Verifiable secure computation of linear fractional programming using certificate validation

Nedal M. Mohammed[1], Laman R. Sultan[2], Ahmed A. Hamoud[3], Santosh S. Lomte[4]
[1,4]Department of Computer Science, Dr. Babasaheb Ambedkar Marathwada University, Aurangabad, India
[2]Department of Power Mechanics, Basra Technical Institute, Southern Technical University, Al-Basrah, Iraq
[3]Department of Mathematics, Dr. Babasaheb Ambedkar Marathwada University, Aurangabad, India
[1]Department of Computer Science, Taiz University, Taiz, Yemen.

Article Info

Article history:

Received Mar 28, 2019
Revised Jul 8, 2019
Accepted Jul 31, 2019

Keywords:

Certificate validation LFP
Computation outsourcing
Verifiable computation
Verifiable secure computation of LFP

ABSTRACT

Outsourcing of scientific computations is attracting increasing attention since it enables the customers with limited computing resource and storage devices to outsource the sophisticated computation workloads into powerful service providers. However, it also comes up with some security and privacy concerns and challenges, such as the input and output privacy of the customers, and cheating behaviors of the cloud. Motivated by these issues, this paper focused on privacy-preserving Linear Fractional Programming (LFP) as a typical and practically relevant case for verifiable secure multiparty computation. We will investigate the secure and verifiable schema with correctness guarantees, by using normal multiparty techniques to compute the result of a computation and then using verifiable techniques only to verify that this result was correct.

Corresponding Author:

Nedal M. Mohammed,
Department of Computer Science,
Taiz University,
Taiz, Yemen.
Email: dr.nedal.mohammed@gmail.com

1. INTRODUCTION

The powerful advantage of cloud computing is called outsourcing, where the customers with limited computing resource and storage devices can outsource the sophisticated computation workloads into powerful service providers. Despite the tremendous benefits, there are many challenges and security concerns because the cloud server and customer are not in the same trusted domain, to avoid these problems [1-4]. First, to combat the security concern is applying encryption techniques to customer's sensitive information before outsourcing to the cloud but still, there is a challenge how makes the task of computation over encrypted data [5,6]. Second, no guarantee from the cloud on the quality of the computed data and results. For instance, solving financial linear programs is useful for optimizing global profits confidentiality is important because the inputs are sensitive information from multiple companies but correctness is important because the outcome represents financial value.

In theory, correctness and privacy can be achieved by producing cryptographic proofs of correctness in a multi-party way [7,8]. In [9] They achieved Correctness by replicating a computation and comparing the results this done against uncorrelated failure. Without assuming uncorrelated failure or trusted hardware the correctness can be done e.g., [10] by instead producing cryptographic proofs of correctness. Also, privacy can be done when the computation achieved by multiple computation parties using multiparty computation protocols e.g. [11,12].

In this paper, we want to be sure that the results are correct and with the multiple mutually distrusting in putters, also we want to guarantee the privacy of the inputs. We present certificate validation as a general technique for achieving verifiable secure computation of linear fractional programming. We use of El-Gamal encryption [13-15] by combining the computation stage and the validation stage rather than using expensive encryption schemes such as Paillier's cryptosystem.

The rest of the paper is organized as follows: section 2. Shows verifiable computation schema. In section 3. We describes the system model of our proposed Protocol for privacy-preserving outsourcing LFP. In section 4. We provide experimental result analysis for the proposed schema. At last the work conclusion is presented in section 5.

2. VERIFIABLE COMPUTATION

Verifiable computation has been studied by plenty of researchers in various application scenarios. They researched widely how to verify the correctness of computations performed by untrusted parties (without privacy) [16-21].

