Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8892838, 9 pages
https://doi.org/10.1155/2020/8892838

Research Article

Efficient and Privacy-Preserving Outsourcing of 2D-DCT and 2D-IDCT

Dezhi An, Shengcai Zhang, Jun Lu, and Yan Li

School of Cyber Security, Gansu University of Political Science and Law, Lanzhou 730070, China

Correspondence should be addressed to Dezhi An; adz6199@gsli.edu.cn

Received 12 May 2020; Revised 2 June 2020; Accepted 16 June 2020; Published 27 July 2020

Academic Editor: Wei Wang

As a subset of discrete Fourier transform (DFT), discrete cosine transform (DCT), especially two-dimensional discrete cosine transform (2D-DCT), is an important mathematical tool for digital signal processing. However, the computational complexity of 2D-DCT is quite high, which makes it impossible to meet the requirements in some signal processing fields with large signal sizes. In addition, to optimize the 2D-DCT algorithm itself, seeking help from a cloud platform is considered to be an excellent alternative to dramatically speeding up 2D-DCT operations. Still, there are three key challenges in cloud computing outsourcing that need to be addressed, including protecting the privacy of input and output data, ensuring the correctness of the returned results, and ensuring adequate local cost savings. In this paper, we explore the design of a practical outsourcing protocol for 2D-DCT and 2D-IDCT, which well solves the above three challenges. Both theoretical analysis and simulation experiment results not only confirm the feasibility of the proposed protocol but also show its outstanding performance in efficiency.

1. Introduction

DFT is a common tool for frequency domain analysis of discrete signals and systems, but it is inconvenient to process image and voice data due to the need for complex domain operations. To solve this problem, a real domain transform, called DCT, is constructed based on DFT by preserving only the cosine term in the Fourier series. In addition to the general orthogonal transform properties of DCT, the basis vector of its transform matrix can well describe the relevant features of image signals and human voice signals. Therefore, DCT is considered to be a quasioptimal tool for transforming image and voice signals and is widely used in various fields such as media compression [1]–[3], digital watermarking [4]–[6], and wireless communication [7, 8]. 2D-DCT can directly transform two-dimensional data, so it is quite suitable for the analysis and processing of two-dimensional signals, such as static images. For example, 8×8 2D-DFT is adopted as a standard by the JPEG still image compression algorithm [9].

The computational complexity of 2D-DCT and 2D-IDCT is dominated by computing the product of three matrices. Such high-computational overhead makes it impossible to perform high-efficiency processing on signals when the local computing equipment is insufficient or the signals are large in size. Wireless camera sensor networks, for example, are widely used in habitat monitoring, target detection, and espionage [10, 11], where nodes typically use processors with low cost, micro power consumption, and poor performance. In contrast, these nodes are often burdened with complex image processing tasks, a considerable portion of which require the participation of 2D-DCT or 2D-IDCT.

In order to improve the efficiency of the transformation, the usual method is to divide the signal into blocks, then perform 2D-DCT or 2D-IDCT operations in each block, and finally merge the blocks. Nevertheless, the method of using blocks is not a once and for all solution, because smaller blocks are necessary for greater efficiency, but too small blocks will lead to serious block effects. In addition, various fast 2D-DCT and 2D-IDCT algorithms have been proposed [12]–[14], which can reduce the computational complexity by more than half on the basis of using a block method.

We emphasize that in addition to efficiency optimization from an algorithmic perspective, requesting computing assistance from the cloud platform is also an excellent alternative.

For example, the image processing tasks of nodes in a wireless sensor network can be delegated to a public cloud server. The cloud platform has a large amount of hardware and software resources, and the temporary use rights of partial computing resources it owns are transferred to clients by way of fee lease. The process by which clients rent cloud resources to help them complete their own computing is called cloud outsourcing. On the one hand, the client can achieve significant computing overhead savings and efficiency improvement through computing outsourcing, so as to quickly complete high-complexity computing tasks. On the other hand, the client can significantly reduce costs by eliminating the expense of purchasing and maintaining large amounts of computing equipment. At the same time, cloud platforms can also reap considerable economic benefits from resource rentals. Therefore, computing outsourcing is considered a win-win move for both the client and the cloud.

However, computing outsourcing between the client and the cloud faces three key challenges. First, we must protect the client's privacy from being stolen. The client's input and output data contains private information, which may be personally identifiable information, trade secrets, or core technical parameters. Since the cloud is untrusted, we must protect the plaintext of the input and output data from being obtained by the cloud. Meanwhile, the high concentration of information makes the cloud platform vulnerable, which is also a possible way to leak private information. Second, we must verify the results returned by the cloud. The cloud platform is run for profit, and a malicious cloud will deliberately return random error results to the client to extract cost savings. Even if the cloud is honest, calculation errors may occur due to software bugs or hardware errors. Therefore, it is necessary to design an efficient verification algorithm to strictly control the correctness of the returned results. Third, we must ensure that the client can realize significant savings from computing outsourcing. In other words, the total complexity of decryption, encryption, and result verification algorithms must be far less than that of solving the original problem directly; otherwise, there is no need for the client to outsource the computing task. In general, a qualified computing outsourcing protocol must be secure, verifiable, and efficient.

In this paper, a protocol capable of solving the above three challenges is designed for the outsourcing of 2D-DCT and 2D-IDCT. Before giving the protocol, we discuss two possible designs in an exploratory way. Multiround communication and block encryption severely damage the efficiency of both designs. In response to the problems in these two designs, we propose the formal outsourcing protocol, which greatly reduces the client's communication overhead and key management overhead. In the proposed protocol, the original two-dimensional signal matrix is integrally encrypted without affecting the block flexibility of the cloud, i.e., the cloud can still perform 2D-DCT or 2D-IDCT of any block size according to the client's will. This is an important point that affects the versatility of the protocol, because different application scenarios may require the operation of 2D-DCT or 2D-IDCT with different block sizes. Subsequently, we carried out theoretical analysis to confirm that the proposed protocol meets the requirements of security, verifiability, and efficiency. Finally, the simulation results show that the proposed protocol is not only better than block 2D-DCT and 2D-IDCT in efficiency but also faster than the corresponding fast 2D-DCT and 2D-IDCT algorithms.

