Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Efficiently and securely outsourcing compressed sensing reconstruction to a cloud

Yushu Zhang[a,b,c,d], Yong Xiang[b,*], Leo Yu Zhang[b], Lu-Xing Yang[b], Jiantao Zhou[e]

[a] *College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China*
[b] *School of Information Technology, Deakin University, Victoria 3125, Australia*
[c] *Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China*
[d] *Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 210023, China*
[e] *Department of Computer and Information Science, Faculty of Science and Technology, University of Macau, Macau, China*

ARTICLE INFO

Article history:
Received 29 September 2018
Revised 28 April 2019
Accepted 10 May 2019
Available online 11 May 2019

Keywords:
Compressed sensing
Compressed sensing reconstruction
outsourcing
Cloud computing
Asymmetric verification

ABSTRACT

Compressed sensing has considerable potential for utilization in various fields owing to its efficient sampling process, but its reconstruction complexity is extremely high. For resource-constrained users, performing the compressed sensing reconstruction (CSR) task is impractical. In particular, the emergence of big data makes this task increasingly time-consuming. Cloud computing resources are abundant and can be employed to solve this task. However, owing to the lack of trust in the cloud, it is necessary to outsource the CSR task without privacy leakages. In this study, we design an efficient secure outsourcing protocol for the CSR task. In the basic outsourcing service model, a client samples a signal via a secure measurement matrix and then sends the acquired measurements to the cloud for CSR outsourcing. The reconstructed signal can not only be utilized by the client, but also by other users. The proposed outsourcing scheme is highly efficient and privacy-preserving, based on three aspects. First, the sensing matrix employed for reconstruction is assumed to be public, because it has a significantly larger size than the signal and consumes considerable resources if encrypted and transmitted. Second, a secret orthogonal sparsifying basis is contained only in the measurement matrix, rather than the sensing matrix. Third, a user can verify the reconstructed signal by leveraging the keys, which are the unique information shared between the client and user. We also demonstrate the privacy and analyze the efficiency of the proposed CSR outsourcing protocol, both theoretically and experimentally.

1. Introduction

In recent years, cloud computing has been attracting widespread attention in both academia and industry [1,13–16]. Owing to resource restrictions or to save costs, a client can transfer their computational tasks to the cloud for processing. In particular, along with the emergence and development of big data, the demand for such transfers is continuously increasing. However, in view of task sensitivity and cloud distrust, security concerns must be taken into account [7,25–27,29]. This encourages the client to consider how to let the cloud fulfill dependable task processing over an encrypted domain, which inevitably introduces some new challenges [34]. The first challenge is to protect the privacy of both the client's input tasks

* Corresponding author.
E-mail address: yxiang@deakin.edu.au (Y. Xiang).

https://doi.org/10.1016/j.ins.2019.05.024
0020-0255/© 2019 Elsevier Inc. All rights reserved.

and the cloud's output results. The client suitably encrypts the input task and then outsources the encrypted task to the cloud for processing. After completing the task, the cloud should not be able to infer sensitive information from the result. Hence, it is desirable that the encryption operation ensures not only the input/output privacy, but also the successful completion of the task. The second challenge is to be able to verify the result returned by the cloud. For various reasons, there may exist hardware faults and software bugs, resulting in incorrect computing. In addition, for intentional reasons the cloud may become lazy and want to muddle through, thus simply returning an arbitrary result. The cloud may even leverage this incorrect result to access useful knowledge on the task for financial incentive. Consequently, this requires the client to be able to determine whether the cloud behaves faithfully and returns a correct result. The third challenge concerns the efficiency of handling the computational task. This demands that the time required for the client to perform encryption operations should be substantially less than that required to complete the unencrypted task on their own. Meanwhile, the time required for the cloud to process the encrypted task should be kept as close as possible to that required for the client to complete the unencrypted task themselves.

Compressed sensing (CS) is an efficient signal sampling technique [4,12]. Given two necessary conditions, it can recover a signal from a set of far fewer samples than required by the Shannon–Nyquist sampling theorem. These two conditions include sparsity, which requires the signal to be sparse in some domain, and the restricted isometry property (RIP), which should be satisfied by the sensing matrix. (CS) has a low linear encoding complexity, but a very high cubic reconstruction complexity. Because reconstructing and storing signals will consume a large amount of computational resources and occupy considerable storage space, it is unpractical, or even impossible, for this to be performed by resource-constrained devices, such as sensors and mobile terminals. Motivated by these challenges, in this study we propose outsourcing the compressed sensing reconstruction (CSR) task to the cloud.

A basic system service model involves a client sampling an original signal based on the CS technique to generate measurements, which are then sent to the cloud for storage and CSR task solving. When the client or other users make a request, the cloud can solve the CSR task and return the reconstructed signal. In general, a measurement matrix is handled by the client, while a sensing matrix, which is a product of the measurement matrix and an orthogonal sparsifying basis, is utilized on the cloud side. To protect the privacy of the signal, the measurement matrix and the generated measurements must be encrypted ahead of transmission to the cloud. However, in real-time applications it is not inadvisable for each signal to encrypt and transmit the measurement matrix, because the measurement matrix is significantly large, and will occupy too many resources if encrypted and transmitted. To address this issue, we adopt the primary assumption that the measurement matrix is public knowledge, which is known by both the client and cloud.

Keeping the measurement matrix public presents a new issue of how to encrypt the signal in the sampling process. To handle this problem, we can migrate the function of the orthogonal sparsifying basis from the sensing matrix to the measurement matrix. Then, we set the secret orthogonal sparsifying basis to provide privacy protection. As a result, the secret orthogonal sparsifying basis is now part of the measurement matrix rather than the sensing matrix. Specifically, the newly generated measurement matrix is a product of the original measurement matrix and the inverse matrix of the secret basis, and the new sensing matrix is the original measurement matrix. It is worth noting that in our model the sensing matrix used on the cloud side stays public, and the measurement matrix utilized by the client remains secret. Furthermore, the encryption mechanism taken by this function migration should not affect the sparsity and the RIP, otherwise the solving of the CSR task would be compromised. In Section 4.2, we show that neither the sparsity nor the RIP are affected.

Furthermore, other users face the issue of how to verify the reconstructed signal. The client can easily verify the reconstructed signal by leveraging the auxiliary information, including the measurement matrix and measurements, whereas other users do not have this auxiliary information. It is natural to ask whether the client can send the auxiliary information to other users. Unfortunately, this is not practical, as the amount of information is often very large, indeed significantly larger than the size of the signal to be sampled, and would consume considerable communication resources. Verification without using the auxiliary information is called *asymmetric verification*. In our work, we adopt keys, which are the unique information shared between the client and other users, to achieve a feasible asymmetric verification scheme. This also represents a deterministic verification approach, i.e., the reconstructed signal can be accurately distinguished as correct or incorrect. More interestingly, it is a partial verification scheme, meaning that a part of the reconstructed signal passing the verification can be used to reconstruct the signal to a satisfactory quality.

The main contributions of this paper can be summarized as follows.

- We propose two efficient CSR task-outsourcing protocols, simultaneously achieving the goals of correctness, privacy, client/user verification, and efficiency.
- We propose the novel idea of a publicly known sensing matrix, avoiding the transmission of the sensing matrix between the client and cloud, and thus saving communication resources.
- We devise an encryption approach by migrating the function of the orthogonal sparsifying basis into the original measurement matrix, while maintaining the sparsity and RIP.
- We design an asymmetric and partial verification approach for users, which is simple to use.