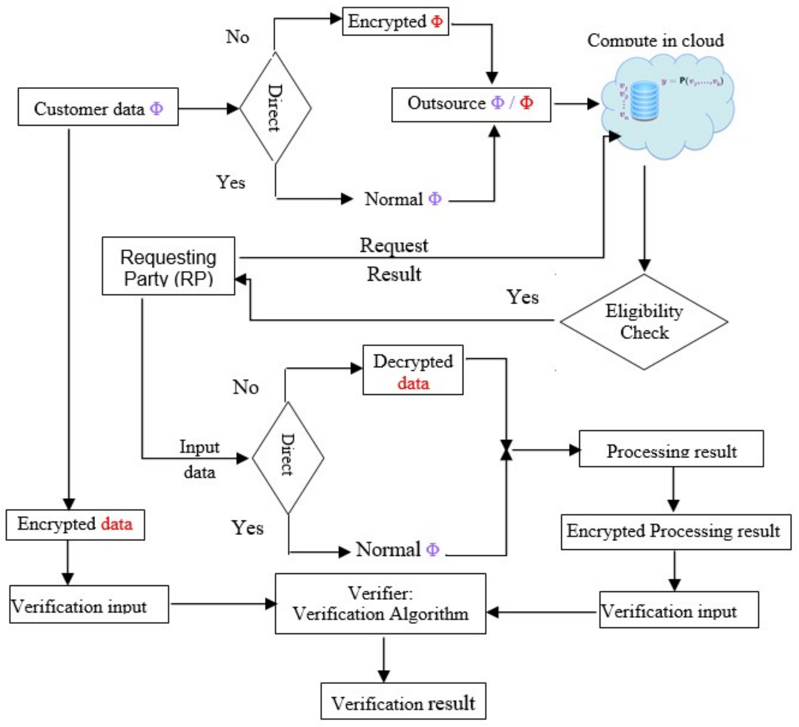

Figure 1. System Structure of Verifiable Computation Scheme.

Verifiable computation schemes are normally based on either computation complexity theory or cryptographic algorithms. Data and computations can be outsourced to another party in order to obtain a processing result in return. However, whether the result is right or wrong could cause a potential risk for a data processing result requester. For outsourced data processing and computations, verification of the computation results is a critical issue to ensure the trust of Computation-as-a-Service [22].

3. PROTOCOL FOR PRIVACY-PRESERVING OUTSOURCING LINEAR FRACTIONAL PROGRAMMING

We present main protocol for privacy-preserving outsourcing with correctness guarantees. We compute a solution and a so-called certificate using normal multiparty computation, and then produce a proof that the solution is valid with respect to the certificate using the El-Gamal-based proofs [23].

3.1. Functions of certificates and validating

To efficiently validate a computation result, we use certificates. In complexity theory, a certificate is a proof that a value lies in a certain set that can be verified in polynomial time.

Let S_1, S_2 be sets and $Y \subseteq S_1$. A polynomial time computable predicate $[\varphi \subseteq S_1 \times S_2]$ is called a validating function for Y if $Y = \{w \in S_1 | \exists c \in S_2 : \varphi(w,c)\}$. If $\varphi(w,c)\, w \in Y$. In our case, a computation is given by a computation function $\varphi(y,a,r)$, and a validating function $\varphi(y,a,r)$. Here, on input x, function f computes function output r and certificate a; validating function φ checks that r is a valid output with respect to x and a. We require that if $(a,r) = f(y)$, then $\varphi(y,a,r)$, but we do not demand the converse: the outcome of the computation might not be unique, and might merely check that some correct solution was found, not that it was produced according to algorithm f. (For instance, a square root finder may return the positive square root while negative square root is also valid.) In our case study, we use that the optimality of a solution to a LFP can be efficiently validated using a certificate.

3.2. The verifiable multiparty computation protocol by certificate validation

We present Verifiable Multiparty computation protocol by certificate validation (VerMPC) protocol to compute $(a,r) = f(x)$, and prove this result X_i is correct. We use passively secure multiparty computation protocols based on (t,n) Shamir sharing with $n = 2t+1$. In these protocols, the input parties encrypt and announce their inputs, then makes a proof of knowledge of the corresponding plaintext then broadcast for this encryption and proof. Next, the parties provide the plaintext and randomness of the encryption to the two computation parties who will later prove the result is correct. The two computation parties check if the provided sharing of the input is consistent with the encryptions that were broadcast for preventing corrupted input parties learns information about both their encrypted and their secret shared inputs, this done by encrypting their shares of the inputs then using the homomorphic property of the cryptosystem for checking correctness. Then, the actual computation takes place in the third computation party. The two parties holding additive shares of the input Shamir-share them between all three computation parties, then the computation is performed between the three parties. These two of the computation parties produce the encrypted result and prove its correctness [24]. The computation parties send their additive shares of the result and the randomness of their encryption shares results to the resulting party (the encryptions of the certificate and proof of correctness) [13,25-27]. The result party checks the proofs of knowledge provided by the in putters computes the encrypted results from its shares and use Verify algorithm to verify the correctness.

