Fast Tamper Location of Batch DNA Sequences Based on Reversible Data Hiding

Juntao Fu, Weiming Zhang*, Nenghai Yu, Guoli Ma, Qi Tang
Key Laboratory of Electromagnetic Space Information, Chinese Academy of Sciences
University of Science and Technology of China
Hefei, 230027, Anhui, P.R China
Email: fujuntao@mail.ustc.edu.cn
*zhangwm@ustc.edu.cn

***Abstract*—Nowadays, more and more DNA sequences are stored in databases of NCBI and other organizations. Based on these databases, much research has been conducted. Once tampered, the research results will be useless and even lead to disastrous consequences. It is significant to protect the integrity of the DNA sequences. This paper proposes a fast tamper location method of batch DNA sequences based on reversible data hiding. A fragile watermarking combining with the Merkle Hash Tree is presented with respect to achieving fast tamper location of batch DNA sequences. With the method in the paper, the tampered blocks can be fast located, especially for dealing with batch DNA sequences. The method can be applied to large databases, where providers can quickly check whether the DNA sequences are intact or not. Besides, users can solve the problem that the sequences may be tampered during transmission.**

Keywords-DNA sequence; fast tamper location; Merkle Hash Tree; reversible data hiding; fragile watermarking

I. INTRODUCTION

DNA, representing the Deoxyribonucleic Acid, is a kind of vital biological macromolecules. Generally a DNA sequence is an arrangement of four nitrogenous bases, which are adenine (*A*), thymine (*T*), guanine (*G*), and cytosine (*C*). As the carrier of genetic code, DNA plays an important role in all kinds of life and the research on DNA sequences can make a deep understanding of life, so a perfect protection on DNA sequences is necessary. Nowadays, many organizations have kept storage of DNA sequences in their databases for researchers to download. Once the sequences stored in these databases are tampered or modified by the evil attackers, the researchers could not get the correct results. What's worse, it could lead to disasters.

A method that can detect the integrity of DNA sequences and even locate the tampered regions will be surely beneficial. The verification of data integrity has been in research for many years, especially in large databases and cloud storage. The widely adopted is message digest, of which the hash value is the typical. However, as to DNA sequences, what researchers usually do is just to calculate the hash values of DNA sequences and then compare them, which is not so efficient. It is necessary to find new ways to fast locate the tampered regions of a DNA sequence, especially when users are faced with a DNA sequence with a long length or a large number of DNA sequences.

A data embedded scheme is essential when the authentication information is obtained. In digital data filed, reversible watermark [1] is a wonderful way to do this. Based on the reversible data hiding method, the authentication information can be embedded into DNA sequences. When DNA sequences are downloaded from the databases by users, users can extract the authentication information to detect the integrity of DNA sequences.

Nowadays, hiding information in DNA has become an interesting research topic because of its biological property. Many different ways [2][3] have been put forward to hide information, such as the authentication information, some confidential information. In [4], Tai et al. embedded the authentication code into the DNA sequence to detect the integrity. However, this method could not locate the tampered regions. In [5], Ma et al. put forward a tamper restoration method based on reversible data hiding. Their method achieved the embedding of the authentication information and could locate and restore the tampered regions. However, the tampered regions of a DNA sequence is detected just by comparing the hash value of each block one by one, making the number of comparison equal to the number of blocks. When a longer DNA sequence is required to verify, the number of comparison will be much larger and cost much more time.

In this paper, we propose a fast tamper location method of batch DNA sequences based on reversible data hiding. By our method, the tampered regions can be quickly localized and no reference DNA sequences are demanded. The method can be suitable for not only a single DNA sequence but also batch DNA sequences.

The paper is organized as follows: the paper introduces how to generate the watermark in section 2. In section 3, a scheme of watermark embedding, extraction, tamper location and DNA sequence restoration is elaborated. In Section 4, the paper shows experiment results of the proposed method applied to a single DNA sequence and batch DNA sequences. This paper is concluded with a discussion in Section 5.

II. Watermark Generation

A. The Construction of Merkle Hash Tree

The Merkle Hash Tree [9][10] is a kind of Hash Tree, whose nodes are all hash values. The general form is complete Binary Hash Tree. As an efficient way to achieve fast tampered location, the Merkle Hash Tree is widely used in integrity authentication.

Firstly, divide a data file into *N* blocks and calculate the hash value of each block. All the hash values of the blocks are acted as the leaf nodes. Based on these leaf nodes, a Merkle Hash Tree can be constructed. The whole process is as follows:

1) Let h_i denote the hash value of $i\text{-}th$ block, the total number of blocks is *N*.

