

# An Energy-efficient Computing Offloading Framework for Blockchain-enabled Video Streaming Systems

Shijing Yuan\*, Jie Li\*, Yuxuan Zhu<sup>†</sup>, Chentao Wu\*, Yue Ding<sup>§</sup>,

Department of Computer Science and Engineering, Shanghai Jiao Tong University, China

<sup>†</sup>University of Michigan-Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University, China

<sup>§</sup>School of Software, Shanghai Jiao Tong University, China

Email: \*{2019ysj,zyx-max-sjtu,lijiecs}@sjtu.edu.cn,wuct@cs.sjtu.edu.cn, <sup>§</sup>dingyue@sjtu.edu.cn,

**Abstract**—Blockchain and edge computing have been widely applied in video streaming systems. However, previous works lack a joint consideration of video redundancy and full utilization of edge resources (bandwidth resources, CPU frequency), resulting in sub-optimal performance of video streaming systems. In this paper, we propose a computing offloading framework for blockchain-enabled video streaming systems to fully exploit edge resources and reduce energy consumption. Specifically, we formulate computing offloading, resource allocation, and adaptive compression as a joint optimization problem. We transform and decompose the original non-convex problem and propose an algorithm based on the alternating direction method of multipliers (ADMM) to solve the decomposed problem in a distributed manner. Simulation results demonstrate that our scheme can effectively reduce energy consumption and fully utilize the bandwidth and computational resources.

**Index Terms**—Edge computing, blockchain, computing offloading, resource allocation, adaptive compression

## I. INTRODUCTION

As the demand for video streaming services grows significantly, video streaming technologies (e.g., video transcoding, real-time target detection, etc.) are also evolving rapidly. Currently, the traditional methods of video stream processing in the multimedia internet of things mainly rely on local computing and cloud computing, both of which result in significant latency. Therefore, multi-access edge computing is widely used in video streaming processing to provide low-latency video streaming services [1]. Meanwhile, blockchain, which has been widely used to ensure the immutability of processing results, is integrated as a promising technology into the edge video streaming system [2]. Video processing records cannot be tampered with once they are stored on the blockchain ledger.

Edge computing and blockchain-integrated video streaming systems still faces the problem that video tasks are always redundant [3]. Implementing high video

streaming services (e.g., high video analysis accuracy) does not always require retaining all of the original video information [3]. Video tasks that are offloaded to mobile edge servers (MEC) without reasonable compression waste strained edge resources (e.g., bandwidth resources, CPU frequencies), and thereby the first question is: *How to compress video tasks adaptively to reduce their redundancy.* Besides, resources at the edge are often limited, and offloading decisions and resource allocation schemes can greatly affect the performance of the video streaming system [4]. The second question is: *How to reasonably design the offloading scheme and resource allocation to fully exploit the computational and bandwidth resources of the edge devices and MEC servers?*

Despite the fact that recent efforts [3] [5]–[9] have been made to improve the performance of edge video streaming systems, the problems mentioned above cannot be tackled in a satisfying way. To be specific, researchers develop resource allocation and computing offloading schemes for edge video transcoding [5]–[7] or focus on resource allocation in edge blockchain systems [8] [9]. But none of these works consider the joint design of video compression, computing offloading, and resource allocation, yielding a sub-optimal result. To bridge this gap, in this paper we propose a joint optimization framework for edge video streaming systems that combines the above factors to guarantee low energy consumption and high resource utilization.

However, to achieve the above goal, the following challenges are confronted. 1) The first challenge is the lack of credible incentive allocation. Due to the heterogeneous nature of edge devices, they may be reluctant to participate in computationally intensive video processing due to mistrust and selfishness [5]. Therefore, it is a huge challenge to provide credible incentive allocation for edge video streaming systems to recruit more edge devices to participate in computing. 2) The second challenge is the complexity of the optimization problem. Joint computing offloading, resource allocation and adaptive compression for video tasks involve multiple optimization variables (e.g., offloading decision, compression ratio, CPU frequency, etc.), leading to a non-convex optimization problem.

To address these challenges, we propose an energy-efficient computing offloading framework for blockchain

We thank anonymous reviewers for their insightful comments. This work was supported by National Key R&D Program of China (No. 2020YFB1806700), and Research Collaboration Grant from NII, Japan. Jie Li is the corresponding author.

edge streaming systems. Notably, video tasks are adaptively compressed to reduce energy consumption before being offloaded. Furthermore, the compressed video tasks can be offloaded to a nearby MEC server or edge groups to fully exploit computational and bandwidth resources. The main contributions in this paper is summarized as follows.

- **Novel Offloading Framework.** To the best of our knowledge, this is the first work to design a joint computing offloading, resource allocation, and adaptive compression framework for blockchain-enabled video streaming systems. Meanwhile, we introduce an incentive mechanism to encourage edge nodes to participate in computation.
- **Efficient algorithm.** We formulate computing offloading, resource allocation and adaptive compression as an optimization problem and propose an alternating direction method of multipliers (ADMM) based algorithm to solve it in a distributed way. Besides, we analyze the complexity and the convergence of the proposed algorithm.
- **Simulation experiments.** By conducting simulation experiments, we evaluate the performance of the proposed scheme. Compared with the baseline scheme, the experimental results demonstrate that the proposed scheme can effectively reduce energy consumption and resource overhead.