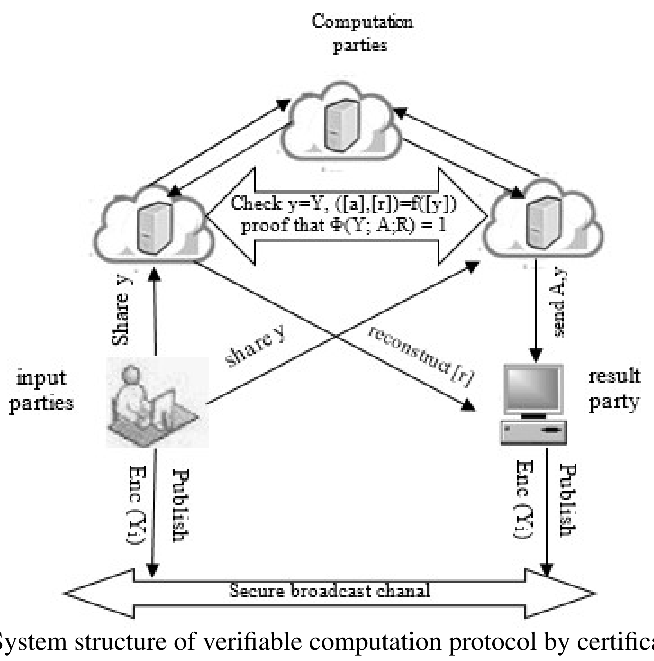

Figure 2. System structure of verifiable computation protocol by certificate validation

3.3. Secure and verifiable linear fractional programming

The LFP is a special class of mathematical optimization expressed in the following standard form [28]:

$$max \quad Z = \frac{cy + \delta}{dy + \xi}$$

$$\text{S.t.} \quad Ay \leq b, \quad By \geq 0, \tag{1}$$

where (1) the objective function is a linear fractional function (ratio of two linear functions) y is an $n \times 1$ vector of variables which are to be determined, c and d are $n \times 1$ column vectors of coefficients, and set of constraints are a system of linear equalities and inequalities (affine constraints) A is $m \times n$ matrix of coefficients, b is $m \times 1$ column vector of coefficients and δ, ξ are constants. B is $n \times n$ nonsingular matrix. For instance, the LFP represents the problem to find x_1, x_2 satisfying

$$\begin{aligned} &max \quad Z = \frac{2x_1 + 3x_2}{x_1 + x_2 + 1} \\ &\text{S.t.} \\ &\quad x_1 + x_2 \leq 3 \\ &\quad x_1 + 2x_2 \leq 3 \\ &\quad x_1, x_2 \geq 0. \end{aligned}$$

To find the optimal solution of a fractional linear program, typically an iterative algorithm called the simplex algorithm is used after convert LFP to LP [29].

$$\begin{aligned} &max \quad F(y) = 2y_1 + 3y_2 \\ &\text{S.t.} \\ &\quad 4y_1 + 4y_2 \leq 3 \\ &\quad 4y_1 + 5y_2 \leq 3 \\ &\quad y_1, y_2 \geq 0. \end{aligned}$$

$$A = \begin{pmatrix} 4 & 4 \\ 4 & 5 \end{pmatrix}, b = \begin{pmatrix} 3 \\ 3 \end{pmatrix}, c = \begin{pmatrix} 2 \\ 3 \end{pmatrix}.$$

Each iteration involves several comparisons and a Gaussian elimination step, making it quite heavy for multiparty computation. For relatively small instances, passively secure linear fractional programming is feasible [11], but actively secure MPC much less so when including pre-processing.

Theorem: We prove that y it is optimal using the optimal solution p of the so-called dual LP maximise $\quad b \cdot p \quad$ such that $\mathbf{A} \cdot p \leq c, p \leq \mathbf{0}$.

Proof: The solutions $(\frac{y_1}{q}, \cdots, \frac{y_n}{q}) \quad and \quad (\frac{p_1}{q}, \cdots, \frac{p_m}{q}) \quad (y \in \mathbb{Z}^n, p \in \mathbb{Z}^m, q \in \mathbb{N}^+)$ are both optimal if the following conditions hold:

$$q \geq 1; \quad p \cdot b = c \cdot y; \qquad \mathbf{A} \cdot y \leq q \cdot b; \quad y \geq 0;$$
$$\mathbf{A}^T \cdot p \leq q \cdot c; \quad p \leq 0.$$

Also, the simplex algorithm for finding y turns out to also directly give p. To turn the above criterion into a set of polynomial equations, we define the certificate to consist of bit decompositions of $(q \cdot b - \mathbf{A} \cdot y)_i$, y_i, $(q \cdot c - \mathbf{A}^T \cdot p)_i$, $and \ -p_i$, and prove that each bit decomposition $b_0, b_1, ...$ sums up to the correct value v (with equation $v = b_0 + 2 \cdot b_1 + \cdots$) and contains only bits (with equations $b_i \cdot (1 - b_i) = 0$).

