A Novel Resource Management Scheme for Blockchain-based Video Streaming with Mobile Edge Computing

Menting Liu, Yinglei Teng, Victor C.M. Leung, and Mei Song

***Abstract*—Blockchain-based video streaming systems aim to build decentralized peer-to-peer networks with flexible monetization mechanisms. In this paper, we propose a novel blockchain-based framework for video streaming with mobile edge computing (MEC). With the trust and traceability features of the blockchain, we design an incentive mechanism to facilitate the collaborations among content creators, video transcoders and consumers, without the intervention of any third party. In addition, we present a block size adaptation scheme for blockchain-based video streaming. Moreover, we consider two offloading modes, i.e., offloading to the nearby MEC nodes or a group of device-to-device (D2D) users, to avoid the overload of MEC nodes. Then, we formulate the issues of resource allocation, scheduling of offloading, and adaptive block size as an optimization problem. We employ a low-complexity alternating direction method of multipliers (ADMM)-based algorithm to solve the problem in a distributed fashion. Simulation results are presented to show the effectiveness of the proposed scheme.**

***Index Terms*—Blockchain, video transcoding, mobile edge computing.**

I. Introduction

With the skyrocketing growth of the demands for online streaming services, video streaming platforms like Netflix and YouTube have become popular over the past decade [1]. However, these traditional online streaming platforms suffer from several disadvantages: 1) Low profit for content creators. 2) High monthly charge and low privacy for consumers. 3) Less-than-ideal advertising effects for advertisers. Recently, several startups (e.g., Theta, Stream, Viewly, Livepeer, Flixxo, VirtuTV, etc.) are employing *blockchain technology* to solve these pitfalls [2]–[7]. Leveraging blockchain technology, these emerging platforms are able to build decentralized peer-to-peer access networks with flexible monetization mechanisms using *smart contract* [8] [9]. In this new era of blockchain-based video streaming, content creators, consumers, and advertisers can fully support each other without the intervention of any third party.

Nevertheless, blockchain-based video streaming systems are also faced with some challenges. One of the main challenges is video *transcoding*. Similar to the traditional video streaming platforms, the original video contents on these emerging platforms are also required to be transcoded/converted into multiple representations in different bitrates, resolutions, qualities, video codec, etc. to cater to heterogeneous users [10]. However, video transcoding is computation-intensive and time-consuming, which sets an extremely high demand of computational resources [11], [12]. In addition, the block size in the blockchain has significant impacts on the performance of blockchain-based video streaming systems. With a larger block size, more transactions can be included on a block, and the throughput of the blockchain can be higher [13]. However, a larger block size will result in higher block propagation delays and higher orphaning probability, which will degrade the performance of the blockchain [14]–[16].

Recent advances in *mobile edge computing* (MEC) [10], [17]–[20] can provide possible solutions addressing the above challenges in blockchain-based video streaming systems. With MEC, computation-intensive transcoding tasks can be offloaded to the network edge, which is equipped with computing and storage resources to accelerate video streaming services [10], [21]. A novel transcoding framework is presented in [10] to offload the transcoding workload to the network edge. [21] adopts a network virtualization framework for video transcoding with MEC.

Although some excellent works have been done on MEC-enabled video streaming, most existing works are carried out in traditional video streaming systems without considering the blockchain. However, the distinct features of the blockchain introduce non-trivial challenges to the MEC-enabled video streaming systems. To the best of knowledge, blockchain-based video streaming systems with MEC have not been well studied. The key contributions of this paper are summarized as follows:

- We propose a novel blockchain-based framework for

M. Liu and V. C. M. Leung are with Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC V6T 1Z4, Canada. (e-mail: liumengtingsophia@gmail.com; vleung@ece.ubc.ca)

Y. Teng and M. Song are with Beijing Key Laboratory of Space-ground Interconnection and Convergence, Beijing University of Posts and Telecommunications, Beijing, 100876, China (e-mail: lilytengtt@gmail.com; songm@bupt.edu.cn)

video streaming with MEC. With the trust and traceability features of the blockchain, we design an incentive mechanism to facilitate the collaborations among content creators, video transcoders and consumers, without the intervention of any third party.

- We present a block size adaptation scheme for blockchain-based video streaming, where the block size can dynamically changed to accommodate the time-varying nature of video streaming and its low-latency requirements.
- Unlike the previous works only considering one single computation offloading mode, we consider two offloading modes, i.e., offloading to the nearby MEC nodes or a group of device-to-device (D2D) users, to avoid the overload of MEC nodes.
- We formulate the issues of resource allocation, scheduling of offloading, and adaptive block size as an optimization problem. Then, we employ a low-complexity alternating direction method of multipliers (ADMM)-based algorithm to solve the problem in a distributed fashion. Simulation results are presented to show the effectiveness of the proposed scheme.

The rest of this paper is organized as follows. Section II presents the system model. The offloading framework and incentive mechanism are introduced in Section III. In Section IV, we formulate the resource allocation, scheduling of offloading, and adaptive block size into an optimization problem and adopt an ADMM-based algorithm to solve it. Simulation results are discussed in Section V. Finally, Section VI concludes this paper.

II. System Model

In this section, we first describe the system architecture to provide some necessary backgrounds. Following that, we present the related models adopted in this paper, including video transcoding model, computation offloading model and network model. For the clarity of the following discussion, the key notations are summarized in Table I.

A. System Architecture

As shown in Fig. 1, we consider an MEC-enabled framework for blockchain-based video streaming systems[1] with adaptive block size S_B, where there are one macro base station (MBS), M small cell base stations (SBSs) and N users. The users, also referred as nodes, have to bond an amount of *token* to join the video streaming system. The SBSs and nodes are distributed according to two independent homogeneous Poisson point processes (HPPPs) with density λ_s and λ_u, respectively. We denote

[1]In this paper, *Livepeer* serves a prototype for the model. However, this model also matches most other blockchain-based video streaming platforms.

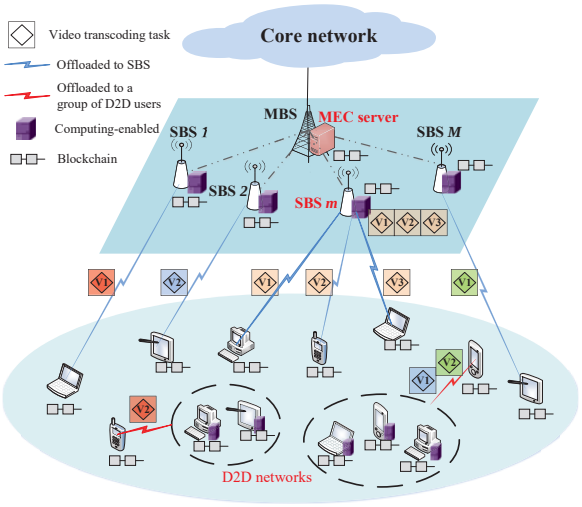


Fig. 1. An illustration of blockchain-based video streaming systems with mobile edge computing (MEC).

the set of SBSs as $\mathcal{M} = \{1, 2, \ldots, M\}$ and use m to refer to the m-th small cell or SBS. Assuming each node accesses to the nearest SBS, there are N_m nodes in small cell m. An MEC server is placed in the MBS, and all the SBSs are connected to the MBS as well as the MEC server. Note that in this MEC-enabled framework, the SBSs and nodes with certain computation capability can assist the transcoding job. In this sense, the transcoders can offload the computating intensive transcoding task to either the nearby SBS or a group of D2D nodes.

