded with watermark, D_{AW} is the attacked database after insertion, deletion and modification, and D_R is the restored database same as the initial database. Specific steps are as follows:

Step1. Data preprocessing.

Step1.1. selecting multiple integer columns from the database that can identify the features of things and then sorting the selected columns in ascending order

Step1.2. defining the semantic distortion range of the jth column as $[min[j], max[j]]$

according to the value of the column and calculating the tolerance of the prediction error of the jth column. The equation of calculating the tolerance is

$$\hat{y} = \lfloor (max[j] + min[j]) / 2 \rfloor \tag{2.1}$$

In the Eq. (2.1), $max[j]$ and $min[j]$ individually represent the maximum and minimum values of the jth column. $\hat{y}$ is the value determined by the maximum and minimum of the column.

Step 2. Grouping tuples. Ks, a secret grouping key, is set by a random method and the tuples in the database are divided into a set of non-overlapping groups $\{G_i\} i = 1, 2, \ldots\ldots, N_g$, and the value of N_g is determined by the number of the bits of the watermark to be embedded. Eq. (2.2) should be used to determine the group in which each tuple is located.

$$n_u = H(Ks \,|\, H(Ks \,|\, t_u.PK)) \bmod N_g \tag{2.2}$$

In the Eq. (2.2), n_u represents the group number, '|' represents the concatenation operation, $H()$ is a hash function, Ks is the secret grouping key and $t_u.PK$ is the primary key of the tuple as parameters.

Step 3. Constructing histogram.

Step 3.1. Calculating p_e and p_h with Eq. (2.3) and Eq. (2.4) respectively

$$p_e = y - \hat{y} \tag{2.3}$$

$$p_h = | p_e | \tag{2.4}$$

In the Eq. (2.3) and Eq. (2.4), y represents the value located in jth column and in certain tuple, p_e represents the prediction error value corresponding to y. p_h is the absolute value of the prediction error of the original database.

Step 3.2. Constructing a histogram for each group with p_h and marking the p_h occurring most frequently as p_i in ith group (i.e., the peak bin in the histogram) and storing p_i to array *pa*.

Step 4. Shifting and embedding.

That related research found that the discrete database has histogram with gaps. Take the right shift as an example. For the histogram column, you don't need to move all on the right side of p_i Just clear the side of p_i a vacant position and then embed the watermark at p_i.

Step4.1. In the ith group, the method will start from p_i and search the left side and the right side, find the first p_h whose frequency is zero, record the corresponding position as p_{iL} and p_{iR}, record the sum of heights of rectangles from p_{iL} to p_i as hs_{iL}, record the sum of heights of rectangles from p_i to p_{iR} as hs_{iR}, and then respectively calculate the distance from the position of p_i to p_{iL} and to p_{iR}. The equations are as follows:

$$d_{iL} = | p_i - p_{iL} |, \quad d_{iR} = | p_i - p_{iR} | \tag{2.5}$$

In the Eq. (2.5), d_{iL} is the value calculated by the distance from the position of p_i to p_{iL}. d_{iR} is the value calculated by the distance from the position of p_i to p_{iR}.

Step4.2. Shifting histogram and embedding watermark. According to the results of comparing hs_{iL} with hs_{iR}, the Eq. (2.6) and Eq. (2.7) can be concluded.

if $hs_{iL} \geq hs_{iR}$,

$$p'_h = \begin{cases} p_h + 1, & p < p_h < p_i + d_{iR} \\ p_h + w, & p_h = p_i \\ p_h, & otherwise \end{cases} \tag{2.6}$$

if $hs_{iL} < hs_{iR}$,

$$p'_h = \begin{cases} p_h - 1, & p_i - d_{iL} < p_h < p_i \\ p_h - w, & p_h = p_i \\ p_h, & otherwise \end{cases} \tag{2.7}$$