To summarize, our main contributions include the following:

(i) To the best of our knowledge, we are the first to propose such an outsourcing protocol for 2D-DCT and 2D-IDCT, and we are also the first to accelerate the operational efficiency of 2D-DCT and 2D-IDCT from the perspective of cloud computing outsourcing

(ii) Through carrying out theoretical analysis and simulation experiments, it is shown that the proposed protocol handles the three challenges faced in computing outsourcing well

(iii) In the proposed protocol, only a single round of communication is required and the key management work is simple, which is suitable for the outsourcing of 2D-DCT and 2D-IDCT with arbitrary block size

The rest of this paper is organized as follows. Section 2 describes the related work, and Section 3 gives the problem statement. 2D-DCT and 2D-IDCT are briefly introduced in Section 4. Section 5 presents the possible designs and the formal outsourcing protocol. Section 6 provides theoretical analysis, followed by experiments in Section 7. Finally, Section 8 concludes this paper.

2. Related Work

There are two ways to implement outsourcing for complexity calculations. On the one hand, the theoretical cryptography community considers designing a universal design that covers all problems, i.e., any outsourcing of computing can be realized through this design. The basic approach to achieve this goal is to use some sophisticated basic cryptographic tools, such as Yao's garbled circuits [15] and Gentry's fully homomorphic encryption (FHE) schemes [16]. In these designs [17]–[20], the original problem is converted into a Boolean circuit over {0, 1}, and then, the client encrypts the converted problem using a FHE algorithm and sends the encrypted problem to the cloud, who solves the problem homomorphically and returns the result. Finally, the client decrypts the result and checks its correctness. However, such designs are far from practical applications because of the extremely high complexity of FHE operations and the pessimistic circuit sizes.

On the other hand, the security engineering community focuses on designing different outsourcing protocols to deal with different practical problems. Under this idea, the encryption of the original problem is generally achieved through some ingenious data conversion. Meanwhile, in order to face practical applications, the efficiency of the protocol is given special consideration to ensure that the client

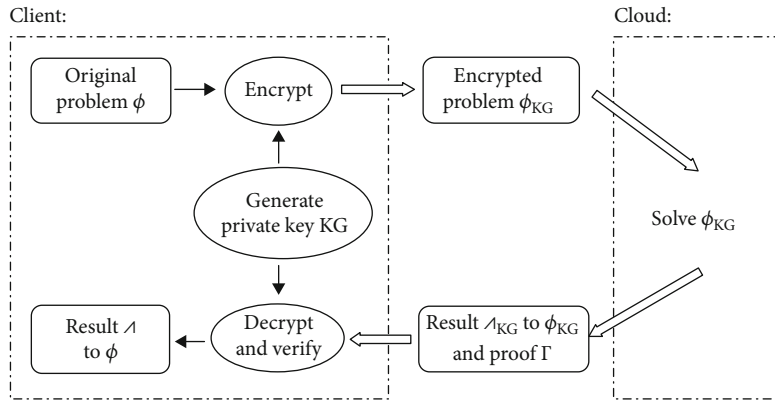

Figure 1: System model.

can acquire considerable cost savings from outsourcing. According to this trend, a large number of practical outsourcing protocols have been proposed. Among these protocols, there are outsourcing of basic mathematical calculations. Lei et al. proposed an outsourcing protocol for matrix inversion [21], Chen et al. proposed an outsourcing protocol for linear regression [22], and Li et al. proposed an outsourcing protocol for ID3 decision tree [23]. Several different outsourcing protocols for linear equations were presented in [24–26]. In addition, some outsourcing protocols are directly oriented to practical engineering application scenarios. An outsourcing protocol for Markowitz portfolio was proposed by Zhang et al. in [27], an outsourcing protocol for data classification was proposed by Li et al. in [28], an outsourcing protocol for the storage and statistics of smart meter data was proposed by Zhang et al. in [29], and an outsourcing protocol for biometric identification was proposed by Zhu et al. in [30]. Moreover, Li et al. proposed an optimal GPU-accelerated multimedia processing service pricing strategy in [31].

By using a homomorphic cryptography algorithm, some digital signal processing operations are transferred to the ciphertext domain to protect the privacy of the signal. Bianchi et al. implemented ciphertext domain operations on DFT and DCT in [32, 33], respectively. Zheng and Huang proposed a method for implementing discrete wavelet transform (DWT) and multiresolution analysis (MRA) in the homomorphic ciphertext domain [34]. Pedrouzo-Ulloa et al. realized secure number theoretic transform (NTT) in a distrustful environment [35]. Recently, Han et al. improved the homomorphic DFT with batch homomorphic encryption [36]. However, because of the high complexity of homomorphic operations, outsourcing with these solutions cannot meet the requirements of efficiency, i.e., the client fails to acquire the required savings. By contrast, data processing tools can also be efficiently outsourced, mainly to speed up signal processing. For example, Xiao et al. realized efficient and secure outsourcing of DFT, IDFT, and circular convolution [37], and Zhang et al. designed two outsourcing protocols for compressed sensing and sparse robustness decoding service in [38, 39], respectively. Nevertheless, the efficient outsourcing of 2D-DCT and 2D-IDCT has not been designed by predecessors.

3. Problem Statement

3.1. System Model. As illustrated in Figure 1, the system consists of two entities: the client and the cloud. Their roles are elaborated as follows.