B. Video Transcoding Model

Video segment is considered as the unit of video streams, which is a time-sliced chunk of multiplexed audio and video [7]. We assume that there is only one broadcaster $BC_m, m \in \mathcal{M}$ in each small cell m. After publishing the original streams, each broadcaster BC_m would submit a transcoding job transaction TJ_{v_m} onto the blockchain. We use $\langle I_m, V_m, Q_{v_m}, L_{v_m} \rangle$ to denote TJ_{v_m}, where I_m (in *bit*) is the size of the original video stream file, and $v = 1, 2, \ldots, V_m$ represents V_m different targeted versions with Q_{v_m} sliced video segments in length L_{v_m} (in s). As soon as these transactions are mined on the blockchain, the next blockhash will be used to pseudo-randomly determine the transcoders for these jobs. Assume that for each transaction $TJ_{v_m}, \forall m, v$, there is only one corresponding transcoder TC_{v_m} chosen to perform the transcoding task [6]. Then we adopt $\langle r_{v_m}, \tau_{v_m} \rangle$ to describe the requirements (bitrate r_{v_m} and delay tolerance τ_{v_m}) of the v_m-th targeted version.

TABLE I
NOTATIONS

Symbol	Definition	Symbol	Definition
λ_a	The density of SBSs	λ_u	The density of nodes
M	The number of SBSs	N	The number of nodes
I_m	The size of the original video stream file released by broadcaster BC_m	V_m	The number of targeted versions of the video released by BC_m
L_{v_m}	The length of video segments of the v_{th} version w.r.t. the video released by BC_m	Ω_{v_m}	The profit for TC_{v_m} by accomplishing transcoding task
φ_{v_m}	The input size of each video segment in task Φ_{v_m}	X_{v_m}	The workload/intensity of task Φ_{v_m}
Q_{v_m}	The number of video segments in task Φ_{v_m}	r_{v_m}/τ_{v_m}	The requirements (required bitrate/delay tolerance) of task Φ_{v_m}
P_S^T/P_S^I	The power consumption of SBSs in active/idle state	P_U^T	The transmit power of nodes
α	The pathloss exponent	B_S/B_U	The available bandwidth for cellular/D2D networks
σ_w^2	The noise power	ϖ_e	The unit price of energy
S_B	The adaptive block size	$\boldsymbol{\delta}$	Offloading mode selection indicator
$\boldsymbol{b}$	Spectrum allocation vector	$\boldsymbol{c}$	Computational resource allocation profile

C. Computation Offloading Model

For clarity, we use $\Phi_{v_m} = \langle \varphi_{v_m}, X_{v_m}, Q_{v_m}, L_{v_m}, r_{v_m}, \tau_{v_m} \rangle$ to describe the transcoding task w.r.t. TJ_{v_m}, which includes several main parameters: the input size of video segment φ_{v_m} (in *bit*), workload/intensity X_{v_m} (in *CPU cycles/bit*), the number of video segments Q_{v_m} (in *segment*), the length of video segment L_{v_m} (in *s*), required bitrate r_{v_m} (in *bit/s*) and delay tolerance τ_{v_m} (in *s*). Assuming all the SBSs and nodes have a computation capacity to be a transcoding helper, we consider two offloading modes where the transcoding task is:

1) ***offloaded to a nearby SBS (mode 0)***: The transcoder TC_{v_m} offloads the full task Φ_{v_m} to its serving SBS m and the MEC server assigns c_{v_m} CPU cores to SBS m to complete the transcoding job.

2) ***offloaded to a group of D2D nodes (mode 1)***: The transcoder TC_{v_m} divides task Φ_{v_m} into K_{v_m} equal parts $\left\{\Phi_1, \Phi_2, \ldots, \Phi_{K_{v_m}}\right\}$ and separately dispatches them to a group of nearby D2D nodes $U_1, U_2, \ldots, U_{K_{v_m}}$[2].

We denote $\delta_{v_m} \in \{0, 1\}$ as the task offloading decision of $TC_{v_m}, \forall m, v$. Specifically, we have $\delta_{v_m} = 0$ if TC_{v_m} selects *mode 0*, i.e., offloading the whole transcoding task to SBS m. We have $\delta_{v_m} = 1$ if TC_{v_m} chooses *mode 1*, i.e., offloading the task to a group of D2D nodes. So $\boldsymbol{\delta} = \{\delta_{v_m}\}, \forall m, v$ can be considered as the offloading decision profile. Besides, in the case of *mode 0*, we use $\boldsymbol{c} = \{c_{v_m}\}, c_{v_m} \in \{0, 1, \ldots, C\}, \forall m, v$ to denote the computational resource allocation profile, where C is the total number of available CPU cores at the MEC server.

[2]In this paper, we assume that the transcoding task is divided into several equal parts and further offloaded to the D2D nodes. More general cases can be considered in future works.

D. Network Model

The channel radio propagation between nodes and SBSs/nodes is assumed to comprise both path loss and Rayleigh fading. Note that in order to study the performance of transcoding task offloading, we only focus on the channel condition between the transcoders and SBSs or D2D nodes. Specifically, the path loss fading of the channel between TC_{v_m} and SBS m (D2D node $U_{k_{v_m}}$) is $r_{m,v}{}^{-\alpha}$ ($l_{m,v,k}{}^{-\alpha}$) where $r_{m,v}$ ($l_{m,v,k}$) represents the distance between TC_{v_m} and SBS m (D2D node $U_{k_{v_m}}$). Meanwhile, the Rayleigh fading of the link between TC_{v_m} and SBS m (D2D node $U_{k_{v_m}}$) is denoted by $h_{m,v}$ ($g_{m,v,k}$) where $h_{m,v}$ and $g_{m,v,k}$ follow an exponential distribution with mean $1/\varsigma$, i.e., $h_{m,v} \sim \exp(\varsigma)$ and $g_{m,v,k} \sim \exp(\varsigma)$. Note that $h_{m,v}, g_{m,v,k}, \forall m, v, k$ are mutually independent with each other. The transmit power density of the SBSs and nodes with are P_S^T and P_U^T, respectively.

After the trancoding job is done, the SBS and D2D nodes need to send the transcoded version of the video segment back to the transcoder through the downlinks. In downlinks, we assume that two separate channels with bandwidth B_S and B_U are assigned for the cellular networks and D2D networks while an orthogonal frequency-division multiplexing access (OFDMA) transmission mechanism is adopted within each small cell. Thus, we use $b_{v_m}^{(0)} \in [0, 1]$ and $b_{k_{v_m}}^{(1)} \in [0, 1]$ to describe the spectrum allocation in the downlink cellular network and D2D network, respectively. Hence, we have $\boldsymbol{b} = \left\{b_{v_m}^{(0)}, b_{k_{v_m}}^{(1)}\right\}, \forall m, v, k$ as the spectrum allocation profile.

III. OFFLOADING FRAMEWORK AND INCENTIVE MECHANISM

In this section, we first present the offloading framework and the performance analysis for each offloading mode, and

then introduce the incentive mechanism for the blockchain based video streaming systems.

A. Offloading Framework

In order to release the burden of transcoders, we utilize the recent advances in MEC by allowing the video transcoding tasks to be offloaded to the nearby SBS (*mode 0*) or a group of D2D users (*mode 1*). To facilitate the offloading framework, we first derive some important performance metrics including output size, delay and energy consumption as follows.

1) Offloaded to a Nearby SBS (Mode 0):

(i) **Output Size:** The output size refers to the size of transcoded video segments, which is the key indicator for streaming overhead and can be calculated by the product of bitrates and video segment length as $O_{v_m}^{(0)} = r_{v_m} L_{v_m}$.

(ii) **Delay:** The total delay is defined as the time from the arrival of a video stream segment to the completion of transcoding task, which consists of queuing time, transcoding time and downloading time[3] [1]. For transcoder TC_{v_m} in *mode 0*, the system delay can be expressed by

$$D_{v_m}^{(0)} = D_{v_m}^{(0,t)} + D_{v_m}^{(0,q)} + D_{v_m}^{(0,d)}, \tag{1}$$

where the transcoding delay $D_{v_m}^{(0,t)}$, queuing delay $D_{v_m}^{(0,q)}$ and downloading delay $D_{v_m}^{(0,d)}$ are derived as follows.

