A Rational Delegating Computation Protocol Based on Smart Contract

Juan Ma[1], Yuling Chen[1(✉)], Guoxu Liu[2], Hongliang Zhu[3], and Changgen Peng[1]

[1] State Key Laboratory of Public Big Data, College of Computer Science and Technology, Guizhou University, Guiyang, China
ylchen3@gzu.edu.cn

[2] Blockchain Laboratory of Agricultural Vegetables, Weifang University of Science and Technology, Shouguang, China

[3] Engineering Lab for Cloud Security and Information Security Center of BUPT, Beijing, China

Abstract. The delegating computation has become an irreversible trend, together comes the pressing need for fairness and efficiency issues. To solve this problem, we leverage game theory to propose a smart contract-based solution. Firstly, according to the behavioral preferences of the participants, we design an incentive contract to describe the motivation of the participants. Secondly, to satisfy the fairness of the rational delegating computation, we propose a delegating computation protocol based on the smart contract. Specifically, rational participants are to gain the maximum utility and reach the Nash equilibrium in the protocol. Besides, we design a reputation mechanism with a reputation certificate, which measures the reputation from multiple dimensions. And the reputation is used to assure the client's trust in the computing party to improve the efficiency of the protocol. Finally, the analysis results show that the proposed protocol solves the complex traditional verification problem. Meanwhile, we prove the fairness and correctness of the protocol.

Keywords: Rational delegating computation · Smart contract · Nash equilibrium · Incentive contract · Reputation mechanism

1 Introduction

The delegating computation is that some of these devices are computationally weak due to various resource constraints. As a consequence, there are tasks, which potentially could enlarge a device's range of application, that are beyond its reach. A solution is to delegate computations that are too expensive for one device, to other devices which are more powerful or numerous and connected to the same network [1]. The traditional delegating computation protocols generally assume that participants need to be honest or malicious. In fact, a lot of participants are rational in the execution process, rational participants always choose a strategy to maximize their utilities. Besides, owing to the high cost of computation and communication complexity in the verification process of delegating computation, the efficiency of the protocol will be reduced.

J. Xiong et al. (Eds.): MobiMedia 2021, LNICST 394, pp. 389–398, 2021.
https://doi.org/10.1007/978-3-030-89814-4_28

Combining the game theory and traditional delegating computation, the rational delegating computation protocol is a part of rational cryptography. From the perspective of the participant's self-interest, the protocol utilizes the utility functions to ensure the correctness and reliability of the calculation results. In recent years, many scholars have conducted researches on rational delegating computation. Li et al. combined game theory and fully homomorphic encryption technology to construct a rational delegating computation protocol, which guarantees the interests of both participants [2]. Tian et al. constructed a rational delegating computation protocol based on Yao's garbled circuit and fully homomorphic encryption technology [3]. But in the process of delegating computation, if the malicious participants don't follow the rules in the protocol, the interests of honest participants will be reduced. Therefore, how to ensure the fairness of the protocol, which need to be considered.

Recently, the fairness of delegating computation is one of the hot topics in current research, and the existing researches utilize a third-party (e.g., bank [4], semi-trusted third-party [5, 6], trusted third-party [7, 8]) to overcome these issues. However, in the computation process, with a third-party, potential security problems will inevitably occur, e.g., unreasonable Nash equilibrium, privacy leakage, and low efficiency. To eliminate the drawbacks, many researchers adopt smart contract to realize the Peer-to-Peer transaction between the clients and the computing parties. Zhou et al. combined the game theory with traditional delegating computation and proposed a three-party game rational delegating computation protocol based on smart contracts [9]. Dong et al. combined game theory and smart contract to design a reasonable prisoner contract, collusion contract, and betrayal contract [10]. Song et al. researched the application of game theory in blockchain and proposed that the application of smart contract in delegating computation effectively prevent the collision problem of computing parties [11]. The payment scheme based on smart contract technology guarantees the fairness of participants. However, in the process of delegating computation, different tasks or utility have different impacts on the strategies of the computing parties, leading the clients to get an incorrect result. Therefore, how to select a reliable computing party is an urgent problem.

In the process of delegating computation, the reputation mechanism is designed to improve the reputation of honest participants and reduce the reputation of malicious participants. Jiang et al. proposed a rational delegating computation based on reputation and contract theory, which ensures that the client selects a reliable computing party [12]. Li et al. proposed a fair payment protocol based on Bitcoin time commitment, which ensures the fairness of participant's payment by using Bitcoin time commitment technology [13]. However, in the process of selecting the computing party, the reputation of the computing party will be updated every round. Therefore, how to efficiently view the latest reputation of the computing party is a key issue that needs to be resolved.