(1) *Client*, The client has a complex problem Φ that needs to be solved. In this paper, the problem Φ is considered to be the 2D-DCT or 2D-IDCT operation. In order to acquire computing overhead savings, the client plans to outsource the solution of Φ to the cloud platform. To achieve the purpose of privacy protection, the client firstly uses the locally generated private key KG to encrypt the original problem Φ and generate the encrypted problem Φ_{KG}. Subsequently, the user sends Φ_{KG} to the cloud, who solves Φ_{KG} to obtain the result Λ_{KG} and a proof Γ used for verification and returns Λ_{KG} and Γ to the client. After the client receives Λ_{KG}, the private key KG is used to decrypt Λ_{KG} to obtain the result Λ of problem Φ. Finally, the client uses Γ to verify the correctness of Λ. If the verification passes, the client accepts Λ; otherwise, the client rejects Λ.

(2) *Cloud*. The cloud has a large number of hardware and software resources used for computing. It charges the rental fee of the client and provides computing support services for the client with part of its own resources. After receiving the encrypted problem Φ_{KG}, the cloud solves it and returns Λ_{KG} and Γ.

3.2. Threat Model and Design Goals. The threat mainly comes from the cloud's noncredibility, and here, we assume that the cloud is malicious. On the one hand, the cloud tries to obtain plaintext about problem Φ and result Λ. On the other hand, the cloud tries to save resources by returning a random erroneous result in the expectation that it will not be discovered by the client. In this case, we summarize the design goals of an efficient and privacy-preserving outsourcing protocol as follows.

(1) *Correctness*. The client must be able to acquire the correct answer if both the client and the cloud follow the protocol carefully.

(2) *Privacy*. The cloud cannot steal any private information of the client from the input and output data.

(3) *Soundness*. The client must be able to verify the correctness of the returned results.

(4) *Efficiency*. The client must be able to realize significant savings from computing outsourcing.

3.3. Framework. Syntactically, an outsourcing protocol includes the following five algorithms.

(1) *KeyGen* (1^λ). On input of a security parameter λ, the client uses this algorithm to generate a private key KG.

(2) *ProbEnc* (Φ;*KG*). On input of the original problem Φ and the key KG, the client uses this algorithm to encrypt Φ to generate the encrypted problem Φ_{KG} and then sends Φ_{KG} to the cloud.

(3) *ProbSolve* (Φ_{KG}). On input of the encrypted problem Φ_{KG}, the cloud uses this algorithm to solve Φ_{KG}. Then, the cloud returns the result Λ_{KG} back, together with a proof Γ.

(4) *ResultDec* (Λ_{KG};*KG*). On input of the result Λ_{KG} of Φ_{KG} and the key KG, the client uses the algorithm to decrypt Λ_{KG} to obtain the result Λ of the original problem Φ.

(5) *ResultVerify* (Λ;Γ). On input of the result Λ of Φ and the proof Γ, the client uses the algorithm to verify the correctness of Λ.

4. 2D-DCT and 2D-IDCT

4.1. 2D-DCT. Suppose that the original two-dimensional signal can be represented by a matrix $f \in R^{N\times N}$. In practice, if the original signal is not square, the transformation is usually done after the complement, and the original signal is subsequently obtained by removing the complement after the reconstruction. We consider performing block 2D-DCT of $K \times K$ on the matrix f

, where N is divisible by K.

First, the matrix f needs to be partitioned as

$$f = \begin{pmatrix} f(1,1) & f(1,2) & \cdots & f(1,N) \\ f(2,1) & f(2,2) & \cdots & f(2,N) \\ \cdots & \cdots & \cdots & \cdots \\ f(N,1) & f(N,2) & \cdots & f(N,N) \end{pmatrix} = \begin{pmatrix} f_{1,1} & f_{1,2} & \cdots & f_{1,N/K} \\ f_{2,1} & f_{2,2} & \cdots & f_{2,N/K} \\ \cdots & \cdots & \cdots & \cdots \\ f_{N/K,1} & f_{N/K,2} & \cdots & f_{N/K,N/K} \end{pmatrix}, \tag{1}$$

where $f(i,j)$, $i,j \in [1,N]$ and $f_{i,j} \in R^{K\times K}$, $i,j \in [1,N/K]$ are the elements and the blocks of f, respectively, and the relation between them is

$$f_{i,j} = \begin{pmatrix} f(Ki-K+1, Kj-K+1) & \cdots & f(Ki-K+1, Kj) \\ \cdots & \cdots & \cdots \\ f(Ki, Kj-K+1) & \cdots & f(Ki, Kj) \end{pmatrix}. \tag{2}$$

Subsequently, an orthogonal matrix $A \in R^{K\times K}$ is calculated by

$$A(i,j) = c(i)\cos\left[\frac{(j+0.5)\pi}{N} i\right], \tag{3}$$

where $c(i)$ is defined by

$$c(i) = \begin{cases} \sqrt{\dfrac{1}{N}}, & i = 0, \\ \sqrt{\dfrac{2}{N}}, & i \neq 0. \end{cases} \tag{4}$$

Finally, the operation result $F \in R^{N\times N}$ of $K \times K$ 2D-DCT on f can be expressed as

$$F = \begin{pmatrix} A\cdot f_{1,1}\cdot A^T & A\cdot f_{1,2}\cdot A^T & \cdots & A\cdot f_{1,N/K}\cdot A^T \\ A\cdot f_{2,1}\cdot A^T & A\cdot f_{2,2}\cdot A^T & \cdots & A\cdot f_{2,N/K}\cdot A^T \\ \cdots & \cdots & \cdots & \cdots \\ A\cdot f_{N/K,1}\cdot A^T & A\cdot f_{N/K,2}\cdot A^T & \cdots & A\cdot f_{N/K,N/K}\cdot A^T \end{pmatrix}. \tag{5}$$

