IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received November 4, 2020, accepted December 2, 2020, date of publication December 7, 2020, date of current version December 18, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3043042

CVT: A Crowdsourcing Video Transcoding Scheme Based on Blockchain Smart Contracts

YULING CHEN[1,2], HONGYAN YIN[3], YUEXIN XIANG[4], WEI REN[4,2], (Member, IEEE), YI REN[5], AND NEAL NAIXUE XIONG[6], (Senior Member, IEEE)

[1]College of Computer Science and Technology, Guizhou University, Guiyang 550025, China
[2]State Key Laboratory of Public Big Data, Guiyang 550025, China
[3]Technical Center of Beijing Customs, Beijing 100176, China
[4]School of Computer Science, China University of Geosciences, Wuhan 430074, China
[5]School of Computing Science, University of East Anglia, Norwich NR4 7TJ, U.K.
[6]Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, OK 74464, USA

Corresponding author: Wei Ren (weirencs@cug.edu.cn)

This work was supported in part by the National Key Research and Development Program of China Project under Grant 2018YFF0212106, in part by the National Natural Science Foundation of China under Grant 61972366 and Grant 61962009, in part by the Major Scientific and Technological Special Project of Guizhou Province under Grant 20183001, and in part by the Foundation of Guizhou Provincial Key Laboratory of Public Big Data under Grant 2018BDKFJJ009, Grant 2019BDKFJJ003, and Grant 2019BDKFJJ011.

ABSTRACT Streaming media has been largely used by millions of users every day. The number of customers and programs, e.g., TV series, movies, and various shows, are still growing fast. However, the demand for video transcoding for various personal terminal devices results in the shortage of computing resources and the prolongation of processing delay in centralized video transcoding systems. To solve this issue, we propose a blockchain, especially, smart contract based scheme that can achieve decentralized and on-demand crowdsourcing for video transcoding, which remarkably mitigates the transcoding overhead. Specifically, our scheme consists of four key components such as employers, workers, task allocation, and payment. An employer initializes the smart contract, releases the task, and initiates the smart contract. Workers bid for the task, and the successful bidder will obtain the task and execute the task. The task allocation mechanism and the payment mechanism can guarantee the profits of both and encourage both as well. Moreover, the smart contract consists of the bidding contract and the task execution contract. The extensive analysis of our proposed scheme justified the feasibility, security for defending against typical threats, applicability in realistic situations, and portability for most multimedia such as videos and audios.

INDEX TERMS Blockchain, crowdsourcing, video transcoding, smart contracts.

I. INTRODUCTION

Recently, streaming media such as Netflix, Disneyland+, and Twitch attract increasing subscribers since it enables conveniently and pervasively video accessing. However, current streaming media still imposes several inevitable problems. The most critical issue is to guarantee the efficiency of video transcoding during the video transmission process in the centralized systems. The sharp increment of the number of users and videos requires more computation resources for video transcoding, which may slow down or even terminate the process of video transcoding and so as to remarkably damage the user experiences.

In order to solve the weaknesses of current centralized systems, crowdsourcing is proposed as a promising approach. Crowdsourcing is a modern business model to distribute the assignments to the volunteers. For example, a company can hardly complete a mass of processing and computing work (e.g., video transcoding) due to the limit of several factors, such as computation resources and the performance of the devices. Because of the characteristics of crowdsourcing, it is an efficient and beneficial way to solve this problem. But we still need to think about how to protect the interests of each entity during the crowdsourcing process while maintaining the efficiency of crowdsourcing.

Researchers have proposed some methods to implement higher efficient video transcoding approaches [1]–[3], and other scholars have proposed applying crowdsourcing to deal with the drawbacks of centralized systems [4]–[6]. Also

The associate editor coordinating the review of this manuscript and approving it for publication was Chuan Heng Foh.

there are several methods presented to solve the common issues in the multimedia field [7]–[9]. According to the work mentioned above, we consider using blockchain-based Ethereum (ETH) smart contracts to achieve a novel crowdsourcing video transcoding mechanism for improving the efficiency of video transcoding that can reduce the stress of the centralized systems and also create profits for crowdsourcing volunteers.

The contributions of our work in this paper are shown as follows:

- We propose four essential components for achieving crowdsourcing video transcoding tasks, including the employer, the worker, the task allocation mechanism and the payment mechanism.
- We design a kind of smart contract combined with our proposed four components that can realize the bidding function and task execution function at the same time during the crowdsourcing video transcoding process.

We will introduce the related work in Section II. The proposed scheme and the detailed design are shown in Section III. The analysis of our scheme is illustrated in Section IV. The experimental results are shown in Section V. Finally, we will conclude our paper in Section VI.

II. RELATED WORK

We divide related work into three categories according to the domain and characteristics of the work and we will introduce these three classifications respectively below.

A. VIDEO TRANSCODING

Koziri *et al.* [10] investigated the current situation of the development of video transcoding related cloud service and summarized challenges and future opportunities. The Cloud-based Video Streaming Services (CVS2) is presented by Li *et al.* [11] for enhancing the robustness of the system while keeping the cost of the service provider from fluctuating significantly when users use cloud services for video on demand. Liu *et al.* [12] proposed a blockchain-based mobile edge computing (MEC) architecture, which offers video streaming distributedly and securely for blockchain based video transcoding architecture to satisfy the demands of the customers.

Also, Costero *et al.* [1] proposed using a multiagent machine learning based method to achieve dynamic optimization during the video transcoding process to greatly improved service quality and reduced power consumption. Jiang *et al.* [2] presented an enhanced architecture for dealing with the problems in large-scale video transcoding in the public cloud through the application of the message queue layer. He *et al.* [3] proposed a new framework called CrowdTranscoding that utilizes the surplus of personal terminal computing resources to execute video transcoding tasks. And they also put forward Viewer Crowdsourcing Transcode Scheduler (VCTS) to solve the issues of workload during the task allocation.

Furthermore, Wei *et al.* [13] proposed a cloud-based online video transcoding system (COVT) targeting at providing the quality-of-service and economical solution for large-scale video transcoding that they also performed simulation evaluation to verify their proposed scheme. Yoon *et al.* [14] proposed to run a low-cost video transcoding scheme on the edge of the wireless. Comparing with using costly server-based transcoding, they designed a transparent, low-cost, scalable video transcoding system that is realized on a hardware Raspberry PI at a low cost.