The remainder of this paper is organized as follows. The next section discusses related work. Section 3 introduces the basics of CSR, the system and threat model, and the design goals. Section 4 describes the protocol construction in detail, including the basic idea, key design and generation, client/user verification, and formal protocols. Section 5 provides further investigations in terms of privacy guarantees and a theoretical analysis, followed by a performance evaluation in Section 6. Finally, the last section concludes the paper.

2. Related work

CSR outsourcing is closely connected with *matrix computation* and *equation solving* outsourcing. Lei et al. outsourced large matrix inversion [24], large matrix multiplication [22], and large matrix determinant computations [23] to the cloud by employing some transformations including random permutations and value altering. A randomized Monte Carlo verification algorithm with a one-sided error is employed in these outsourcing mechanisms. Wang et al. proposed large-scale systems of linear equation [38,40] and linear programming outsourcing [36,37] by applying some matrices/vectors to the original equations. Chen et al. presented some protocols to improve these approaches [36,38,40] and acquire significant performance gains [9]. Unlike in [9,38,40], a sparse matrix is utilized in [10] to construct a novel linear equation solving outsourcing method, which only requires one round of communication between the client and cloud and can detect cloud misbehavior with a probability of one. To reduce the huge number of memory input/output operations, Sergio et al. exploited the sparsity in linear systems of equations to obtain an efficient and secure outsourcing scheme [30]. Ding et al. analyzed and improved a privacy-preserving conjugate gradient method algorithm to securely outsource large-scale systems of linear equations [11]. Chen et al. devised a linear regression outsourcing technique, and designed two protocols for different application scenarios [8]. Luo et al. proposed a secure outsourcing algorithm for nonlinearly constrained nonlinear programming, and then parallelized this algorithm to accelerate computations in the cloud [28].

As demonstrated above, various matrix computation and equation solving outsourcing protocols have been studied over recent years [32]. However, no efficient schemes for secure CSR task outsourcing have so far been proposed. If utilized in CSR outsourcing, the transformation techniques in the abovementioned outsourcing schemes may lead to the severe problem that sparsity or the RIP no longer holds. Meanwhile, these protocols verify the results based on the task itself, and do not achieve asymmetric verification. In contrast, our protocol can efficiently outsource CSR to the cloud with effective encryption and asymmetric verification techniques.

CSR outsourcing represents a crossover study area of CS and information security. The area of CS applied in information security was reviewed in our studies [43,44,47] from various aspects, including theoretical security, application security, wireless communications, multimedia data, cloud computing, and the Internet of Things. CSR outsourcing combined with CS and cloud security belongs to the application security and cloud computing domains. Kang et al. investigated a series of privacy-preserving compressive multimedia applications in the cloud, including multimedia compression, adaptation, editing/manipulation, enhancement, retrieval, and recognition [20]. Wang et al. proposed a privacy-preserving watermark detection framework in cloud computing, based on CS and secure multiparty computation [42]. In [20,42], the authors did not consider the problem of how to outsource the CSR task to the cloud. Wang et al. scheduled privacy-assured image reconstruction service outsourcing [41] and healthcare signal reconstruction service outsourcing [39], both of which involved the problem of CSR outsourcing. In [19], Hu et al. introduced an outsourced image reconstruction and identity authentication service based on a CS technique in the cloud. The studies [19,39,41] considered a solution to linear programming outsourcing through a sequence of transformations on the CSR task itself. However, these transformations are impractical for real-time applications, because after each signal is sampled using the CS technique only the measurements are expected to be outsourced to the cloud, not including the measurement matrix. In addition, support-set-assured sparse reconstruction service outsourcing was discussed in [48], which is related to CSR outsourcing. However, the authors did not consider the verification problem. In the proposed CSR outsourcing protocol, the client samples each signal and then only sends the corresponding measurements to the cloud, and a reasonable verification method is proposed. Both the verification method and encryption mechanism are effective but simple. In addition, there have been some studies concerning scheduling algorithms for cloud computing and big data [2,31,33,35], but these have not involved secure computation outsourcing.

3. Problem formulation

3.1. Compressed sensing reconstruction

The fundamental Shannon–Nyquist sampling theory describes sampling from the perspective of a limited signal band, and has been widely considered as key for acquiring and reconstructing data. Nevertheless, the resulting number of required measurements can be so large that the storage becomes infeasible, and the acquisition time becomes too long. CS is a new sampling theory, providing theoretical conditions to ensure the exact recovery of signals from a small number of linear projections below the Nyquist rate. The underlying property allowing the development of this idea is the sparsity of the signal. A signal $\mathbf{x}$ of length N is said to be k-sparse or compressible if $\mathbf{x}$ can be well approximated using only $k \ll N$ coefficients under some orthogonal sparsifying basis $\mathbf{D}$, as follows:

$$\mathbf{x} = \mathbf{D}\mathbf{s}, \tag{1}$$

where $\mathbf{s}$ is the transform coefficient vector, which has at most k significant nonzero entries. This theory means that $\mathbf{x}$ can be acquired by the following random measurement:

$$\mathbf{y} = \mathbf{A}\mathbf{x}, \tag{2}$$

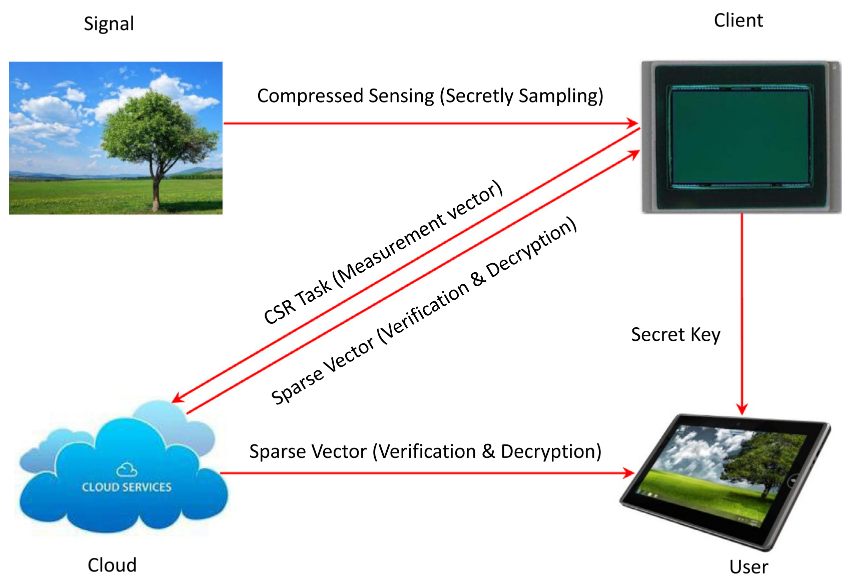

Signal
Client
Compressed Sensing (Secretly Sampling)
CSR Task (Measurement vector)
Sparse Vector (Verification & Decryption)
Secret Key
CLOUD SERVICES
Sparse Vector (Verification & Decryption)
Cloud
User

Fig. 1. System model of CSR task outsourcing in the cloud.

where **A** is an $M \times N$ $(M < N)$ random measurement matrix, and **y** represents the measurement coefficient vector. Here, **x** can be faithfully recovered from only a very small number of measurements through l_1-minimization:

$$\min \ \|\mathbf{s}\|_1 \quad s.t. \ \mathbf{y} = \mathbf{\Theta s}, \tag{3}$$

where $\mathbf{\Theta}$ is the sensing matrix, with $\mathbf{\Theta} = \mathbf{AD}$, and the measurement matrix **A** should be highly incoherent with the orthogonal sparsifying basis **D**.