4.2. 2D-IDCT. 2D-IDCT is the inverse process of 2D-DCT. Thus, similarly, the 2D-IDCT operation is represented as

$$f = \begin{pmatrix} A^T\cdot F_{1,1}\cdot A & A^T\cdot F_{1,2}\cdot A & \cdots & A^T\cdot F_{1,N/K}\cdot A \\ A^T\cdot F_{2,1}\cdot A & A^T\cdot F_{2,2}\cdot A & \cdots & A^T\cdot F_{2,N/K}\cdot A \\ \cdots & \cdots & \cdots & \cdots \\ A^T\cdot F_{N/K,1}\cdot A & A^T\cdot F_{N/K,2}\cdot A & \cdots & A^T\cdot F_{N/K,N/K}\cdot A \end{pmatrix}. \tag{6}$$

5. Protocol Construction

5.1. Attempt One. Since the efficiency of 2D-DCT is mainly limited by the calculation of three matrix multiplication, we naturally think that the key of design is to realize the outsourcing of $A\cdot f_{i,j}\cdot A^T$, where $i,j \in [1,N/K]$. Several different outsourcing protocols for the multiplication of two matrices have been proposed [40]–[42]. Therefore, a direct assumption is to first use these protocols to outsource $A\cdot f_{i,j}$ to obtain the product S and then outsource $S\cdot A^T$, thereby acquiring calculation result of $A\cdot f_{i,j}\cdot A^T$.

Apart from other flaws of the protocol, the most obvious drawback of the protocol is the need for two rounds of communication between the cloud and the client. This will bring not only multiplied communication overhead to the client but also multiplied computing overhead for the client, because the client needs to perform independent encryption, decryption, and result verification operations for each round of outsourcing. As a result, the existence of two-round communication will make it difficult for the outsourcing protocol to meet the requirements of high efficiency.

5.2. Attempt Two. We attempt to implement an efficient outsourcing protocol that required only one round of communication by using the privacy-preserving matrix multiplication as adopted in [40, 43, 44]. First, to protect the privacy of input and output data, the private key matrices $P_1, P_2, \cdots, P_{4N^2/K^2} \in R^{K\times K}$ are locally generated by the client as follows:

$$P_n(i,j) = p_i \delta_{\pi_n(i),j}, \quad n \in \left[1, \frac{4N^2}{K^2}\right], \tag{7}$$

where p_i is a random number and $\pi_n(\cdot)$ is a permutation function that maps an original index i to its permuted index. Besides, $\delta_{i,j}$ is the Kronecker delta function given by

$$\delta_{i,j} = \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases} \tag{8}$$

We emphasize that the inverse of P_n can be accessed simply by

$$P_n^{-1}(i,j) = \frac{1}{P_i} \delta_{\pi_n^{-1}(i),j}. \tag{9}$$

Afterwards, for any block $A \cdot f_{i,j} \cdot A^T$ of the matrix F in (5), the client encrypts the matrices A, $f_{i,j}$, and A^T as

$$\begin{cases} E(A) = P_{(4N/K)(i-1)+4j-3} \cdot A \cdot P^{-1}_{(4N/K)(i-1)+4j-2}, \\ E\left(f_{i,j}\right) = P_{(4N/K)(i-1)+4j-2} \cdot f_{i,j} \cdot P_{(4N/K)(i-1)+4j-1}, \\ E\left(A^T\right) = P^{-1}_{(4N/K)(i-1)+4j-1} \cdot A^T \cdot P_{(4N/K)(i-1)+4j}. \end{cases} \tag{10}$$

The encrypted matrices $E(A)$, $E(f_{i,j})$, and $E(A^T)$ are then sent to the cloud, who perform

$$\begin{aligned} E\left(F_{i,j}\right) &= E(A) \cdot E\left(f_{i,j}\right) \cdot E\left(A^T\right) \\ &= P_{(4N/K)(i-1)+4j-3} \cdot F_{i,j} \cdot P_{(4N/K)(i-1)+4j}. \end{aligned} \tag{11}$$

Finally, to decrypt the returned $E(F_{i,j})$, the client only needs to calculate

$$F_{i,j} = P^{-1}_{(4N/K)(i-1)+4j-3} \cdot E\left(F_{i,j}\right) \cdot P^{-1}_{(4N/K)(i-1)+4j}. \tag{12}$$

This design avoids the trouble of multiround communication existing in Attempt One and seems to be feasible. However, this is not the case, because the complex key management severely slows down the efficiency of the protocol. In Attempt Two, for an original data matrix with dimension $N \times N$, the client needs to generate $4N^2/K^2$ mutually independent key matrices with size $K \times K$ for encryption. Meanwhile, $2N^2/K^2$ key matrices in them need to be saved in order to be invoked during decryption. Besides, we emphasize that in 2D-DCT and 2D-IDCT operations, only f and F are private in the input and output data, while A and A^T are public and can be generated independently by anyone. Therefore, an efficient protocol should manage to encrypt only f or F on input.

5.3. Formal Protocol

5.3.1. Outsourcing Protocol for 2D-DCT. We now present the formal protocol, which solves the flaws of Attempt One and Attempt Two well, i.e., not only requires only one round of communication but also greatly simplifies the key management workload.

First, in terms of privacy protection, we adopted the privacy-preserving matrix addition as discussed in [26, 41, 45]. The client generates two random vectors u and v with size $N \times 1$ as the private key. To protect the privacy information in the original two-dimensional signal, the client encrypts the matrix f as follows:

$$E(f) = f + u \cdot v^T. \tag{13}$$

After the encryption of (13), any block $f_{i,j}$ in the matrix f satisfies

$$E\left(f_{i,j}\right) = f_{i,j} + u_i \cdot v_j^T, \tag{14}$$

where vector $u_i \in R^{K\times 1}$ is a vector consisting of the $Ki - K + 1$th to Kith elements of vector u and vector $v_j^T \in R^{1\times K}$ is a vector consisting of the $Kj - K + 1$th to Kjth elements of vector v^T.