In the Eq. (2.6) and Eq. (2.7), p'_h is the absolute value of new prediction error of watermarked database. According to Eq. (2.4), if $p_e \geq 0$, p_h equals to p_e. if $p_e < 0$, p_h equals to $-p_e$. Then substituting the two results respectively into the Eq. (2.6) and Eq. (2.7), the followings Eq. will be concluded.

if $hs_{iL} \geq hs_{iR}$ and $p_e \geq 0$,

$$p'_e = \begin{cases} p_e + 1, & p_i < p_e < p_i + d_{iR} \\ p_e + w, & p_e = p_i \\ p_e, & otherwise \end{cases} \tag{2.8}$$

if $hs_{iL} \geq hs_{iR}$ and $p_e < 0$,

$$p'_e = \begin{cases} p_e - 1, & -(p_i + d_{iR}) < p_e < -p_i \\ p_e - w, & p_e = -p_i \\ p_e, & otherwise \end{cases} \tag{2.9}$$

if $hs_{iL} < hs_{iR}$ and $p_e \geq 0$,

$$p'_e = \begin{cases} p_e - 1, & p_i - d_{iL} < p_e < p_i \\ p_e - w, & p_e = p_i \\ p_e, & otherwise \end{cases} \tag{2.10}$$

if $hs_{iL} < hs_{iR}$ and $p_e < 0$,

$$p'_e = \begin{cases} p_e + 1, & -p_i < p_e < -(p_i - d_{iL}) \\ p_e + w, & p_e = -p_i \\ p_e, & otherwise \end{cases} \tag{2.11}$$

In the Eq. (2.8)–(2.11), p'_e is the new prediction error of watermarked database. According to different situations, the histogram shifting and watermark embedding can be finally solved with Eq. (2.8), Eq. (2.9), Eq. (2.10), and Eq. (2.11). Besides, there are two special situations which should be noted. If $p_i = 0$, Eq. (2.8) is the only one which should be used to solve the situation. If $p_i = 1$, Eq.

(2.8) and Eq. (2.9) should be the only choice. This reduces the data distortion caused by the shifted.

Step5. Watermark extraction and data recovery.

Step5.1. The method selects the columns in which the watermark is embedded, and sorts the columns. The selection method and the sorting method are the same as Step1. Using the secret grouping key generated from the watermark embedding stage, the tuples are divided into non-overlapping groups by Eq. (2.2). Calculating p'_e and p'_h with Eq. (2.12) and Eq. (2.13) respectively. The histogram is constructed with p'_h on each group.

$$p'_e = y' - \hat{y} \tag{2.12}$$

$$p'_h = | p'_e | \tag{2.13}$$

In the Eq. (2.12), y' represents any value in D_W, $\hat{y}$ which can be calculated with Eq. (2.3) represents the tolerance of the column including y'. p'_e is the prediction error corresponding to y'. After the histogram is constructed, the method will scan p'_e one by one, judge the position in the histogram, extract the watermark and restore the database.

Step5.2. Watermark extraction. Each watermark is extracted by using the distance and direction of the shifting in each histograms stored in the array *pa*. The method performs the above detection on all tuples of the same group, and separately records the number of all ‘0’ and ‘1’ detected in the group, and then use the majority voting mechanism to determine the final watermark bit of the group. The watermark bit which has a larger number is treated as the final detected watermark bit.

If $| p'_e | = p_i$, meaning the attribute value is not modified and there is a gap at the side of the histogram, the detected watermark bit is ‘0’. If $hs_{iL} < hs_{iR}$, two situations will appear. If $p'_e > 0$ and $p'_e = p_i - 1$, the detected watermark bit is ‘1’ and if $p'_e < 0$ and $p'_e = p_i + 1$, the detected watermark bit is ‘1’. If $hs_{iL} \geq hs_{iR}$, two situations will also appear. If $p'_e > 0$ and $p'_e = p_i + 1$, the detected watermark bit is ‘1’ and if $p'_e < 0$ and $p'_e = p_i - 1$, the detected watermark bit is ‘1’. Corresponding to the embedded watermark, two special situations will appear while detecting watermark bit ‘1’. If $p_i = 0$ and $p'_e = p_i + 1$, the detected watermark bit is ‘1’. If $p_i = 1$, if $p'_e > 0$ and $p'_e = p_i + 1$, the detected watermark bit is ‘1’ and if $p'_e < 0$ and $p'_e = p_i - 1$, the detected watermark bit is ‘1’. It has been verified by experiments that the watermark bits extracted by the majority voting mechanism can also be used to prove the owner of the copyright.