3.2. System and threat model

As shown in Fig. 1, in the system model a client samples a signal of interest using a measurement matrix, and acquires the corresponding measurement values. Owing to a lack of storage and computing resources, the client wants to utilize the cloud for task processing. The CSR task that the client wants to outsource can be expressed as a convex optimization problem, denoted as $\Pi = (\mathbf{y}, \mathbf{\Theta})$, as in (3). To ensure the task privacy, the client needs to suitably encrypt Π with a key K, and then outsource the encrypted version Π_K to the cloud, where Π_K is solved in a privacy-preserving manner to obtain the result, denoted as $\hat{\mathbf{s}}$. This result can be returned to the client itself, or sent to a user if requested. Regardless of whether the client or a user receives the result, it must be verified. If the verification succeeds, then decrypting the result reveals the real solution for the original task.

In the security threat model, we consider a cloud with malicious behavior. The cloud may have a direct incentive to reveal sensitive information of the input Π_K and output $\hat{\mathbf{s}}$, and even deliberately sabotage the computation, e.g., by returning a random result to save computing resources and attempting not to be caught.

3.3. Design goals

For the aforementioned model, a practical and efficient protocol should be designed to achieve the following design goals.

- Correctness
  The protocol can achieve success in fulfilling the CSR task, as long as both the client and cloud follow the protocol honestly.
- Privacy
  The protocol can protect the input/output privacy of the CSR task. In other words, the cloud cannot obtain meaningful information from either the encrypted input task or encrypted output result.
- Client/User verification
  The protocol can verify whether the output result of the cloud is correct, and verify that it does not cheat either the client or user. Even though the user does not have the original task in hand, no false result can pass the verification with a non-negligible probability.
- Efficiency

The protocol can substantially reduce the local computational burden on the client/user side compared to that required for the client to complete the CSR task on their own. Furthermore, the amount of computation required to process the encrypted task in the cloud can be made as close as possible to that required for the original unencrypted task.

4. Protocol construction

4.1. Basic idea

Assume that a client wants to harness the CS technique to capture a signal $\mathbf{x}$ from the natural world. Let us first consider a general case. After sampling, the client transmits the measurement matrix $\mathbf{A}$ and measurement vector $\mathbf{y}$ to the cloud, which calculates this CSR task $\Pi = (\mathbf{y}, \mathbf{\Theta})$. In this scenario, the client has the measurement matrix $\mathbf{A}$ in hand, while the cloud owns the sensing matrix $\mathbf{\Theta} = \mathbf{AD}$, and both sides have a common measurement vector $\mathbf{y}$. In order to guarantee the privacy of $\mathbf{x}$, the client must utilize a secret measurement matrix $\mathbf{A}$. Nevertheless, $\mathbf{A}$ must be shared with the cloud for successful decoding, and thus it seems almost impossible to directly handle the task in this general case. In addition to being unable to share the secret measurement matrix, sending an updated $\mathbf{A}$ immediately or frequently is impractical to achieve security enhancement and save the communication cost, as it contains a much larger amount of data than $\mathbf{x}$. However, it is interesting that the orthogonal sparsifying basis $\mathbf{D}$ is only used in the sensing matrix of the cloud, but not in the measurement matrix of the client. Can we consider a *function migration* for $\mathbf{D}$, i.e., a scenario where $\mathbf{D}$ is only dealt with by the client rather than the cloud? If so, $\mathbf{D}$ will no longer be shared with the cloud. As a result, we can make the measurement matrix $\mathbf{A}$ public, similar to a public key in asymmetric cryptography, which is always known to the cloud. This avoids frequently transmission of $\mathbf{A}$ to the cloud. In the following, we explain how this idea works.

Let the measurement matrix $\mathbf{A}$ used for signal sampling be publicly known. If $\mathbf{D}_K$ is a secret matrix, then the signal $\mathbf{x}$ and the sparse representation $\mathbf{s}$ will consist of plaintext and ciphertext, respectively, because $\mathbf{s} = \mathbf{D}_K^{-1}\mathbf{x}$. This is a simple cryptosystem, which is secure against some potential attacks, such as known/chosen plaintext attacks with $\mathbf{D}_K$ being a session key. It is highly expected that the cloud can solve the task with respect to $\mathbf{A}$. Without loss of generality, this task should take the following form:

$$\min \|\mathbf{s}\|_1 \quad s.t.\, \mathbf{y} = \mathbf{As}. \tag{4}$$

Apparently, after the cloud fulfills the above task $\Pi_K = (\mathbf{y}, \mathbf{A})$, the result $\mathbf{s}$ remains in an encrypted form, such that the output privacy is protected. Note that the received result is not necessarily the same as the original result, and here we use the original expression for convenience in stating the problem. On the other hand, the task can be simplified as $\Pi_K = (\mathbf{y})$, because $\mathbf{A}$ is publicly known. Further, plugging $\mathbf{s} = \mathbf{D}_K^{-1}\mathbf{x}$ into $\mathbf{y} = \mathbf{As}$, we obtain

$$\mathbf{y} = \mathbf{AD}_K^{-1}\mathbf{x} = \mathbf{\Theta}_K\mathbf{x}, \tag{5}$$

where $\mathbf{\Theta}_K = \mathbf{AD}_K^{-1}$. This indicates that the client can leverage $\mathbf{\Theta}_K$ to sample the signal $\mathbf{x}$. The proposed CSR outsourcing technique will employ the basic idea of this function migration. It seems obvious that this function migration appears to exchange the measurement and sensing matrices between the client and cloud. However, whether this is reasonable and further details must be investigated.

4.2. Key design and generation

In CS theory, the sensing matrix $\mathbf{\Theta}$ should satisfy the RIP of order k if there exists a constant $\delta_k \in (0, 1)$ such that

$$(1 - \delta_k)\|\mathbf{s}\|_2^2 \le \|\mathbf{\Theta s}\|_2^2 \le (1 + \delta_k)\|\mathbf{s}\|_2^2, \tag{6}$$

for all k-sparse signals $\mathbf{s}$ [5]. For a stronger security guarantee, the design of the key $\mathbf{D}_K$ (or $\mathbf{D}_K^{-1}$) is likely to break the RIP for $\mathbf{\Theta}_K$, i.e., $\mathbf{\Theta}_K$ will no longer satisfy the RIP. Moreover, the measurement matrix $\mathbf{\Theta}_K$ used for sampling in the proposed CSR outsourcing approach looks like a sensing matrix. Therefore, we should determine how to approach the relationship between $\mathbf{D}_K$ and $\mathbf{\Theta}_K$. We can give a satisfactory answer by observing the equation

$$\mathbf{y} = \mathbf{AD}_K^{-1}\mathbf{x} = \mathbf{AD}_K^{-1}\mathbf{D}_K\mathbf{s} = \mathbf{As}. \tag{7}$$

Even though the design of $\mathbf{D}_K$ leads $\mathbf{\Theta}_K$ to not satisfy the RIP, it does not affect the RIP of $\mathbf{A}$ utilized in the cloud, which can be a Gaussian or Bernoulli random matrix with independent and identically distributed entries from a sub-Gaussian distribution [6]. However, it is worth mentioning that the design of $\mathbf{D}_K$ may impact the sparsity of $\mathbf{s}$, resulting in a poor reconstruction performance.