The rest of this paper is organized as follows. The related works are listed in Section II. The system model is described in Section III. In Section IV, the problem formulation, the transformation and decomposition of the problem are given. In Section V, the experiments and simulation results are presented. Finally, the conclusions are given in VI.

## II. RELATED WORK

Multi-access edge computing (MEC) is regarded as a promising solution of video streaming and has been extensively studied. Researchers [6], [10], [11] maximize system revenue and reduce latency by jointly and cooperatively computing task offloading, resource allocation, and content caching. Some works [12], [13] dedicate to incorporating the advantage of MEC into different scenarios such as virtual reality (VR) and mobile video streaming. A telepathology system called LiveMicro is established in [14] based on edge computing, achieving the goals of high-throughput and low latency for remote diagnosis. Aiming at enhancing the scalability of drones, Wang *et al.* [15] take advantage of deep neural networks (DNNs) to economize considerable wireless bandwidth. Ming *et al.* [16] develop a graph-assisted reinforcement learning algorithm to improve the performance of an edge-based video surveillance system. Also, there exist some researches focusing on the combination of MEC and blockchain. Liu *et al.* devise the MEC-enabled framework in a blockchain-based system with a incentive mechanism to foster collaboration [5]. Feng *et al.* [7] propose a design that integrates the

optimization of MEC with blockchain to achieve a trade-off between energy consumption of MEC system and the speed of block generation. While the works [5], [6] and [7] apply the ADMM-based algorithm to lower the complexity after transforming the formulated problem into a convex one. Jiang *et al.* [9] model its joint optimization problem as a Markov decision process (MDP) and deploys the asynchronous advantage actor-critic (A3C) algorithm to achieve a high performance. Although the above works provide an in-depth study of MEC applications or MEC-enabled blockchain systems, joint optimization together with further analysis upon each factor to promote the performance of video streaming systems is ignored.

## III. SYSTEM MODEL

In this section, we first introduce the system framework, and then describe the computing offloading and compression model respectively, followed by the resource allocation model.

### A. System Framework

The system framework is shown in Fig. 1. We consider a video streaming system enabled by blockchain, which is composed of two parts: a blockchain sub-system and a MEC sub-system, including a macro base station (MBS),  $m$  MEC-enabled small base stations (SBSs),  $v_m$  edge devices under the  $m$ -th SBS, and  $g_{k_{v_m}}$  edge groups under the  $v_m$ -th video task, where  $m = \{1, 2, \dots, |\mathcal{M}|\}$ ,  $v_m = \{1, 2, \dots, |\mathcal{V}_m|\}$  and  $g_{k_{v_m}} = \{1, 2, \dots, |\mathcal{G}_{k_{v_m}}|\}$ . In the MEC sub-system, the video stream task captured by the edge device is compressed into  $|\mathcal{G}_{k_{v_m}}|$  sub tasks according to the adaptive video compression ratio, and then offloaded to edge groups (Mode 0) or a nearby MEC server (Mode 1) through the wireless channel. During which, the offloading records and processing results of video tasks are recorded in the distributed ledger of blockchain in the form of transaction. In the blockchain system, all video processing participants work together as miners to maintain the blockchain system by applying a low-difficulty proof-of-work (PoW) consensus [17].

### B. Computing Offloading and Compression Model

We define  $VT_{v_m} = \langle v_m, L_{v_m}, C_{k_{v_m}}, \alpha_{k_{v_m}}^{(0)}, \beta_{v_m}^{(1)} \rangle$  as the video processing task, where  $v_m$  represents the version of the video task,  $C_{k_{v_m}}$  donates the size of the sub chunk after being equally divided by  $k_{v_m}$ ,  $k_{v_m} \in \mathcal{K}_{v_m}$ , and  $\alpha_{k_{v_m}}^{(0)}$ ,  $\beta_{v_m}^{(1)}$  are the compression ratio in Mode 0 and the Mode 1, respectively. Video processors  $P_{v_m}$ , MEC servers and edge groups  $g_{k_{v_m}}$  work together to complete the offloading process of video task  $VT_{v_m}$ , during which each video stream processing task  $VT_{v_m}$  is compressed by a corresponding  $P_{v_m}$ . Also, the noise and redundant information in video tasks  $\{VT_{v_m}, v_m \in \mathcal{V}_m\}$  are inevitable, thus rational compression of video tasks is necessary. To reduce the delay and energy consumption of video processing without damaging the quality of it,  $P_{v_m}$  compresses video

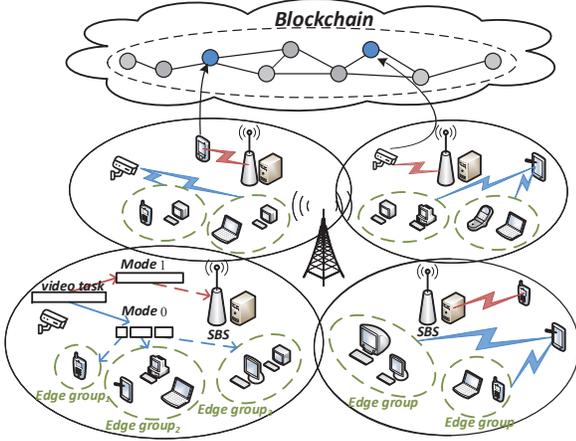


Fig. 1. The proposed computing offloading framework considering adaptive compression, hybrid offloading and resource allocation.