Step5.3. Data recovery. According to the positive and negative values of p'_e, and the results of comparing hs_{iL} with hs_{iR}, the following data recovery is performed.

if $hs_{iL} \geq hs_{iR}$ and $p'_e \geq 0$,

$$y^r = \begin{cases} y' - 1, & p_i + 1 < p'_e \leq p_i + d_{iR} \\ y' - 1, & p'_e = p_i + 1, w = 1 \\ y', & otherwise \end{cases} \tag{2.14}$$

if $hs_{iL} \geq hs_{iR}$ and $p'_e < 0$,

$$y^r = \begin{cases} y' + 1, & -(p_i + d_{iR}) \leq p'_e < -(p_i + 1) \\ y' + 1, & p'_e = -(p_i + 1), w = 1 \\ y', & otherwise \end{cases} \tag{2.15}$$

if $hs_{iL} < hs_{iR}$ and $p'_e \geq 0$,

$$y^r = \begin{cases} y' + 1, & p_i - d_{iL} \leq p'_e < p_i - 1 \\ y' + 1, & p'_e = p_i - 1, w = 1 \\ y', & otherwise \end{cases} \tag{2.16}$$

if $hs_{iL} < hs_{iR}$ and $p'_e < 0$,

$$y^r = \begin{cases} y' - 1, & -(p_i - 1) < p'_e \leq -(p_i - d_{iL}) \\ y' - 1, & p'_e = -(p_i - 1), w = 1 \\ y', & otherwise \end{cases} \tag{2.17}$$

In the above equations, y^r is the restored attribute value. Corresponding to the special situation while embedding the watermark, if $p_i = 0$, Eq. (2.14) is the only one which should be used to solve the situation. If $p_i = 1$, Eq. (2.14) and Eq. (2.15) should be the only choice.

3. Experimental results and analysis

Experiments were conducted on a workstation with Intel Core i5 with CPU of 2.40 GHz and RAM of 8 GB. The method is implemented and tested.

The test database is the Forest Cover Type dataset provided by the University of California. The dataset contains 581,012 tuples and 54 properties. In this paper, 10 integer-type columns are selected for experiments and compared with the existing GAHSW reversible watermarking method. For the sake of comparison, the method of this paper, same as the experiment of GAHSW in [17], will generate a synthetic column as the primary key. All of the following experiments embed watermarks under the same conditions.

3.1. Statistical distortion analysis

In this section, the data distortion rate (DDR) is used to evaluate the distortion effects of the HGW and GAHSW on the database after embedding the watermark. The DDR is calculated as followings:

$$DDR = \frac{T_{dis}}{TD} \tag{3.1}$$

In the Eq. (3.1), T_{dis} is the total amount of data distorted and *TD* is the total amount of data in the database. The larger the DDR, the greater the distortion of the data will appear. Conversely, the distortion of the data is small. Experiments verify that the distortion rate of HGW is much smaller than the distortion rate of GAHSW.

With different length of the watermark (the number of groups) which are 24, 48 and 72, the distortion rates of HGW and GAHSW were compared. During each comparing process, the total number of tuples of 1000, 1200, 1400, 1600, 1800, and 2000 are respectively verified with same length of the watermark (the number of groups), and the results are plotted in Figure 2, 3 and 4. In the figure, the vertical axis is DDR which indicates the ratio of data distortion. When DDR is zero, it means that the data of the database is not distorted and the horizontal axis represents the different number of the tuples. Since both HGW and GAHSW rely on stochastic optimization, the distortion rate is the average of the 10 runs of the two methods.