For different types of signals, the client can select the matching $\mathbf{D}_K$. Note that although the integral of $\mathbf{AD}_K^{-1}$ is used for sampling, in principle it can be interpreted as an encryption operation for $\mathbf{D}_K$ in advance, and subsequently a subsampling operation for $\mathbf{A}$. In general, there are two common methods for constructing $\mathbf{D}_K$ on the premise of ensuring the good sparsity of $\mathbf{s}$. One method is to utilize some optical transforms with parameters, such as the discrete fractional Fourier [3] and multiple-parameter discrete fractional transforms [18,21], where the parameters (orders) often indicate the rotation angle of the transform in the time–frequency plane, and can serve as keys. These have mathematical properties such as rotations, periodicity, zero rotation, and so on. They also possess similar properties to the cases with no parameters, thus simultaneously guaranteeing the sparsity and providing security. So far, optical transforms with parameters have been extensively employed in data encryption [45,46,49,50]. The other method is to utilize secret elementary transformations on classical orthogonal transform matrices, such as the discrete cosine and discrete wavelet transforms. The secret elementary transformations involve randomly exchanging two rows, multiplying some row by a secret non-zero number, and adding some row multiplied by a secret number to another row. It is not difficult to prove that these secret transformations do not alter the sparsity of **s**. This is because the transformations are equivalent to operations on **s**, i.e., exchanging two entries, multiplying some entry by a secret non-zero number, and adding an entry multiplied by a secret number to another entry, respectively. The two abovementioned methods can easily be implemented in practice by optical hardware. Meanwhile, the generation of $\mathbf{D}_K$ can be controlled by only one or a few keys.

4.3. Client/user verification

On the client side, an instinctive approach is that upon receiving $\hat{\mathbf{s}}$ returned by the cloud, the client decrypts this to obtain $\hat{\mathbf{x}} = \mathbf{D}_K\hat{\mathbf{s}}$, and then checks whether $\mathbf{y} = \mathbf{\Theta}_K\hat{\mathbf{x}}$ holds using randomized verification. This approach is frequently employed in some matrix computation outsourcing methods [23,24,40]. It aims to acquire a faithful result with an overwhelming probability using few resources, because checking all the entries of $\hat{\mathbf{s}}$ will have a very high computational complexity, which is unpractical, especially for resource-constrained clients. Generally, randomly selecting a only small number of entries, considerably smaller than the size of $\hat{\mathbf{x}}$, verifies whether a false result from a cheating cloud can pass the verification process with a non-negligible probability. In contrast, a simple and absolute verification approach is employed in the proposed outsourcing method. Upon receiving $\hat{\mathbf{s}}$, it is unnecessary for the client to decrypt this prior to verification, as decryption may waste computational resources if the result is incorrect. Instead, the client can directly check whether $\mathbf{y} = \mathbf{A}\hat{\mathbf{s}}$ is correct. Considering the sparsity of $\hat{\mathbf{s}}$, the client just needs to handle the k non-zero elements. Because $k < M < N$ and $k \ll N$, a very low computational complexity is guaranteed. More importantly, this is an absolute verification approach, i.e., the result can be verified with 100% completeness. Once $\hat{\mathbf{s}}$ passes verification, the client computes $\hat{\mathbf{x}} = \mathbf{D}_K\hat{\mathbf{s}}$.

On the user side, no prior information on either **y** or **A** contributes to the verification, and therefore it is an extremely hard task to verify. However, we can still solve this by virtue of a unique $\mathbf{D}_K$ in the user's hand, which is a key shared by the client and user. Here, we assume a unique key to facilitate the analysis, despite the fact that $\mathbf{D}_K$ could possibly be generated by several keys. In addition to the key K_1 used for the outsourcing of a signal, we assign a different key K_2 for the signal. Hence, the client performs sampling twice, as follows:

$$\begin{cases} \mathbf{y}_1 = \mathbf{A}\mathbf{D}_{K_1}^{-1}\mathbf{x} \\ \mathbf{y}_2 = \mathbf{A}\mathbf{D}_{K_2}^{-1}\mathbf{x} \end{cases}. \tag{8}$$

The two tasks outsourced to the cloud are

$$\min \|\mathbf{s}_1\|_1 \quad s.t.\ \mathbf{y}_1 = \mathbf{A}\mathbf{s}_1 \tag{9}$$

and

$$\min \|\mathbf{s}_2\|_1 \quad s.t.\ \mathbf{y}_1 = \mathbf{A}\mathbf{s}_2, \tag{10}$$

which are masked as $\Pi_{K_1} = (\mathbf{y}_1)$ and $\Pi_{K_2} = (\mathbf{y}_2)$, respectively. After receiving $\mathbf{s}_1$ and $\mathbf{s}_2$, the user can implement the verification using the following equation:

$$\mathbf{D}_{K_1}\hat{\mathbf{s}}_1 \overset{?}{=} \mathbf{D}_{K_2}\hat{\mathbf{s}}_2. \tag{11}$$

In fact, we take advantage of two sparse representations, $\hat{\mathbf{x}} = \mathbf{D}_{K_1}\hat{\mathbf{s}}_1 = \mathbf{D}_{K_2}\hat{\mathbf{s}}_2$, for a signal **x**. In this manner, because $k \ll N$, handling $2k$ non-zero entries still has a low complexity, and the verification is absolute. One might consider a repeated verification technique to check two results corresponding to the same key. However, this is not feasible, because the cloud could fake the same result for the same **y**.

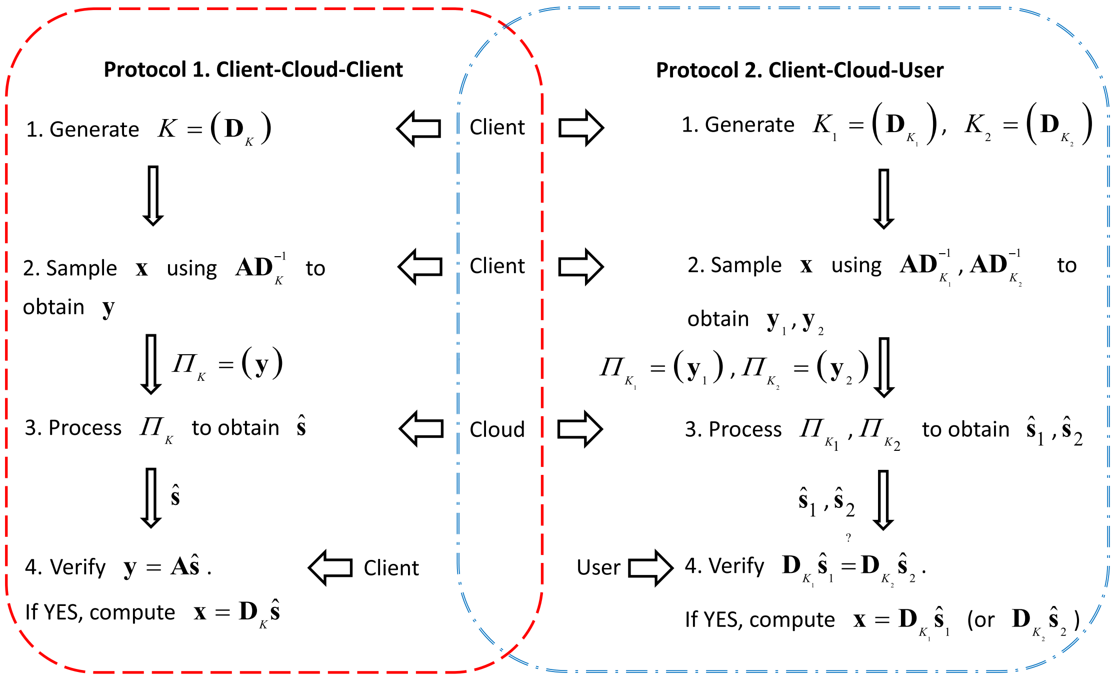
4.4. Formal protocols

Now, we introduce two formal protocols for the proposed CSR outsourcing approach, which are aimed at the client and user sides, respectively. Both of these contain four sub-algorithms (*KeyGen, TaskGen, TaskProc*, and *ResultVerify*). The two protocols are separately referred to as *Client-Cloud-Client* and *Client-Cloud-User*, and their difference lies in whether the client or user receives the CSR result from the cloud. Fig. 2 presents a flow chart for a better comparative understanding of the two protocols.