task  $VT_{v_m}$  into sub-tasks, and then chose one of the two offloading modes, e.g, offloading to the MEC server (Mode 1), or, offloading to the edge group (Mode 0). In Mode 1,  $P_{v_m}$  compresses video task  $VT_{v_m}$  into a whole chunk by  $\beta_{v_m}^{(1)}$ , and then offloads it to the MEC server for further processing. In Mode 0,  $P_{v_m}$  splits the video task  $VT_{v_m}$  into  $K$  equal sub-tasks  $\{C_{k_{v_m}}, k_{v_m} \in \mathcal{K}_{v_m}\}$ , and then compresses these equal sub-tasks into  $\{C'_{k_{v_m}}, k_{v_m} \in \mathcal{K}_{v_m}\}$  by  $\{\alpha_{k_{v_m}}^{(0)}, k_{v_m} \in \mathcal{K}_{v_m}\}$ , after which they will be offloaded to edge groups  $\{g_{k_{v_m}}, k_{v_m} \in \mathcal{K}_{v_m}\}$ .

### C. Resource Allocation Model

We consider a two-layer heterogeneous cellular network model supporting D2D communication mode. Since orthogonal frequency division multiplexing access (OFDMA) is within easy access, the edge node can reuse the  $SBS_m$  frequency band. The spectrum allocation in D2D and cellular networks are represented by  $b_{k_{v_m}}^{(0)}$  and  $b_{v_m}^{(1)}$ , respectively. The computational resources allocated to  $VT_{v_m}$  are denoted as  $c_{v_m}^{(v)}$  and those allocated to the miners of the blockchain system for consensus is represented as  $c_{v_m}^{(b)}$ .

## IV. PROBLEM FORMULATION

In this section, we first give the revenue of the blockchain sub-system. Then, we analyze the energy consumption and revenue of the MEC sub-system.

### A. The Blockchain Sub-system

The video processor  $P_{v_m}$  also acts as the miner maintaining the blockchain. We use  $\eta_{v_m}$  to denote the probability that a mined block is successfully recorded on the blockchain, which can be expressed as [8]  $\eta_{v_m} = \frac{c_{v_m}^{(b)}}{\sum_{v_m \in \mathcal{V}_m} c_{v_m}^{(b)}} e^{-\mu s_{v_m}}$ , where  $c_{v_m}^{(b)}$  is the computational resources allocated for mining,  $s_{v_m}$  is the block size of the mined block, and  $\mu$  is the average orphaning-rate.

Hence, the block generation reward [8] can be calculated as  $(\dot{R} + s_{v_m})\eta_{v_m}$ , in which  $\dot{R}$  is the fixed reward for mining. The energy consumption  $E_{v_m}^{mine}$  of mining can be given as [18]  $E_{v_m}^{mine} = \kappa c_{v_m}^{(b)}$ , where  $\kappa$  is the energy coefficient that depends on the chip architecture. Hence, the revenue  $\Phi_{v_m}$  of the blockchain sub-system can be expressed as

$$\Phi_{v_m} = (\dot{R} + s_{v_m})\eta_{v_m} - \kappa c_{v_m}^{(b)}. \quad (1)$$

### B. The MEC sub-system

1) *Mode 0*: We analyze the energy consumption and the processing reward of Mode 0. Total latency  $D^{(0)}$  for completing the video task  $VT_{v_m}$  consists of the offloading time  $D_{v_m}^{(0,off)}$  and the processing time  $D_{v_m}^{(0,ps)}$ .

(a) *Offloading time*: The time to transfer  $VT_{v_m}$  is equal to the sum of the transfer delays of all sub-tasks

$$D_{v_m}^{(0,off)} = \sum_{k_{v_m} \in \mathcal{K}_{v_m}} \frac{C_{k_{v_m}} \alpha_{k_{v_m}}}{b_{k_{v_m}}^{(0)} B^{(0)} e_{k_{v_m}}^{(0)}}, \quad (2)$$

where  $b_{k_{v_m}}^{(0)}$  represents the proportion of spectrum allocated to group  $g_{k_{v_m}}$ ,  $B^{(0)}$  donates the bandwidth of D2D communication mode, and  $e_{k_{v_m}}^{(0)}$  is the spectrum efficiency from video processor  $P_{v_m}$  to edge group  $g_{k_{v_m}}$ .

(b) *Processing time*: In Mode 0, each compressed chunk is processed by edge group  $\{g_{k_{v_m}}, k_{v_m} \in \mathcal{K}_{v_m}\}$  in parallel. To simplify calculation, the processing time  $D_{v_m}^{(0,ps)}$  of video task  $VT_{v_m}$  can be expressed as the average processing time of each sub chunk:

$$D_{v_m}^{(0,ps)} = \frac{1}{K_{v_m}} \sum_{k_{v_m} \in \mathcal{K}_{v_m}} \frac{\epsilon_1 C_{k_{v_m}} \alpha_{k_{v_m}}}{f_{k_{v_m}}}, \quad (3)$$

Where,  $\epsilon_1$  is the CPU cycle required to process video tasks of unit size, and  $f_{k_{v_m}}$  is the computational capacity of edge group  $g_{k_{v_m}}$ .