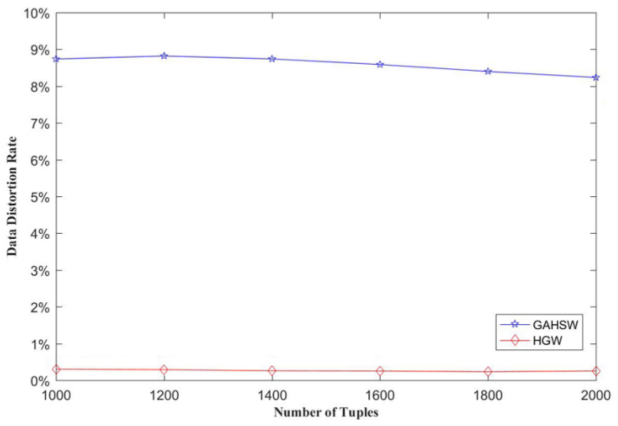

Figure 2. The comparison of data distortion rates caused by 24-bit watermarks.

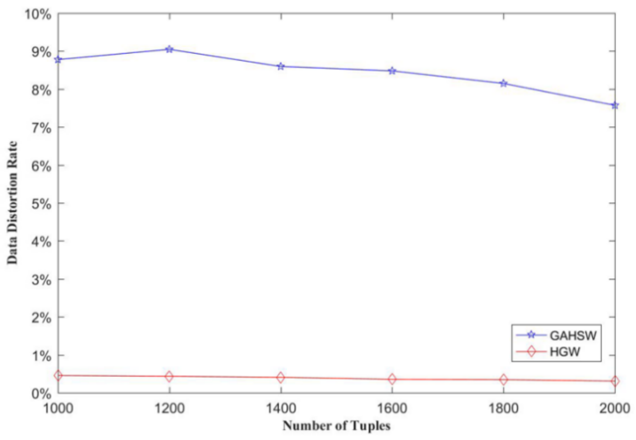

Figure 3. The comparison of data distortion rates caused by 48-bit watermarks.

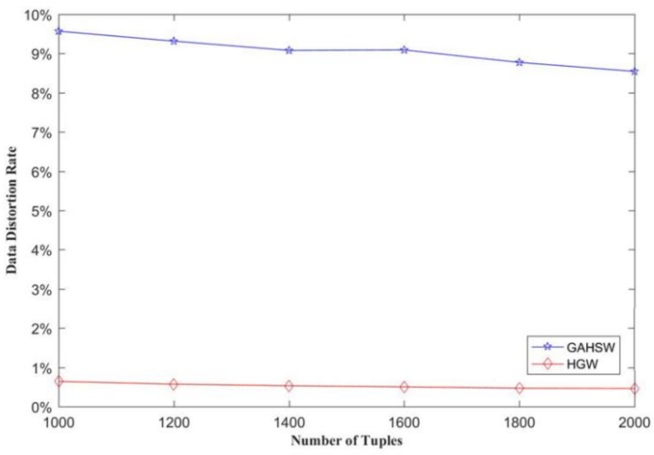

Figure 4. The comparison of data distortion rates caused by 72-bit watermarks.

As can be seen from the figure 2, 3 and 4, when the same watermark information is embedded in the same database, the DDR generated by the HGW method is much lower than the GAHSW, which is less than 0.7%.

3.2. Histogram shifting quantity analysis

In this section, the total amount of distortion data from the shift (without the distortion caused by embedding the watermark bits) is used to evaluate the distortion effects of the HGW and GAHSW on the database after embedding the watermark.