Afterwards, the encrypted matrix $E(f)$ is sent to the cloud who perform $K \times K$ 2D-DCT as

$$\begin{aligned} E\left(F_{i,j}\right) &= A \cdot E\left(f_{i,j}\right) \cdot A^T = A \cdot \left(f_{i,j} + u_i \cdot v_j^T\right) \cdot A^T \\ &= A \cdot f_{i,j} \cdot A^T + A \cdot u_i \cdot v_j^T \cdot A^T = F_{i,j} + A \cdot u_i \cdot v_j^T \cdot A^T. \end{aligned} \tag{15}$$

We emphasize that matrices A and A^T are not private and can be generated independently by the cloud itself by (3). The calculated results $E(F_{i,j}) \forall i, j \in [1, N/K]$ are then combined into matrix $E(F)$ and returned to the client, who uses the private key vectors u and v to decrypt $E(F)$ efficiently through

$$F_{i,j} = E\left(F_{i,j}\right) - (A \cdot u_i) \cdot \left(v_j^T \cdot A^T\right). \tag{16}$$

Finally, the client quickly verifies the correctness of $F_{i,j}$ by the inference of Parseval's theorem, i.e.,

$$\sum_{m=1}^{K}\sum_{n=1}^{K} f_{i,j}^2(m,n) = \sum_{m=1}^{K}\sum_{n=1}^{K} F_{i,j}^2(m,n). \tag{17}$$

If (17) holds within the allowable error range, block $F_{i,j}$ passes the verification. As long as all the blocks in the matrix F pass the verification, the client accepts the calculation result F; otherwise, the client rejects F.

We summarize the outsourcing protocol for 2D-DCT as follows.

(1) *KeyGen* (1^λ). On input of a security parameter λ, the client generates the private key $\mathrm{KG} = (u, v)$.

(2) *ProbEnc* (Φ;*KG*). On input of the original problem $\Phi = f$ and the key KG, the client performs (13) to encrypt Φ to generate the encrypted problem $\Phi_{\mathrm{KG}} = E(f)$ and then sends Φ_{KG} to the cloud.

(3) *ProbSolve* (Φ_{KG}). On input of the encrypted problem Φ_{KG}, the cloud solves Φ_{KG} by (15). Then, the cloud returns the result $\Lambda_{\mathrm{KG}} = E(F)$ back, together with an empty proof Γ.

(4) *ResultDec* (Λ_{KG};*KG*). On input of the result Λ_{KG} of Φ_{KG} and the key KG, the client performs (16) to decrypt Λ_{KG} to obtain the result $\Lambda = F$ of the original problem Φ.

(5) *ResultVerify* (Λ;Γ). On input of the result Λ of Φ and the proof Γ, the client verifies the correctness of Λ by (17).

Compared with Attempt One and Attempt Two, on the one hand, the formal protocol requires only one round of communication. On the other hand, the difficulty of key management is greatly reduced because the key is just two vectors of length N. Meanwhile, only the private matrix f is encrypted on input, eliminating the encryption and transmission of A and A^T, which further improves the efficiency of outsourcing. Besides, the original signal matrix f is encrypted in its entirety, making the encryption process more efficient and concise compared with block encryption. However, the integral encryption operation does not affect the block flexibility of the cloud, i.e., the cloud can still perform 2D-DCT of any block size based on negotiation with the client.

5.3.2. Outsourcing Protocol for 2D-IDCT. The outsourcing protocol of 2D-IDCT is similar to that of 2D-DCT. First, the client also generates vectors u and v locally as the private key. To protect the privacy of the input privacy matrix F, the client performs

$$E(F) = F + u \cdot v^T. \tag{18}$$

Afterwards, the encrypted matrix $E(F)$ is sent to the cloud who perform $K \times K$ 2D-IDCT as

$$E\left(f_{i,j}\right) = A^T \cdot E\left(F_{i,j}\right) \cdot A. \tag{19}$$

The calculated result $E(f)$ is then returned to the client who uses the private key to decrypt $E(f)$ efficiently through

$$f_{i,j} = E\left(f_{i,j}\right) - \left(A^T \cdot u_i\right) \cdot \left(v_j^T \cdot A\right). \tag{20}$$

Finally, the correctness of resulting f is also verified by equation (17). We summarize the outsourcing protocol for 2D-IDCT as follows.

(1) *KeyGen* (1^λ). On input of a security parameter λ, the client generates the private key $\mathrm{KG} = (u, v)$.

(2) *ProbEnc* (Φ;*KG*). On input of the original problem $\Phi = F$ and the key KG, the client performs (18) to encrypt Φ to generate the encrypted problem $\Phi_{\mathrm{KG}} = E(F)$ and then sends Φ_{KG} to the cloud.

(3) *ProbSolve* (Φ_{KG}). On input of the encrypted problem Φ_{KG}, the cloud solves Φ_{KG} by (19). Then, the cloud returns the result $\Lambda_{\mathrm{KG}} = E(f)$ back, together with an empty proof Γ.

(4) *ResultDec* (Λ_{KG};*KG*). On input of the result Λ_{KG} of Φ_{KG} and the key KG, the client performs (20) to decrypt Λ_{KG} to obtain the result $\Lambda = f$ of the original problem Φ.

(5) *ResultVerify* (Λ;Γ). On input of the result Λ of Φ and the proof Γ, the client verifies the correctness of Λ by (17).

6. Protocol Construction

6.1. Privacy Analysis. For the outsourcing of 2D-DCT, the input privacy matrix is f and the output privacy matrix is F. In the proposed protocol, f is encrypted with two key vectors u and v by the privacy-preserving matrix addition. According to the theoretical proof in [26], the resulting matrix $E(f)$ is computationally indistinguishable in value from a random matrix R. That is, any probabilistic polynomial time distinguisher cannot distinguish the element $E(f)(i,j)$ of $E(f)$ from the element $R(i,j)$ of R for any $\forall\, i, j \in [1, N]$, except with negligible success probability. The success probability of the distinguisher decreases linearly with the expansion of the value range of the key vector. Therefore, the cloud cannot access any private plaintext contained in f from $E(f)$, thereby successfully protecting input privacy. Subsequently, the cloud performs block 2D-DCT of $K \times K$ on the encrypted matrix $E(f)$, resulting in $E(F)$. F is the result of performing $K \times K$ 2D-DCT operation on f. Since the polynomial adversary cannot find any valuable association between $E(f)$ and f in terms of value, the association between $E(F)$ and F is believed to be untraceable, i.e., the output privacy is also successfully protected.