2) Use || to concatenate adjacent two values from beginning to end and calculate the new hash value denoted as $h(h_i \,||\, h_{i+1})$, where *h* is a hash function, such as SHA-1, MD5.

3) The new hash values are generated, then do step *2)* on the basis of the new hash values till the final number of the new hash value is just only one, which represents the root of a Merkle Hash Tree. In the end, a Merkle Hash Tree is constructed.

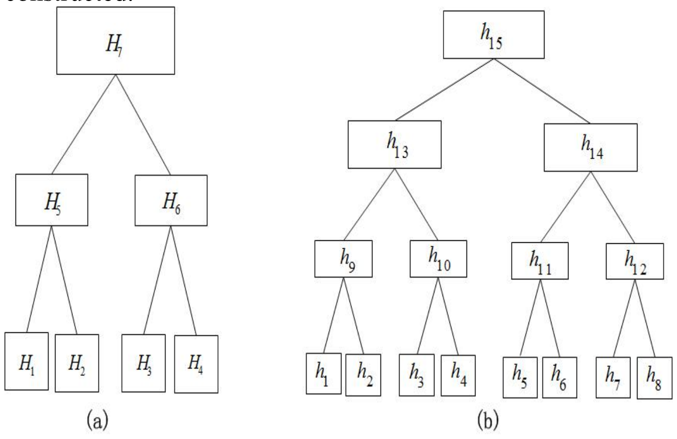

Fig. 1. (a) Construction of Merkle Hash Tree based on batch DNA sequences, (b) Construction of Merkle Hash Tree based on a single DNA sequence.

Take *N*=8 for an example, a Merkle Hash Tree is constructed as in Fig. 1.(b). From h_1 to h_8, they are hash values of the origin blocks. The rest hash values are generated based on the hash values of the origin eight blocks. All the fifteen hash values together form a Merkle Hash Tree.

B. Watermark Generation

The watermark generation is at the lever of the block because the DNA sequence is always divided into blocks. Firstly divide a DNA sequence into blocks and do permutation operation on each block. Then use the bits after permutation to create the hash bits and construct the Merkle Hash Tree. Finally, generate the watermark and embed it into the DNA sequence.

1) Block permutation: Divide an origin DNA sequence into *N* blocks, where *N* is an integer. In order to create a complete binary Hash Tree, set *N* as the integer power of 2, such as *N*=4, 8, 16. Here denote the origin sequence block as $a_i, i = 1, 2 \ldots N$. Then a permutation operation, such as a key-dependent method, is applied to the origin blocks. The blocks after permutation are denoted as b_i.

2) Hash bits generation: The hash bits are generated in the following way:

$$h_i = h(b_i), i = 1, 2, 3, \ldots, N \tag{1}$$

where *h* represents a common hash function, h_i represents the hash bits of the $i\text{-}th$ block, b_i represents the block bits after permutation.

All the *N* hash values are used to construct the Merkle Hash Tree, and then there are $2 \times N - 1$ hash values in the Merkle Hash Tree. The final *N* hash values used for watermark generation and embedding are obtained in detail as follows:

a) Arrange all values of the Hash Tree in a row according to the rule form left to right and top to down. There are $2 \times N - 1$ hash values in the row, denoted as $h_j^{'}$, *j*=1, 2,…, $2 \times N - 1$.

b) Denote mh_i as final hash bits and set $mh_1 = h_1^{'}$. The rest mh_i is defined as follows:

$$mh_i = h_{2\times i-2}^{'} || h_{2\times i-1}^{'}, i = 2, 3 \ldots, N \tag{2}$$

3) Watermark generation: To fit the feature of the information hiding method, the watermark should be generated and embed it into the DNA sequence. The watermark is designed as follows:

$$W_i = mh_i \tag{3}$$

After watermark generation, the watermark W_i is then embedded into the block a_i by use of the reversible data hiding method. What just described is the situation of one single DNA sequence. As for batch DNA sequences, the principle is the same. A Merkle Hash Tree of two levels is constructed. The first lever is based on the DNA sequences, while the second level is based on the blocks of a single DNA. From the first level, the tampered DNA can be quickly localized, while from the second level, the tampered regions in the tampered DNA can also be quickly localized. Fig. 1 shows the details, where the number of all DNA sequences is four and the block number of a single DNA is eight. The Fig. 1.(a) is a Merkle Hash Tree built on batch DNA sequences, while Fig. 1.(b) is built on a single DNA sequence.

