Secure Outsourced Computation of the Characteristic Polynomial and Eigenvalues of Matrix

Xing Hu and Chunming Tang

School of Mathematics and Information Science, Guangzhou University, Guangzhou 510006,China
huxing@gzhu.edu.cn; ctang@gzhu.edu.cn

Abstract. Linear algebra plays an important role in computer science, especially in cryptography.Numerous cryptog-raphic protocols, scientific computations, and numerical computations are based on linear algebra. Many linear algebra tasks can be reduced to some core problems, such as matrix multiplication, determinant of matrix and the characteristic polynomial of matrix. However, it is difficult to execute these tasks independently for client whose computation abilities are weaker than polynomial-time computational ability. Cloud Computing is a novel economical paradigm which provides powerful computational resources that enables resources-constrained client to outsource their mass computing tasks to the cloud. In this paper, we propose a new verifiable and secure outsourcing protocol for the problem of computing the characteristic polynomial and eigenvalues of matrix. These protocols are not only efficient and secure, but also unnecessary for any cryptographic assumption.

Keywords: Cloud Computing;Outsourced computation;Matrix;
Characteristic polynomial;Eigenvalues

1 Introduction

1.1 Cloud computing

Cloud computing provides an appropriate on-demand network access to a shared pool of configurable computing resources which could be rapidly deployed with much more great efficiency and with minimal overhead to management. It is becoming an important topic in both of industry and academic communities. A user can access cloud services as a utility service and use them almost instantly. The rise of cloud computing also raises several potential risks. The most serious concerns are the possibility of lack of confidentiality, integrity and authentication among the cloud users and service providers. The key intent of this research work is to investigate the existing security schemes and to ensure data confidentiality, integrity and authentication. One of the main challenges is how to achieve a pair of apparently conflicting requirements simultaneously: efficiency in communication, storage and computation on both client and server side, and security against outside and internal attackers.

Two main services provided by cloud servers are storage and computation in cloud computing. Outsourced computation is used when client needs to execute a task, but does not have the appropriate computation power to perform it. The task is then outsourced to the external server, which has the sufficient computing power. Generally, an outsourced computation protocol is correct if the final outputs for client are valid. An outsourced computation protocol is secure if it is done without revealing to the external server either the actual data or the actual answer to the computations. An outsourced computation protocol is verifiable if the final outputs received from cloud server can be verified by client. An outsourced computation protocol is efficient if the computational work done locally by client is less than that of solving the original problem on his own.

1.2 Outsourced computation model

Our outsourcing protocols involve two different entities: cloud client(s) and cloud server(s). Client has many matrix calculations tasks, which needs expensive computation and exceeds his computational abilities. Hence all of calculations are outsourced to cloud server, which has significant computation resources to perform all matrix calculations.

To achieve input/output privacy, the key idea is that some local preprocessing should be done on the original problem and/or data before sending it to the cloud server, and also the client needs to do some local post-processing on the answer received from the cloud server to recover the true answer. More specifically, in our model, instead of directly sending original problem φ , we use some disguising techniques to transform original problem φ into a random problem ϕ , then outsource problem ϕ to a cloud server. The server then conducts computation to get the answer of ϕ and provides a proof that the evaluation has been carried out correctly, but it cannot derive anything of the sensitive information contained in φ from the disguised problem ϕ . After receiving the solution of ϕ from the server, the client should be able to verify the validity of the answer via the appended proof. If it's valid, he recovers the desired answer for the original problem.

The dishonest behaviors of cloud server can be divided into semi-honest and malicious in our model. A semi-honest cloud server corrupted by an adversary is one who follows the protocol with the exception that it keeps recodes of all its intermediate results, he wishes to learn more sensitive information than they should obtain from the running of the protocol. A malicious cloud server is one who can deviate from the protocol description, they also can tamper with the correct data to make the honest client calculate the wrong output, even suspend (or abort) the execution in any desired point in time. In this paper, we assume that cloud server is malicious.

Outsourcing protocols for some linear algebra problems with the following properties are to be designed:

- **Correctness** Any cloud server that faithfully follows the mechanism must produce an valid output.
- **Soundness** No cloud server can generate an incorrect output that can be verified successfully by client with non-negligible probability.
- **Input/output privacy** No sensitive information from client's private data can be derived by cloud server during performing the computation.
- **Verifiability** The protocol should allow client to verify the correctness of results received from an honest server with non-negligible probability, and also to detect the wrong results received from a dishonest server with non-negligible probability.
- **Efficiency** The local computations done by client should be substantially less than that of solving the original problem on his own.