B. CROWDSOURCING

The concept of crowdsourcing was put forward earlier by Howe [15] which contributes a lot to the establishment of the distributed task assignment system for various demands. Then, Vukovic [16] took crowdsourcing in the field of software development as an example, deduced the need to provide general crowdsourcing services in the cloud environment, and made crowdsourcing further developed. Moreover, Estellés-Arolas and González-Ladrón-De-Guevara [17] modified definition of crowdsourcing, established the basic characteristics of crowdsourcing and presented a more complete and consistent definition of crowdsourcing.

Chittilappilly *et al.* [5] did an investigation, the goal of which is to provide a comprehensive picture of the current state of general crowdsourcing technology. On the basis of their concerns, the works are divided into three parts that are incentive design, task assignment and quality control respectively. In the similar way, Mao *et al.* [18] did a survey that investigated the application of crowdsourcing in software engineering and concluded the challenges, opportunities and future trends of crowdsourcing. Garcia-Molina *et al.* [19] provided an overview of data crowdsourcing and the processes involved collecting, managing, and analyzing the data, and described the essential design steps when implementing a crowdsourced scheme.

As crowdsourcing evolves, many researchers proposed novel methods to apply crowdsourcing. For example, Chatzimilioudis *et al.* [20] proposed applying smart phones to implement the crowdsourcing according to the typical characteristics of the smart phone. Chawla *et al.* [4] studied the design of crowdsourcing and analyzed the approximation of optimal crowdsourcing competitions. Additionally, they presented a scheme of the optimal crowdsourcing competition that reflects the optimal auction design theory. Franklin *et al.* [6] the proposed CrowdDB solves the problem that neither database systems nor search engines can adequately answer by crowdsourcing human input, and they also put forward an important way to develop crowdsourced query processing systems in the future.

Gu *et al.* [21] proposed a distributed online crowdsourcing mechanism based on smart contracts, the characteristics of which are decentralized, unalterable, and truthful. Kittur *et al.* [22] proposed a generic framework utilizing microtask markets for crowdsourcing complex and interdependent tasks, and described the framework and analyzed the advantages and flaws of the scheme. CrowdAsm proposed by Pan *et al.* [23] that is useful for collaborative crowdsourcing systems to maximize the quality of expected results, at the same time, minimizing expected delays by combining the expertise of available workers. Also, Tong *et al.* [24] and Wang *et al.* [25] proposed a scheme for large-scale crowdsourced task decomposition and a human-computer hybrid to improve crowdsourcing efficiency respectively.

C. BLOCKCHAIN FOR CROWDSOURCING

Blockchain has emerged in recent years, several researchers combine this new technology with crowdsourcing according to the characteristics of them.

Feng and Yan [26] proposed a blockchain-based mobile crowdsourcing chain system to enable distributed and decentralized trust management to address the inefficiencies of traditional blockchain-based crowdsourcing systems, and a new block-generated consensus mechanism that significantly reduces computational overhead. A framework based on blockchain is presented by Han *et al.* [27] for sustaining crowdsourcing platforms. The proposed framework has two main advantages, one is the transparent incentives are provided by the framework, the other is trusted personal data can be shared in a cross-platform system. Kogias *et al.* [28] proposed applying blockchain and smart contract technologies to improve the existing crowdsourcing scheme from the perspective of security. Li *et al.* [29] proposed a decentralized CrowdBC framework based on blockchain, which can preserve the privacy of customers. ZebraLancer, a private and anonymous decentralized crowdsourcing system, is designed and implemented by Lu *et al.* [30], and the issues of data leakage and identity breach in the decentralized crowdsourcing system are solved. Wang *et al.* [31] developed an optimized blockchain-based architecture aiming at peer-to-peer energy transactions, for administering and managing the operation of crowdsourced energy systems (CES). Also, some scholars proposed their own ideas towards combing the blockchain with crowdsourcing [32]–[34].

Different from the proposed schemes above, we combine the crowdsourcing, the video transcoding and the blockchain smart contract to achieve a novel crowdsourcing video transcoding mechanism for solving the shortage of computing resources during the video transcoding process in the centralized system that can also prevent from the loss of participants' interests due to the common malicious behaviors.

III. PROPOSED SCHEME - CVT

In this section, we introduce the system model and workflow of the smart contracts based crowdsourcing video transcoding mechanism, in which the main roles and rules of the proposed scheme will be explained in detail. Before formulating the issue, we will show the notations used in this paper below in Table 1.

TABLE 1. Notions.

Symbol	Definition
D_v	the bidding valid time duration of the crowdsourcing task
E_{ads}	the employer's address
M	the quotation message set for $task_e$
P_e	the earnest money for the employer
P_w	the earnest money for the successful bidder
P	the payment result
Q	the parameter standards related to video quality
T_r	the time when the crowdsoucing task is released
T_d	the agreed delivery time
W	the workers that have successfully offer their price
$c_{i,j}$	the price that the w_j offers for $task_e$
$m_{i,j}$	the quotation message of the w_j towards $task_e$
max_p	the maximum amount the employer can accept to pay
max_t	the maximum task time duration acceptable to the employer
$t_{i,j}$	the time duration for finishing $task_e$ that the w_j has promised
tag_j	the natural number for marking the order of w_j
$task_e$	the task publically released by the employer
w_j	the worker that participates the bidding
$w_{j,ads}$	the w_j's address

A. SYSTEM COMPONENTS

There are four essential components in our designed model that need to be presented and described here, including employers, workers, task allocation mechanism and payment mechanism. We will describe these components in detail as follow:

- **Employer.** In a decentralized system, employers need to issue smart contracts for distributing crowdsourcing tasks for decreasing the computing resource consumption of video transcoding.
- **Worker.** The workers tend to receive and complete the task the employers publically issue in the decentralized crowdsourcing community for gaining profits. Any w_j can choose any published task and propose its quotation with other required parameters for the task. We define the set of workers is S that includes s homogeneous workers with no correlation. Also, the structure of the quotation message of the workers from S can be formalized as the set M:

$$M = \begin{bmatrix} t_{1,j} & c_{1,j} \\ \dots & \dots \\ t_{s,j} & c_{s,j} \end{bmatrix}, \tag{1}$$