III. Watermark Embedding and Fast Tamper Location

The scheme about how to embed the watermark information into a DNA sequence based on reversible data hiding is proposed in this section. Besides, this section will also discuss how to locate the tampered regions fast when a watermarked DNA sequence is downloaded by users.

A. Reversibly embedding of watermark

There is an interesting phenomenon in the natural DNA sequences that large redundancy exists in DNA sequences, whose bases usually repeatedly appear. This phenomenon makes DNA sequences fit for embedding and extracting authentication information by using the reversible data hiding method. The following is a piece of DNA sequence taken from the NCBI database: TTTTGGTTGTTTGAGGAGTCTAGAGGGAAATACCTCT CAG. For the piece TTTT, if the positions of the first T and the end T are known, the middle two T can embed data. However, the piece CTAG is not suitable for embedding data because of none repeated bases.

Therefore, a DNA sequence should be divided into embeddable segments and non-embeddable segments before applied to embed data. Then construct the beginning and ending position labels for all embeddable segments. In order to construct the labels, two operations are defined, which are adding (+) operation and subtracting (—) operation. The operations and the other symbols are defined in IUPAC [8]. By the rules in [5] and the operations, the labels can be constructed. Once the labels constructed, the embeddable segments can be applied to embed information.

A base substitution method is required to embed 2 bits per repeated base [6] [7]. Its main goal is to create a one-to-one mapping. Table I shows a base substitution method. The base substitution strategy is: (00, A), (01, C), (10, G), (11, T), where 00, 01, 10, 11 are binary forms, A, C, G, T are Nucleotides.

Table I. THE BASE SUBSTITUTION METHOD

Nucleotide	Binary bits
A	00
C	01
G	10
T	11

Besides, a local map is required to indicate whether the base in the corresponding position is a degenerate base [8] though they are little. If so set the bit to 1 otherwise 0. The length of the local map is equal to the length of the sequence, but it is can be lossless-compressed with high compression ratio. All degenerate bases are collected to form a recording R and replaced by any normal bases. The data to be embedded is like:

$$M = L \,||\, R \,||\, P \tag{4}$$

where L is the compressed location map, R is the record of degenerate bases, P is the payload data, such as authentication information.

The data embedding is operated at the level of block. The embedding strategy is in the following way. For each block, first generate the data to be embedded and divide the block into segments. Then for each segment, if non-embeddable, just skip to the next segment, while if embeddable, construct the beginning and ending labels and embed some bits.

When a user receives a watermarked DNA sequence, via the subtraction operation and the one-to-one map, the starting and ending labels can be retrieved as well as the embedded data (that is M). Then L can be exactly separated because of the message symbol at the end of the bit stream. Via the Local Map by decompressing L, all positions and the number of the original degenerate bases will be known. In the end, the whole original sequence can be restored by replacing R.

B. Fast tamper location

When a user downloads a DNA sequence from the database, the user first extracts the watermark $W_i = mh_i$ by the reversible data hiding method and then the sequence block a_i can be obtained. From mh_i the user can get all $h_j^{'}$. Based on $h_j^{'}$, a Merkle Hash Tree can be created, which represents the origin sequence. Do permutation operation on block a_i to obtain b_i, calculate the hash value of b_i and then the user can construct another new Merkle Hash Tree, which represents the sequence received. By comparing the two Merkle Hash Trees, all the tampered blocks can be fast located.

If the sequence is not tampered, the user can efficiently achieve integrity authentication just by comparing the root value of the two Merkle Hash Trees. The origin DNA sequence can be recovered by concatenating all the N origin blocks. If the tampered regions exist, turn to the left and right children nodes of the parent node until all the tampered regions are located. It is the same suitable for the Batch DNA sequences. For example via Fig. 1, if the first block (its hash value h_1) of the second DNA sequence (its hash value H_2) is tampered, we can locate the second DNA sequence by the first level Merkle Hash Tree at first. Then locate the first block of the second DNA sequence by the second level Merkle Hash Tree. The whole tampered location process is also quite efficient.

IV. Experimental Results

The proposed method was tested on a set of DNA sequences collected from the NCBI database. In our paper, we first show the embedding capacity of our reversible data hiding method. Then we discuss the efficiency of the proposed method for locating tampered regions. During the integrity authentication of the DNA sequence, the number of comparison is used to test the fast tamper location method. The assessment is applied to a single DNA sequence as well as batch DNA sequences.