PROTOCOL 1: Client-Cloud-Client.

KeyGen$(1^\lambda) \rightarrow$ A key generation algorithm that is run by the client. This takes a security parameter λ as input and returns a key that is a secret orthogonal basis, $K = (\mathbf{D}_K)$.

TaskGen$(\mathbf{A}, K, \mathbf{x}) \rightarrow$ A task generation algorithm that is run by the client. This takes a measurement matrix $\mathbf{A}\mathbf{D}_K^{-1}$ and signal **x** as inputs, and returns a task that is a measurement vector, $\Pi_K = (\mathbf{y})$.



Protocol 1. Client-Cloud-Client
1. Generate $K = (\mathbf{D}_K)$
2. Sample $\mathbf{x}$ using $\mathbf{AD}_K^{-1}$ to obtain $\mathbf{y}$
$\Pi_K = (\mathbf{y})$
3. Process Π_K to obtain $\hat{\mathbf{s}}$
$\hat{\mathbf{s}}$
4. Verify $\mathbf{y} = \mathbf{A}\hat{\mathbf{s}}$.
If YES, compute $\mathbf{x} = \mathbf{D}_K\hat{\mathbf{s}}$
Client
Client
Cloud
Client
User
Protocol 2. Client-Cloud-User
1. Generate $K_1 = (\mathbf{D}_{K_1})$, $K_2 = (\mathbf{D}_{K_2})$
2. Sample $\mathbf{x}$ using $\mathbf{AD}_{K_1}^{-1}, \mathbf{AD}_{K_2}^{-1}$ to obtain $\mathbf{y}_1, \mathbf{y}_2$
$\Pi_{K_1} = (\mathbf{y}_1), \Pi_{K_2} = (\mathbf{y}_2)$
3. Process Π_{K_1}, Π_{K_2} to obtain $\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2$
$\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2$
4. Verify $\mathbf{D}_{K_1}\hat{\mathbf{s}}_1 \overset{?}{=} \mathbf{D}_{K_2}\hat{\mathbf{s}}_2$.
If YES, compute $\mathbf{x} = \mathbf{D}_{K_1}\hat{\mathbf{s}}_1$ (or $\mathbf{D}_{K_2}\hat{\mathbf{s}}_2$)

Fig. 2. Flow chart of two protocols.

TaskProc$(\mathbf{A}, \Pi_K) \rightarrow$ A task processing algorithm that is run by the cloud. This takes a sensing matrix $\mathbf{A}$ and task Π_K as inputs, and returns a result $\hat{\mathbf{s}}$.

ResultVerify$(\mathbf{A}, \mathbf{y}, \hat{\mathbf{s}}) \rightarrow$ A result verification algorithm that is run by the client. This takes a sensing matrix $\mathbf{A}$, measurement vector $\mathbf{y}$, and result $\hat{\mathbf{s}}$ as inputs, and returns YES or NO. If YES, then the client computes $\mathbf{x} = \mathbf{D}_K\hat{\mathbf{s}}$. If NO, then the client rejects the result.

PROTOCOL 2. Client-Cloud-User.

KeyGen$(1^\lambda) \rightarrow$ A key generation algorithm that is run by the client. This takes a security parameter λ as input and returns two keys $K_1 = (\mathbf{D}_{K_1})$ and $K_2 = (\mathbf{D}_{K_2})$, each of which is a secret orthogonal basis.

TaskGen$(\mathbf{A}, K_1, K_2, \mathbf{x}) \rightarrow$ A task generation algorithm that is run by the client. This takes two measurement matrices $\mathbf{AD}_{K_1}^{-1}$ and $\mathbf{AD}_{K_2}^{-1}$ and a signal $\mathbf{x}$ as inputs, and returns two tasks $\Pi_{K_1} = (\mathbf{y}_1)$ and $\Pi_{K_2} = (\mathbf{y}_2)$, each of which is a measurement vector.

TaskProc$(\mathbf{A}, \Pi_{K_1}, \Pi_{K_2}) \rightarrow$ A task processing algorithm that is run by the cloud. This takes a sensing matrix $\mathbf{A}$ and two tasks Π_{K_1} and Π_{K_2} as inputs, and returns two results $\hat{\mathbf{s}}_1$ and $\hat{\mathbf{s}}_2$.

ResultVerify$(K_1, K_2, \hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2) \rightarrow$ A result verification algorithm that is run by the user. This takes two keys $K_1 = (\mathbf{D}_{K_1})$ and $K_2 = (\mathbf{D}_{K_2})$ and two results $\hat{\mathbf{s}}_1$ and $\hat{\mathbf{s}}_2$ as inputs, and returns YES or NO. If YES, then the user computes $\mathbf{x} = \mathbf{D}_{K_1}\hat{\mathbf{s}}_1$ (or $\mathbf{D}_{K_2}\hat{\mathbf{s}}_2$). If NO, then the user rejects the results.

5. Further investigations

5.1. Privacy guarantee

Concerning the input privacy of the CSR task, we assume that the cloud that receives $\mathbf{y}$ attempts to directly reveal $\mathbf{x}$ based on the formula $\mathbf{y} = \boldsymbol{\Theta}_K\mathbf{x}$. However, because $\boldsymbol{\Theta}_K$ is unknown the input privacy of $\mathbf{x}$ can easily be ensured. Regarding the output privacy of the CSR task, we assume that the cloud that acquires $\hat{\mathbf{s}}$ attempts to disclose $\mathbf{x}$ according to $\mathbf{x} = \mathbf{D}_K\hat{\mathbf{s}}$. To do so, the cloud must crack $\mathbf{D}_K$. We previously mentioned two common methods for constructing $\mathbf{D}_K$. First, when parameterized optical transforms serve as $\mathbf{D}_K$, the parameters act as the keys. Second, when secret elementary transformations on the classical orthogonal transform matrices are used as $\mathbf{D}_K$, rather than determining these secret elementary transformations it is easier for the cloud to discover the corresponding equivalent operations on $\hat{\mathbf{s}}$. These operations consist of random swaps of two values and alterations of single values. We assume that the corresponding precision for the number utilized to alter a single value is μ. Then, the key space to recover all the operations is $\binom{N}{k} k! 10^{\mu k}$.[1]

[1] Utilizing all the operations, the cloud can obtain the original unencrypted version of $\hat{\mathbf{s}}$, denoted by $\tilde{\mathbf{s}}$, so that the original signal is revealed by calculating $\mathbf{x} = \mathbf{D}\tilde{\mathbf{s}}$.

Table 1
Theoretical complexity analysis.