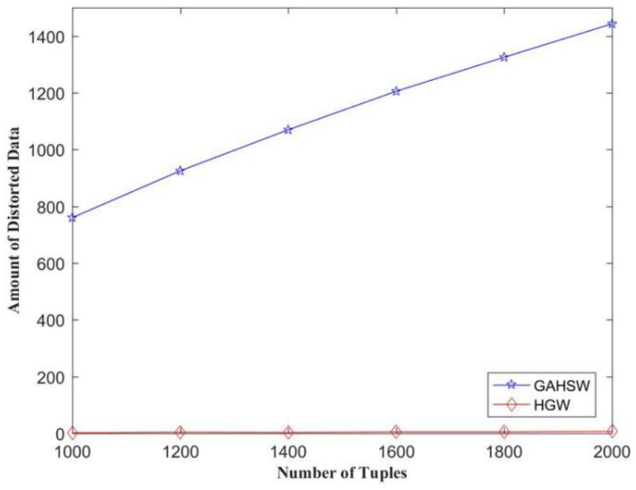

Figure 5. The comparison of the total amount of the data distortion caused by shift with 24-bit lengths of watermark.

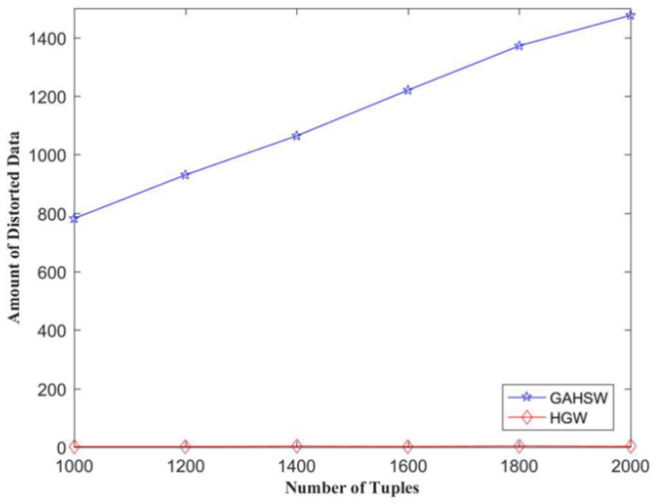

Figure 6. The comparison of the total amount of the data distortion caused by shift with 48-bit lengths of watermark.

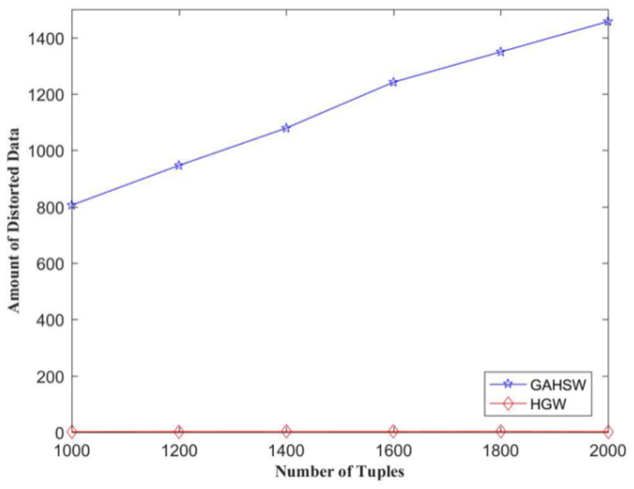

Figure 7. The comparison of the total amount of the data distortion caused by shift with 72-bit lengths of watermark.

With different length of the watermark (the number of groups) which are 24, 48 and 72, the distortion rates of HGW and GAHSW were compared. During each comparing process, the total number of tuples of 1000, 1200, 1400, 1600, 1800, and 2000 are respectively verified with same length of the watermark (the number of groups), and the results are plotted in Figure 5, 6 and 7. In the figure, the vertical axis is the total amount of distortion data, indicating the number of data of distortion. When the total amount of translation distortion data is zero, it means that the data of the database is not distorted and the horizontal axis represents the different number of the tuples. Since both HGW and GAHSW rely on stochastic optimization, the total amount of distortion data is the average of the results of 10 runs of the two methods. It can be seen from the figure 5, 6 and 7 that when the same watermark information is embedded in the same database, the total amount of translational distortion data generated by the HGW method is much lower than GAHSW, which is at least 1.5 and 7 at the maximum and almost overlaps with the horizontal axis.