  In addition, we set the unit of M as:

$$m_{i,j} = (t_{i,j}, c_{i,j}), \quad \forall i \in S, \tag{2}$$

  where $m_{i,j}$ is the information for the intention of the w_j towards $task_e$.
- **Task Allocation Mechanism.** We define W as the set of all workers who have successfully offered their prices in $(T_r, T_r + D_v]$ for $task_e$. After the time $T_r + D_v$, quotations are no longer accepted by the smart contract. Additionally, any worker who has joined W will be marked a tag_j of a natural number according to the order. The default value of tag_j is 0, for every one more w_j that participates in the bidding, tag_j plus one. Furthermore, C is the quotation set and T is the time duration set. After the employer initiates the task allocation mechanism, the

strategy for selecting the successful bidder of $task_e$ is as follow:

1) If w_j from S has offered its price in $(T_r, T_r + D_v]$, then $\{w_j\} \cup W \rightarrow W$; Otherwise, refusing w_j for joining W.
2) Assuming the task published by the employer can be completed, then the task allocation target is formulated below:

$$target = arg_{i \in S} min(c_{i,j} + r \cdot t_{i,j}), \tag{3}$$

where r is a parameter set by the employer for adjusting the weight between c_i and t_i according to the actual demands of the employer.
3) For each w_j in W bidding for $task_e$, if $m_{i,j}$ offered by w_j can satisfy $c_{i,j} \leq max_p$ and $t_{i,j} \leq max_t$, then maintaining w_j in W; Otherwise removing w_j from W.
4) Selecting w_j from W as the successful bidder that meets the condition $w_j = arg_{i \in S} min\,(c_{i,j} + r \cdot t_{i,j})$:
   a) If there is no element in W, the smart contract terminates.
   b) If there is only one w_j in W, w_j is the successful bidder, marking the current W (the task allocation result) as W'.
   c) If there are more than one w_j from W are chosen, the final successful bidder is the worker that satisfies $w_j = argmin\, tag_j$. Then recording $m_{i,j}$ offered by the only w_j in W i.e. the successful bidder, marking the current W as W'.
5) Updating the quotation set C and the time duration set T as $c_{i,j} \cup C \rightarrow C$ and $t_{i,j} \cup T \rightarrow T$ respectively according to $m_{i,j}$ that the successful bidder offered.
6) Recording the task allocation result W', the quotation set C, and the time duration set T as the promise of the successful bidder for $task_e$ by the smart contract.

- **Payment Mechanism.** It is easy to find out the successful bidder should be paid according to the task allocation result W'. The payment mechanism is used to confirm the time and the amount that the successful bidder should be paid. We stipulate the payment result of the successful bidder for $task_e$ is P. On the basis of W', the payment mechanism is shown as follows:
  1) The agreed delivery time for $task_e$ is determined as $T_d = T_r + D_v + t_{i,a_i}$.
  2) If there is no element in W', the employer does not need to pay, i.e. $P = 0$.
  3) If w_j exists in W, the successful bidder needs to complete $task_e$ before T_d:
     a) If the successful bidder completes $task_e$ before T_d, the payment result $P = c_{i,j}$.
     b) If the successful bidder fails to finish $task_e$ before T_d, the payment result $P = -P_w$.

B. SYSTEM OVERALL ARCHITECTURE

The main thought of our designed system is to combine the above four components and smart contracts for implementing crowdsourcing video transcoding. To enhance the connections among the above four components, we add more components, and our whole scheme is shown in Figure 1.

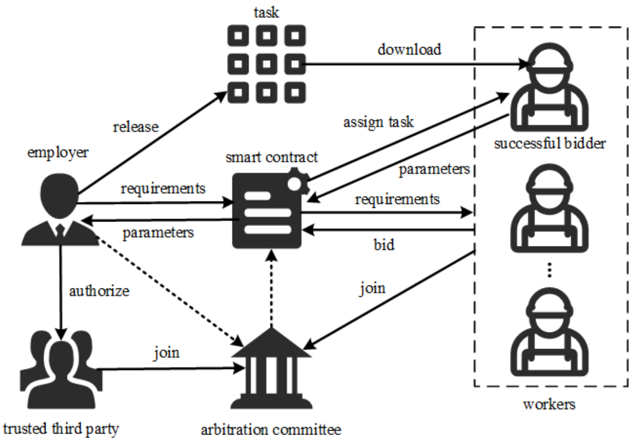


FIGURE 1. Overall architecture of our scheme.

The employer sets up the smart contract according to the specific requirements of the task that needs to be transcoded. Also, the smart contract includes the bidding part and the task execution part to ask for proper workers to complete the video transcoding task. Then the workers who are willing to accept the task can participate in the bidding. Finally, the smart contract will select a successful bidder from the bidding part according to the task allocation mechanism who will start the trading part with the employer then, at the same time the employer should release the corresponding task.

Moreover, the trusted third party is needed in the system to check the transcoded task join the arbitration committee, at the same time workers as the same number of the trusted third party will join the committee too. We will further introduce this designed part in Section III-G.

The next section focuses on the design of the smart contract in our scheme.

C. SMART CONTRACT DESIGN

The designed smart contract needs to satisfy two functions: one is bidding for the tasks marked as the bidding part, and the other is to ensure the process of the task execution between the employer and the successful bidder correct for guaranteeing the profits of both parties marked as the task execution part. The specific design of the smart contract is shown in Algorithm 1.

Besides, there are four steps for achieving crowdsourcing video transcoding on the basis of the designed smart contract, we will describe each process in the following sections in detail.

D. SMART CONTRACT INITIALIZATION

Before the smart contract starts to initiate the bidding process, the employer should upload the basic parameters of the bidding part which contain D_v, max_p and max_t.