For the outsourcing of 2D-IDCT, the input privacy matrix is F and the output privacy matrix is f. Based on the

consistency of the implementation approach between 2D-IDCT outsourcing and 2D-DCT outsourcing, we conclude that the privacy of input and output is also well protected in the outsourcing of 2D-IDCT.

6.2. Verification Analysis. In terms of result verification, two effects should be achieved. On the one hand, the correct results returned from the cloud should be accepted by the client with a probability close to 1. On the other hand, the erroneous results returned from the cloud should be rejected by the client with a probability close to 1.

In the proposed outsourcing protocol for 2D-DCT and 2D-IDCT, the returned results are all verified by the inference of Parseval's theorem as shown in (17). On the one hand, any correct 2D-DCT or 2D-IDCT transformation result obviously satisfies the inference of Parseval's theorem, thus reaching the first required effect. On the other hand, we emphasize that the cloud is unable to intentionally generate an erroneous result satisfying (17), because the cloud cannot even obtain the plaintext of the input privacy matrix and therefore cannot know the correct sum of squares of elements. In addition, we discuss the situation where the cloud returns an erroneous result at random, but it just meets (17). In this case, the sum of squares of the elements in the erroneous result matrix happens to be correct. For a grayscale block image with $K \times K$ pixels, since the grayscale value of each pixel is taken from the interval [0, 255], the possible values of the sum of squares of elements are in the interval $[0, K^2 \cdot 255^2]$. As long as K is not too small, the probability that the sum of squares of the elements in a randomly selected result matrix happens to be equal to the correct value is minimal. Moreover, the cloud will not try to avoid verification at any cost, because the motivation for the cloud to return erroneous results is considered to be cost-saving. Therefore, we believe that the proposed protocol also conforms to the second required effect.

6.3. Efficiency Analysis. In this section, we conduct a theoretical evaluation for the efficiency performance of the protocol. The evaluation is carried out from three parts: client-side overhead, cloud-side overhead, and communication overhead. Since the efficiency of 2D-DCT outsourcing is consistent with that of 2D-IDCT outsourcing, we take the outsourcing of 2D-DCT as an example for evaluation.

6.3.1. Client-Side Overhead. First, the client takes time $O(2N)$ on generating the private key vectors u and v and $O(N)$ on encrypting the two-dimensional signal matrix f. Then, the client performs (16) to decrypt the result $E(F)$ returned by the cloud, which takes time $O((2 + (1/K))N^2)$. Finally, the client verifies the decrypted result by (17), which takes time $O(2N^2)$. In general, the client needs to spend a total of $O((4 + (1/K))N^2 + 3N)$ time on the outsourcing of 2D-DCT.

6.3.2. Cloud-Side Overhead. The only operation required in the cloud is to perform block 2D-DCT operation of $K \times K$ on the encrypted matrix $E(f)$, which takes time $O(2KN^2)$.

6.3.3. Communication Overhead. During the outsourcing process, only the matrix f and matrix F need to be transmitted between the client and the cloud; thus, the communication overhead is quite small.

Note that the computation overhead in the cloud is the same as it would be for the client to perform the 2D-DCT locally. Based on the above analysis, we conclude that as long as $K \geq 3$, the client can get computing savings from outsourcing. Moreover, the larger the value of K, the more cost savings the client can achieve. For the most common case of $K = 8$, the client can reduce the computing overhead to about one quarter by outsourcing.

7. Simulation Experiment

In this section, we carry out a simulation experiment to further confirm the efficiency of the proposed protocol. The experiment was performed using MATLAB 2016b on a laptop with an Intel Core i5 processor and 8 GB RAM simulating a client. We simulate the situations of users performing 8×8 2D-DCT, fast 8×8 2D-DCT, and computing outsourcing, respectively. The fast 2D-DCT algorithm used for comparison here is proposed by Cho and Lee in [13]. We, respectively, calculated the time required for the client to perform 2D-DCT operation on a two-dimensional signal of different sizes in the three cases. We also assume that the communication between the client and the cloud can be achieved in a negligible time, so the experiment does not count the time spent on communication. The experimental results are shown in Table 1, which is the average value of 10 repeated experiments. The meaning of the parameters in Table 1 is as follows.

(i) $t_{\text{2D–DCT}}$ represents the time taken by the client to perform block 2D-DCT of 8×8 using the definition method

(ii) $t_{\text{fast 2D–DCT}}$ represents the time taken by the client to perform block 2D-DCT of 8×8 using the fast algorithm method

(iii) $t_{\text{outsourcing}}$ represents the time taken by the client to perform block 2D-DCT of 8×8 using the outsourcing method

(iv) $t_{\text{2D–DCT}}/t_{\text{fast 2D–DCT}}$ represents the performance gain to the client using the fast algorithm method compared to the definition method

(v) $t_{\text{2D–DCT}}/t_{\text{outsourcing}}$ represents the performance gain to the client using the outsourcing method compared to the definition method

(vi) $t_{\text{fast 2D–DCT}}/t_{\text{outsourcing}}$ represents the performance gain to the client using the outsourcing method compared to the fast algorithm method

As can be seen from Table 1, on the one hand, compared with the definition method, regardless of the value of size N, the outsourcing protocol can bring the client a performance gain of more than 4.43 times, which is consistent with the result of previous theoretical analysis. On the other hand, even compared with the fast 2D-DCT algorithm, outsourcing

Table 1: Experimental results.