Before diving into the derivation, we first introduce the queuing model. In this blockchain-based system, we assume the video segments generated from the video stream source are maintained in a queue. Then we model the video transcoding process at SBS m with c_{v_m} CPU cores working in parallel as an M/G/F queue. Specifically, the queue length ℓ_{v_m} increases by one when a video segment arrives and decreases one after the completion of one video segment's transcoding. The arrival of video segments follows a Poisson distribution with rate $\lambda^{(0)}$ while the service time follows a general distribution with mean value $\mu^{(0)}$.

(a) *Transcoding delay*: For transcoder TC_{v_m} who selects *mode 0*, the transcoding time for SBS m allocated with c_{v_m} CPU cores to complete task Φ_{v_m} is $D_{v_m}^{(0,t)} = \frac{1}{\mu^{(0)} c_{v_m}}$.

(b) *Queueing delay*: Average queueing delay is considered in this paper, which comprises two parts, i.e., the remaining processing time of the current transcoding tasks at the SBS and the sum of the transcoding time of all the video segments in the queue, which can be written as

$$D_{v_m}^{(0,q)} = D_{v_m}^{(0,r)} + \frac{\mathbb{E}(\ell_{v_m})}{\mu^{(0)} c_{v_m}}, \tag{2}$$

where $D_{v_m}^{(0,r)}$ represents the remaining processing time at SBS m and $\mathbb{E}(\ell_{v_m})$ is the average queue length.

[3]Since the input size of the video segment is much smaller than the output one, the time cost for uploading the video segment to the SBS can be neglected. And the same assumption is considered in *mode 1*.

According to the Little's formula, we have $\mathbb{E}(\ell_{v_m}) = \lambda^{(0)} \mathbb{E}\left[D_{v_m}^{(0,q)}\right]$ and $D_{v_m}^{(0,q)} = \frac{D_{v_m}^{(0,r)}}{1-\lambda/[\mu^{(0)} c_{v_m}]}$ with the remaining processing time $D_{v_m}^{(0,r)} = \frac{1}{2}\lambda^{(0)}\left[{\sigma_q}^2 + \frac{1}{(\mu^{(0)} c_{v_m})^2}\right]$ and the variance of task commences ${\sigma_q}^2$ (Details and proof can be found in [11]).

(c) *Downloading delay*: The time cost for SBS m to send the transcoded video segment back to TC_{v_m} is calculated by

$$D_{v_m}^{(0,d)} = \frac{O_{v_m}^{(0)}}{b_{v_m}^{(0)} B_S e_{v_m}^{(0)}}, \tag{3}$$

where $e_{v_m}^{(0)}$ is the spectrum efficiency (SE) of the downlink from SBS m to TC_{v_m}, which can be obtained using stochastic geometry methods [22] as

$$\begin{aligned} e_{v_m}^{(0)} &= \ln\left(1 + \frac{P_S h_{m,v} {r_{m,v}}^{-\alpha}}{{\sigma_w}^2 + I_{v_m}^{(0)}}\right) \\ &= \int_0^\infty \exp\left(\begin{array}{l} -\varsigma\beta {r_{m,v}}^{\alpha} {\sigma_w}^2 \\ -2\pi\lambda_S \int_{r_{m,v}}^\infty \left(1 - \frac{1}{1+\varsigma\beta\left(\frac{r_{m,v}}{z}\right)^\alpha}\right) z dz \end{array} \right) d\beta, \end{aligned} \tag{4}$$

where $I_{v_m}^{(0)} = \sum_{x=1, x\neq m}^{M} P_S^T h_{x,v} {r_{x,v}}^{-\alpha}$ is the interference from other SBSs, and ${\sigma_w}^2$ is the power spectrum density of additive white Gaussian noise.

(iii) **Energy Consumption:** In this paper, we ignore the energy consumption for uploading the video segment to the SBS because the input size of the video segment is much smaller than that of the output one. In line with [11], [23], we consider the total energy cost at the SBS as

$$E_{v_m}^{(0)} = P_S^T D_{v_m}^{(0,d)} + P_S^I \left[D_{v_m}^{(0,t)} + D_{v_m}^{(0,q)}\right], \tag{5}$$

where P_S^I is the power consumption in idle state for the SBSs.

2) Offloaded to a Group of D2D Users (Mode 1):

In this part, we conduct the performance analysis for *mode 1*, i.e., the transcoding task is offloaded to a group of D2D nodes.

(i) **Output Size:** The size of the transcoded version w.r.t. one single part $\Phi_{k_{v_m}}$ can be calculated by $O_{k_{v_m}}^{(1)} = \frac{r_{v_m} L_{v_m}}{K_{v_m}}$.

(ii) **Delay:** Differ from *mode 0*, only the transcoding delay and downloading delay are considered in *mode 1*, and the queueing delay is left out due to the small computational capacity of each D2D node. For TC_{v_m} who chooses *mode 1*, the total delay is

$$D_{k_{v_m}}^{(1)} = D_{k_{v_m}}^{(1,t)} + D_{k_{v_m}}^{(1,d)}, \tag{6}$$

where the transcoding delay $D_{k_{v_m}}^{(1,t)}$ and downloading delay $D_{k_{v_m}}^{(1,d)}$ are specified as follows.

(a) *Transcoding delay*: For the part of video segment $\Phi_{k_{v_m}}$ offloaded to $U_{k_{v_m}}$ by TC_{v_m}, the transcoding delay is

$D_{k_{v_m}}^{(1,t)} = \frac{\varphi_{v_m} X_{v_m}}{K_{v_m} f_{k_{v_m}}}$ with the computational capability $f_{k_{v_m}}$ for $U_{k_{v_m}}$.

(b) *Downloading delay*: The time cost for $U_{k_{v_m}}$ to send the transcoded video segments back to TC_{v_m} is calculated as

$$D_{k_{v_m}}^{(1,d)} = \frac{O_{k_{v_m}}^{(1)}}{b_{k_{v_m}}^{(1)} B_U e_{k_{v_m}}^{(1)}}, \tag{7}$$

where $e_{k_{v_m}}^{(1)}$ is the SE of the downlink from $U_{k_{v_m}}$ to TC_{v_m}, which can also be obtained with stochastic geometry methods [22] as: where $I_{k_{v_m}}^{(1)} = \sum_{x=1, x\neq m}^{M} \sum_{v=1}^{V_x} \sum_{k=1}^{K_{vx}} P_U^T g_{x,v,k} l_{x,v,k}{}^{-\alpha}$ is the interference from the nodes in other small cells, and ϕ is the second dimension linear contact distribution [24] whose probability density function (PDF) is calculated as

(iii) **Energy Consumption:** The total energy consumption for the D2D nodes $\left\{U_1, U_2, \ldots, U_{K_{v_m}}\right\}$ to transcode the whole video segment is $E_{v_m}^{(1)} = \sum_{k=1}^{K_{v_m}} \left[\kappa f_{k_{v_m}}{}^3 D_{k_{v_m}}^{(1,t)} + P_U^T D_{k_{v_m}}^{(1,d)}\right]$ with the computation energy efficiency coefficient κ.

B. Incentive Mechanism

It's worth noticing that the nodes in the video streaming systems may not be willing to become transcoders if there is no reward for the transcoding service. Accordingly, we introduce an incentive mechanism for the proposed framework, which is based on the virtual currency circulated in the blockchain-based video streaming systems, i.e., token.