	KeyGen	*TaskGen*	*TaskProc*	*ResultVerify*
Protocol 1	$\mathbf{D}_K$	$\mathcal{O}(MN^2)$	$\mathcal{O}(N^3)$	$\mathcal{O}(kM)$
Protocol 2	$\mathbf{D}_{K_1}$, $\mathbf{D}_{K_2}$	$\mathcal{O}(2MN^2)$	$\mathcal{O}(2N^3)$	$\mathcal{O}(2kM)$

In the first method, $\mathbf{D}_K$ is directly generated by inputting the parameters, and therefore is often unpredictable in terms of the structure. If the parameters are frequently updated, then plaintext attacks, such as a known plaintext attack or chosen plaintext attack, can be effectively resisted. In the second method, $\mathbf{D}_K$ evolves from some known orthogonal transform matrix. Correspondingly, the structure can possibly be predicted and broken by plaintext attacks. However, the second method can be made more resistant against brute-force attacks than the first, as its key space is sufficient to withstand an exhaustive search. However, the first method only has a few parameters as keys, thus being easily successfully captured. Overall, the first method is suitable for resisting plaintext attacks, while the second is effective for resisting brute-force attacks. Note that these two methods can easily be combined to simultaneously deal with both plaintext and brute-force attack. In addition, $\mathbf{D}_K$ can also be elaborately and independently decorated by the client, in addition to the above two methods of constructing $\mathbf{D}_K$.

5.2. Computational analysis

The computational analysis results of the proposed protocols are presented in Table 1. For the key generation algorithm, the secret matrix $\mathbf{D}_K$ can easily be constructed, because it is generated by modifying some existing orthogonal transforms, and this modification only has linear complexity. Meanwhile, note that the inverse matrix calculation often has cubic complexity, but the calculation of $\mathbf{D}_K^{-1}$ does not. This is because $\mathbf{D}_K$ is an orthogonal matrix. Utilizing the transpose of the orthogonal matrix as its inverse, one can exploit the equivalent modification for the transpose of the orthogonal matrix, and obtain $\mathbf{D}_K^{-1}$ with linear complexity. Regarding the task generation and result verification algorithms, in Protocol 1 both have the complexity $\mathcal{O}(MN^2 + kM)$ on the client side. This means that the proposed Protocol 1 reduces the original complexity $\mathcal{O}(N^3)$ to $\mathcal{O}(MN^2)$ for the client, while not augmenting the computational burden of the CSR task in the cloud. The complexity of the result verification algorithm alone is only $\mathcal{O}(kM)$. One advantage compared with randomized verification is that it is not necessary to adopt a completely negative attitude to the result once NO is returned. The client can still utilize the remaining correct non-zero entries in $\hat{\mathbf{s}}$ to reconstruct the original signal, which may be acceptable owing to signal redundancy. The above investigations were also performed for Protocol 2, giving similar analysis results. Finally, we would like to point out that in both proposed protocols the client just transmits one or two vectors of length N, rather than matrices, which involves a very low communication cost.

6. Performance evaluation

We select an image and divide it into a certain number of blocks, each of which is regarded as a one-dimensional signal, to simulate the original signal for a visual view. The Lena image of size 512×512 is utilized. This is partitioned into 32×32 blocks, and so each signal has a length of 1024 and each image consists of 256 signals. With respect to the key generation, we apply secret row exchange operations to the DCT matrix, specifically a permutation operation to each row, to generate $\mathbf{D}_K$. The measurement matrix $\mathbf{A}$ is composed of independent identically distributed ensembles, yielding a Gaussian distribution. The client samples each block with $\mathbf{AD}_K^{-1}$ to generate the outsourcing task, and then sends this to the cloud for processing. The cloud leverages the convex optimization toolbox [17], which is a basis pursuit algorithm, to process the CSR task. The result is fed back to the client/user for verification and decryption. The communication latency between the client/user and the cloud is ignored. The workstation is a computer with Intel Core i7-4790 CPU and 8 GB RAM, and the platform is MATLAB.

The efficiency of the proposed CSR task outsourcing scheme is reported in Table 2, where the computation time is the average value from 256 signals, and is shown against the sampling rate (SR). This table corresponds to Protocol 2, while the results for Protocol 1 can easily be obtained by adding the user's time cost to the client side. Here, t_{client} and t_{user} represent the time costs for the client and user, respectively, and $t_{original}$ and t_{cloud} are the time costs for processing the original and encrypted CSR tasks, respectively. We note that t_{cloud} is found to have the consistent value of $t_{original}$ through the experiments. This is because of the fact that the encrypted CSR task shown in (4) has the same structure as the original CSR task shown in (3). The proposed CSR task outsourcing scheme provides both the client and user with a good amount of computation saving, and this *asymmetric speedup* is measured as $t_{original}/(t_{client} + t_{user})$. As can be observed in Table 2, the speedup becomes greater as the sampling rate increases. Furthermore, this increase tends to follow a linear trend, which can also be visually validated by the curve in Fig. 3. Note that SR=1 means that the system of linear equations to be solved is well-posed rather than underdetermined. It is easier to solve a well-posed equation, and thus the corresponding computation time will be smaller, as shown in Fig. 3. The proposed outsourcing protocol can achieve savings of over $317\times$, meaning that a considerable amount of computational overhead is successfully shifted from the local client or user to the cloud. It

Table 2
Efficiency of the proposed CSR task outsourcing protocol (time in seconds).

		Original Case	CSR outsourcing		Speedup
No.	SR	$t_{original}$	t_{client}	t_{user}	$t_{original}/(t_{client}+t_{user})$
1	0.10	0.6029	0.0510	0.0002	11.7608
2	0.15	1.1402	0.0455	0.0002	24.9571
3	0.20	1.8240	0.0477	0.0002	38.0638
4	0.25	2.6863	0.0468	0.0003	57.0619
5	0.30	3.9248	0.0478	0.0002	81.6153
6	0.35	5.3263	0.0493	0.0003	107.3091
7	0.40	7.0283	0.0513	0.0002	136.4638
8	0.45	9.2424	0.0491	0.0003	186.7909
9	0.50	11.6846	0.0533	0.0003	218.2319
10	0.55	14.1603	0.0499	0.0003	281.9289
11	0.60	16.8704	0.0569	0.0003	295.0222
12	0.65	20.3118	0.0507	0.0003	398.2623
13	0.70	22.9309	0.0529	0.0002	431.5208
14	0.75	26.6909	0.0521	0.0002	510.1789
15	0.80	30.4167	0.0529	0.0003	572.5472
16	0.85	33.9167	0.0536	0.0002	630.1536
17	0.90	36.5922	0.0539	0.0003	675.7654
18	0.95	41.4766	0.0543	0.0003	759.9221
19	1.00	34.6867	0.0562	0.0002	614.0842
Average		16.9217	0.0513	0.0003	317.4548

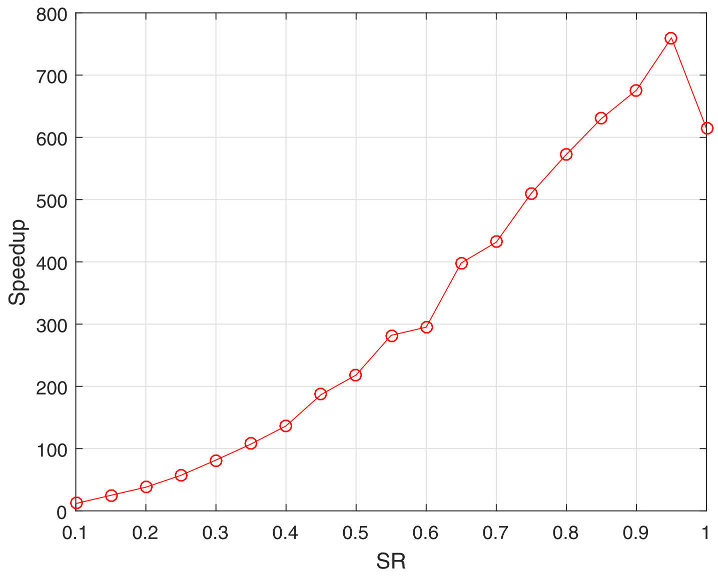


Fig. 3. Speedup versus sampling rate.

is worth noting that any existing CSR algorithm can be utilized on the cloud side, not limited to the convex optimization toolbox.