Algorithm 1 Smart Contract for Bidding and Task Execution

```
The employer uploads $Q$ to the smart contract, $ads_A = E_{ads}$ and $m_0 = (max_t, max_p)$;
The employer releases $task_e$;
The employer sends $P_e$ to the smart contract, $T_r = now$ and $tag = tag_j = 0$;
while $time \in [T_r, T_r + D_v]$ do
    $tag_j = tag_j + 1$;
    $w_j$ sends $m_{i,j}$ to the smart contract;
    if $c_{i,j} + r \cdot t_{i,j} < c_0 + r \cdot t_0$ then
        $m_0 = m_{i,j}$;
        $ads_B = w_{j,ads}$;
        $tag = tag_j$;
    else if $c_{i,j} + r \cdot t_{i,j} = c_0 + r \cdot t_0$ & $tag_j < tag$ then
        $m_0 = m_{i,j}$;
        $ads_B = w_{j,ads}$;
        $tag = tag_j$;
    else
        continue;
    end if
end while
while $time \in (T_r + D_v, T_d]$ do
    The successful bidder is $ads_B$ that needs to send $P_w$ to the smart contract to initiate the task execution;
    $ads_B$ uploads $Q'$ to the smart contract and sends completed $task_e$ to the trusted third party;
    if $Q' = Q$ & $task_e = true$ then
        The smart contract sends $P_w$ to $ads_B$ and refunds $P_w$ to $ads_B$ and $P_e - P_w$ to the employer;
    else if $Q' \neq Q$ || $task_e = false$ then
        The employer sends a request to the smart contract;
        The smart contract starts the arbitration mechanism; break;
    else
        The smart contract terminates the task execution;
        The smart contract refunds $P_e$ to the employer;
        The smart contract sends $P_w$ to the employer; break;
    end if
end while
```

Then, the employer should upload the requirements of the transcoding task and specific parameter standards that the worker needs to meet for the transcoding task to the smart contract for the task execution part. The parameters towards different video transcoding tasks include coding scheme, video bitrate, frame rate, audio sample rate, length-width ratio and container format. We mark these parameter standards related to video quality as Q (i.e. the employer uploads the requirements of the task and Q to the smart contract).

Also, the employer should send P_e to the smart contract to launch the bidding and the task execution after the bidding. The amount of P_e is equal to max_p for protecting the rights and interests of workers.

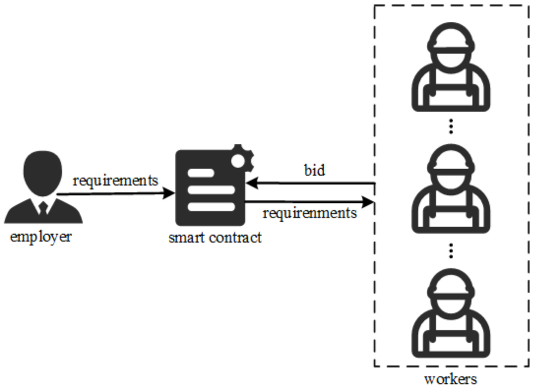


FIGURE 2. Smart contract bidding part.

E. BIDDING FOR THE TASK

After the employer initializes the smart contract and starts the bidding, the workers can see the employer's requirements, according to which the workers can choose whether to take part in the bidding. The process is shown in Figure 2.

For maximizing the interests of the participants, we follow the task allocation mechanism and payment mechanism mentioned in Section III-A, and the priority of $(c_{i,j} + r \cdot t_{i,j})$ is higher than tag_j. From the bidding part of Algorithm 1 we can see that if any worker decides to join the bidding, the worker has to send its $m_x = (t_x, c_x)$ to the smart contract, and the smart contract marks the current tag_j to the worker. The address of the worker is w_{add}.

Then the smart contract compares the $m_x = (t_x, c_x)$ and tag_j the worker offers with the current saved parameters $m_0 = (t_0, c_0)$ and tag. There are essential situations as follow during the time period $(T_r, T_r + D_v]$:

1) If $c_x + t_x < c_0 + t_0$, the smart contract substitutes current m_0 to m_x, current tag to tag_j, and current ads_B to $w_{j,ads}$.
2) If $c_x + t_x = c_0 + t_0$ and $tag_j < tag$, the smart contract reacts the same as the previous point.
3) If there are other situations, the smart contract preserves the current parameters.

Besides, if there are no workers bid for the task until the time period is passed, the smart contract will refund P_e to the employer and end the process. Otherwise, the worker corresponding to ads_B is the successful bidder who will start the next process with the employer through the smart contract.

We will detailedly describe the task execution between the employer and the successful bidder through the task execution part of the smart contract in the next section.

F. TASK EXECUTION

After finishing the bidding part and choosing the successful bidder, the task execution part of the smart contract starts. The whole process of this part is illustrated in Figure 3 and the steps of task execution part are shown as follow:

1) The successful bidder sends P_w to the smart contract and the amount of P_w is equal to c_x that the successful bidder announced in the bidding part;

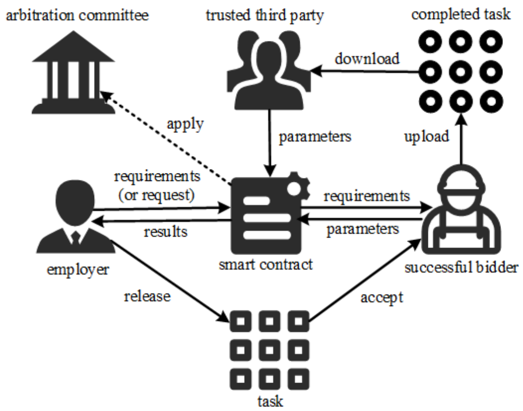


FIGURE 3. Smart contract task execution part.

2) The employer randomly chooses a hash algorithm and uploads it to the smart contract;
3) The employer releases the task and and uses the selected hash algorithm to calculate the hash value of each video block and the hash value of each video block is marked as H_b, then uploads H_b and Q to the smart contract;
4) The successful bidder downloads the transcoding task and calculates the hash value of the task as H_b' by the hash algorithm selected by the employer:
   a) If $H_b = H_b'$, the smart contract randomly sends one of the addresses of the trusted third party (marked as ads_t) for checking the transcoded video to the successful bidder and starts the transcoding task;
   b) If $H_b \neq H_b'$, the successful bidder terminates the process, and the smart contract refunds P_e to the employer and P_w to the successful bidder.
5) The successful bidder calculates the hash value of each transcoded video block marked as H_{tb} and uploads H_{tb} and Q' to the smart contract and sends the transcoded task to ads_t:
   a) If $Q = Q'$, the smart contract continues the process;
   b) If $Q \neq Q'$, the smart contract terminates the process and sends P_w as the compensation to the employer and refunds P_e to the employer.
6) The trusted third party downloads the transcoded task from ads_t and calculates the hash value of the transcoded video block marked as H_{tb}' and uploads H_{tb}' of the transcoded video blocks to the smart contract:
   a) If $H_{tb} = H_{tb}'$, the smart contract sends P_e to the successful bidder, and refunds P_w to the successful bidder and $P_e - P_w$ to the employer;
   b) If $H_{tb} \neq H_{tb}'$, the smart contract terminates the process and raises a punishment to the successful bidder that sends P_w as the compensation to the employer and refunds P_e to the employer.