(c) *Energy consumption*: In Mode 0, the total energy consumption  $E^{(0)}$  includes the energy consumption of video task offloading and computing:

$$E_{v_m}^{(0)} = P^{(0,off)} D_{v_m}^{(0,off)} + P^{(0,ps)} D_{v_m}^{(0,ps)}, \quad (4)$$

where  $P^{(0,off)}$  and  $P^{(0,ps)}$  are the power consumption of edge groups in idle state and active state respectively.

2) *Mode 1*: In Mode 1, the video task is compressed into a large video block by video processor  $P_{v_m}$  at the compression ratio  $\beta_{v_m}$ , and then offloaded to MEC servers.

(a) *Transmission delay*: The time consumed in offloading  $VT_{v_m}$  from video processor  $P_{v_m}$  to  $SBS_m$  is calculated by

$$D_{v_m}^{(1,off)} = \frac{\epsilon_2 L_{v_m} \beta_{v_m}}{b_{v_m}^{(1)} B^{(1)} e_{v_m}^{(1)}}, \quad (5)$$

where  $b_{v_m}^{(1)}$  represents the spectrum proportion allocated to  $P_{v_m}$ ,  $B^{(1)}$  is the transmission bandwidth under Mode 1, and  $e_{v_m}^{(1)}$  donates the spectrum efficiency from video processor  $P_{v_m}$  to MEC servers.

(b) *Processing delay*: The time consumed by MEC in processing large video block can be calculated by

$$D_{v_m}^{(1,ps)} = \frac{\epsilon_3 L_{v_m} \beta_{v_m}}{c_{v_m}^{(v)}}, \quad (6)$$

where  $\epsilon_3$  is the CPU cycle required to calculate the video block of unit size and  $c_{v_m}^{(v)}$  is the computational resources allocated to  $SBS_m$ .

(c) *Energy consumption*: Once Mode 1 is selected by the video processor  $P_{v_m}$ , the video task  $VT_{v_m}$  will be offloaded to  $SBS_m$ , the total energy consumption required to complete the video processing can be calculated by:

$$E_{v_m}^{(1)} = P^{(1,off)} D_{v_m}^{(1,off)} + P^{(1,ps)} D_{v_m}^{(1,ps)}, \quad (7)$$

where  $P^{(1,off)}$  and  $P^{(1,ps)}$  is the power of MEC servers in idle and running state, respectively.

3) *Incentive mechanism*: To encourage edge nodes to become  $P_{v_m}$  and participate in the collaborative processing, we introduce a reward mechanism, which can be implemented with smart contract. According to [19] [20], the reward of video processing is related to the size of the task and the processing accuracy, which can be quantified as  $R_{v_m} = L_{v_m} \phi_{v_m}$ , where  $\phi$  is the accuracy function

$$\phi_{v_m} = (1 - a_{v_m}) \sum_{k_{v_m} \in \mathcal{K}_{v_m}} \frac{\theta_{v_m}}{K_{v_m}} \log \left( \frac{\alpha_{k_{v_m}}}{d_{k_{v_m}}} \right) + a_{v_m} \theta_{v_m} \log \left( \frac{\beta_{v_m}}{d_{v_m}} \right), \quad (8)$$

where  $\{\theta_{v_m}, d_{v_m}\}$  are the constant coefficient associated with the video task and the convolutional neural network (CNN) used for video detection, which can be obtained by least squares calibration.  $P_{v_m}$  needs to pay for the energy consumption of video processing, and pays the corresponding processing remuneration to the edge groups. Hence, the revenue of  $P_{v_m}$  can be given as:

$$\Psi_{v_m} = R_{v_m} - \varpi \left[ (1 - a_{v_m}) E_{v_m}^{(0)} + a_{v_m} E_{v_m}^{(1)} \right], \quad (9)$$

where  $\varpi$  is the unit price of energy consumption.

### C. Problem formulation and Decomposition

1) *Problem formulation*: We set the goal as maximizing the revenue of all video processing nodes. The optimization variables include the offloading decision  $a$ , the computational resource allocation vector  $\{c^{(b)}, c^{(v)}\}$ , the computing offloading mode  $a$ , the spectrum allocation vector  $\{b^{(0)}, b^{(1)}\}$ , and the compression ratio vector  $\{\alpha, \beta\}$ . The optimization problem can be expressed as follows:

$$\begin{aligned} P1: & \max_{\substack{\{a, c^{(v)}, c^{(b)}, \\ b^{(0)}, b^{(1)}, \alpha, \beta\}}} \sum_{m \in \mathcal{M}} \sum_{v_m \in \mathcal{V}_m} \lambda \Phi_{v_m} + (1 - \lambda) \Psi_{v_m} \\ C1: & a_{v_m} \in \{0, 1\}, v_m \in \mathcal{V}_m, m \in \mathcal{M}, \\ C2: & \alpha_{k_{v_m}}, \beta_{v_m} \in [0, 1], v_m \in \mathcal{V}_m, m \in \mathcal{M}, \\ C3: & \sum_{v_m \in \mathcal{V}_m} \sum_{k_{v_m} \in \mathcal{K}_{v_m}} b_{k_{v_m}}^{(0)} \leq 1, \sum_{v_m \in \mathcal{V}_m} b_{v_m}^{(1)} \leq 1, \\ C4: & \sum_{m \in \mathcal{M}} \sum_{v_m \in \mathcal{V}_m} c_{v_m}^{(v)} + c_{v_m}^{(b)} \leq \dot{C}, \\ C5: & (1 - a_{v_m}) D_{v_m}^{(0)} + a_{v_m} D_{v_m}^{(1)} \leq \tau DDL, v_m \in \mathcal{V}_m, \end{aligned} \quad (10)$$

where  $C1$  ensures the effectiveness of the offloading decision,  $C2$  ensures that the compression ratio does not surpass the limited range,  $C3$  guarantees that the sum of the allocated spectrum ratios can not exceed the total spectrum resources,  $C4$  indicates that the allocated computational resources does not exceed the total computing resource, and  $C5$  make sure that the total processing latency is within the deadline.