3.3. Robustness analysis

In this section, the robustness of the HGW watermarking method under well-known database attacks is reported. For convenience, the article carries on the comparing test by setting the length of the watermark (the number of groups) as 48 just like the test mentioned in the reference [17]. With analyzing the attack results, this paper verifies the robustness of the HGW method. This article mainly tests three types of attacks: inserts, deletes, and changes. Robustness is evaluated by using bit error rate (BER). That is, the ratio of the number of bits extracted by error to the number of embedded watermark bits. The BER is calculated as follows:

$$BER = \frac{\sum_{i=1}^{N_g} w_i \oplus w_i^{det}}{N_g} \tag{3.2}$$

In the Eq. (3.2), w_i is the embedded watermark bit and w_i^{det} is the detected watermark bit. We can see that the lower the BER value appears, the higher the watermark robustness will be. It is verified by experiments that the robustness of HGW is comparable to that of GAHSW. In the followings, we conduct attack experiments under the best situation and the worst situation, demonstrate watermark detection and data recovery results, and compare the two results.

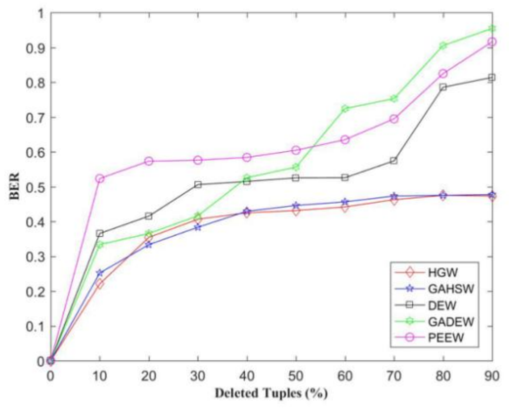

Figure 8. Comparison of watermark extraction BER of HGW with GAHSW, PEEW, DEW and GADEW after deletion attacks.

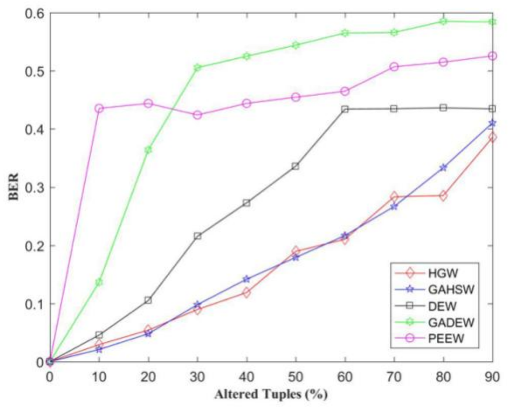

Figure 9. Comparison of watermark extraction BER of HGW with GAHSW, PEEW, DEW and GADEW after modification attacks.

Simulating as an attacker, this article attempts to insert, delete, and modify data for 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, and 90%, and the results are plotted respectively in Figures 8 and 9. The vertical axis in the figure is BER, indicating that the ratio of watermarks has not been successfully detected. When the BER is close to zero, it means that the watermark is correctly detected from the database. The horizontal axis represents the changes' percentage (%) of the tuples in the database after attacking. Since the HGW relies on random optimization, the BER of the watermark extraction is the average of 10 runs of the method.

Both GAHSW and HGW are robust to insertion attacks, and the BER under the insertion attack is always zero no matter how many tuples are inserted. To this end, the comparison diagram of the insertion attacks will not be drawn. It can be seen from Figure 8 and Figure 9 that the robustness of the GAHSW and HGW methods is roughly equivalent under the Deletion attack and modification attacks. As the attack strength increases, the BER of the watermark extracted by these two methods also increases. The experiments also show that the performance of HGW and GAHSW have better watermark detection rate under this attack compared with other methods.