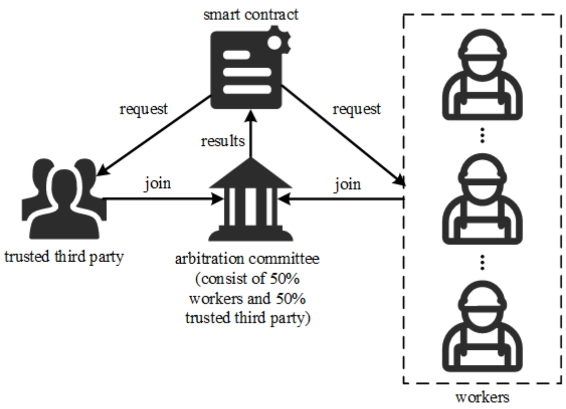


FIGURE 4. The arbitration committee system.

7) The trusted third party sends the transcoded video to the employer, and the employer checks whether the content of the transcoded video block is correct before the time T_d:
   a) If the content of the transcoded video blocks is correct, the employer terminates the task execution that means the task is completed.
   b) If the content of the transcoded video blocks is incorrect, the employer can send a request to the smart contract to launch an arbitration, and the smart contract will invite the arbitration committee to judge.
   c) If the current time exceeds T_d and the employer has not sent any request, the smart contract terminates the process that means the task is completed.

When the employer sends a request to the smart contract, the arbitration mechanism is activated to deal with the dispute between the employer and the successful bidder, and we will introduce the arbitration mechanism in the following section.

G. ARBITRATION MECHANISM

The arbitration mechanism consists of two parts: one is the arbitration committee, and the other is the rules corresponding to different situations that happen in the voting of the arbitration committee.

We introduce the composition of the arbitration committee at first. From Figure 4, we can see that the smart contract randomly requires the trusted third party and the same number of workers to join the arbitration committee. After that, each member of the arbitration committee will do as the steps below before the time T_d, for realizing the arbitration mechanism:

1) The member downloads the original task released by the employer and calculates the hash value of the video blocks as H_o:
   a) If $H_o = H_b$, the smart contract continues the process;
   b) If $H_o \neq H_b$, the smart contract searches for a new member to join the arbitration committee from the

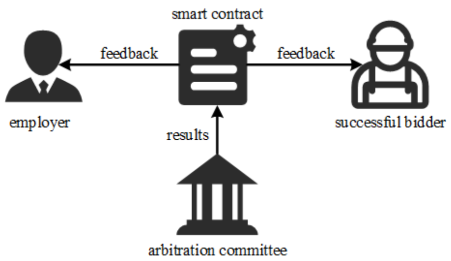


FIGURE 5. Feedback from the smart contract.

same party and repeats point 1) until the number of members of both parties is equal;

2) The member downloads the transcoded task uploaded by the successful bidder and calculates the hash value of the video blocks as H_t:
   a) If $H_t = H_{tb}$, the smart contract continues the process;
   b) If $H_t \neq H_{tb}$, the smart contract searches for a new member to join the arbitration committee from the same party and repeats points 1) and 2) until the number of members of both parties is equal;
3) After verifying the original video blocks and the transcoded video blocks, the member of the arbitration committee can compare the content of these two tasks:
   a) If the contents of these two tasks are different, the member of the committee sends 0 to the smart contract;
   b) If the contents of these two tasks are the same, the member of the committee sends 1 to the smart contract;
4) The smart contract counts the number of 0 and 1 marked as num_0 and num_1 sent by the arbitration committee member and sends the feedback (see Figure 5) to each party according to the rules as follow:
   a) If $num_0 > num_1$, the smart contract raises a punishment for the successful bidder that it sends P_w to the employer as the compensation and refunds P_e to the employer.
   b) If $num_0 < num_1$, the smart contract raises a punishment for the employer that it sends P_e to the successful bidder as the compensation and refunds P_w to the successful bidder.
   c) If $num_0 = num_1$, the smart contract refunds P_e to the employer and P_w to the successful bidder.

After finishing the above steps, the crowdsourcing video transcoding task is completed. According to the different situations that happen during the process, we give the corresponding strategy towards dealing with them. In Section IV our scheme will be analyzed in detail on the basis of each essential part during the process.

IV. SECURITY ANALYSIS

In this section, we list several threats towards our system and analyze the security and the feasibility of our scheme. We analyze the system as three parts respectively the bidding part, the task execution part and the arbitration part.

A. BIDDING ANALYSIS

During the bidding, there are two essential issues that should be considered:

1) **The dishonest quotation (including $c_{i,j}$ and $t_{i,j}$).**
   a) $c_{i,j}$ *the worker offers is too high.* The employer sets max_p before the smart contract starts for limiting the workers to offer a too high price that the employer can not afford.
   b) $c_{i,j}$ *the worker offers is too low.* If one worker offers a too low price, that means other workers will not compete with this task, and the worker will work for the employer and earns only little that almost makes no sense for the worker.
   c) $t_{i,j}$ *the worker offers is too long.* The employer sets max_t before the smart contract starts for limiting the workers to offer too long time that the employer can not accept.
   d) $t_{i,j}$ *the worker offers is too short.* If one worker offers too short time and the worker does not finish it on time during the task execution, the smart contract will send the worker's P_w to the employer as compensation. Thus, it is unnecessary for workers to complete for a task by offering a too short time that the workers can not finish the task on time.
2) **The successful bidder or the employer does not continue to enforce the process after the bidding.** If one of the parties stops to implement the process after the bidding part, when the time reaches T_d the smart contract terminates the process and refunds P_e to the employer and P_w to the successful bidder, that will not affect the interests of both parties.

B. TASK EXECUTION ANALYSIS

Also, three important issues have to be analyzed during the task execution part:

1) **The employer or the successful bidder uploads the video blocks that are not the same as the video blocks they announced to upload.** Before each party uploads the video blocks, the smart contract will ask it to provide the corresponding hash value of each video block as the evidence. If one party cheats during the task execution process, it is easy to find out by the hash value. Then the smart contract will punish the cheated party and send its earnest money to the other party.
2) **The successful bidder uploads the incorrect video with correct Q.** The employer can initiate arbitration and send a request to the smart contract if the employer finds the contents of transcoded video blocks are different from the video blocks the employer releases.

a) *The voting result of arbitration is that the employer is correct.* The smart contract sends P_w to the employer and refunds P_e to the employer.

b) *The voting result of arbitration is that the employer is incorrect.* The smart contract sends P_e to the successful bidder as compensation and refunds P_w to the successful bidder.