To solve the aforementioned problems, we propose a rational delegating computation protocol based on smart contract, which realizes the optimal utility of all rational participants, and guarantees the correctness of the calculation results and the fairness of the payment process. Besides, we design a reputation mechanism for measuring the reputation from different dimensions to select the high-quality computing parties. The main contributions are as follows:

1. According to the behavioural preferences of the participants, we design an incentive contract to motivate the participants to choose the strategy honestly, which reaches a reasonable Nash equilibrium result.
2. Based on smart contract, we propose a rational delegating computation protocol, which realizes the fairness of rational delegating computation. And we design a reputation mechanism for the client to choose high-quality computing parties. In addition, we prove the fairness and correctness of the protocol.

The rest of the paper is structured as follows. Section 2 introduces concepts such as game theory, Nash equilibrium, and smart contract. Section 3 proposes an incentive contract based on a smart contract. Section 4 establishes a reputation mechanism that is convenient for the client to choose the computing party. Section 5 proves the fairness and correctness of the protocol. Section 6 calculates the cost of the contract and analyzes the performance of the protocol. Section 7 summarizes the paper and the layout of future research.

2 Preliminaries

2.1 Game Theory

Definition 1 (Standard Game): The standard form of a n-player game is composed of three elements: player set P, strategy space S and utility function u, denoted as $G = \{P, S, u\}$, where $P = \{P1, ..., Pn\}$, $S = \{S1, ..., Sn\}$, $u = \{u1, ..., un\}$. Any specific strategy $si \in Si$ indicates that strategy si is the key element of the strategy set Si, and utility function $ui : S \to R$ (R represents the real number space) denotes the profits of the players i under different strategy profiles [14].

Definition 2 (Nash Equilibrium): A strategy profile $s^* = \{s1^*, ..., sn^*\}$ is a Nash Equilibrium of game $G = \{P, S, u\}$, if $u_i(si^*, s - i^*) \geq u_i(sj^*, s - j^*)$ holds for each player $Pi(i = 1, ..., n)$ and all $sj \in Si$. Obviously, if player $i \neq j$ complies with the strategy si^*, then the player j will not deviate from the strategy sj^*, as it will not benefit at all. In principle, there may be multiple Nash equilibrium in a game [15].

2.2 Smart Contract

Definition 3 (Smart contract): *Smart contracts* are in the blockchain environment, allowing the definition and execution of contracts signed on the blockchain. It is the automated execution of the contract, and its essence is a piece of code written on the blockchain. Smart contract is the basis of the program-ability of blockchain, and each node does not rely on a third-party to automatically execute the contract. Broadly speaking, a smart contract is a set of rules encoded in a programming language. Once the execution requirements of the code are met, the script will be automatically executed to realize the operation, and this process does not require the participation of a trusted third-party [16].

2.3 System Model

The system model considered in our construction comprises one client denoted by Cd and multiple computing parties denoted by $\{C1, C2, ..., Cn\}$.

As illustrated in Fig. 1, the system model consists of eight steps. Firstly, the client broadcasts the task to the computing party, and the interested computing party returns the response request. Next, the client will select two computing parties based on the reputation to perform outsourcing tasks and sign a contract with them. According to the content of the signed contract, the client and the computing party respectively deposit the deposit stipulated in the contract into the smart contract. Then, the computing party performs the task and sends the calculation result to the client. The client judges whether the computing party has executed the task correctly according to the calculation result, and makes the corresponding response and payment. Finally, according to the interactive behavior of the computing party, the client updates and uploads the reputation of the computing party.

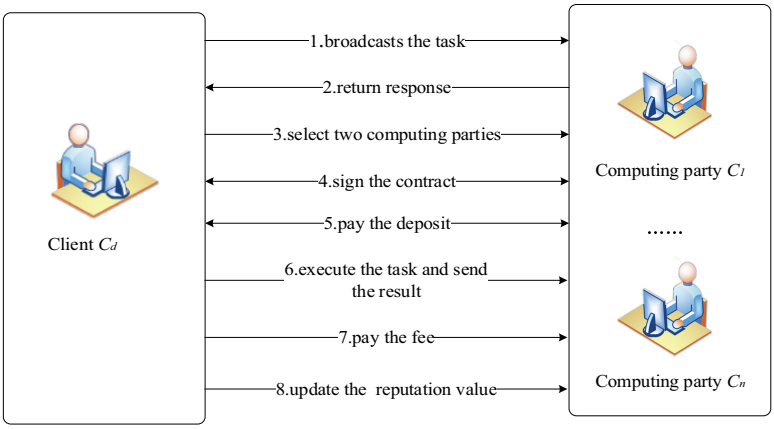

Fig. 1. System model

3 Incentive Contract

3.1 Deposit Variables

Since the incentive contract is introduced, we need to define more deposit variables. They are all non-negative.