2) *Transformation of P1*: Observing  $P1$ , we find that it is a discrete non-convex problem. The existence of high-order terms and coupled variables induce a high complexity of the solution. Therefore we transform  $P1$  by the following steps.

(a) *Relax discrete variables*: We use the relaxation method of discrete variables to make discrete variables continuous:  $0 \leq \tilde{a}_{v_m} \leq 1$ ,  $0 \leq \tilde{c}_{v_m}^{(b)} \leq \dot{C}$ ,  $0 \leq \tilde{c}_{v_m}^{(v)} \leq \dot{C}$ ,  $0 \leq \tilde{\alpha}_{k_{v_m}}^{(0)} \leq 1$ ,  $0 \leq \tilde{\beta}_{v_m} \leq 1$ .

(b) *Substitution product term*: We adopt the product term replacement method [21] to tackle the high order terms  $\{(1 - a)\alpha, a\beta\}$ . We define:  $X_{k_{v_m}}^2 = (1 - \tilde{a}_{v_m}) \tilde{\alpha}_{k_{v_m}}^{(0)}$ ,  $Y_{k_{v_m}}^2 = \tilde{a}_{v_m} \tilde{\beta}_{v_m}^{(1)}$ . Then,  $P1$  is transformed into  $P2$ :

$$\begin{aligned} P2: & \max_{\{\tilde{a}, \tilde{c}, b, X, Y\}} \sum_{m \in \mathcal{M}} \sum_{v_m \in \mathcal{V}_m} \lambda \Phi'_{v_m} + (1 - \lambda) \Psi'_{v_m} \\ C1'': & C1 - C5 \text{ holds}, \\ C4'': & X_{k_{v_m}}^2 \leq 1 - \tilde{a}_{v_m}, v_m \in \mathcal{V}_m, m \in \mathcal{M}, \\ C5'': & Y_{v_m}^2 \leq \tilde{a}_{v_m}, v_m \in \mathcal{V}_m, \\ C6'': & \sum_{k_{v_m} \in \mathcal{K}_{v_m}} \left( \Lambda_{k_{v_m}} X_{k_{v_m}}^2 + \Lambda'_{k_{v_m}} \frac{X_{k_{v_m}}^2}{b_{k_{v_m}}^{(0)}} \right) \\ & + \left( \varpi_1 \frac{Y_{v_m}^2}{b_{v_m}} + \varpi_2 \frac{Y_{v_m}^2}{\tilde{c}_{v_m}^{(v)}} \right) \leq \tau DDL, v_m \in \mathcal{V}_m, \end{aligned} \quad (11)$$

where  $\Lambda'_{k_{v_m}}$ ,  $\Lambda_{k_{v_m}}$ ,  $\varpi_1$  and  $\varpi_2$  are variables independent of optimization, which can be given as  $\Lambda'_{k_{v_m}} = \frac{\epsilon_1 P^{(0,off)} C_{k_{v_m}}}{KB^{(0)} e_{k_{v_m}}^{(0)}}$ ,  $\Lambda_{k_{v_m}} = \frac{\epsilon_1 P^{(0,ps)} C_{k_{v_m}}}{K f_{k_{v_m}}}$ ,  $\varpi_1 = \frac{\epsilon_2 L_{v_m} P^{(1,off)}}{B^{(1)} e_{v_m}^{(1)} P^{(1,off)}}$ ,  $\varpi_2 = \frac{\epsilon_3 L_{v_m} P^{(1,ps)}}{P^{(1,ps)}}$ .  $\Psi'_{v_m}$  and  $\Phi'_{v_m}$  are the objective after converted.

3) *Decomposition of P2*: Observing  $P2$ , we notice that the existence of global variables  $\{\tilde{c}_{v_m}^{(v)}, \tilde{c}_{v_m}^{(b)}, \mathbf{U}_m, \mathbf{W}_m\}$  make  $P2$  non-separable. Therefore, we introduce the local copies  $\{\hat{c}_{v_m}^{(v)}, \hat{c}_{v_m}^{(b)}\}$  of them to split it into independent sub-problems. Meanwhile, we maintain the synchronization of local copies and global variables  $\hat{c}_{v_j, m}^{(v)} = \tilde{c}_{v_j}^{(v)}$ ,  $\hat{c}_{v_j, m}^{(b)} = \tilde{c}_{v_j}^{(b)}$ ,  $\forall v, j, m$ . Then, we can rewrite  $P2$  as optimization problem  $P3$ :

$$P3: Q_m = \begin{cases} - \left( \sum_{v_m \in \mathcal{V}_m} \Phi'_{v_m} + \Psi'_{v_m} \right), & \text{when } C1'' \sim C6'' \text{ holds} \\ \infty, & \text{otherwise} \end{cases} \quad (12)$$