1) The Revenue of Transcoding Service:

The revenue TC_{v_m} can receive by finishing the transcoding job is denoted by $R_{v_m}^{SC}$, whose details are predeterminded in the smart contract and coded on the blockchain [6]. If TC_{v_m} successfully completed the transcoding job, it would receive a percentage ($\eta_{v_m}^b$) of block reward $R_{v_m}^B$ shared by the bonded nodes who stake on it. Other rewards and punishment are denoted by $R_{v_m}^O$ which comprises the transcoding reward, fee share, slashed reward, etc.[4] In conclude, we can quantify the transcoding revenue $R_{v_m}^{SC}$ of TC_{v_m} as

$$R_{v_m}^{SC} = \eta_{v_m}^b \left(1 - \mathbb{P}_{orphan}\right) R_{v_m}^B + R_{v_m}^O, \tag{10}$$

where two key factors are outlined as follows:

- **Orphaning probability:** As is shown in (10), the expected block reward $R_{v_m}^B$ is discounted by the chances that the block is orphaned, $\mathbb{P}_{orphan}$, which largely depends on the block size S_B and block generation time T_G. Using the fact that block times follow a Poisson distribution, the orphaning probability can be approximated as [14], [25]:

$$\mathbb{P}_{orphan} = 1 - e^{-\frac{t_P}{T_G}} = 1 - e^{-\frac{\xi S_B}{T_G}}, \tag{11}$$

where t_P is the block propagation time that is assumed to be linear with block size, i.e., $t_P = \xi S_B$ where ξ (*second/kB*) is the marginal time needed to reach consensus [26].

- **Block reward:** According to [25] and [26], block reward includes a fixed reward $\dot{R}$ and a variable reward decided by the number of transactions on a block. Therefore, we have:

$$R_{v_m}^B = \dot{R} + \varepsilon \frac{S_B}{\chi_{v_m}}, \tag{12}$$

where ε is a given variable reward factor and χ_{v_m} denotes the average size of transactions w.r.t. task Φ_{v_m}.

Observing (11) and (12), we can find that the block size issue introduces a tradeoff between *orphaning probability* and *block reward*. Specifically, the larger the block is, it would take more time to propagate the mined block to the whole network, thus increases the chance of "orphaning". However, with larger block size, the block reward becomes higher due to more transactions included on one block at a time.

2) The Cost of Transcoding Service:

In this paper, we assume the transcoders need to pay for the energy cost for the transcoding task. By introducing the unit price of energy ϖ_e (*token/J*), the payment for the energy cost of transcoding per video segment can be quantified as

$$Z_{v_m}^e = \varpi_e \left[\left(1 - \delta_{v_m}\right) E_{v_m}^{(0)} + \delta_{v_m} E_{v_m}^{(1)}\right]. \tag{13}$$

Taking the transcoding service revenue as well as cost into consideration, the profit TC_{v_m} can receive by transcoding the whole video file is given by (14) according to (10)-(13). where the part reward $R_{v_m}^O$ not relevant to the optimization variables is eliminated in (a), and these denotations are defined for ease of description: $\Lambda_{v_m}^{(0)} = \frac{P_S^T O_{v_m}^{(0)}}{B_S e_{v_m}^{(0)}}$ and $\Lambda_{k_{v_m}}^{(1)} = \frac{P_U^T O_{k_{v_m}}^{(1)}}{B_U e_{k_{v_m}}^{(1)}}$.

IV. Problem Formulation and Solution

In this section, we formulate the maximization of average transcoding service profit (i.e., average reward for the transcoders) as an optimization problem by jointly considering the issues of adaptive block size, scheduling of offloading, computational resource allocation and spectrum allocation $(S_B, \boldsymbol{\delta}, \boldsymbol{c}, \boldsymbol{b})$. To obtain the solution, we first relax the discrete variables and use the reformulation-linearization-technique (RLT) to reformulate the problem, and then adopt an ADMM-based algorithm to solve it in a distributed fashion.

A. Problem Formulation

In this paper, we aim to maximize the average reward for the transcoders by seeking the optimal offloading scheduling and resource allocation scheme for video transcoding with adaptive block size. Therefore, we formulate

[4]Since other rewards (excepting the block reward) and punishment $R_{v_m}^O$ has no correlation with the block size issue, the details are omitted.

$$
\begin{aligned}
e_{k_{v_m}}^{(1)} &= \frac{O_{k_{v_m}}^{(1)}}{b_{k_{v_m}}^{(1)} B_U \ln\left(1+\frac{P_U^T g_{m,v,k} l_{m,v,k}{}^{-\alpha}}{\sigma_w{}^2 + I_{k_{v_m}}^{(1)}}\right)} \\
&= \int_0^\infty \int_{l_{m,v,k}}^\infty \exp\left(\begin{array}{l} -\varsigma\beta l_{m,v,k}{}^{\alpha}\sigma_w{}^2 \\ -2\pi\frac{\lambda_U}{K_{v_m}}\int_\phi^\infty\left(1-\frac{1}{1+\varsigma\beta\left(\frac{l_{m,v,k}}{z}\right)^{\alpha}}\right) z dz \end{array}\right) f(\phi)\, d\phi d\beta,
\end{aligned} \tag{8}
$$

$$
\begin{aligned}
&f(\phi) = 1 - \tfrac{1}{2}\exp\left(-4\pi\lambda_U\phi^2\right) + \tfrac{1}{2}\exp\left(-6\pi\lambda_U\phi^2\right) - \\
&2\pi\lambda_U \int_\phi^\infty y \exp\left\{-\lambda_U\left[\ \left(4\phi^2+2y^2\right)\left(\pi - \arccos\tfrac{\phi}{y}\right) + 6\phi\sqrt{y^2-\phi^2}\ \right]\right\} dy\ .
\end{aligned} \tag{9}
$$

$$
\begin{aligned}
\Omega_{v_m} &= R_{v_m}^{SC} - Z_{v_m}^e Q_{v_m} \\
&\overset{(a)}{=} \eta_{v_m}^b e^{-\frac{\varepsilon}{T_G}S_B}\left(\dot{R} + \frac{\varepsilon}{\chi_{v_m}}S_B\right) - \varpi_e Q_{v_m}\left[\begin{array}{l} (1-\delta_{v_m})\left(\frac{1}{c_{v_m}}\frac{P_S^I}{\mu^{(0)}} + \frac{\Lambda_{v_m}^{(0)}}{b_{v_m}^{(0)}} + P_S^I D_{v_m}^{(0,q)}\right) \\ +\delta_{v_m}\sum_{k=1}^{K_{v_m}}\left(\frac{\Lambda_{k_{v_m}}^{(1)}}{b_{k_{v_m}}^{(1)}} + \kappa f_{k_{v_m}}{}^3 D_{k_{v_m}}^{(1,t)}\right)\end{array}\right],
\end{aligned} \tag{14}
$$

the optimization problem as $\mathcal{P}1$, where the optimization variables $(S_B, \boldsymbol{\delta}, \boldsymbol{c}, \boldsymbol{b})$ describe the adaptive block size S_B, the offloading decision indicator vector $\{\delta_{v_m}, \forall m, v\}$, the computational resource allocation profile $\{c_{v_m}, \forall m, v\}$, and the spectrum allocation profile $\left\{b_{v_m}^{(0)}, b_{k_{v_m}}^{(1)}, \forall m, v, k\right\}$. where constraints $C1$ guarantee that the transcoding task offloading mode selection is valid. Constraints $C2$ ensure the validity of computational resource allocation. Constraints $C3$ and $C4$ guarantee that the sum of spectrum allocated to all the downlinks between SBS/D2D nodes and the transcoders cannot exceed the total available spectrum bandwidth B_S/B_U. In order to meet the delay requirement, $C5$ are put forward. Finally, constraint $C6$ makes sure that the block size does not excess the block size limit $\dot{S}$.

Observing $\mathcal{P}1$, we can find this problem is extended from the Knapsack problem, which is faced with two major challenges: **1)** It's a mixed discrete and second order (in form of x/y in the objective and $C5$, where x and y are optimization variables) optimization problem, which is notoriously difficult to solve [27]. **2)** $\mathcal{P}1$ has a quite large size. In order to solve $\mathcal{P}1$ in a centralized way, the MEC server needs to access the channel state information (CSI) of all the nodes, which is very challenging, especially for dense networks.