1.3 Contributions

In this paper, we use disguising technology as a tool to construct efficiently and verifiably secure outsourced protocol for the computation of the characteristic polynomial of matrix. Our protocol achieve several desired features, such as privacy, verifiability and efficiecy. And no any cryptographic assumption is needed in this protocol.

Moreover, this is the first time a verifiable and secure outsourcing protocol for computing the characteristic polynomial and eigenvalues of matrix is proposed.

2 Related Work

In 2009, Gentry firstly constructed fully homomorphic encryption scheme based on ideal lattice theory, which is a significant work. Efficiently and verifiably secure outsourcing computation can be constructed for any function if efficiently fully homomorphic encryption scheme exists. However, it is impossible to construct efficiently fully homomorphic encryption scheme in recent years. Some improved fully homomorphic encryption scheme were proposed [03-09], which are far away from practically fully homomorphic encryption scheme.

Atallah[10] proposed verifiably outsourcing computation protocol for expensive linear algebraic computation by using a homomorphic semantically secure encryption system. However, their design is built on the assumption of two non-colluding servers and thus vulnerable to colluding attacks. In their protocols, the computation cost for client is $O(n^2)$, where n is the size of the input matrix. Then, based on Shamir's secret sharing scheme, Atallahet al. [11]constructed a verifiably outsourcing computation protocol for matrix multiplication, which only needs one un-trusted cloud server. In their protocols, the computation cost for client is $O(t^2n^2)$, where n and t are the size of matrix and the threshold in Shamir's secret sharing scheme respectively. Based on Yao's Garbled Circuits and fully homomorphic encryption scheme, verifiably non-interactive outsourcing computation for any function was proposed in [12], however, client needs to do an expensive preprocessing stage, and his computation complexity is related to the size of the Boolean circuit representing function .

To construct outsourcing computation protocol for solving linear equation $Ax = b$, C.Wang et al.[13]realized it by using iteration from Jacobi method and additive homomorphic encryption with semantic security. In their protocols, the computation complexity for client is $O(n)$, however, their solutions are approximate and the input matrix A is constrained. In 2011, P. Mohassel[14] designed non-interactive and secure protocols for delegating matrix multiplication, based on a number of encryption schemes with limited homomorphic properties where the client only needs to perform $O(n^2)$ work. But their verification algorithm works correctly with an all but negligible probability in k , the probability is over the random coins of the verification algorithm (see section4.2 in[14]).

In [15], a new protocol for publicly verifiable secure outsourcing of matrix multiplication was presented. Their scheme was in the amortized model [12], in which the client invests a one-time expensive computation phase when storing a large matrix with the server, which was used to make the verification of matrix multiplication fast. However, their design was built upon some cryptographic assumptions, such as the co-CDH assumption and the Decision Linear assumption. And their result also relied on the use of pseudo-random functions with closed-form efficiency and bilinear maps.

In 2007, Eike Kiltz et al.[16]presented secure two-party protocols for various core problems in linear algebra, such as computing the determinant and minimal polynomial of a matrix. Their protocols were based on an algorithm by Wiedemann for "black-box linear algebra" [17] which was efficient when applied to sparse matrices. Their techniques exploited certain nice mathematical properties of linearly recurrent sequences and their relation to the minimal and characteristic polynomial of the input matrix. Nevertheless, their constructions were secure under the assumption of the existence of a homomorphic public-key encryption scheme and a secure instantiation of Yao's garbled circuit protocol. And their protocol for computing determinant and the minimal polynomial of an encrypted matrix needs $O(n^2 \log n \log |F|)$ communication complexity and $O(\log n)$ rounds respectively.

Another important way to construct secure outsourcing protocols is to use mathematical disguise (or blinding) methods. In 2001, Atallahet al.[18]investigated the outsourcing of numerical and scientific computations. Their schemes applied disguise techniques to science computational problems, which guaranteed the data security and privacy. But they did not handle the important case of verification of validity of final result. In 2005, Hohenberger S, Lysyanskaya A[19] presented practical outsource-secure scheme for modular exponentiation where the honest party may use two exponentiation programs. The exponentiation programs were un-trusted and cannot communicate with each other, after deciding on an initial strategy. In 2012, several new methods of secure outsourcing of numerical and scientific computations were proposed by Yerzhan N. Seitkulov[21]. They had presented different methods of finding approximate solutions to some equations solved by an external computer. Most of their methods were verifiable.