Deletion attack is to randomly delete the tuple to destroy the watermark. Figure 8 shows the BER of the watermark extracted by the HGW, GAHSW, PEEW, DEW and GADEW methods after the Deletion attack. When the database is suffered from severe deletion attack, for example, the number of tuples deleted in the database becomes 90%, DEW, GADEW, and PEEW extract the watermark with the BER value of 0.814, 0.955 and 0.915, respectively. However, the BER of the HGW extraction watermark is 0.472, and the BER of the GAHSW watermark extraction is 0.477. The HGW can also recover at least half of the watermark. If most of the watermark contained in the tuples is deleted, the watermark will not be able to be recovered. Deletion attacks have the greatest impact on database watermarks.

Modification attacks are to randomly modify the attributes of the database. Figure 9 shows the BER of the watermark extracted by the HGW, GAHSW, PEEW, DEW and GADEW methods after modification attacks. We can see that the BER of the extracted watermark increases as the amount of data modification increases. The number of tuples modified in the database becomes 90%, DEW, GADEW, and PEEW extract the watermark with the BER value of 0.434, 0.584 and 0.526, respectively. However, the BER of the HGW extraction watermark is 0.383, and the BER of the GAHSW watermark extraction is 0.411. The HGW can also recover at least half of the watermark. If the watermark contained in most tuples is modified, then the watermark tuple affected by this attack will be difficult to extract from the remaining unaffected data.

In general, the HGW method, comparing to GAHSW, produces a lower BER when embedding the same watermark information into the same database. It is worth noting that the HGW method sorts the names of the attribute columns in alphabetical order before the watermark is embedded and extracted, and groups the tuples according to the primary key of the tuple. Therefore, the rearrangement has no effect on the database.

4. Conclusions and future work

In order to reduce the data distortion rate of the carrier, a reversible database watermarking method based on histogram gap low distortion is proposed in this article. With this method, the vacancy provided by the histogram gap is used to reduce the number of columns which are shifted when the watermark is embedded, and therefore the modification of the carrier database data will be reduced, which also meets the need of reducing data distortion. The characteristic of HGW significantly reduces the number of histogram column shift while embedding watermarks, greatly reduces the changes to the carrier data, and effectively reduces the database data distortion caused by watermark embedding. Comparing HGW with GAHSW, the results show that HGW is superior to GAHSW in the aspects of reducing the amount of the histogram shifting and the data distortion. In the case of no attack, the watermark extraction error rate of HGW is better than that of GAHSW. In the face of attack, the bit error rate of watermark extraction of HGW nearly equals to that of

GAHSW. Basing on the present methods, my future work is to propose better methods to maximally reduce the total amount of distortion data.

Acknowledgments

The work presented in this paper is supported by the National Natural Science Foundation of China (No.U1536104, 61572052, U1636219, 61772549, U1736214, U1804263,), the National Key R&D Program of China (No. 2016QY01W0105, 2016YFB0801303), and Plan for Scientific Innovation Talent of Henan Province (No.2018JR0018).

Conflict of interest

The authors declare no conflict of interest.