B. Problem Reformulation

To address **the first difficulty** (i.e., discrete and second order property), we can use discrete variable relaxation and RLT to reformulate the original problem.

Before doing the transformation, we first introduce two microscales, θ_c and θ_b to avoid the *divide-by-zero* error induced by $\boldsymbol{c}$ and $\boldsymbol{b}$, thus $\mathcal{P}1$ is converted into where Ω_{v_m} in the objective function turns into

Lemma 1. Problem $\mathcal{P}_1{}'$ is a lower bound of problem $\mathcal{P}_1$.

Proof. For better illustration, we define the feasible set of $\mathcal{P}_1$ and $\mathcal{P}_1{}'$ as $\mathcal{F}_1$ and $\mathcal{F}_1{}'$, respectively. Compared with $\mathcal{P}_1$, $\mathcal{P}_1{}'$ only shows the difference in the objective and $C5'$. Due to the fact that $\frac{1}{x} > \frac{1}{x+\theta}, \theta > 0, \forall x$, the objective function of $\mathcal{P}_1$ is larger than that of $\mathcal{P}_1{}'$, i.e., $\Omega_{v_m} > \Omega_{v_m}{}', \forall m, v$. For the same reason, we can infer that $C5$ is stricter than $C5'$, then we have $\mathcal{F}_1 \subset \mathcal{F}_1{}'$. In conclude, the optimal result of $\mathcal{P}_1{}'$ is smaller than that obtained from $\mathcal{P}_1$. In other words, $\mathcal{P}_1{}'$ provides a lower bound solution for $\mathcal{P}_1$. □

Next, we define $\hat{c}_{v_m} := \frac{1}{c_{v_m}+\theta_c}$, $\hat{b}_{v_m}^{(0)} := \frac{1}{b_{v_m}^{(0)}+\theta_b}$ and $\hat{b}_{k_{v_m}}^{(1)} := \frac{1}{b_{k_{v_m}}^{(1)}+\theta_b}$ as the auxiliary variables, and reformulate $\Omega_{v_m}{}'$, C_2, C_3, C_4 and $C_5{}'$ as follows:

and

$$
\begin{aligned}
&C_2{}': \sum_{m=1}^{M}\sum_{v=1}^{V_m}\frac{1}{\hat{c}_{v_m}} \le C + N\theta_c,\ \frac{1}{(1-\delta_{v_m})C+\theta_c} \le \hat{c}_{v_m} \le \frac{1}{\theta_c}, \forall m, v \\
&C_3{}': \sum_{v=1}^{V_m}\frac{1}{\hat{b}_{v_m}^{(0)}} \le 1 + V_m\theta_b,\ \frac{1}{(1-\delta_{v_m})+\theta_b} \le \hat{b}_{v_m}^{(0)} \le \frac{1}{\theta_b}, \forall m, v \\
&C_4{}': \sum_{v=1}^{V_m}\sum_{k=1}^{K_{v_m}}\frac{1}{\hat{b}_{k_{v_m}}^{(1)}} \le 1 + N\theta_b,\ \frac{1}{(1-\delta_{v_m})+\theta_b} \le \hat{b}_{k_{v_m}}^{(1)} \le \frac{1}{\theta_b}, \forall m, v, k \\
&C_5{}'': (1-\delta_{v_m})\left(\frac{\hat{c}_{v_m}}{\mu^{(0)}} + \frac{\hat{b}_{v_m}^{(0)}\Lambda_{v_m}^{(0)}}{P_S^T} + D_{v_m}^{(0,q)}\right) + \\
&\qquad \delta_{v_m}\left(\frac{\hat{b}_{k_{v_m}}^{(1)}\Lambda_{k_{v_m}}^{(1)}}{P_U^T} + D_{k_{v_m}}^{(1,t)}\right) \le \tau_{v_m}, \forall m, v, k.
\end{aligned} \tag{19}
$$

Then we conduct the transformation of $\mathcal{P}_1{}'$ by the following two steps:

1) ***Discrete Variable Relaxation:*** We first relax the discrete variables $\boldsymbol{\delta}$ and $\boldsymbol{c}$ into continuous variables as $0 \le \delta_{v_m} \le 1$ and $0 \le c_{v_m} \le C$, respectively.

2) ***Reformulation Linearization Technique (RLT) based Transformation:*** For the second order terms in the form of

$$
\begin{aligned}
\mathcal{P}1: \quad & \max_{S_B,\boldsymbol{\delta},\boldsymbol{c},\boldsymbol{b}} \quad \tfrac{1}{M}\sum_{m=1}^{M}\sum_{v=1}^{V_m}\Omega_{v_m} \\
s.t.\ & C_1: \delta_{v_m}\in\{0,1\},\forall m,v \\
& C_2: \sum_{m=1}^{M}\sum_{v=1}^{V_m} c_{v_m}\le C, c_{v_m}\in\{0,1,\ldots,(1-\delta_{v_m})C\},\forall m,v \\
& C_3: \sum_{v=1}^{V_m} b_{v_m}^{(0)} B_S \le B_S, 0\le b_{v_m}^{(0)}\le 1-\delta_{v_m},\forall m,v \\
& C_4: \sum_{v=1}^{V_m}\sum_{k=1}^{K_{v_m}} b_{k_{v_m}}^{(1)} B_U \le B_U, 0\le b_{k_{v_m}}^{(1)}\le \delta_{v_m},\forall m,v,k \\
& C_5: (1-\delta_{v_m})\left(\frac{1/\mu^{(0)}}{c_{v_m}}+\frac{\Lambda_{v_m}^{(0)}/P_S^T}{b_{v_m}^{(0)}}+D_{v_m}^{(0,q)}\right)+\delta_{v_m}\left(\frac{\Lambda_{k_{v_m}}^{(1)}/P_U^T}{b_{k_{v_m}}^{(1)}}+D_{k_{v_m}}^{(1,t)}\right)\le\tau_{v_m},\forall m,v,k \\
& C_6: S_B\le\dot{S}
\end{aligned}
\tag{15}
$$

$$
\begin{aligned}
\mathcal{P}_1': \quad & \max_{S_B,\boldsymbol{\delta},\boldsymbol{c},\boldsymbol{b}} \tfrac{1}{M}\sum_{m=1}^{M}\sum_{v=1}^{V_m}\Omega_{v_m}' \\
s.t.\ & C_1-C_4, C_6 \\
& C_5': (1-\delta_{v_m})\left(\frac{1/\mu^{(0)}}{c_{v_m}+\theta_c}+\frac{\Lambda_{v_m}^{(0)}/P_S^T}{b_{v_m}^{(0)}+\theta_b}+D_{v_m}^{(0,q)}\right)+\delta_{v_m}\left(\frac{\Lambda_{k_{v_m}}^{(1)}/P_U^T}{b_{k_{v_m}}^{(1)}+\theta_b}+D_{k_{v_m}}^{(1,t)}\right)\le\tau_{v_m},\forall m,v,k
\end{aligned}
\tag{16}
$$

$$
\Omega_{v_m}' = \eta_{v_m}^b e^{-\frac{\xi}{T_G}S_B}\left(\dot{R}+\frac{\varepsilon}{\chi_{v_m}}S_B\right)-\varpi_e Q_{v_m}\left[\begin{array}{l}(1-\delta_{v_m})\left(\frac{1}{c_{v_m}+\theta_c}\frac{P_S^I}{\mu^{(0)}}+\frac{\Lambda_{v_m}^{(0)}}{b_{v_m}^{(0)}+\theta_b}+P_S^I D_{v_m}^{(0,q)}\right)\\ +\delta_{v_m}\sum_{k=1}^{K_{v_m}}\left(\frac{\Lambda_{k_{v_m}}^{(1)}}{b_{k_{v_m}}^{(1)}+\theta_b}+\kappa f_{k_{v_m}}{}^3 D_{k_{v_m}}^{(1,t)}\right)\end{array}\right]. \tag{17}
$$

$$
\Omega_{v_m}''=\eta_{v_m}^b e^{-\frac{\xi}{T_G}S_B}\left(\dot{R}+\frac{\varepsilon}{\chi_{v_m}}S_B\right)-\varpi_e Q_{v_m}\left[\begin{array}{l}(1-\delta_{v_m})\left(\frac{\hat{c}_{v_m}P_S^I}{\mu^{(0)}}+\hat{b}_{v_m}^{(0)}\Lambda_{v_m}^{(0)}+P_S^I D_{v_m}^{(0,q)}\right)\\ +\delta_{v_m}\sum_{k=1}^{K_{v_m}}\left(\hat{b}_{k_{v_m}}^{(1)}\Lambda_{k_{v_m}}^{(1)}+\kappa f_{k_{v_m}}{}^3 D_{k_{v_m}}^{(1,t)}\right)\end{array}\right], \tag{18}
$$