- g: the client agrees to pay to a computing party for computing the task.
- v: the computing party's cost for computing the task.
- r: the deposit to invoke the TTP.
- c: the deposit a computing party pays to the client in order to get the task.
- $2g + r$: the deposit of a client in the smart contract.

The following relations is obvious:
(1) $g - v > 0$ (2) $r - 2c > 0$

3.2 Contract Content

We will introduce the specific content of Incentive contract, and we analyze each step in detail by studying the utility function of rational players. The contract is as follows.

Contract content

**Step1(Publish task):* The client C_d publishes the task.

**Step1.1:* C_d broadcasts the task X to all computing parties;

**Step1.2:* If C decides to accept task will return response request P.

**Step2:* C_d selects $C_i (i \in \{1, 2\})$ based on the reputation.

**Step3(Sign contract):* C_d, C_1 and C_2 must sign the *contract* to start it, otherwise *contract* terminates.

**Step4(Pay deposit):* C_d, C_1 and C_2 pay the deposit $(2g + r, c, c)$; otherwise *contract* terminates.

**Step5(Verification the result).* The client checks the results returned by the computing party.

**Step5.1:* If both C_1, C_2 deliver the results within the specified time, and the results are equal. Transferring $Cd's$ deposit g to each C_i, returning $C_i's$ deposit c.

**Step5.2:* Otherwise, the TTP is invoked and one of the following steps is performed;

**Step5.2.1:* If both C_1, C_2 are one party honest (return the correct result C_h) and one party malicious (return the wrong result C_f or return the random result C_r). Transferring $Cd's$ deposit g to each C_h and $C_f's$ (or $Cr's$) deposit c to C_d, and pays fee r to TTP;

**Step5.2.2:* If both C_1, C_2 return wrong result. Transferring deposit c to C_d, returning $Cd's$ deposit $2g$, and pays fee r to TTP;

**Step5.2.3:* If both C_1, C_2 return random result. Transferring C_i deposit c to C_d, returning $Cd's$ deposit $2g$, and pays fee r to TTP;

**Step5.3:* For each C_i, if C_i fail to deliver the result within the specified time. Transferring deposit c to C_d, returning $Cd's$ deposit $2g$.

**Step6:* If the C_d deviates from the protocol, transferring $Cd's$ deposit g to each C_i, returning each $C_i's$ deposit c.

4 The Proposed Reputation Scheme

In this part, the reputation mechanism mainly includes five stages. The first stage, the content of the reputation certificate is defined. The second stage, the client selects the appropriate computing party to compute the task according to the reputation. The third stage, the client evaluates the current reputation based on the computing party's honesty. The fourth stage, the client computes the global reputation. In the fifth step, the computing party updates the reputation certificate.

4.1 Reputation Certificate

In the delegating computation process, the client selects the appropriate computing party according to its task requirements. In a reputation mechanism with a reputation certificate, the computing party has a high-quality reputation as the basis for obtaining the task. Generally, the reputation of the computing party is higher, the probability of being selected is greater. The reputation certificate format is shown in Table 1.

Table 1. Reputation certificate.

Symbol	Describe
IDcd	*ID* of the client
IDc	*ID* of the computing party
Rn	Number of rounds
Sigcd	Signature of the client
Sigc	Signature of the computing party
HRVc	The historical reputation of the computing party

4.2 Search of Historical Reputation

After the client publishes the task, the computing party needs to return a response and generates the latest reputation certificate to get the task. For the request returned by the computing party, the client selects two appropriate computing party based on the reputation. The format of the new reputation certificate is as follows.

$$CRCCn + 1 = IDC \| Rn + 1 \| HRVn \| Sign\,(IDC \| Rn + 1 \| HRVn)$$

4.3 Evaluation of Current Reputation

Usually, the computing party's strategies are affected by different tasks or utilities, which cause them to have different reputations. Reputation is whether the computing party's

behavior is honest or not after each round of tasks. In general, the computing party has two reputation statuses: honest or malicious. Specifically, When the computing party's reputation value in round $t+1$ is higher than that in round t, i.e. $Lrept+1 \geq Lrept$, we consider the computing party to be honest. On the contrary, When the computing party's reputation value in round $t+1$ is lower than round t, i.e. $Lrept+1 \leq Lrept$, we consider the computing party to be malicious. The evaluation of reputation is as follows Eq. (1).