In2013, verifiable and secure outsourcing protocols for matrix multiplication and matrix inversion have been proposed in [22]. These protocols used disguising matrix to tackle privacy-preserving problem.

3 Review

3.1 Mathematical notation

Without loss of generality, we will use capital letter, such as A, to denote a matrix, and the notation a_{ij} represents the element at the i^{th} row and j^{th} column of the matrix A. The notation F_q denotes a finite field which has q elements, and $F_q^{n\times n}$ denotes the set of $n \times n$ matrices over F_q. In what follows, we also use $\delta_{x,y}$ to denote the Kronecker delta function that equals 1 if $x = y$ and 0 if $x \neq y$.

Moreover, we use $E(i, j(k))$ to denote an elementary matrix which differs from the identity matrix I by one single elementary row operation. The elementary matrix $E(i, j(k))$ is the identity matrix but with a randomly chose $k \in F_q$ in the (i, j) position. We note immediately that left multiplication (premultiplication) by $E(i, j(k))$ represents elementary row operations. So $E(i, j(k))\cdot A$ is the matrix produced from A by adding k times row j to row i .

3.2 Secure outsourcing protocol MP

In[22], a verifiable and secure outsourcing protocol MP has been proposed for matrix multiplication, which will be used as a building block in later section.

Let $M_1, M_2 \in F_q^{n\times n}$ be two matrices, client wants to obtain M_1M_2.Cloud server is not allowed to learn any sensitive information such as the input matrices or the actual output,etc. We briefly review this protocol which proceeds as follow:

Step 1 Client generates a secret value for verification. He chooses two numbers $1 \leq i, j \leq n$ randomly and picks two vectors $(a_{_{i1}}, ..., a_{_{in}})and(b_{1j}, ..., b_{nj})^T$ which corresponding to the i^{th} row of M_1 and the j^{th} column of M_2 respectively, then computes the secret value $c = \sum_{t=1}^{n} a_{_{it}} b_{_{tj}}$.

Step 2 Client proceeds with the following sub-steps:

(a) Generates six private random permutations $\pi_1, ..., \pi_6$,where $\pi_i \in \{1, ..., n\}$, $i = 1, 2, ..., 6$.

(b) Chooses $6n$ non-zero numbers $\{a_1, ..., a_n\}$,$\{b_1, ..., b_n\}$, $\{c_1, ..., c_n\}$, $\{d_1, ..., d_n\}$,$\{e_1, ..., e_n\}$,$\{f_1, ..., f_n\}$from F_q randomly.

(c) Generates six matrices $P_1, ..., P_6$ where $P_1(i, j) = a_i\delta_{\pi_1(i),j}$,$P_2(i, j) = b_i\delta_{\pi_2(i),j}$, $P_3(i, j) = c_i\delta_{\pi_3(i),j}$, $P_4(i, j) = d_i\delta_{\pi_4(i),j}$, $P_5(i, j) = e_i\delta_{\pi_5(i),j}$, $P_6(i, j) = f_i\delta_{\pi_6(i),j}$.These matrices are readily invertible, e.g. $P_1^{-1}(i, j) = a_j^{-1}\delta_{\pi_1^{-1}(i),j}$.

Step 3 Client computes 4 matrices

$$X_1 = P_1M_1P_2^{-1},\ Y_1 = P_2M_2P_3^{-1},\ X_2 = P_4M_1P_5^{-1},\ Y_2 = P_5M_2P_6^{-1}.$$

and then sends to cloud server two pairs (X_1, Y_1) and (X_2, Y_2).

Step 4 The sever computes $Z_1 = X_1Y_1$,$Z_2 = X_2Y_2$ and return the computed output (Z_1, Z_2) to client.

Step 5 Client recover the result by computing $T_1 = P_1^{-1}Z_1P_3 \text{and} T_2 = P_4^{-1}Z_2P_6$.

Step 6 After recovering, client needs to verify the result T_1 and T_2. That is, if $T_1 = T_2$ and $T_k(i, j) = c$ $(k = 1, 2)$ hold simultaneously, which means the server is honest, client obtain the correct output $P_1^{-1}Z_1P_3$ which actually equals to M_1M_2 , otherwise, client refuses the received outputs and terminates the protocol.