$x\cdot y$, we use the RLT [28] to linearize the objective function, C_3', C_4' and C_5'' by defining $\boldsymbol{X}=\{X_{v_m}\}, X_{v_m}:=\delta_{v_m}\hat{c}_{v_m}$, $\boldsymbol{Y}=\left\{Y_{v_m}^{(0)},Y_{k_{v_m}}^{(1)}\right\}, Y_{v_m}^{(0)}:=\delta_{v_m}\hat{b}_{v_m}^{(0)}, Y_{k_{v_m}}^{(1)}:=\delta_{v_m}\hat{b}_{k_{v_m}}^{(1)}$.

After the discrete variable relaxation, δ_{v_m}, $\hat{c}_{v_m}$, $\hat{b}_{v_m}^{(0)}$ and $\hat{b}_{k_{v_m}}^{(1)}$ are bonded by $0\le\delta_{v_m}\le 1$, $\frac{1}{C+\theta_c}\le\hat{c}_{v_m}\le\frac{1}{\theta_c}$, $\frac{1}{1+\theta_b}\le\hat{b}_{v_m}^{(0)}\le\frac{1}{\theta_b}$, and $\frac{1}{1+\theta_b}\le\hat{b}_{k_{v_m}}^{(1)}\le\frac{1}{\theta_b}$, respectively, thus we can obtain the *RLT bound-factor product constraints* for X_{v_m} as

$$
\Xi_{v_m}^{c}=\begin{cases}X_{v_m}-\frac{1}{C+\theta_c}\delta_{v_m}\ge 0,\\ \frac{1}{\theta_c}-\hat{c}_{v_m}-\frac{1}{\theta_c}\delta_{v_m}+X_{v_m}\ge 0,\\ \frac{1}{\theta_c}\delta_{v_m}-X_{v_m}\ge 0,\\ \hat{c}_{v_m}-\frac{1}{C+\theta_c}-X_{v_m}+\frac{1}{C+\theta_c}\delta_{v_m}\ge 0.\end{cases} \tag{20}
$$

Similarly, we can obtain the *RLT bound-factor product constraints* for $Y_{v_m}^{(0)}$ as

$$
\Xi_{v_m}^{(b,0)}=\begin{cases}Y_{v_m}^{(0)}-\frac{1}{1+\theta_b}\delta_{v_m}\ge 0,\\ \frac{1}{\theta_b}-\hat{b}_{v_m}^{(0)}-\frac{1}{\theta_b}\delta_{v_m}+Y_{v_m}^{(0)}\ge 0,\\ \frac{1}{\theta_b}\delta_{v_m}-Y_{v_m}^{(0)}\ge 0,\\ \hat{b}_{v_m}^{(0)}-\frac{1}{1+\theta_b}-Y_{v_m}^{(0)}+\frac{1}{1+\theta_b}\delta_{v_m}\ge 0.\end{cases} \tag{21}
$$

The constraints for $Y_{k_{v_m}}^{(1)}$ is denoted by $\Xi_{k_{v_m}}^{(b,1)},\forall m,v,k$, which has the similar form with $\Xi_{v_m}^{(b,0)}$. After substituting X_{v_m}, $Y_{v_m}^{(0)}$ and $Y_{k_{v_m}}^{(1)}$ into Ω_{v_m}'', C_3', C_4' and C_5'', the original optimization problem is reformulated as

C. Problem Decomposition

To address **the second difficulty** (i.e., large-scale issue), we attempt to solve $\mathcal{P}2$ in a distributed manner with the ADMM algorithm. Observing $\mathcal{P}_2$, we can find that S_B, $\hat{\boldsymbol{c}}$ and its related varibale $\boldsymbol{X}$ are global variables that make the optimization problem inseparable. Intuitively, we can introduce the local copies of $S_B,\hat{\boldsymbol{c}},\hat{\boldsymbol{X}}$ for each small cell,

$$
\begin{aligned}
&\mathcal{P}2: \max_{\substack{S_B,\boldsymbol{\delta},\hat{\boldsymbol{c}},\hat{\boldsymbol{b}}\\ \boldsymbol{X},\boldsymbol{Y}}} \frac{1}{M}\sum_{m=1}^{M}\eta_{v_m}^{b} e^{-\frac{\xi}{T_G}S_B}\left(\dot{R}+\frac{\varepsilon}{\chi_{v_m}}S_B\right)-\\
&\varpi_e Q_{v_m}\left[\begin{array}{l}(\hat{c}_{v_m}-X_{v_m})\frac{P_S^I}{\mu^{(0)}}+\left(\hat{b}_{v_m}^{(0)}-Y_{v_m}^{(0)}\right)\Lambda_{v_m}^{(0)}\\ +(1-\delta_{v_m})P_S^I D_{v_m}^{(0,q)}+\sum_{k=1}^{K_{v_m}}\left(Y_{k_{v_m}}^{(1)}\Lambda_{k_{v_m}}^{(1)}+\delta_{v_m}\kappa f_{k_{v_m}}{}^3 D_{k_{v_m}}^{(1,t)}\right)\end{array}\right]\\
&s.t.\ C_6\\
&\quad C_1': 0\le\delta_{v_m}\le 1,\forall m,v\\
&\quad C_2'': \sum_{m=1}^{M}\sum_{v=1}^{V_m}\frac{1}{\hat{c}_{v_m}}\le C+N\theta_c, \hat{c}_{v_m}\le\frac{1}{\theta_c}, (\hat{c}_{v_m}-X_{v_m})C+\hat{c}_{v_m}\theta_c\ge 1,\forall m,v\\
&\quad C_3'': \sum_{v=1}^{V_m}\frac{1}{\hat{b}_{v_m}^{(0)}}\le 1+V_m\theta_b, \hat{b}_{v_m}^{(0)}\le\frac{1}{\theta_b}, \hat{b}_{v_m}^{(0)}-Y_{v_m}^{(0)}+\hat{b}_{v_m}^{(0)}\theta_b\ge 1,\forall m,v\\
&\quad C_4'': \sum_{v=1}^{V_m}\sum_{k=1}^{K_{v_m}}\frac{1}{\hat{b}_{k_{v_m}}^{(1)}}\le 1+N\theta_b, \hat{b}_{k_{v_m}}^{(1)}\le\frac{1}{\theta_b}, Y_{k_{v_m}}^{(1)}+\hat{b}_{k_{v_m}}^{(1)}\theta_b\ge 1,\forall m,v,k\\
&\quad C_5''': (\hat{c}_{v_m}-X_{v_m})\frac{1}{\mu^{(0)}}+\left(\hat{b}_{v_m}^{(0)}-Y_{v_m}^{(0)}\right)\frac{\Lambda_{v_m}^{(0)}}{P_S^T}+\\
&\qquad\qquad (1-\delta_{v_m})D_{v_m}^{(0,q)}+Y_{k_{v_m}}^{(1)}\frac{\Lambda_{k_{v_m}}^{(1)}}{P_U^T}+\delta_{v_m}D_{k_{v_m}}^{(1,t)}\le\tau_{v_m},\forall m,v,k\\
&\quad C_7: X_{v_m}\in\Xi_{v_m}^{c}, Y_{v_m}^{(0)}\in\Xi_{v_m}^{(b,0)}, Y_{k_{v_m}}^{(1)}\in\Xi_{k_{v_m}}^{(b,1)},\forall m,v,k
\end{aligned}
\tag{22}
$$