$$Lrept+1 = \alpha i, t\left[\frac{H}{S} * D(r) * A(r) * T(r)\right] \tag{1}$$

Let $\alpha i, t$ be the client Cd interactive evaluation to the computing party Ci in round t, here, $\alpha i, t \in \{0, 1, -1\}$. In order to reward the computing party who performed well in the execution of the task and punish the computing party who failed to complete the task as required, we propose the setting as shown in Eq. (2).

$$\begin{cases} \alpha i, t=1, Lrepi, t=\frac{H}{S} * D(r) * A(r) * T(r) \\ \alpha i, t=0, Lrepi, t=0 \\ \alpha i, t=-1, Lrepi, t=-\left[\frac{H}{S} * D(r) * A(r) * T(r)\right] \end{cases} \tag{2}$$

If and only if there is no any interaction between the client Cd and the computing party Ci, $Lrepi, t=0$. Obviously, for any Ci, when $t=0$, $\alpha i, t=0$, then $Lrepi, 0=0$. If and only if the computing party Ci honestly performs the delegation task in round t, $\alpha i, t=1$, then $Lrepi, t=\frac{H}{S} * D(r) * A(r) * T(r)$. Conversely, if and only if the computing party Ci is behaves dishonestly in round t, $\alpha i, t=-1$, then $Lrepi, t=-\left[\frac{H}{S} * D(r) * A(r) * T(r)\right]$.

Where, $Lrepi, t$ satisfies $-1 \leq Lrepi, t \leq 1$, $Lrepi, t$ is the reputation of the current round, H represents the number of honest calculations by the computing party,S represents the total number of calculations by the computing party, $D(r)$ represents the complexity coefficient of the delegating computation and satisfies $0 \leq D(r) \leq 1$, $A(r)$ represents the benefit coefficient of the delegating computation and satisfies $0 \leq A(r) \leq 1$, $T(r)$ represents the time to submit the result of the calculation.

(1) When $Lrepi, t=-1$, it means that Ci is completely malicious in the process of delegating computation of round t;
(2) When $Lrepi, t=0$, it means that there is no interaction between Cd and Ci in the process of delegating computation of round t;
(3) When $Lrepi, t=1$, it means that Ci is completely honest in the delegating computation of round t.

4.4 Computation of Global Reputation

In this stage, after the client evaluates the reputation of the current round, the client computes the global reputation according to the historical reputation and the reputation of the current round. The computation of global reputation is as follows Eq. (3).

$$Grep_{n+1} = Grep_n + Grep_{n+1} \tag{3}$$

$Grepn+1$ represents the global reputation of the computing party, and $Grepn$ represents the global reputation of the computing party in the last round.

4.5 Update of Reputation Certificate

Finally, the client signs the global reputation of the computing party and uploads it to the blockchain. Then, the computing party updates the content of its reputation certificate, which facilitates the generation of the latest reputation certificate for the next round of tasks. The format of the latest reputation certificate is as follows.

$$CRCC, n+1 = CRCC - Cd, n+1 \| Sign\,(CRCC - Cd, n+1)$$

5 Protocol Proof

In this section, we prove the agreement in detail from the two aspects of fairness and correctness of the protocol. More details are given in Sects. 5.1 and 5.2.

5.1 Fairness Proof

Theorem 1. The rational delegating computation protocol based on smart contract is fair.

Proof. To ensure the fairness of the protocol, at the initial stage of the protocol, the client and the computing party respectively pay a deposit of to the smart contract. In the payment process of delegating computation, there are four cases as follows:

Case 1: When the computing party is honest, the smart contract transfers the deposit g to the computing party's account;

Case 2: When the computing party is malicious, the smart contract confiscates the malicious computing party's deposit c and transfers deposit c to the client's account;

Case 3: When the client is malicious, the smart contract confiscates the deposit $2g$ and transfers deposit g to the two computing parties' account respectively;

Case 4: When the client is honest, the client can get the correct calculation result.

In order to avoid the situation that the third-party is dishonest or bought off, based on the smart contract technology, we construct a rational delegating computation protocol to achieve fairness.

5.2 Correctness Proof

Theorem 2. The rational delegating computation protocol based on smart contract is correct.

Proof. There are three cases for the proof as following:

(1) When $C1$ chooses the honest strategy, if $C2$ chooses the honest strategy, the utility is $u(g-v)$; else if $C2$ chooses the random values strategy, the utility is $u(-c)$; else $C2$ chooses the malicious strategy, the utility is $u(-c-v)$; which is $u(g-v) > u(-c) > u(-c-v)$.