3) **The successful bidder accepts the task but does not finish it as the employer requires.**

a) *The successful bidder does not finish the task before T_d.* When the current time exceeds T_d, but the successful bidder does not upload related parameters or only uploads part of parameters, the smart contract sends P_w to the employer as compensation and refunds P_e to the employer.

b) *Q' is not the same as Q uploaded by the employer.* At least one parameter in Q' is not the same as the corresponding parameter in Q that the employer uploaded, the smart contract sends P_w to the employer and refunds P_e to the employer.

C. ARBITRATION ANALYSIS

Moreover, two essential issues have existed during the arbitration process as follow:

1) **The employer sends a request that is incorrect about the successful bidder to the smart contract.** There is only one situation that the employer should send a request to the smart contract that is the video blocks the successful bidder uploaded have the same Q' as Q, however, the content of these video blocks is incorrect. Thus, after the employer starts the arbitration mechanism, each member of the arbitration can obtain the original task released by the employer and the transcoded task uploaded by the successful bidder. Then, they can compare the contents of the original task and the transcoded task, and send their the results to the smart contract. If $num_0 < num_1$, that means the employer cheats, the smart contract sends P_e to the successful bidder and refunds P_w to the successful bidder.

2) **The trusted third party colludes with either party.** For the arbitration mechanism, every time the employer sends the request to the smart contract, the smart contract will randomly select the members from the trusted third party and the workers. That means, it is hard for each party to find enough members of the randomly chosen arbitration committee on the anonymous networks for collusion in order to gain profits by the incorrect result of the arbitration committee sent to the smart contract.

V. PERFORMANCE ANALYSIS

We tested our scheme on the private chain to show the utility of the scheme in real situations. We implemented the experiment as the steps presented in Section III.

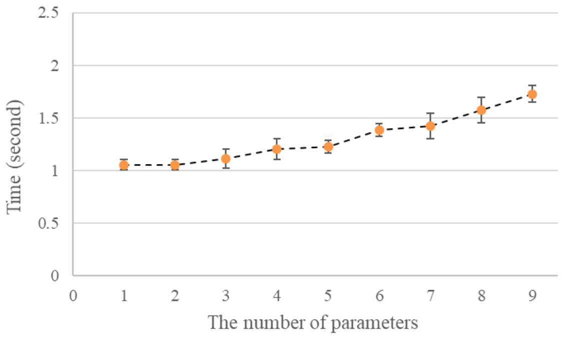


FIGURE 6. Performance of the bidding part.

Specifically, we applied *Remix*[1] to compile and test our scheme. First, we chose the compiler with the version 0.4.25+commit to compile the smart contract codes of our scheme.[2] Then, we built a private chain on the local host and create several accounts on the chain. Next, we started mining on the private chain. After the mining began, we set the environment Web3 Provider on *Remix* to connect the private chain we built. Finally, we deployed the smart contract on the private chain to test.

We selected testing the factors (steps) that have the greatest impact on the performance of our scheme during the interactive process between users and the smart contract. In order to show the results clearer, we divided the experiment into two parts - the bidding part performance and the task execution part performance. We will illustrate and analyze the experimental results below.

A. BIDDING PART PERFORMANCE

During the bidding, the most important factor is the speed of uploading parameters. The extensibility of our scheme also should be considered that means changing the number of uploaded parameters in the bidding situation. Therefore, we tested the speed of uploading the different numbers of parameters at once by each participated worker.

The experimental results are shown in Figure 6. We can see that as the number of parameters that uploaded by each worker at once raises, the time consumption increases slightly. Even if the worker is required to upload nine parameters at one time, the time consumption is within acceptable limits (within 2 seconds).

Therefore, it is sufficient to show that our designed smart contract can deal with the bidding part well that also has the extensibility of changing the number of parameters according to different requirements.

B. TASK EXECUTION PART PERFORMANCE

After the bidding part is finished, the smart contract will implement the execution part. According to the process and analysis of our scheme in Section III, it is uncomplicated to summarize that six steps have a huge impact on the

[1] http://remix.ethereum.org/

[2] https://github.com/Y-Xiang-hub/Crowdsourcing-With-Smart-Contract

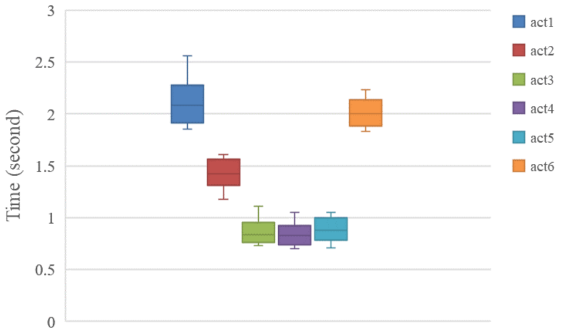


FIGURE 7. Performance of the task execution part.

performance during the task execution part. The specific six steps are listed as follows:

1) Deploy the smart contract.
2) Send the earnest money to the smart contract.
3) Upload the parameters to the smart contract by the successful bidder.
4) Upload the parameters to the smart contract by the trusted third party.
5) Send the request to the smart contract.
6) Select the members of the committee.

We respectively mark the above steps in order as act_i ($1 \leq i \leq 6$). The experimental results are shown in Figure 7. The results demonstrate that deploying the smart contract and selecting the members of the committee take the most time (generally around 2 seconds). Additionally, it can be seen that the time consumption of sending tokens to the smart contract (around 1.5 seconds) is higher than the other three uploading parameter operations (generally within 1 second).

Thus, the experimental results illustrate that the performance of the task execution part of the scheme can fit the real-world application demands.

VI. CONCLUSION

In this paper, we propose a blockchain smart contract based scheme combined with the four essential components, which is designed to achieve crowdsourcing video transcoding for solving the lack of computing resources and enhances the efficiency of video transcoding.

According to the security analysis and performance analysis, our proposed scheme (i.e., CVT) is justified to be feasible and sufficient to deal with the common security issues in crowdsourcing video transcoding.