and then try to make these copies agree. For small cell m, we denote $\tilde{\boldsymbol{S}}_{\boldsymbol{B}} := \left\{\tilde{S}_B(m)\right\}, \tilde{\boldsymbol{c}} := \{\tilde{c}_{v_i}(m)\}, \tilde{\boldsymbol{X}} := \left\{\tilde{X}_{v_i}(m)\right\}, v=1,\ldots,V_i, m, i=1,\ldots,M$ as the local copy of $S_B, \hat{\boldsymbol{c}}$, and $\hat{\boldsymbol{X}}$, respectively, so we have

$$\tilde{S}_B(m)=S_B, \tilde{c}_{v_i}(m)=\hat{c}_{v_i}, \tilde{X}_{v_i}(m)=X_{v_i},\forall m,i,v. \tag{23}$$

Then the equivalent global consensus version of $\mathcal{P}_2$ is where constraints C_8 ensure the consistency between the local variables and the corresponding global ones, and $\boldsymbol{F}^l = \left\{\mathcal{F}_m^l\right\} = \left\{\delta,\tilde{c},b,\tilde{X},Y\right\}_m$ and $\boldsymbol{F}^g = \left\{\mathcal{F}_m^g\right\} = \{\hat{c},X\}_m$ are denoted for convenience.

Note that $\tilde{\boldsymbol{S}}_{\boldsymbol{B}}, S_B$ (block size related) and $\boldsymbol{F}^l, \boldsymbol{F}^g$ (offloading scheduling and resource allocation related) are decoupled, so we can divide the local utility function into two parts, (25) and (26):

With (25) and (26), we can rewrite $\mathcal{P}_3$ as

$$
\begin{aligned}
&\mathcal{P}_3': \min_{\tilde{\boldsymbol{S}}_{\boldsymbol{B}},S_B;\boldsymbol{F}^l,\boldsymbol{F}^g}\sum_{m=1}^{M}U_m^{(1)}+U_m^{(2)}\\
&\quad C_8: \tilde{S}_B(m)=S_B, \tilde{c}_{v_i}(m)=\hat{c}_{v_i}, \tilde{X}_{v_i}(m)=X_{v_i},\forall m,i,v
\end{aligned}
\tag{27}
$$

Thanks to the introduction of local copies, $\mathcal{P}_3'$ can be decoupled into M sub-problems that can be independently solved in each small cell. Therefore, the ADMM can be used to solve $\mathcal{P}_3'$ in a distributed fashion, thus the computational complexity can be significantly reduced.

D. Problem Solutions with ADMM

According to [29] and [30], $\mathcal{P}_3'$ is so called *global consensus problems* that needs a network-wide consensus. First, we derive the augmented Lagrangian for $\mathcal{P}_3'$ as where $\boldsymbol{w}^S = \left\{w^S(m)\right\}, \boldsymbol{w}^c = \left\{w_{v_i}^c(m)\right\}, \boldsymbol{w}^X = \left\{w_{v_i}^X(m)\right\}, v=1,\ldots,V_i, m, i=1,\ldots,M$ are the Lagrange multipliers w.r.t. C_8, and $\rho\in\mathbb{R}_{++}$ is the penalty coefficient to promote consensuses.

Then based on ADMM, the the global consensus of block size, offloading scheduling and resource allocation for video transcoding can be achieved by the following updates.

1) Adaptive Block Size:

- The adaptive block size for SBS m:

To obtain the optimal block size, each SBS needs to solve the following local problem:

$$\mathcal{SP}_{1_L}: \min_{\tilde{\boldsymbol{S}}_{\boldsymbol{B}}} U_m^{(1)}+\left\{\left[w^S(m)\right]^{[t]}\tilde{S}_B(m)+\frac{\rho}{2}\left[\tilde{S}_B(m)-S_B{}^{[t]}\right]^2\right\}. \tag{29}$$

Observing $\mathcal{SP}_{1_L}$, we can find it's a non-convex problem due to the non-convexity of $U_m^{(1)}$, but a nearly-optimal solution can be obtained using the gradient descent method [27].

- Reach the global consensus on the block size:

The updating of global variables can be considered as taking the average of all local copies w.r.t. the global variables to achieve a global consensus.

$$S_B=\frac{1}{M\rho}\sum_{m=1}^{M}\left[w^S(m)\right]^{[t]}+\frac{1}{M}\sum_{m=1}^{M}\left[\tilde{S}_B(m)\right]^{[t+1]}. \tag{30}$$

- Multiplier updating at SBS m:

$$\left\{\boldsymbol{w}^S(m)\right\}^{[t+1]}=\left[\boldsymbol{w}^S(m)\right]^{[t]}+\rho\left(\left[\tilde{\boldsymbol{S}}_{\boldsymbol{B}}(m)\right]^{[t+1]}-S_B^{[t+1]}\right). \tag{31}$$

2) Offloading Scheduling and Resource Allocation:

$$
\begin{aligned}
&\mathcal{P}_3: \max_{\tilde{\boldsymbol{S}}_B, S_B; \boldsymbol{F}^l, \boldsymbol{F}^g} \frac{1}{M} \sum_{m=1}^{M} \sum_{v=1}^{V_m} \eta_{v_m}^b e^{-\frac{\varepsilon \tilde{S}_B(m)}{T_G}} \left[\dot{R} + \frac{\varepsilon \tilde{S}_B(m)}{\chi_{v_m}}\right] - \\
&\varpi_e Q_{v_m} \left\{ \begin{array}{l} \left[\tilde{c}_{v_i}(m) - \tilde{X}_{v_i}(m)\right] \frac{P_S^I}{\mu^{(0)}} + \left(\hat{b}_{v_m}^{(0)} - Y_{v_m}^{(0)}\right) \Lambda_{v_m}^{(0)} \\ + (1-\delta_{v_m}) P_S^I D_{v_m}^{(0,q)} + \sum_{k=1}^{K_{v_m}} \left(Y_{k_{v_m}}^{(1)} \Lambda_{k_{v_m}}^{(1)} + \delta_{v_m} \kappa f_{k_{v_m}}{}^3 D_{k_{v_m}}^{(1,t)}\right) \end{array} \right\} \\
&s.t.\ C_1', C_3'', C_4'' \\
&\quad C_2''': \sum_{m=1}^{M} \sum_{v=1}^{V_m} \frac{1}{\tilde{c}_{v_i}(m)} \le C + N\theta_c, \tilde{c}_{v_i}(m) \le \frac{1}{\theta_c}, \\
&\qquad \left[\tilde{c}_{v_i}(m) - \tilde{X}_{v_i}(m)\right] C + \tilde{c}_{v_i}(m)\theta_c \ge 1, \forall m, i, v \\
&\quad C_5'''': \left[\tilde{c}_{v_i}(m) - \tilde{X}_{v_i}(m)\right] \frac{1}{\mu^{(0)}} + \left(\hat{b}_{v_m}^{(0)} - Y_{v_m}^{(0)}\right) \frac{\Lambda_{v_m}^{(0)}}{P_S^T} + \\
&\qquad (1-\delta_{v_m}) D_{v_m}^{(0,q)} + Y_{k_{v_m}}^{(1)} \frac{\Lambda_{k_{v_m}}^{(1)}}{P_U^T} + \delta_{v_m} D_{k_{v_m}}^{(1,t)} \le \tau_{v_m}, \forall m, i, v, k \\
&\quad C_6': \tilde{S}_B(m) \le \dot{S}, \forall m \\
&\quad C_7': \tilde{X}_{v_i}(m) \in \Xi_{v_m}^c, Y_{v_m}^{(0)} \in \Xi_{v_m}^{(b,0)}, Y_{k_{v_m}}^{(1)} \in \Xi_{k_{v_m}}^{(b,1)}, \forall m, i, v, k \\
&\quad C_8: \tilde{S}_B(m) = S_B, \tilde{c}_{v_i}(m) = \hat{c}_{v_i}, \tilde{X}_{v_i}(m) = X_{v_i}, \forall m, i, v
\end{aligned} \tag{24}
$$