TABLE I  
SIMULATION PARAMETERS

parameters	value
Density of small base stations	2-100 /km <sup>2</sup> km
The compression vector	0.05-0.95
Available Bandwidth resources	2-65MHz
Computational capability of MEC servers	3-60GHz
Computational capability of edge groups	0.8-2.7GHz

$$\min_{\{\tilde{\mathbf{a}}, \alpha, \beta, \tilde{\mathbf{c}}^{(b)}, \tilde{\mathbf{c}}^{(v)}, \tilde{\mathbf{b}}^{(0)}, \tilde{\mathbf{b}}^{(1)}\}} \sum_{m \in \mathcal{M}} Q_m \quad (13)$$

$$C1''' : C1'' \sim C6'' \quad \text{holds}$$

$$C2''' : \tilde{c}_{v_m}^{(v)} = \hat{c}_{v_m}^{(v)}, \tilde{c}_m^{(b)} = \hat{c}_m^{(b)}, \forall v, m.$$

4) *Solution of P3*: According to [6], P3 is called a global consensus problem, whose solution process includes three steps, *local variables updating*, *global variables updating*, and *Lagrange multipliers updating*. First, we write the Lagrangian of P3

$$\begin{aligned} L_\rho & \left( \{\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \tilde{\mathbf{a}}, \mathbf{X}, \mathbf{Y}\}, \{\tilde{\mathbf{c}}^{(b)}, \tilde{\mathbf{c}}^{(v)}\}^{(global)} \right) \\ & \left( \{\hat{\mathbf{c}}^{(b)}, \hat{\mathbf{c}}^{(v)}\}^{(local)}, \{\boldsymbol{\delta}, \boldsymbol{\sigma}\}^{(dual)} \right) \\ & = \sum_{m \in \mathcal{M}} Q_m + \sum_{m \in \mathcal{M}} \sum_{j \in \mathcal{M}} \sum_{v_m \in \mathcal{V}_m} \sigma_{v_j, m} \left( \hat{c}_{v_j, m}^{(v)} - \tilde{c}_{v_j}^{(v)} \right) \\ & + \sum_{m \in \mathcal{M}} \sum_{j \in \mathcal{M}} \sum_{v_m \in \mathcal{V}_m} \delta_{v_j, m} \left( \hat{c}_{v_j, m}^{(b)} - \tilde{c}_{v_j}^{(b)} \right) \\ & + \frac{\rho}{2} \sum_{m \in \mathcal{M}} \sum_{j \in \mathcal{M}} \sum_{v_m \in \mathcal{V}_m} \left( \hat{c}_{v_j, m}^{(v)} - \tilde{c}_{v_j}^{(v)} \right)^2 \\ & + \frac{\rho}{2} \sum_{m \in \mathcal{M}} \sum_{j \in \mathcal{M}} \sum_{v_m \in \mathcal{V}_m} \left( \hat{c}_{v_j, m}^{(b)} - \tilde{c}_{v_j}^{(b)} \right)^2, \end{aligned} \quad (14)$$

where  $\{\boldsymbol{\delta}, \boldsymbol{\sigma}\}$  are the Lagrange multipliers and  $\rho$  is the penalty parameter which determines the convergence property. Then, we propose an ADMM-based algorithm in **Algorithm 1** to solve P3 in a distributed way.

- *Update Local Variables* : The transformed problem (13) is decomposed into  $m$  sub-problems which can be solved distributively by each SBS. First, we update local variables at each iteration  $[t+1]$

$$\begin{aligned} \{\hat{c}_{v_j, m}^{(b)}, \hat{c}_{v_j, m}^{(v)}\}_{m \in \mathcal{M}}^{[t+1]} & = \arg \min_{\{\hat{c}_{j, m}^{(b)}, \hat{c}_{v_j, m}^{(v)}\}} \\ Q_m + \sum_{j \in \mathcal{M}} \sum_{v_m \in \mathcal{V}_m} & \left[ \sigma_{v_j, m}^{[t]} \hat{c}_{v_j, m}^{(v)} + \frac{\rho}{2} \left( \hat{c}_{v_j, m}^{(v)} - \tilde{c}_{v_j}^{(v)[t]} \right)^2 \right] \\ + \sum_{j \in \mathcal{M}} \sum_{v_m \in \mathcal{V}_m} & \left[ \delta_{v_j, m}^{[t]} \hat{c}_{v_j, m}^{(b)} + \frac{\rho}{2} \left( \hat{c}_{v_j, m}^{(b)} - \tilde{c}_{v_j}^{(b)[t]} \right)^2 \right]. \end{aligned} \quad (15)$$

The decomposed sub-problem (15) can be solved by gradient descent method [22] or the newton method [23].