In the verification process, each non-zero entry in $\hat{\mathbf{s}}$ is checked, and then a deterministic result is given, i.e., this entry is either correct or incorrect. However, it is still possible to accept $\hat{\mathbf{s}}$ when some entry returns NO. If the number of non-zero entries returning YES is sufficient that the reconstructed signal has satisfactory quality, then it can be acceptable. This case of partial verification is suitable for a scenario in which the application requirement is not sufficiently strict. For example, in our experiment removing 10% of the non-zeros in $\hat{\mathbf{s}}$ with a sampling of 0.7 produces a reconstructed image of satisfactory visual quality, as shown in Fig. 4. Of course, if all the non-zero entries pass the verification, then the best quality will be obtained. In addition, Fig. 4 also demonstrates the correctness of the proposed outsourcing scheme.

In the proposed outsourcing protocols, the design goals are to achieve correctness, privacy, client/user verification, and efficiency, as previously mentioned. These goals have been demonstrated through theoretical discussions. In the experimental evaluation, we have further validated the correctness, client/user verification, and efficiency. The proposed protocols can correctly outsource the CSR task and verify the output result returned by the cloud. Computational savings can clearly be obtained for the client. A greater number of signals (images) have been utilized to draw unanimous outcomes. Although the

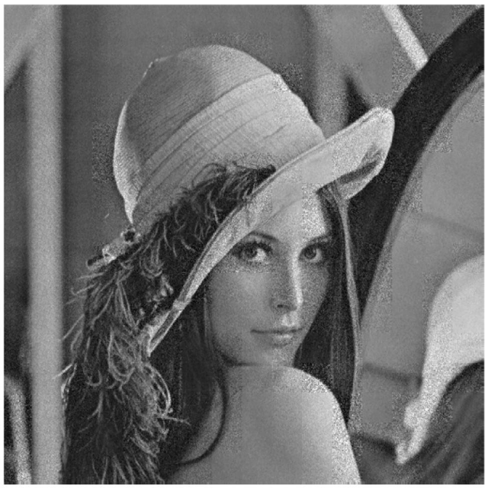


Fig. 4. An example of partial verification.

client can successfully shift a considerable amount of computational overhead to the cloud, the cloud can still improve the effectiveness without affecting the returned result, based on some scheduling algorithms in distributed systems [2,31,33,35].

7. Concluding remarks

In this paper, we have proposed two CSR outsourcing protocols, which aim towards client and user verification, respectively. The client can generate a secret measurement matrix based on the secrecy of the orthogonal basis, which is then utilized for signal sampling. The generated measurement vector is sent to the cloud, where the CSR task using the public sensing matrix is solved. After receiving the result returned by the cloud, the client performs verification and finally decryption. If the user requires this signal, then they perform the verification with the same procedure and two different keys. The encryption approach is based on the function migration of the orthogonal sparsifying basis and does not affect the sparsity or RIP. The asymmetric verification represents a deterministic and partial verification approach. Both the encryption and asymmetric verification methods are effective and simple to use. Theoretical and experimental analyses were conducted to validate the security and efficiency of the proposed CSR outsourcing protocols.

Acknowledgment

This work was supported by Guangxi Key Laboratory of Trusted Software (kx201904), Macau Science and Technology Development Fund (FDCT/022/2017/A1 and FDCT/077/2018/A2), National Natural Science Foundation of China (61702221), and Chongqing Basic Research and Front Exploration Project (cstc2018jcyjA3093).

References

[1] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, et al., A view of cloud computing, Commun. ACM 53 (4) (2010) 50–58.
[2] N. Bessis, S. Sotiriadis, F. Pop, V. Cristea, Using a novel message-exchanging optimization (meo) model to reduce energy consumption in distributed systems, Simul. Modell. Pract. Theory 39 (2013) 104–120.
[3] Ç. Candan, M.A. Kutay, H.M. Ozaktas, The discrete fractional Fourier transform, IEEE Trans. Signal Process. 48 (May 5) (2000) 1329–1337.
[4] E.J. Candès, J. Romberg, T. Tao, Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information, IEEE Trans. Inf. Theory 52 (2) (2006) 489–509.
[5] E.J. Candès, T. Tao, Decoding by linear programming, IEEE Trans. Inf. Theory 51 (December 12) (2005) 4203–4215.
[6] E.J. Candes, T. Tao, Near-optimal signal recovery from random projections: universal encoding strategies? IEEE Trans. Inf. Theory 52 (December 12) (2006) 5406–5425.
[7] F. Chen, T. Xiang, X. Fu, W. Yu, User differentiated verifiable file search on the cloud, IEEE Trans. Serv. Comput. (2016). in press
[8] F. Chen, T. Xiang, X. Lei, J. Chen, Highly efficient linear regression outsourcing to a cloud, IEEE Trans. Cloud Comput. 2 (4) (2014) 499–508.
[9] F. Chen, T. Xiang, Y. Yang, Privacy-preserving and verifiable protocols for scientific computation outsourcing to the cloud, J. Parallel Distrib. Comput. 74 (March 3) (2014) 2141–2151.
[10] X. Chen, X. Huang, J. Li, J. Ma, W. Lou, D. Wong, New algorithms for secure outsourcing of large-scale systems of linear equations, IEEE Trans. Inf. Foren. Secur. 10 (1) (2015) 69–78.
[11] Q. Ding, G. Weng, G. Zhao, C. Hu, Efficient and secure outsourcing of large-scale linear system of equations, IEEE Trans. Cloud Comput. (2018). in press
[12] D.L. Donoho, Compressed sensing, IEEE Trans. Inf. Theory 52 (4) (2006) 1289–1306.
[13] K. Gai, M. Qiu, Blend arithmetic operations on tensor-based fully homomorphic encryption over real numbers, IEEE Trans. Ind. Inf. 14 (8) (2018) 3590–3598.
[14] K. Gai, M. Qiu, H. Zhao, Cost-aware multimedia data allocation for heterogeneous memory using genetic algorithm in cloud computing, IEEE Trans. Cloud Comput. (2016). in press, 2016