A. Capacity of reversible data hiding method

Table II. THE CAPACITY OF SOME DNAS

GI	Number of nucleotides	Capacity (bits)	bpn (bit/base)
7768689	118343	73006	0.612
7768731	140513	83160	0.592
27807848	74032	45220	0.611
32453484	155285	87572	0.564
209553988	772998	543956	0.704
223667572	483802	258330	0.534

Here we define two metrics to evaluate our reversible data hiding method. One is the Capacity, which represents the total length of the embedded data a DNA sequence can hold. The other is the bpn, which represents the embedding data each base can hold. About two thousand DNA sequences are tested. The length of the sequences ranges from hundreds to millions. Table II shows the result of six DNA sequences.

From the results, we conclude that the bpn of the method is nearly 60 percent. Although the bpn is lower than the compression method in [11], our method is fit for the sequence not that long.

B. Fast location on tampered regions of one single DNA sequence

The Merkle Hash Tree method is compared with the general method introduced in [5]. In this section the number of comparison is used to evaluate our algorithm.

In the experiment, we select a DNA sequence with the length of 200118, and we divide it into 512 blocks. When a base of a DNA sequence is tampered, it means the block the base belongs to is also tampered. As the tampered bases of the DNA sequence will not fill the whole DNA sequence, the DNA sequence is tampered at the lever of block. A random function is adopted to tamper the blocks. The number of the tampered blocks ranges from 0 to 100. Repeat one hundred times for each tampered situation to avoid special cases. The tampered rate is defined as follows:

$$\alpha = \frac{M}{N} \tag{5}$$

where M represents the number of tampered blocks and N represents the number of all blocks.

By the method introduced in [5], the number of comparison is N, no matter what the number of tampered blocks. N can be replaced by:

$$N = 2^n \tag{6}$$

where n is an integer.

The number of tampered block is M and the height of the Merkle Hash Tree is n, so the comparison number by the proposed method is $M \times n$. Define *eff* to reflect the result:

$$eff = \frac{Mn}{2^n} = \alpha \times \log N \tag{7}$$

The *eff* is determined by α if N is a constant value. In practical application, N is always fixed, so the higher the tampered rate, the larger the *eff*, which means the Merkle Hash Tree method is appropriate for situation of low tampered rate.

The situation in (7) is just theoretical situation. Actually, if two tampered blocks share the same parent, the comparison number will be nearly one times lower. Fig. 2 shows the efficiency of the Merkle Hash Tree method over the general method.

It can be concluded that when the tampered rate is low (in the figure lower than 18%) via Fig. 2, the Merkle Hash Tree method is more efficient than the method by checking the integrity of a DNA sequence as block one by one. The method in [5] can reconstruct the whole sequence in high probability if the tampered rate is not extensive (below 20%). So if the tampered rate is too high, the DNA sequence will be useless. Our method is helpful in practical application with a low tampered rate.

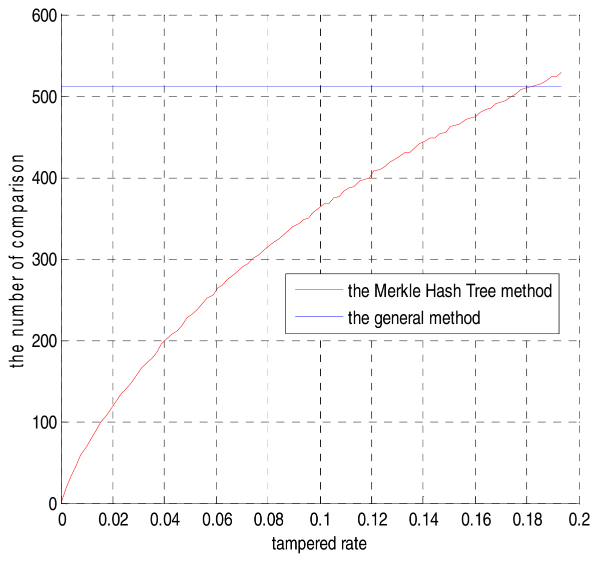

Fig. 2 the comparison of the Two methods for single DNA

C. Fast location on tampered regions of Batch DNA sequences

In this part, we construct a two lever Merkle Hash Tree to deal with batch DNA sequences. The first lever is based on all the DNA sequences, while the second based on the blocks of a single DNA sequence.