References

1. N. Gursale and A. Mohanpurkar, A robust, distortion minimization fingerprinting technique for relational database, *IJRITCC*, **2**(2014), 1737–1741.
2. R. Agrawal and J. Kiernan, Watermarking relational databases, *VLDB*, Elsevier, Hong Kong, China, **28** (2002), 155–166.
3. R. Sion, Proving ownership over categorical data, *ICDE*, Boston, (2004), 584–596.
4. R. Sion, M. Atallah and S. Prabhakar, Rights protection for categorical data, *IEEE Trans. Knowl. Data Eng.*, **17**(2005), 912–926.
5. S. Liu, S. Wang, R. Deng, et al, A block oriented fingerprinting scheme in relational database, *ISCISC*, Seoul, Korea, (2004), 145–192.
6. M. Shehab, E. Bertino and A. Ghafoor, Watermarking relational databases using optimization-based techniques, *IEEE Trans. Knowl. Data Eng.*, **20** (2008), 116–129.
7. Y. Zhang, B. Yang and X. Niu, Reversible watermarking for relational database authentication, *JCP*, **17** (2006), 59–65.
8. J. Chang and H. Wu, Reversible fragile database watermarking technology using difference expansion based on SVR prediction, *IS3C*, Taiwan, China, (2012), 690–693.
9. E. Mahmoud, H. Farfoura and X. Wang, A novel blind reversible method for watermarking relational databases, *JCIE*, **36** (2013), 87–97.
10. V. Khanduja, S. Chakraverty and O. Verma, Enabling information recovery with ownership using robust multiple watermarks, *JISA*, **29** (2016), 80–92.
11. G. Gupta and J. Pieprzyk, Reversible and blind database watermarking using difference expansion, *IJDCF*, **1** (2008), 42–54.
12. G. Gupta and J. Pieprzyk, Database relation watermarking resilient against secondary watermarking attacks, *Inform. System. Secur.*, Springer, (2009), 222–236.
13. S. Bhattacharya and A. Cortesi, A distortion free watermark framework for relational databases, *ICSDT*, Sofia, Bulgaria, (2013), 229–234.
14. K. Jawad and A. Khan, Genetic algorithm and difference expansion based reversible watermarking for relational databases, *J. System Software*, **86** (2013), 2742–2753.

15. M. Farfoura and A. Horng, A novel blind reversible method for watermarking relational databases, *ISPA*, (2010), 563–569.
16. S. Iftikhar, M. Kamran and Z. Anwar, RRW-a robust and reversible watermarking technique for relational data, *IEEE Trans. Knowl. Data Eng.*, **27** (2015), 1132–1145.
17. D. H. Hu, D. Zhao and S. L. Zheng, A new robust approach for reversible database watermarking with distortion control, *IEEE Trans. Knowl. Data Eng.*, (2018), DOI: 10.1109/TKDE.2018.2851517.
18. Y. Cao, Z. L. Zhou, X. M. Su, et al., Coverless information hiding based on the molecular structure images of material, *CMC*, **54** (2018), 197–207.
19. W.J. Xu, S.J. Xiang and V. Sachnev, A Cryptograph Domain Image Retrieval Method Based on Paillier Homomorphic Block Encryption, *CMC*, **55** (2018), 285–295.
20. J. W. Wang, T. Li, X. Y. Luo, et al., Identifying computer generated images based on quaternion central moments in color quaternion wavelet domain, *TCSVT*, (2018), DOI: 10.1109/TCSVT.2018.2867786.
21. Y. Du, Z. X. Yin and X. P. Zhang, Improved lossless data hiding for jpeg images based on histogram modification, *CMC*, **55** (2018), 495–507.
22. Y. Zhang, C. Qin, W. M. Zhang, et al., On the fault-tolerant performance for a class of robust image steganography, *Signal Process.g*, **146** (2018), 99–111.
23. X. Y. Luo, X. F. Song, X. L. Li, et al., Steganalysis of HUGO steganography based on parameter recognition of syndrome-trellis-codes, *Mult Tool Appl.*, **75** (2016), 13557–13583.
24. T. Qiao, R. Shi, X. Y. Luo, et al., Statistical model-based detector via texture weight map: Application in re-sampling authentication, *TMM*, (2018), DOI: 10.1109/TMM.2018.2872863.
25. Y. Y. Ma, X. Y. Luo, X. L. Li, et al., Selection of rich model steganalysis features based on decision rough set α-positive region reduction, *TCSVT*, **29** (2019), 336–350.