- *Update Global Variables* :

$$\begin{aligned} \tilde{c}_{v_j}^{(v)[t+1]} & = \frac{1}{M} \sum_{m \in \mathcal{M}} \left( \hat{c}_{v_j, m}^{(v)[t+1]} + \frac{1}{\rho} \sigma_{v_j, m}^{[t]} \right) \\ \tilde{c}_j^{(b)[t+1]} & = \frac{1}{M} \sum_{m \in \mathcal{M}} \left( \hat{c}_{v_j, m}^{(b)[t+1]} + \frac{1}{\rho} \delta_{v_j, m}^{[t]} \right) \end{aligned} \quad (16)$$

- *Update Lagrange Multipliers*:

$$\begin{aligned} \sigma_{v_j, m}^{[t+1]} & = \sigma_{v_j, m}^{[t]} + \rho \left( \hat{c}_{v_j, m}^{(v)[t+1]} - \tilde{c}_{v_j}^{(v)[t+1]} \right) \\ \delta_{v_j, m}^{[t+1]} & = \delta_{v_j, m}^{[t]} + \rho \left( \hat{c}_{v_j, m}^{(b)[t+1]} - \tilde{c}_j^{(b)[t+1]} \right) \end{aligned} \quad (17)$$

5) *The Convergence and the Complexity Analysis*: The optimization objective and all optimization variables of (13) are bounded, hence  $\sum_{m \in \mathcal{M}} Q_m < \infty$ . Thus, when  $t \rightarrow \infty$ , the objective and dual variables meet converge. By applying the proposed algorithm, the complexity of (13) is reduced from  $O(\dot{N}\dot{M}\dot{K} + \dot{K}\dot{L})^x$  to  $O(\dot{N}\dot{K} + \dot{K}\dot{L})^x H$ ,  $x > 0$ .  $x = 0$  represents linear time algorithm, and  $x > 1$  represents polynomial time algorithm.  $\dot{L}$  represents the average number of collections in each SBS,  $H$  is the number of iterations to achieve convergence.

**Algorithm 1** The Proposed Algorithm for solving (13)

**1:Initialization**

1) initialize  $\{\tilde{\mathbf{c}}^{(b)}, \tilde{\mathbf{c}}^{(v)}\}^{[0]}$ ,  $\{\boldsymbol{\delta}, \boldsymbol{\sigma}\}^{[0]}$ ,  $\rho$ , and  $\vartheta_{dual}$ .

**2:Repeat**

- 2) Update local variables  $\{\hat{\mathbf{c}}^{(b)}, \hat{\mathbf{c}}^{(v)}\}^{[t]}$  by solving (15).
- 3) Updates the global variables  $\{\tilde{\mathbf{c}}^{(b)}, \tilde{\mathbf{c}}^{(v)}\}^{[t]}$  by solving (16), and update Lagrange multipliers  $\{\boldsymbol{\delta}, \boldsymbol{\sigma}\}^{[t]}$  by solving (17) until

$$\| \hat{\mathbf{c}}^{(b)[t+1]} - \hat{\mathbf{c}}^{(b)[t]} \|_2 \leq \vartheta_{dual}, \| \hat{\mathbf{c}}^{(v)[t+1]} - \hat{\mathbf{c}}^{(v)[t]} \|_2 \leq \vartheta_{dual}$$

**3:Output** the solution  $\{\tilde{\mathbf{a}}, \mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \mathbf{X}, \mathbf{Y}, \tilde{\mathbf{c}}^{(v)}, \tilde{\mathbf{c}}^{(b)}\}$ , and recovery the binary variables and discrete variables  $\{\mathbf{a}, \alpha, \beta, \mathbf{c}^{(v)}, \mathbf{c}^{(b)}\}$ .

V. EXPERIMENT AND DISCUSSION

In this section, we evaluate the performance of our scheme through simulation experiments. The important parameters are listed in Table I.

Fig. 2 illustrates the convergence performance of our scheme. We can find that the objective value converges rapidly in the first 20 iterations, which demonstrates the good convergence of the proposed scheme.

Fig. 3 describes the impact of the weight parameter  $\lambda$  on the performance of the blockchain sub-system and the MEC sub-system. The revenue of the blockchain sub-system and the MEC sub-system show an alternating upward trend. This is because, to achieve the maximum weighted result, computing resources are uniformly allocated to the blockchain system or MEC system in different intervals, rather than evenly distributed to both, which demonstrates that the proposed scheme fully exploits the computing resources.

In Fig. 4(a), we compare the revenue of the MEC sub-system with several baseline schemes, including [6] and [7]. The results illustrated that our proposed scheme performs

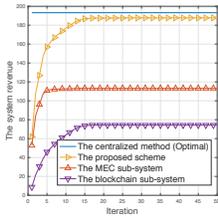


Fig. 2. Convergence of the proposed scheme.

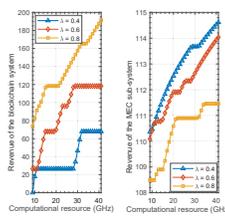
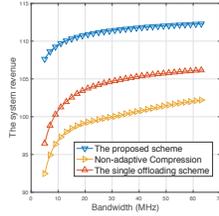
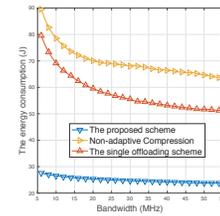


Fig. 3. The impact of the weight parameter  $\lambda$ .



(a) Revenue under different schemes



(b) Energy consumption under different schemes

Fig. 4. The impact of the available bandwidth resources.

better than the other two schemes. This is because the proposed scheme compresses video tasks adaptively, effectively reduce the energy consumption in the transmission and processing stage, thus reducing the expenditure.

Fig. 4(b) reveals the effect of bandwidth resources on the energy consumption. As is illustrated, the more bandwidth resources, the smaller the energy consumption of all schemes. Compared with the other two schemes, the energy consumption of the proposed scheme is the lowest, especially when the bandwidth resources are limited.