Also, the scheme has good portability, which means our scheme can be applied to other similar multimedia crowdsourcing scenarios by modifying a few settings and parameters.

REFERENCES

[1] L. Costero, A. Iranfar, M. Zapater, F. D. Igual, K. Olcoz, and D. Atienza, ''MAMUT: Multi-agent reinforcement learning for efficient real-time multi-user video transcoding,'' in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2019, pp. 558–563.

[2] Q. Jiang, Y. C. Lee, and A. Y. Zomaya, ''Scalable video transcoding in public clouds,'' in *Proc. 19th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput. (CCGRID)*, May 2019, pp. 70–75.

[3] Q. He, C. Zhang, and J. Liu, ''CrowdTranscoding: Online video transcoding with massive viewers,'' *IEEE Trans. Multimedia*, vol. 19, no. 6, pp. 1365–1375, Jun. 2017.

[4] S. Chawla, J. D. Hartline, and B. Sivan, ''Optimal crowdsourcing contests,'' *Games Econ. Behav.*, vol. 113, pp. 80–96, Jan. 2019.

[5] A. I. Chittilappilly, L. Chen, and S. Amer-Yahia, ''A survey of general-purpose crowdsourcing techniques,'' *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 9, pp. 2246–2266, Sep. 2016.

[6] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin, ''CrowdDB: Answering queries with crowdsourcing,'' in *Proc. Int. Conf. Manage. Data (SIGMOD)*, 2011, pp. 61–72.

[7] L. Shu, Y. Zhang, Z. Yu, L. T. Yang, M. Hauswirth, and N. Xiong, ''Context-aware cross-layer optimized video streaming in wireless multimedia sensor networks,'' *J. Supercomput.*, vol. 54, no. 1, pp. 94–121, Oct. 2010.

[8] W. Guo, N. Xiong, A. V. Vasilakos, G. Chen, and C. Yu, ''Distributed k–connected fault–tolerant topology control algorithms with PSO in future autonomic sensor systems,'' *Int. J. Sensor Netw.*, vol. 12, no. 1, pp. 53–62, 2012.

[9] Z. Wan, N. Xiong, N. Ghani, A. V. Vasilakos, and L. Zhou, ''Adaptive unequal protection for wireless video transmission over ieee 802.11 e networks,'' *Multimedia Tools Appl.*, vol. 72, no. 1, pp. 541–571, 2014.

[10] M. G. Koziri, P. K. Papadopoulos, N. Tziritas, T. Loukopoulos, S. U. Khan, and A. Y. Zomaya, ''Efficient cloud provisioning for video transcoding: Review, open challenges and future opportunities,'' *IEEE Internet Comput.*, vol. 22, no. 5, pp. 46–55, Sep. 2018.

[11] X. Li, M. A. Salehi, M. Bayoumi, N.-F. Tzeng, and R. Buyya, ''Cost-efficient and robust on-demand video transcoding using heterogeneous cloud services,'' *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 3, pp. 556–571, Mar. 2018.

[12] Y. Liu, F. R. Yu, X. Li, H. Ji, and V. C. M. Leung, ''Decentralized resource allocation for video transcoding and delivery in blockchain-based system with mobile edge computing,'' *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11169–11185, Nov. 2019.

[13] L. Wei, J. Cai, C. H. Foh, and B. He, ''QoS-aware resource allocation for video transcoding in clouds,'' *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 1, pp. 49–61, Jan. 2017.

[14] J. Yoon, P. Liu, and S. Banerjee, ''Low-cost video transcoding at the wireless edge,'' in *Proc. IEEE/ACM Symp. Edge Comput. (SEC)*, Oct. 2016, pp. 129–141.

[15] J. Howe, ''The rise of crowdsourcing,'' *Wired Mag.*, vol. 14, no. 6, pp. 1–4, Jun. 2006.

[16] M. Vukovic, ''Crowdsourcing for enterprises,'' in *Proc. Congr. Services I*, Jul. 2009, pp. 686–692.

[17] E. Estellés-Arolas and F. González-Ladrón-de-Guevara, ''Towards an integrated crowdsourcing definition,'' *J. Inf. Sci.*, vol. 38, no. 2, pp. 189–200, Apr. 2012.

[18] K. Mao, L. Capra, M. Harman, and Y. Jia, ''A survey of the use of crowdsourcing in software engineering,'' *J. Syst. Softw.*, vol. 126, pp. 57–84, Apr. 2017.

[19] H. Garcia-Molina, M. Joglekar, A. Marcus, A. Parameswaran, and V. Verroios, ''Challenges in data crowdsourcing,'' *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 4, pp. 901–911, Apr. 2016.

[20] G. Chatzimilioudis, A. Konstantinidis, C. Laoudias, and D. Zeinalipour-Yazti, ''Crowdsourcing with smartphones,'' *IEEE Internet Comput.*, vol. 16, no. 5, pp. 36–44, Sep. 2012.

[21] Y. Gu, J. Chen, and X. Wu, ''An implement of smart contract based decentralized online crowdsourcing mechanism,'' in *Proc. 2nd Int. Conf. Comput. Sci. Artif. Intell. (CSAI)*, 2018, pp. 195–199.

[22] A. Kittur, B. Smus, and R. Kraut, ''CrowdForge: Crowdsourcing complex work,'' in *Proc. Annu. Conf. Extended Abstr. Hum. Fasctors Comput. Syst. (CHI EA)*, 2011, pp. 43–52.

[23] Z. Pan, H. Yu, C. Miao, and C. Leung, ''Efficient collaborative crowdsourcing,'' in *Proc. 13th AAAI Conf. Artif. Intell.*, Feb. 2016, pp. 4248–4249.

[24] Y. Tong, L. Chen, Z. Zhou, H. V. Jagadish, L. Shou, and W. Lv, ''SLADE: A smart large-scale task decomposer in crowdsourcing,'' *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 8, pp. 1588–1601, Aug. 2018.

[25] J. Wang, T. Kraska, M. J. Franklin, and J. Feng, ''CrowdER: Crowdsourcing entity resolution,'' 2012, *arXiv:1208.1927*. [Online]. Available: http://arxiv.org/abs/1208.1927

[26] W. Feng and Z. Yan, ''MCS-chain: Decentralized and trustworthy mobile crowdsourcing based on blockchain,'' *Future Gener. Comput. Syst.*, vol. 95, pp. 649–666, Jun. 2019.