In our experiment, the total number of the DNA sequences is N_1=32. The block number of a single DNA sequence is N=512. The number of the tampered blocks is defined as M, which ranges from 0 to 3000. If the blocks are checked one by one like in [5], the total comparison number is $N_1 \times N$ =16384. While in our method, the theoretical comparison numbers is denoted as:

$$M \times (\log(N) + \log(N_1)) \tag{8}$$

In the practical application, the values of total sequences number and block number of a DNA can be determined by users, while the tampered rate is not. Therefore from (8), it can be inferred that the comparison number of the proposed method becomes larger with the increasing of the number of the tampered blocks. Via the results in Fig. 3, we can see our method is better than the method in [5] when the tampered rate is lower than 18%. And the proposed method has great advantages over the general method when tampered rate is low.

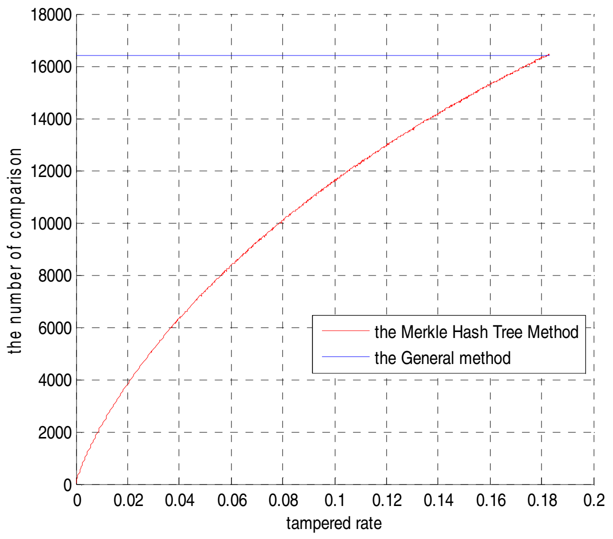

Fig. 3 the comparison of the Two methods for batch DNAs

V. Conclusion

The protection of the DNA sequences is an urgent thing. Based on the reversible data hiding method, users can achieve a good protection on the DNA sequences. In this paper, a fast tamper location of batch DNA sequences is proposed. The experimental results show that the efficiency of locating the tampered regions can be greatly improved when the tampered rate is low. However, it is not enough to just locate the tampered regions, trying to restore the tampered regions is also essential. The future work is attempted to restore the DNA sequences on the basis of the fast tampered location method.

Acknowledgment

This work was supported in part by the Natural Science Foundation of China under Grant 61170234 and Grant 60803155, by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDA06030601, and by the Funding of Science and Technology on Information Assurance Laboratory under Grant KJ-13-003.

Reference

[1] H. Luo, F. X. F X, H. Chen H, et al, Reversible data hiding based on block median preservation, Information Sciences, 2011, 181(2): 308-328.

[2] H. J. Shiu, K. L. Ng, J. F. Fang, R. C. T. Lee and C. H. Huang, Data hiding methods based upon DNA sequences, Information Sciences. Vol. 180, pp. 2196–2208, June 2010.

[3] M. R. Abbasy, A. A. Manaf, M. A. Shahidan M A, Data hiding method based on DNA basic characteristics, Digital Enterprise and Information Systems. Springer Berlin Heidelberg, pp. 53-62, 2011.

[4] W. L. Tai, C. C. N. Wang, P. C. Y. Sheu, J. J. P. Tsai, Data hiding in DNA for authentication of plant variety rights, Journal of Electronic Science and Technology, Vol. 11, No. 1, March 2013.

[5] G. L. Ma, Q. Tang, W.M. Sheu, N. H. Yu, Tamper restoration on DNA sequences based on reversible data hiding, Biomedical Engineering and Informatics, pp.484-489, March 2013.

[6] B. A. Mitrans, A. K. Aboo, Proposed steganography approach using DNA properties, International Journal of Information Technology and Business Management, Vol. 14, No. 1, June 2013.

[7] S. Manna, S. Roy, et al, Modified technique of insertion methods for data hiding using DNA sequences, ACES, 2014 First International Conference on. IEEE, 2014: 1-5.

[8] IUPAC: International Union of Pure and Applied Chemistry, http://en.wikipedia.org/wiki/International_Union_of_Pure_and_Applied_Chemistry

[9] Boris Ederov, Merkle tree traversal techniques, Darmstadt University of Technology, Department of computer science cryptography and computer algebra, April 2007.

[10] M. Szydlo, Merkle tree traversal in log space and time, Advances in Cryptology-EUROCRYPT 2004, Springer Berlin Heidelberg, pp.541-554,2004.

[11] C.C. Chang, T.C. Lu, Y.F. Chang, R.C.T. Lee, Reversible data hiding schemes for deoxyribonucleic acid (DNA) medium, International Journal of Innovative Computing, Information and Control. Vol. 3(5), pp. 1145–1160, 2007.