4. EXPERIMENTAL RESULT

The experimental results are the average of multiple trials. We design numerical experiments to evaluate the efficiency of the mechanism. We ran our mechanism on several LFPs. We measured the time to solve the LFP and to compute the certificate (this depends on the LFP size, number of iterations needed, and the bit length for internal computations), the time for PolyProve to produce a proof, and for PolyVer to verify it (this depends on the LFP size and bit length for the proof). Figure 3. Shows the performance numbers of our experiments.

Table 1. Performance of the proposed scheme for infeasible case

Problem Size	No. of Iterations	Verify Algorithm	Prove Algorithm	Compute Certificate
m=5, n=5	4	11.450	85	110
m=20, n=20	9	79.50	200	347
m=48, n=70	25	300.340	986	1100
m=103, n=150	62	1000	4806	8500

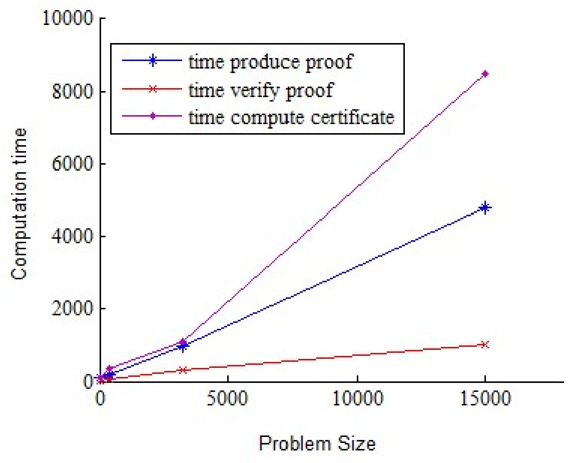

Figure 3. Time cost for each phase of verifiable secure computation of LFP using certificate validation

For the LFPs in our experiments, we find that producing proof adds little overhead to compute the solution and that verifying the proof is much faster than participating in the computation. As a consequence, for the recipient, outsourcing both guarantees correctness and saves time compared to participating in the computation. In general, one expects the difference between computing the solution and proving its correctness to be more pronounced for larger problems: indeed, both the computation and the correctness verification scale in the size of the LFP, but computation additionally scales in the number of iterations needed to reach the optimal solution. This number of iterations typically grows with the LFP size. However, we only found this for the biggest LFP, where proving is over four times faster than computing, for the other programs, this factor was around two. An explanation for this is that also the bit length of solutions (which influences proving time) typically grows with the number of iterations.

5. CONCLUSION

In this paper, we combined passively secure three-party computation with El-Gamal-based proofs. We have shown how to use certificate validation to obtain correctness guarantees for privacy-preserving outsourcing of LFP. The security guarantees of our model lie in between passive security (that does not guarantee correctness in case of active attacks) and active security (that also guarantees privacy in this case). For LFP, verifying results takes much less time than participating in an actively secure computation; in fact, it even takes less time than participating in a passively secure computation without any correctness guarantees. Hence, for computations on inputs of mutually distrusting parties, privacy-preserving outsourcing with correctness guarantees provides a compelling combination of correctness and privacy guarantees.

REFERENCES

[1] C. Hu, A. Alhothaily, A. Alrawais, X. Cheng, C. Sturtivant and H. Liu, ”A secure and variable outsourcing scheme for matrix inverse computation,” *IEEE INFOCOM*, 17(4) (2017), 2304–2312.

[2] N. M. Mohammed and S. S. Lomte, ”Recent advances on secure computations outsourcing in cloud computing,” *Asian Journal of Mathematics and Computer Research*, 24(4) (2017), 192–205.

[3] N. M. Mohammed and S. S. Lomte, ”Secure computations outsourcing of mathematical optimization and linear algebra tasks: Survey,” *International Journal for Research in Engineering Application & Management*, (2019), 6–11.

[4] K. Zhou and J. Ren, ”CASO: Cost-Aware secure outsourcing of general computational problems, ” *IEEE Transactions on Services Computing*, 2018.

[5] C. Gentry, ”Computing arbitrary functions of encrypted data,” *Comm. ACM.*, 53(3) (2010), 97–105.