Note: The server gains no information about the input/output matrix during the execution of the protocol. Moreover, because the matrix $P_t(t = 1, ..., 6)$ is special, which has only n elements rather than the matrix computed by the server which has n^2 elements, the computation time in client side is less than the computing time in the cloud side. According to [22], the protocol MP has been proved to be an efficient and verifiable outsource-secure implementation of matrix multiplication in the single cloud server model.

4 Finding the Characteristic Polynomial and Eigenvalues of Matrix

In this section, we present a secure outsourcing protocol for the characteristic polynomial of a matrix and its eigenvalues. Because the probability of the real random matrix being nonsingular is 1(see Corollary 1.1 in[23]), so we assume that all eigenvalues of a matrix are nonzero.

Problem ME: let $A \in F_q^{n\times n}$ be a private matrix, client needs to calculate the characteristic polynomial of A and its eigenvalues with the help of an untrusted cloud server without revealing any private information,such as the input matrix or the actual output,etc.

Protocol ME: The outsourcing protocol ME proceeded as follows:

Step 1 Client chooses a secret random number $r \in F_q$ and computes $A_1 = rA$.

Step 2 For each $i \in \{i, ..., n\}$,client chooses a random number $1 \leq j \leq n, j \neq i$, and then computes a matrix $B(\lambda) = L(A_1 - \lambda I)$ where $L = \prod_{i=1}^{n} E(i, j(1))$ .Obviously, L is readily invertible.

Step 3 Client chooses a secret random number $1 \leq i \leq n$, then divides each element $b_{ij}(\lambda)$ in the i^{th} row (or the i^{th} column) of $B(\lambda)$ to $m(1 < m < n - 2)$ random pieces. So, client has $|B(\lambda)| = |B_1(\lambda)| + \cdots + |B_m(\lambda)|$.

Step 4 Client picks two secret random numbers $1 \leq i, j \leq n, i \neq j$, then involves the protocol MP to obtain $B_i(\lambda) \cdot B_j(\lambda)$, which denoted by $B_{m+1}(\lambda)$.

Step 5 Client hides matrices $B(\lambda)$ and $B_i(\lambda)(i = 1, ...m + 1)$ by the following substeps:

(a) Generates matrices $P_1, ..., P_{2m+4}$, where P_k is a $n \times n$ matrix for all $k = 1, ...2m + 4$, $P_k(i, j) = a_i^k \delta_{\pi_k(i),j}$, $\pi_k \in \{1, ..., n\}$ are random permutations, and $\{a_1^k, ..., a_n^k\}$ are random numbers over F_q.
Let $P = \{P_1, ..., P_{2m+4}\}$.

(b) For each $0 \leq s \leq m+1$, client picks two matrices $P_l, P_r \in P$ from P,let $P_l = P_l^s, P_r = P_r^s$ and $B_0(\lambda) = B(\lambda)$,then computes matrices $C_s(\lambda) = P_l^s B_s(\lambda) P_r^s$. Note that for each $B_s(\lambda)$, the disguising matrix set $\{P_l^s, P_r^s\}$ is different from each other.

(c) Client sends $C_0(\lambda), ..., C_{m+1}(\lambda)$ to cloud server. Note that the sending order of these matrices $|C_s(\lambda)|$ is random.

Step 6 The server computes $|C_0(\lambda)|, ..., |C_{m+1}(\lambda)|$ and solves $|C_s(\lambda)| = 0$ for each $0 \leq s \leq m+1$. Let $\lambda^s = (\lambda_1^s, ..., \lambda_n^s)$ be the root vector of $|C_s(\lambda)| = 0$. Then the server return $m+2$ two-tuples $(|C_s(\lambda)|, \lambda^s)$ to client. We require that the server's outputs should according to the sending order.

Step 7 After receiving the result from cloud server, client checks the following three equalities:

$$\frac{|C_0(\lambda)|}{|P_l^0 P_r^0|} = \sum_{s=1}^{m} \frac{|C_s(\lambda)|}{|P_l^s P_r^s|} \tag{1}$$

$$\frac{|C_{m+1}(\lambda)|}{|P_l^{m+1} P_r^{m+1}|} = \frac{|C_i(\lambda)|}{|P_l^i P_r^i|} \times \frac{|C_j(\lambda)|}{|P_l^j P_r^j|} \tag{2}$$

$$\prod_{j=1}^{n} (\lambda - \lambda_j^0) = |C_0(\lambda)| \tag{3}$$

If the above three equalities hold simultaneously, client gets $|A_1 - \lambda E| = |B(\lambda)|$ from $|C_0(\lambda)|$. Otherwise, client refuses all answers and terminates the protocol.