(2) When $C1$ chooses the malicious strategy, if $C2$ chooses the honest strategy, the utility is $u(g+c-v)$; else if $C2$ chooses the random values strategy, the utility is $u(-c)$; else $C2$ chooses the malicious strategy, the utility is $u(-c-v)$; which is $u(g+c-v) > u(-c) > u(-c-v)$.

(3) When $C1$ chooses the random value strategy, if $C2$ chooses the honest strategy, the utility is $u(g + c - v)$; else if $C2$ chooses the random values strategy, the utility is $u(-c)$; else $C2$ chooses the malicious strategy, the utility is $u(-c - v)$; which is $u(g + c - v) > u(-c) > u(-c - v)$.

Similarly, $C2$ also has the above three cases. According to analysis, only when participants choose the honesty strategy, maximizing their utility and reach Nash equilibrium.

6 Conclusion

Combining game theory with smart contract, in this paper, we propose a rational delegating computation protocol based on smart contract. More specifically, we analyze the strategies and utilities of participants, then we utilize smart contract to substitute the third-party to ensure the fairness of protocol. Also, we design a reputation scheme, which can measure the reputation from different dimensions to ensure the client chooses a reliable computing party. Our future work will focus on the collusion problem between computing parties in delegating computation.

Acknowledgement. This work was supported by the National Natural Science Foundation of China under Grant No. 61962009, Major Scientific and Technological Special Project of Guizhou Province under Grant No. 20183001, the foundation of Guizhou Provincial Key Laboratory of Public Big Data under Grant No. 2018BDKFJJ008.

References

1. Xue, R., Wu, Y., Liu, M.H., Zhang, L.F., Zhang, R.: Progress in verifiable computation. Scientia Sinica Inf. **45**(11), 1370–1388 (2015)
2. Li, Q.X., Tian, Y.L., Wang, Z.: Rational delegation computation protocol based on fully homomorphic encryption. Acta Electron. Sin. **47**(2), 216–220 (2019)
3. Tian, Y.L., Li, Q.X., et al.: Proven secure rational delegated computing protocol. J. Commun. **40**(07), 135–143 (2019)
4. Carbunar, B, Tripunitara, M.: Fair payments for outsourced computations. IEEE Trans. Parallel Distrib. Syst. **23**(2), 1–9 (2010)
5. Xu, G., Amariucai, G.T., Guan, Y.: Delegation of computation with verification outsourcing: curious verifiers. IEEE Trans. Parallel Distrib. Syst. **28**(3), 717–730 (2017)
6. Huang, H., Chen, X.F., Wu, Q.H., Huang, X.Y., Shen, J.: Bitcoin-Based fair payments for outsourcing computations of fog devices. Future Gener. Comput. Syst. **78**(2), 850–858 (2016)
7. Chen, X.F., Li, J., Susilo, W.: Efficient fair conditional payments for outsourcing computations. IEEE Trans. Inf. Forensics Secur. **7**(6), 1687–1694 (2012)
8. Yin, X., Tian, Y.L., Wang, H.L.: Fair and rational delegati on computation protocol. Ruan Jian Xue Bao/J. Softw. **29**(7), 1953–1962 (2018) (in Chinese).
9. Zhou, Q.X., Li, Q.X., Fan, M.M.: Anti-collusion delegation computation protocol of three-party game based on smart contract. Comput. Eng. **46**(8), 124–131, 138 (2020)
10. Dong, C., Wang, Y., Aldweesh, A., et al.: Betrayal, distrust, and rationality: smart counter-collusion contracts for verifiable cloud computing, pp. 211–227. ACM (2017)

11. Song, L.H., Li, T., Wang, Y.l.: Application of game theory in blockchain. Chinese J. Crypt. **6**(01), 100–111 (2019)
12. Jiang, X, Tian Y. Rational delegation of computation based on reputation and contract theory in the UC framework. In: Yu, S., Mueller, P., Qian, J. (eds.) Security and Privacy in Digital Economy, SPDE 2020. Communications in Computer and Information Science, vol. 1268. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-9129-7_23
13. Li, T., Tian, Y.L., et al.: A fair payment scheme based on blockchain in delegated computing. J. Commun. **41**(03), 80–90 (2020)
14. Tian, Y.L., Guo, J., Wu, Y., et al.: Towards attack and defense views of rational delegation of computation. IEEE Access **7**, 44037–44049 (2019)
15. Osboren, M.: An Introduction to Game Theory, pp. 151–232. Oxford University Press, New York (2004)
16. Liu, F.M., Chen, Y.T.: A review of blockchain technology research. J. Shandong Normal Univ. **35**(03), 299–311 (2020)