[15] K. Gai, M. Qiu, H. Zhao, Energy-aware task assignment for mobile cyber-enabled applications in heterogeneous cloud computing, J. Parallel Distrib. Comput. 111 (2018) 126–135.
[16] K. Gai, M. Qiu, H. Zhao, L. Tao, Z. Zong, Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing, J. Netw. Comput. Appl. 59 (2016) 46–54.
[17] M. Grant, S. Boyd, Y. Ye, CVX: Matlab software for disciplined convex programming, 2008 http://cvxr.com/cvx/.
[18] W.-L. Hsue, W.-C. Chang, Real discrete fractional Fourier, Hartley, generalized Fourier and generalized Hartley transforms with many parameters, IEEE Trans. Circ. Syst. I-Regular Papers 62 (October 10) (2015) 2594–2605.
[19] G. Hu, D. Xiao, T. Xiang, S. Bai, Y. Zhang, A compressive sensing based privacy preserving outsourcing of image storage and identity authentication service in cloud, Inf. Sci. 387 (2017) 132–145.
[20] L.-W. Kang, K. Muchtar, J.-D. Wei, C.-Y. Lin, D.-Y. Chen, C.-H. Yeh, privacy-preserving multimedia cloud computing via compressive sensing and sparse representation, in: Proc. Int. Conf. Inf. Security Intell. Control (ISIC), 2012, pp. 246–249.
[21] X. Kang, R. Tao, F. Zhang, Multiple-parameter discrete fractional transform and its applications, IEEE Trans. Signal Process. 64 (July 13) (2016) 3402–3417.
[22] X. Lei, X. Liao, T. Huang, F. Heriniaina, Achieving security, robust cheating resistance, and high-efficiency for outsourcing large matrix multiplication computation to a malicious cloud, Inf. Sci. 280 (2014) 205–217.
[23] X. Lei, X. Liao, T. Huang, H. Li, Cloud computing service: the case of large matrix determinant computation, IEEE Trans. Serv. Comput. 8 (5) (2015) 688–700.
[24] X. Lei, X. Liao, T. Huang, H. Li, C. Hu, Outsourcing large matrix inversion computation to a public cloud, IEEE Trans. Cloud Comput. 1 (1) (2013). 1–1
[25] X. Li, Y. Zhu, J. Wang, Z. Liu, Y. Liu, M. Zhang, On the soundness and security of privacy-preserving svm for outsourcing data classification, IEEE Trans. Depend. Secure Comput., 15 (5) (2018) 906–912.
[26] X. Liu, R. Choo, R.H. Deng, R. Lu, J. Wen, Efficient and privacy-preserving outsourced calculation of rational numbers, IEEE Trans. Depend. Secure Comput. 15 (1) (2018) 27–39.
[27] X. Liu, R.H. Deng, W. Ding, R. Lu, B. Qin, Privacy-preserving outsourced calculation on floating point numbers, IEEE Trans. Inf. Foren. Secur. 11 (11) (2016) 2513–2527.
[28] C. Luo, J. Ji, X. Chen, M. Li, L.T. Yang, P. Li, Parallel secure outsourcing of large-scale nonlinearly constrained nonlinear programming problems, IEEE Trans. Big Data (2018). in press
[29] B. Qin, R.H. Deng, S. Liu, M. Siqi, Attribute-based encryption with efficient verifiable outsourced decryption, IEEE Trans. Inf. Foren. Secur. 10 (7) (2015) 1384–1393.
[30] S. Salinas, C. Luo, X. Chen, W. Liao, P. Li, Efficient secure outsourcing of large-scale sparse linear systems of equations, IEEE Trans. Big Data 4 (1) (2018) 26–39.
[31] A. Sfrent, F. Pop, Asymptotic scheduling for many task computing in big data platforms, Inf. Sci. 319 (2015) 71–91.
[32] Z. Shan, K. Ren, M. Blanton, C. Wang, Practical secure computation outsourcing: a survey, ACM Comput. Surv. (CSUR) 51 (2) (2018) 31.
[33] A. Sirbu, C. Pop, C. Şerbănescu, F. Pop, Predicting provisioning and booting times in a metal-as-a-service system, Future Gener. Comput. Syst. 72 (2017) 180–192.
[34] H. Takabi, J.B. Joshi, C.-J. Ahn, Security and privacy challenges in cloud computing environments, IEEE Secur. Privacy 8 (6) (2010) 24–31. Nov.-Dec.
[35] M.-A. Vasile, F. Pop, R.-I. Tutueanu, V. Cristea, J. Kołodziej, Resource-aware hybrid scheduling algorithm in heterogeneous distributed computing, Future Gener. Comput. Syst. 51 (2015) 61–71.
[36] C. Wang, K. Ren, J. Wang, Secure and practical outsourcing of linear programming in cloud computing, in: Proc. IEEE Computer Commun. (INFOCOM), IEEE, 2011, pp. 820–828.
[37] C. Wang, K. Ren, J. Wang, Secure optimization computation outsourcing in cloud computing: a case study of linear programming, IEEE Trans. Comput. 65 (1) (2016) 216–229.
[38] C. Wang, K. Ren, J. Wang, K.M.R. Urs, Harnessing the cloud for securely solving large-scale systems of linear equations, in: 31st International Conference on Distributed Computing Systems (ICDCS), IEEE, 2011, pp. 549–558.
[39] C. Wang, B. Zhang, K. Ren, J.M. Roveda, C.W. Chen, Z. Xu, A privacy-aware cloud-assisted healthcare monitoring system via compressive sensing, in: Proc. INFOCOM, 2014, pp. 2130–2138.
[40] C. Wang, B. Zhang, K. Ren, J. Wang, Harnessing the cloud for securely outsourcing large-scale systems of linear equations, IEEE Trans. Parallel Distrib. Syst. 24 (6) (2013) 1172–1181.
[41] C. Wang, B. Zhang, K. Ren, J. Wang, Privacy-assured outsourcing of image reconstruction service in cloud, IEEE Trans. Emerg. Top. Comput. 1 (1) (2013) 166–177.
[42] Q. Wang, W. Zeng, J. Tian, A compressive sensing based secure watermark detection and privacy preserving storage framework, IEEE Trans. Image Process. 23 (3) (2014) 1317–1328.
[43] Y. Zhang, Y. Xiang, L.Y. Zhang, Secure Compressive Sensing in Multimedia Data, Cloud Computing and IoT, Springer, Book, 2018.
[44] Y. Zhang, Y. Xiang, L.Y. Zhang, Y. Rong, S. Guo, Secure wireless communications based on compressive sensing: a survey, IEEE Commun. Surv. Tutorials (2018). in press
[45] Y. Zhang, D. Xiao, Double optical image encryption using discrete chirikov standard map and chaos-based fractional random transform, Opt. Lasers Eng. 51 (April 4) (2013) 472–480.
[46] Y. Zhang, D. Xiao, W. Wen, Y. Tian, Edge-based lightweight image encryption using chaos-based reversible hidden transform and multiple-order discrete fractional cosine transform, Opt. Laser Tech. 54 (2013) 1–6.
[47] Y. Zhang, L.Y. Zhang, J. Zhou, L. Liu, F. Chen, X. He, A review of compressive sensing in information security field, IEEE Access 4 (June) (2016) 2507–2519.
[48] Y. Zhang, J. Zhou, L.Y. Zhang, F. Chen, X. Lei, Support-set-assured parallel outsourcing of sparse reconstruction service for compressive sensing in multi-clouds, in: Proc. Int. Symp. Security Privacy Social Netw. Big Data (SocialSec), 2015, pp. 1–6.
[49] N. Zhou, Y. Wang, L. Gong, Novel optical image encryption scheme based on fractional mellin transform, Opt. Commun. 284 (June 13) (2011) 3234–3242.
[50] B. Zhu, S. Liu, Q. Ran, Optical image encryption based on multifractional fourier transforms, Opt. Lett. 25 (August 16) (2000) 1159–1161.