Step 8 Let $g_B(\lambda) = |B(\lambda)|$, client computes the characteristic polynomial $f_A(\lambda) = \frac{1}{r^n} g_B(r\lambda)$, and the corresponding eigenvalues $\lambda_j^* = \frac{1}{r} \lambda_j^0$ $(j = 1, ..., n)$.

Theorem 1. *In the single cloud server model, protocol ME is a verifiable outsource-secure computation of the characteristic polynomial of a matrix.*

Proof.

- **Correctness:** The correctness property is straight-forward. Obviously, the cloud server's output can be accepted successfully by client if it is honest.
- **Soundness:** Cloud server's dishonest behavior is to return some randomly chosen matrices instead of $C_0(\lambda), ..., C_{m+1}(\lambda)$, and/or return another wrong root vector instead of the right one. However, it is infeasible for the server to satisfy the three equalities in step7 simultaneously. The three equalities will be simultaneously satisfied with probability at most $(\frac{1}{m+2})(\frac{1}{m+1})(\frac{1}{C_m^2})(\frac{1}{q^m})$. That is, any incorrect output generated by cloud server can be detected successfully by client with non-negligible probability.
- **Input/output privacy:** The outsourcing protocol ME does not disclose the input matrix and the real result.

  First of all, client conducts a pre-disguising in step1 and step2, so the real eigenvalues and all the elements of the input matrix A (including the zero elements) are effectually hidden. Second, the second-round disguising has been done in step5. If an attacker wants to deriver any $B_s(\lambda)$ from $C_s(\lambda)$, he has to guess two permutations (from the $(n!)^2$ possible such choices) and $2n$ random numbers before he can determine a $B_s(\lambda)$. That is, for each permutation, the probability is $\frac{1}{n!}$, and for each random number, the probability is $\frac{1}{q}$, so the total probability for the attacker to obtain a correct $B_s(\lambda)$ is $(\frac{1}{n!})^2(\frac{1}{q^n})^2$, which is negligible.

- **Verifiability:** Assume the server is corrupted by a PPT adversary, who wants to cheat client without being caught. He wants to use some wrong data to make the three equalities hold simultaneously. However, he has to find out the correct $|C_0(\lambda)|, |C_i(\lambda)|, |C_j(\lambda)|$ and $|C_{m+1}(\lambda)|$ from $m+2$ determinants $|C_0(\lambda)|, ..., |C_{m+1}(\lambda)|$ and m numbers $\alpha_s \in F_q$ $(s = 1, ..., m)$ to make equality(1)and(2) hold simultaneously. Hence, the chance of the server successfully pass equality(1)and(2)is at most $(\frac{1}{m+2})(\frac{1}{m+1})(\frac{1}{C_m^2})(\frac{1}{q^m})$. Moreover, if the attacker returns to client a randomly chosen vector instead of the right root vector, it is infeasible for him to make equality(5)hold for the reason that the probability of $|C_0(\lambda)|$ found from $|C_0(\lambda)|, ..., |C_{m+1}(\lambda)|$ is $\frac{1}{m+2}$. Therefore, cloud server's dishonest behavior can be caught by client with non-negligible probability.

Theorem 2. *In the single cloud server model, the protocol ME is efficient.*

Proof. The protocol ME achieves not only the privacy but also the efficiency. The disguise conducted in step1 requires $O(n^2)$ local computation, and in step2 incurs close-to-zero additional cost on client side. And disguising matrices $B(\lambda)$ and $B_i(\lambda)$ $(i = 1, ..., m+1)$ in step5 requires $O((m+2)n^2)$ local computation, which could be small if we choose m $(1 < m < n-2)$ properly. In the result verification phase(i.e. step7), client also needs $O(n^2)$ local computation to verify equality $\prod_{j=1}^{n}(\lambda - \lambda_j^0) = |C_0(\lambda)|$. In addition, protocol ME involves the secure outsourcing protocol MP to obtain $B_{m+1}(\lambda)$ which requires $O(n^2)$ local computation[22]. Therefore, the protocol ME is efficient.