Size N	t_{2D-DCT} (sec)	$t_{fast\ 2D-DCT}$ (sec)	$t_{outsourcing}$ (sec)	$\frac{t_{2D-DCT}}{t_{fast\ 2D-DCT}}$	$\frac{t_{2D-DCT}}{t_{outsourcing}}$	$\frac{t_{fast\ 2D-DCT}}{t_{outsourcing}}$
256	0.5530	0.2070	0.1110	2.6715	4.9820	1.8649
512	1.9076	0.7979	0.4301	2.3908	4.4352	1.8551
1024	8.7805	2.8735	1.7091	3.0557	5.1375	1.6813
2048	33.0577	11.9314	6.9559	2.7706	4.7525	1.7153
4096	127.8909	47.5288	28.0907	2.6908	4.5528	1.6920

can still bring more than 1.68 times of performance gain to the client, which further demonstrates the outstanding efficiency of the outsourcing method.

8. Conclusions

In this paper, we speed up 2D-DCT and 2D-IDCT from the point of view of computing outsourcing rather than algorithm optimization. Based on the privacy-preserving matrix addition, we realize the privacy protection of input and output data. Meanwhile, the inference of Parseval's theorem is used to verify the returned results. Furthermore, compared with the two possible solutions Attempt One and Attempt Two, the proposed formal protocol's features of one-round communication and simple key management guarantee the protocol's efficiency performance. The proposed outsourcing protocol is confirmed by theoretical analysis to solve three key challenges in computing outsourcing. Experimental results show that outsourcing is even more efficient than the fast 2D-DCT and 2D-IDCT algorithms. Thus, the adoption of the proposed outsourcing protocol could indeed be an excellent alternative for speeding up 2D-DCT and 2D-IDCT operations. In the future, we will strive to achieve efficient and secure outsourcing of other signal processing tools, such as DWT and NTT.

Data Availability

No data were used to support the findings of the study.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the Team Project of Collaborative Innovation in Universities of Gansu Province (No. 2017C-16) and the Major Project of Gansu University of Political Science and Law (No. 2016XZD12).

References

[1] F. Ernawan, M. N. Kabir, Z. Mustaffa, K. Moorthy, and M. Ramalinga, "An Improved Image Compression Technique using Large Adaptive DCT Psychovisual Thresholds," in *2019 IEEE 2nd International Conference on Knowledge Innovation and Invention (ICKII)*, pp. 561–564, Seoul, Korea (South), 2019.

[2] H. Kaur and R. Kaur, "Speech compression and decompression using DWT and DCT," *International Journal of Computer Technology and Applications*, vol. 3, no. 4, pp. 1501–1503, 2012.

[3] S. Heng, C. So-In, and T. G. Nguyen, "Distributed Image Compression Architecture over Wireless Multimedia Sensor Networks," *Wireless Communications and Mobile Computing*, vol. 2017, 21 pages, 2017.

[4] H.-J. Ko, C.-T. Huang, G. Horng, and S.-J. Wang, "Robust and blind image watermarking in DCT domain using inter-block coefficient correlation," *Information Sciences*, vol. 517, pp. 128–147, 2020.

[5] B. Y. Lei, I. Y. Soon, and Z. Li, "Blind and robust audio watermarking scheme based on SVD-DCT," *Signal Processing*, vol. 91, no. 8, pp. 1973–1984, 2011.

[6] L.-S. Liu, R.-H. Li, and Q. Gao, "A robust video watermarking scheme based on DCT," in *2005 International Conference on Machine Learning and Cybernetics*, vol. 8, pp. 5176–5180, Guangzhou, China, 2005.

[7] F. Cruz-Roldán, J. Piñeiro-Ave, J. L. Rojo-Álvarez, and M. Blanco-Velasco, "Simple Algorithms for Estimating the Symbol Timing Offset in DCT-Based Multicarrier Systems," *Wireless Communications and Mobile Computing*, vol. 2018, 8 pages, 2018.

[8] N. Al-Dhahir, H. Minn, and S. Satish, "Optimum DCT-based multicarrier transceivers for frequency-selective channels," *IEEE Transactions on Communications*, vol. 54, no. 5, pp. 911–921, 2006.

[9] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[10] D. Estrin, D. Culler, K. Pister, and G. Sukhatme, "Connecting the physical world with pervasive networks," *IEEE Pervasive Computing*, vol. 1, no. 1, pp. 59–69, 2002.

[11] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, no. 5, pp. 51–58, 2000.

[12] C. Ma, "A Fast Recursive Two Dimensional Cosine Transform," in *Intelligent Robots and Computer Vision VII*, Boston, MA, USA, 1989.

[13] N. I. Cho and S. U. Lee, "Fast algorithm and implementation of 2-D discrete cosine transform," *IEEE Transactions on Circuits and Systems*, vol. 38, no. 3, pp. 297–305, 1991.

[14] A. C. Hung and T. H.-Y. Meng, "A comparison of fast inverse discrete cosine transform algorithms," *Multimedia Systems*, vol. 2, no. 5, pp. 204–217, 1994.

[15] A. C. Yao, "Protocols for secure computations," in *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, pp. 160–164, Chicago, IL, USA, 1982.

[16] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st annual ACM symposium on Symposium on theory of computing - STOC '09*, pp. 169–178, Bethesda, MD, USA, 2009.

[17] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers," in *Advances in Cryptology – CRYPTO 2010*, pp. 465–482, 2010.

[18] M. Barbosa and P. Farshim, "Delegatable Homomorphic Encryption with Applications to Secure Outsourcing of

Computation," in *Lecture Notes in Computer Science*, pp. 296–312, 2012.

[19] K.-M. Chung, Y. Kalai, and S. Vadhan, "Improved Delegation of Computation Using Fully Homomorphic Encryption," *Advances in Cryptology – CRYPTO 2010*, pp. 483–501, 2010.

[20] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly Practical Verifiable Computation," in *2013 IEEE Symposium on Security and Privacy*, pp. 238–252, Berkeley, CA, USA, 2013.