[27] S. Han, Z. Xu, Y. Zeng, and L. Chen, "Fluid: A blockchain based framework for crowdsourcing," in *Proc. Int. Conf. Manage. Data (SIGMOD)*, 2019, pp. 1921–1924.

[28] D. G. Kogias, H. C. Leligou, M. Xevgenis, M. Polychronaki, E. Katsadouros, G. Loukas, R. Heartfield, and C. Z. Patrikakis, "Toward a blockchain-enabled crowdsourcing platform," *IT Prof.*, vol. 21, no. 5, pp. 18–25, Sep. 2019.

[29] M. Li, J. Weng, A. Yang, W. Lu, Y. Zhang, L. Hou, J.-N. Liu, Y. Xiang, and R. H. Deng, "CrowdBC: A blockchain-based decentralized framework for crowdsourcing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 6, pp. 1251–1266, Jun. 2019.

[30] Y. Lu, Q. Tang, and G. Wang, "ZebraLancer: Private and anonymous crowdsourcing system atop open blockchain," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2018, pp. 853–865.

[31] S. Wang, A. F. Taha, J. Wang, K. Kvaternik, and A. Hahn, "Energy crowdsourcing and peer-to-peer energy trading in blockchain-enabled smart grids," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 8, pp. 1612–1623, Aug. 2019.

[32] X. Xu, Q. Liu, X. Zhang, J. Zhang, L. Qi, and W. Dou, "A blockchain-powered crowdsourcing method with privacy preservation in mobile environment," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 6, pp. 1407–1419, Dec. 2019.

[33] W. Zhang, Z. Hong, and W. Chen, "Hierarchical pricing mechanism with financial stability for decentralized crowdsourcing: A smart contract approach," *IEEE Internet Things J.*, early access, Jul. 7, 2020, doi: 10.1109/JIOT.2020.3007268.

[34] S. Zhu, Z. Cai, H. Hu, Y. Li, and W. Li, "ZkCrowd: A hybrid blockchain-based crowdsourcing platform," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4196–4205, Jun. 2020.

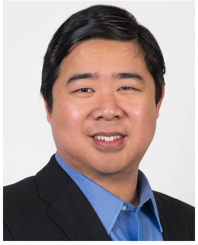
YULING CHEN received the B.S. degree from Taishan University, Taian, China, in 2006, and the M.S. degree from Guizhou University, Guiyang, China, in 2009. She is currently an Associate Professor with the Guizhou Provincial Key Laboratory of Public Big Data, Guizhou University. Her recent research interests include cryptography and information security.

HONGYAN YIN received the master's degree from the School of Economics and Management, UCAS. He is currently a Senior Engineer with the Technical Center of Beijing China Customs. His research interests include EMC and safety of electronic products.

YUEXIN XIANG is currently pursuing the degree with the School of Computer Science, China University of Geosciences, Wuhan, China. His research interests include blockchain, smart contract, and AI security.

WEI REN (Member, IEEE) received the B.S. and M.S. degrees from the University of Science and Technology Beijing, China, in 1999 and 1996, respectively, and the Ph.D. degree in computer science from the Huazhong University of Science and Technology, Wuhan, China, in 2006. He was with the Department of Computer Science, Hong Kong University of Science and Technology, Hong Kong, in 2004 and 2005. He was with the School of Computer Science, University of Nevada Las Vegas, Las Vegas, NV, USA, in 2006 and 2007. He was with the Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, IL, USA, in 2007 and 2008. From 2009 to 2013, he was an Associate Professor with the School of Computer Science, China University of Geosciences, Wuhan. Since 2014, he has been a Full Professor with the School of Computer Science, China University of Geosciences. He has published over 100 refereed articles, two monographs, and four textbooks. He is a Distinguished Member of the China Computer Federation. He was a recipient of the 20 patents and six innovation awards.

YI REN received the Ph.D. degree in information communication and technology from the University of Agder, Norway, in 2012. He was with the Department of Computer Science, National Chiao Tung University (NCTU), Hsinchu, Taiwan, as a Postdoctoral Fellow, and an Assistant Research Fellow, from 2012 to 2017. He is currently a Lecturer with the School of Computer Science, University of East Anglia (UEA), Norwich, U.K. His current research interests include security and performance analysis in wireless sensor networks, ad hoc, and mesh networks, LTE, and e-health security. He received the Best Paper Award in IEEE MDM.



Neal Naixue Xiong (Senior Member, IEEE) received the Ph.D. degree in sensor system engineering from Wuhan University, in 2007, and the Ph.D. degree in dependable communication networks from the Japan Advanced Institute of Science and Technology, in 2008. He worked with Georgia State University, Wentworth Technology Institution, and Colorado Technical University (Full Professor about five years) about ten years. He is currently an Associate Professor with the Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, OK, USA. He has published over 200 international journal articles and over 100 international conference papers. Some of his works were published in the IEEE Journals on Selected Areas in Communications, the IEEE or ACM Transactions, ACM Sigcomm workshop, IEEE INFOCOM, ICDCS, and IPDPS. His research interests include cloud computing, security and dependability, parallel and distributed computing, networks, and optimization theory. Since 2012, he has been a Senior Member of the IEEE Computer Society. He has received the Best Paper Award in the 10th IEEE International Conference on High Performance Computing and Communications (HPCC-08) and the Best student Paper Award in the 28th North American Fuzzy Information Processing Society Annual Conference (NAFIPS2009). He has been a General Chair, a Program Chair, a Publicity Chair, a Program Committee Member, and an Organizing Committee Member of over 100 international conferences, and as a Reviewer of about 100 international journals, including the IEEE Journals on Selected Areas in Communications, the IEEE Transactions on Systems, Man, and Cybernetics: Systems (Park: A/B/C), the IEEE Transactions on Communications, the IEEE Transactions on Mobile Computing, and the IEEE Transactions on Parallel and Distributed Systems. He is serving as an Editor-in-Chief, an Associate Editor or an Editor Member for over ten international journals, including an Associate Editor for IEEE Transactions on Systems, Man, and Cybernetics: Systems and *Information Science*, the Editor-in-Chief for the *Journal of Internet Technology* (JIT) and the *Journal of Parallel & Cloud Computing* (PCC), and a Guest Editor for over ten international journals, including *Sensor Journal*, *WINET*, and *MONET*.