5 Conclusion

In this paper, we present an outsourcing protocol for computing the characteristic polynomial and eigenvalues of matrix.This protocol satisfy privacy,verifiability, efficiency and validity.

References

1. Sun Microsystems. Building customer trust in cloud computing with transparent security.$http : //www.sun.com/offers/details/sun_transparency.xml$,2009.
2. Gentry C. Fully homomorphic encryption using ideal lattices. Proceedings of 41st Annual ACM Symposium on Theory of Computing,2009,169-178.
3. Gentry C. Toward basing fully homomorphic encryption on worst-case hardness. Proceedings of 30th Annual Cryptology Conference,2010,116-137.

4. Van Dijk M, Gentry C, Halevi S, et al. Fully homomorphic encryption over the integers. Advances in CryptologyCEUROCRYPT 2010,Springer Berlin Heidelberg, 2010, 24-43.
5. Smart N P, Vercauteren F. Fully homomorphic encryption with relatively small key and ciphertext sizes. Public Key CryptographyCPKC 2010. Springer Berlin Heidelberg, 2010, 420-443.
6. Stehl D, Steinfeld R. Faster fully homomorphic encryption. Advances in Cryptology-ASIACRYPT 2010. Springer Berlin Heidelberg, 2010,377-394.
7. Lyubashevsky V, Peikert C, Regev O. On ideal lattices and learning with errors over rings. Advances in CryptologyCEUROCRYPT 2010. Springer Berlin Heidelberg, 2010,1-23.
8. Brakerski Z, Vaikuntanathan V. Efficient fully homomorphic encryption from (standard) LWE. Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on. IEEE, 2011, 97-106.
9. Brakerski Z, Vaikuntanathan V. Fully homomorphic encryption from ring-LWE and security for key dependent messages. Advances in CryptologyCCRYPTO 2011. Springer Berlin Heidelberg, 2011, 505-524.
10. Benjamin D, Atallah M J. Private and cheating-free outsourcing of algebraic computations. Privacy, Security and Trust, 2008. PST'08. Sixth Annual Conference on. IEEE, 2008,240-245.
11. Atallah M J, Frikken K B. Securely outsourcing linear algebra computations. Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security. ACM, 2010, 48-59.
12. Gennaro R, Gentry C, Parno B. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. Advances in CryptologyCCRYPTO 2010. Springer Berlin Heidelberg, 2010,465-482.
13. Wang C, Ren K, Wang J, et al. Harnessing the cloud for securely solving large-scale systems of linear equations. Distributed Computing Systems (ICDCS), 2011 31st International Conference on. IEEE, 2011,549-558.
14. Mohassel P. Efficient and Secure Delegation of Linear Algebra. IACR Cryptology ePrint Archive, 2011, 605.
15. Fiore D, Gennaro R. Publicly verifiable delegation of large polynomials and matrix computations, with applications. Proceedings of the 2012 ACM conference on Computer and communications security. ACM, 2012,501-512.
16. Kiltz E, Mohassel P, Weinreb E, et al. Secure linear algebra using linearly recurrent sequences. Springer Berlin Heidelberg, 2007,291-310.
17. Wiedemann D. Solving sparse linear equations over finite fields. Information Theory, IEEE Transactions on, 1986, 32(1):54-62.
18. Atallah M J, Pantazopoulos K N, Rice J R, et al. Secure outsourcing of scientific computations. Advances in Computers, 2002, 54:215-272.
19. Hohenberger S, Lysyanskaya A. How to securely outsource cryptographic computations. Theory of Cryptography. Springer Berlin Heidelberg, 2005,264-282.
20. Clarke D, Devadas S, Van Dijk M, et al. Speeding up Exponentiation using an Untrusted Computational Resource. MEMO 469, MIT CSAIL COMPUTATION STRUCTURES GROUP. 2003.
21. Seitkulov Y N. New methods of secure outsourcing of scientific computations. The Journal of Supercomputing, 2013: 1-14.
22. Xing Hu, Dingyi Pei, Chunming Tang, Duncan S. Wong. Verifiable and Secure Outsourcing of Matrix Calculation and its Application. SCIENTIA SINICA Informationis(in chinese).2013,43(7): 842-852.

23. Feng X, Zhang Z. The rank of a random matrix. Applied mathematics and computation, 2007, 185(1): 689-694.