[6] N. M. Mohammed, L. R. Sultan, S. S. Lomte, ”Privacy preserving outsourcing algorithm for two-point linear boundary value problems,” *Indonesian Journal of Electrical Engineering and Computer Science*, 16(2) (2019), 1065–1069.

[7] W. Shen, Y. Bo, C. Xianghui, C. Yu and S. Xuemin, ”A distributed secure outsourcing scheme for solving linear algebraic equations in ad hoc clouds,” *IEEE Transactions on Cloud Computing*, 4 (2017).

[8] N. Mohammed and S. Lomte, ”Secure and efficient outsourcing of large scale linear fractional programming,” *International Conference on Computing in Engineering and Technology (ICCET)*, (2019), To Appear.

[9] M. Castro and B. Liskov, ”Practical Byzantine fault tolerance and proactive recovery,” *ACM Transactions on Computer Systems*, 20(4) (2002), 398–461.

[10] B. Parno, J. Howell, C. Gentry and M. Raykova, Pinocchio: Nearly practical verifiable computation, *In Proceedings of S & P.*, 13, 2013.

[11] P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. P. Jakobsen, M. Kraigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. I. Schwartzbach and T. Toft, ”Secure multiparty computation goes live,” *In Proceedings of Financial Crypto.*, 09 (2009).

[12] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl and N. Smart, ”Practical covertly secure MPC for dishonest majority - Or: breaking the SPDZ limits,” *In Proceedings of ESORICS.*, 13 (2013).

[13] T. El Gamal, ”A public key cryptosystem and a signature scheme based on discrete logarithms,” *IEEE Transactions on Information Theory*, 31(4) (1985), 469–472.

[14] A. Lopez-Alt, E. Tromer and V. Vaikuntanathan, ”On-the-way multiparty computation on the cloud via multi key fully homomorphic encryption,” *In Proceedings of STOC.*, 12 (2012), 1219–1234.

[15] R. Tamassia and N. Triandopoulos, ”Certification and authentication of data structures,” *In Proceedings of AMW.*, 10 (2010).

[16] P. Ananth, N. Chandran, V. Goyal, B. Kanukurthi and R. Ostrovsky, ”Achieving Privacy in Verifiable Computation with Multiple Servers - With-out FHE and without Pre-processing,” *In Proceedings of PKC.*, 14 (2014).

[17] S. Arora and S. Safra, ”Probabilistic checking of proofs: A new characterization of NP,” *J. ACM.*, 45(1) (1998), 70–122.

[18] C. Baum, I. Damgard and C. Orlandi, ”Publicly auditable secure multi party computation,” *In Proceedings of SCN*, 14 (2014).

[19] R. Canetti, ”Security and composition of multi-party cryptographic protocols,” *Journal of Cryptology*, 13:2000, 1998.

[20] I. Damgard, K. Damgard, K. Nielsen, P. S. Nordholt and T. Toft, ”Condential benchmarking based on multiparty computation,” *IACR Cryptology ePrint Archive*, 2015:1006, 2015.

[21] Y. Ejgenberg, M. Farbstein, M. Levy and Y. Lindell, ”SCAPI: The Secure Computation Application Programming Interface,” *Cryptology ePrint Archive, Report*, 2012/629, 2012.

[22] D. Fiore, R. Gennaro and V. Pastro, ”Efficiently verifiable computation on encrypted data,” *In Proceedings of CCS.*, 14 (2014).

[23] J. Hromkovic, ”Algorithmics for hard problems - introduction to combinatorial optimization, randomization, approximation, and heuristics,” *Springer*, 2001.

[24] B. Schoenmakers and M. Veeningen, ”Universally Verifiable Multiparty Computation from Threshold Homomorphic Cryptosystems,” *Cryptology ePrint Archive, Report,* 2015/058, 2015.

[25] S. Goldwasser, Y. Kalai and G. Rothblum, ”Delegating computation: interactive proofs for muggles,” *In Proc. of STOC.*, 08 (2008), 113–122.

[26] S. Kamara, P. Mohassel and M. Raykova, ”Outsourcing multi-party computation,” *IACR Cryptology ePrint Archive*, 2011:272, 2011.

[27] M. Keller, G. L. Mikkelsen and A. Rupp, ”Efficient threshold zero knowledge with applications to user-centric protocols,” *In Proceedings of ICITS.* 12 (2012).

[28] A. Charnes and W. Cooper, ”Programming with linear fractional functions,” *Naval Re-search Logistics Quarterly*, (1962), 181–186.

[29] EB. Bajalinov, ”Linear-fractional programming theory, methods, applications and software,” *Springer Science & Business Media*, 2013.