## VI. CONCLUSION

In this work, we propose a blockchain-enabled optimization framework for video streaming systems, where adaptive video compression, computing offloading, and resource allocation are jointly considered to formulate as a optimization problem. We then propose an ADMM-based algorithm to solve the problem. Simulation results demonstrate the effectiveness of the proposed scheme. The experimental results show that the proposed scheme can effectively reduce energy consumption and resource overhead.

## REFERENCES

- [1] J. Du, F. R. Yu, G. Lu, J. Wang, J. Jiang, and X. Chu, "Mec-assisted immersive vr video streaming over terahertz wireless networks: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9517–9529, 2020.
- [2] I. Al Ridhawi, M. Aloqaily, A. Boukerche, and Y. Jaraweh, "A blockchain-based decentralized composition solution for iot services," in *2020 IEEE international conference on communications (ICC)*. IEEE, 2020, pp. 1–6.
- [3] L. Qiang and H. Tao, "Dare: Dynamic adaptive mobile augmented reality with edge computing," in *Proc. IEEE International Conference on Network Protocols (ICNP)*, Sept. 2018, pp. 1–11.

- [4] B. Yang, X. Cao, J. Bassey, X. Li, and L. Qian, "Computation offloading in multi-access edge computing: A multi-task learning approach," *IEEE Trans Mob Comput.*, vol. 20, no. 9, pp. 2745–2762, 2020.
- [5] M. Liu, F. R. Yu, Y. Teng, V. C. Leung, and M. Song, "Distributed resource allocation in blockchain-based video streaming systems with mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 695–708, 2018.
- [6] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Computation offloading and resource allocation in wireless cellular networks with mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 16, no. 8, pp. 4924–4938, 2017.
- [7] J. Feng, F. R. Yu, Q. Pei, J. Du, and L. Zhu, "Joint optimization of radio and computational resources allocation in blockchain-enabled mobile edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 4321–4334, 2020.
- [8] Wu, "Revenue-sharing based computation-resource allocation for mobile blockchain," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2020, pp. 56–61.
- [9] X. Jiang, F. R. Yu, T. Song, and V. C. Leung, "Intelligent resource allocation for video analytics in blockchain-enabled internet of autonomous vehicles with edge computing," *IEEE Internet Things J.*, vol. 10, no. 1, pp. 695–708, Sept. 2020.
- [10] "Cooperative computation offloading and resource allocation for delay minimization in mobile edge computing," *J. Syst. Archit.*, vol. 118, p. 102167, 2021.
- [11] S. Xia, Z. Yao, Y. Li, and S. Mao, "Online distributed offloading and computing resource management with energy harvesting for heterogeneous mec-enabled iot," *IEEE Trans. Wireless Commun.*, vol. 20, no. 10, pp. 6743–6757, 2021.
- [12] P. K. Mu, J. Zheng, T. H. Luan, L. Zhu, M. Dong, and Z. Su, "Amis: Edge computing based adaptive mobile video streaming," in *IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2021, pp. 1–10.
- [13] Y. Liu, Q. Chang, M. Peng, T. Dang, and W. Xiong, "Virtual reality streaming in blockchain enabled fog radio access networks," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 1–1, Sept. 2021.
- [14] A. Sacco, F. Esposito, P. Okorie, and G. Marchetto, "Livemicro: An edge computing system for collaborative telepathology," in *2nd {USENIX} Workshop on Hot Topics in Edge Computing (HotEdge 19)*, 2019.
- [15] J. Wang, Z. Feng, Z. Chen, S. George, M. Bala, P. Pillai, S.-W. Yang, and M. Satyanarayanan, "Bandwidth-efficient live video analytics for drones via edge computing," in *2018 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 2018, pp. 159–173.
- [16] Z. Ming, J. Chen, L. Cui, S. Yang, Y. Pan, W. Xiao, and L. Zhou, "Edge-based video surveillance with graph-assisted reinforcement learning in smart construction," *IEEE Internet Things J.*, vol. 99, no. 2, pp. 1–1, June. 2021.
- [17] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.
- [18] J. Li, N. Li, J. Peng, H. Cui, and Z. Wu, "Energy consumption of cryptocurrency mining: A study of electricity consumption in mining cryptocurrencies," *Energy*, vol. 168, pp. 160–168, 2019.
- [19] X. Chen, J.-N. Hwang, D. Meng, K.-H. Lee, R. L. de Queiroz, and F.-M. Yeh, "A quality-of-content-based joint source and channel coding for human detections in a mobile surveillance cloud," *IEEE Trans Circuits Syst Video Technol.*, vol. 27, no. 1, pp. 19–31, 2016.
- [20] S. Yuan, J. Li, C. Wu, Y. Ji, and Y. Zhang, "Dcvc: Distributed collaborative video stream processing in edge computing," in *2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS)*, 2020, pp. 625–632.
- [21] D. Huang, H. Zhou, and Q.-H. Zhao, "A competitive multiple-product newsboy problem with partial product substitution," *Omega*, vol. 39, no. 3, pp. 302–312, 2011.
- [22] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *SIAM Journal on Optimization*, vol. 26, no. 3, pp. 1835–1854, 2016.
- [23] L. W. Ehrlich, "A modified newton method for polynomials," *Communications of the ACM*, vol. 10, no. 2, pp. 107–108, 1967.