[21] X. Lei, X. Liao, T. Huang, H. Li, and C. Hu, "Outsourcing Large Matrix Inversion Computation to A Public Cloud," *IEEE Transactions on Cloud Computing*, vol. 1, no. 1, pp. 1–1, 2013.

[22] F. Chen, T. Xiang, X. Lei, and J. Chen, "Highly Efficient Linear Regression Outsourcing to a Cloud," *IEEE Transactions on Cloud Computing*, vol. 2, no. 4, pp. 499–508, 2014.

[23] Y. Li, Z. L. Jiang, X. Wang, J. Fang, E. Zhang, and X. Wang, "Securely Outsourcing ID3 Decision Tree in Cloud Computing," *Wireless Communications and Mobile Computing*, vol. 2018, 10 pages, 2018.

[24] C. Wang, K. Ren, J. Wang, and Q. Wang, "Harnessing the cloud for securely outsourcing large-scale systems of linear equations," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1172–1181, 2013.

[25] X. Chen, X. Huang, J. Li, J. Ma, W. Lou, and D. S. Wong, "New algorithms for secure outsourcing of large-scale systems of linear equations," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 69–78, 2015.

[26] S. Salinas, C. Luo, X. Chen, W. Liao, and P. Li, "2015 Efficient secure outsourcing of large-scale sparse linear systems of equations," in *EEE Conference on Computer Communications (INFOCOM)*, vol. 4no. 1, pp. 26–39, Kowloon, Hong Kong, 2015.

[27] Y. Zhang, J. Jiang, Y. Xiang, Y. Zhu, L. Wan, and X. Xie, "Cloud-assisted privacy-conscious large-scale Markowitz portfolio," *Information Sciences*, vol. 527, pp. 548–559, 2020.

[28] X. Li, Y. Zhu, J. Wang, Z. Liu, Y. Liu, and M. Zhang, "On the soundness and security of privacy-preserving SVM for outsourcing data classification," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 906–912, 2018.

[29] Z. Zhang, M. Dong, L. Zhu et al., "Achieving Privacy-Friendly Storage and Secure Statistics for Smart Meter Data on Outsourced Clouds," *EEE Transactions on Cloud Computing*, vol. 7, pp. 638–649, 2017.

[30] Y. Zhu, X. Li, J. Wang, and J. Li, "Cloud-assisted secure biometric identification with sub-linear search efficiency," *Soft Computing*, vol. 24, no. 8, pp. 5885–5896, 2020.

[31] H. Li, K. Ota, M. Dong, A. Vasilakos, and K. Nagano, "Multimedia processing pricing strategy in GPU-accelerated cloud computing," *IEEE Transactions on Cloud Computing*, 2017.

[32] T. Bianchi, A. Piva, and M. Barni, "On the implementation of the discrete fourier transform in the encrypted domain," *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 1, pp. 86–97, 2009.

[33] T. Bianchi, A. Piva, and M. Barni, "Encrypted Domain DCT Based on Homomorphic Cryptosystems," *EURASIP Journal on Information Security*, vol. 2009, 12 pages, 2009.

[34] P. Zheng and J. Huang, "Discrete Wavelet Transform and Data Expansion Reduction in Homomorphic Encrypted Domain," *IEEE Transactions on Image Processing*, vol. 22, no. 6, pp. 2455–2468, 2013.

[35] A. Pedrouzo-Ulloa, J. R. Troncoso-Pastoriza, and F. Perez-Gonzalez, "Number Theoretic Transforms for Secure Signal Processing," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 5, pp. 1125–1140, 2017.

[36] K. Han, M. Hhan, and J. H. Cheon, "Improved Homomorphic Discrete Fourier Transforms and FHE Bootstrapping," *IEEE Access*, vol. 7, pp. 57361–57370, 2019.

[37] X. Xiao, J. Huang, Y. Zhang, and X. He, "Efficient and Secure Outsourcing of DFT, IDFT, and Circular Convolution," *IEEE Access*, vol. 7, pp. 60126–60133, 2019.

[38] Y. Zhang, Y. Xiang, L. Y. Zhang, L.-X. Yang, and J. Zhou, "Efficiently and securely outsourcing compressed sensing reconstruction to a cloud," *Information Sciences*, vol. 496, pp. 150–160, 2019.

[39] Y. Zhang, J. Zhou, Y. Xiang et al., "Computation Outsourcing Meets Lossy Channel: Secure Sparse Robustness Decoding Service in Multi-Clouds," *IEEE Transactions on Big Data*, 2017.

[40] X. Lei, X. Liao, T. Huang, and F. Heriniaina, "Achieving security, robust cheating resistance, and high-efficiency for outsourcing large matrix multiplication computation to a malicious cloud," *Information Sciences*, vol. 280, pp. 205–217, 2014.

[41] X. Zhang, S. Liu, H. Cui, and T. Chen, "Reading the Underlying Information From Massive Metagenomic Sequencing Data," *Proceedings of the IEEE*, vol. 105, no. 3, pp. 459–473, 2017.

[42] S. Fu, Y. Yu, and M. Xu, "A Secure Algorithm for Outsourcing Matrix Multiplication Computation in the Cloud," in *Proceedings of the Fifth ACM International Workshop on Security in Cloud Computing - SCC '17*, pp. 27–33, Abu Dhabi United Arab Emirates, 2017.

[43] X. Lei, X. Liao, T. Huang, and H. Li, "Cloud Computing Service: The Case of Large Matrix Determinant Computation," *IEEE Transactions on Services Computing*, vol. 8, no. 5, pp. 688–700, 2015.

[44] W. Liao, C. Luo, S. Salinas, and P. Li, "Efficient Secure Outsourcing of Large-Scale Convex Separable Programming for Big Data," *IEEE Transactions on Big Data*, vol. 5, no. 3, pp. 368–378, 2017.

[45] Y. Zhang, X. Xiao, L.-X. Yang, Y. Xiang, and S. Zhong, "Secure and Efficient Outsourcing of PCA-Based Face Recognition," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1683–1695, 2020.