$$
U_m^{(1)} = \left\{ \begin{array}{ll} -\sum_{v=1}^{V_m} \eta_{v_m}^b e^{-\frac{\varepsilon \tilde{S}_B(m)}{T_G}} \left[\dot{R} + \frac{\varepsilon \tilde{S}_B(m)}{\chi_{v_m}}\right], & if\ C_6'\ holds \\ \infty, & otherwise \end{array} \right. \tag{25}
$$

and

$$
U_m^{(2)} = \left\{ \begin{array}{ll} \sum_{v=1}^{V_m} \varpi_e Q_{v_m} \left\{ \begin{array}{l} \left[\tilde{c}_{v_i}(m) - \tilde{X}_{v_i}(m)\right] \frac{P_S^I}{\mu^{(0)}} + \left(\hat{b}_{v_m}^{(0)} - Y_{v_m}^{(0)}\right) \Lambda_{v_m}^{(0)} + (1-\delta_{v_m}) P_S^I D_{v_m}^{(0,q)} \\ + \sum_{k=1}^{K_{v_m}} \left(Y_{k_{v_m}}^{(1)} \Lambda_{k_{v_m}}^{(1)} + \delta_{v_m} \kappa f_{k_{v_m}}{}^3 D_{k_{v_m}}^{(1,t)}\right) \end{array} \right\}, & \\ & if\ C_1', C_2''', C_3'', C_4'', C_5'''', C_7'\ holds \\ \infty, & otherwise \end{array} \right. \tag{26}
$$

$$
\begin{aligned}
&L_\rho\left(\left\{\tilde{\boldsymbol{S}}_B, \boldsymbol{F}^l\right\}, \{S_B, \boldsymbol{F}^g\}, \left\{\boldsymbol{w}^S, \boldsymbol{w}^c, \boldsymbol{w}^X\right\}\right) = \\
&\sum_{m=1}^{M} \left[U_m^{(1)} + U_m^{(2)}\right] + \sum_{m=1}^{M} w^S(m) \left[\tilde{S}_B(m) - S_B\right] + \sum_{m=1}^{M} \sum_{i=1}^{M} \sum_{v=1}^{V_i} w_{v_i}^c(m) \left[\tilde{c}_{v_i}(m) - \hat{c}_{v_i}\right] \\
&+ \sum_{m=1}^{M} \sum_{i=1}^{M} \sum_{v=1}^{V_i} w_{v_i}^X(m) \left[\tilde{X}_{v_i}(m) - X_{v_i}\right] + \frac{\rho}{2} \sum_{m=1}^{M} \left[\tilde{S}_B(m) - S_B\right]^2 \\
&+ \frac{\rho}{2} \sum_{m=1}^{M} \sum_{i=1}^{M} \sum_{v=1}^{V_i} \left[\tilde{c}_{v_i}(m) - \hat{c}_{v_i}\right]^2 + \frac{\rho}{2} \sum_{m=1}^{M} \sum_{i=1}^{M} \sum_{v=1}^{V_i} \left[\tilde{X}_{v_i}(m) - X_{v_i}\right]^2,
\end{aligned} \tag{28}
$$

• Offloading scheduling and resource allocation decision updating at SBS m:

Similarly, each SBS needs to solve the following local problem to obtain the offloading scheduling and resource allocation solution. It's obvious that $\mathcal{SP}_{2_L}$ is a convex problem, so it can be solved efficiently in polynomial time using standard software such as CVX, SeDuMi, etc.

• Reach global consensus on computational resource allocation:

$$
\hat{c}_{v_i} = \frac{1}{M\rho} \sum_{m=1}^{M} \left[w_{v_i}^c(m)\right]^{[t]} + \frac{1}{M} \sum_{m=1}^{M} \left[\tilde{c}_{v_i}(m)\right]^{[t+1]}, \tag{33}
$$

$$
X_{v_i} = \frac{1}{M\rho} \sum_{m=1}^{M} \left[w_{v_i}^X(m)\right]^{[t]} + \frac{1}{M} \sum_{m=1}^{M} \left[\tilde{X}_{v_i}(m)\right]^{[t+1]}. \tag{34}
$$

• Multiplier updating at SBS m:

$$
\{\boldsymbol{w}^c(m)\}^{[t+1]} = [\boldsymbol{w}^c(m)]^{[t]} + \rho\left([\tilde{\boldsymbol{c}}(m)]^{[t+1]} - \hat{\boldsymbol{c}}^{[t+1]}\right), \tag{35}
$$

$$
\left\{\boldsymbol{w}^X(m)\right\}^{[t+1]} = \left[\boldsymbol{w}^X(m)\right]^{[t]} + \rho\left(\left[\tilde{\boldsymbol{X}}(m)\right]^{[t+1]} - \boldsymbol{X}^{[t+1]}\right). \tag{36}
$$

3) Stopping Criterion: In the implementation process, we employ the stopping criteria as proposed in [30]. Specifically, the residuals for both the primal and dual feasibility condition of small cell m in iteration $[t+1]$ should be small enough such that

$$
\left\|\tilde{\boldsymbol{S}}_{\boldsymbol{B}}(m)^{[t+1]} - S_B{}^{[t+1]}\right\|_2 \le \iota_{pri}, \forall m, \tag{37}
$$

$$\begin{aligned}\mathcal{SP}_{2_L}: \min_{\boldsymbol{F}^l} \quad & U_m^{(2)} + \sum_{i=1}^{M}\sum_{v=1}^{V_i}\left\{\left[w_{v_i}^{c}(m)\right]^{[t]}\tilde{c}_{v_i}(m) + \frac{\rho}{2}\left[\tilde{c}_{v_i}(m) - \hat{c}_{v_i}^{[t]}\right]^2\right\} \\ & + \sum_{i=1}^{M}\sum_{v=1}^{V_i}\left\{\left[w_{v_i}^{X}(m)\right]^{[t]}\tilde{X}_{v_i}(m) + \frac{\rho}{2}\left[\tilde{X}_{v_i}(m) - X_{v_i}{}^{[t]}\right]^2\right\}.\end{aligned} \tag{32}$$

$$\left\|\tilde{\boldsymbol{c}}(m)^{[t+1]} - \hat{\boldsymbol{c}}^{[t+1]}\right\|_2 \le \iota_{pri}, \forall m, \tag{38}$$

$$\left\|\tilde{\boldsymbol{X}}(m)^{[t+1]} - \boldsymbol{X}^{[t+1]}\right\|_2 \le \iota_{pri}, \forall m, \tag{39}$$

and

$$\left\|S_B{}^{[t+1]} - S_B{}^{[t]}\right\|_2 \le \iota_{dual}, \tag{40}$$

$$\left\|\hat{\boldsymbol{c}}^{[t+1]} - \hat{\boldsymbol{c}}^{[t]}\right\|_2 \le \iota_{dual}, \tag{41}$$

$$\left\|\boldsymbol{X}^{[t+1]} - \boldsymbol{X}^{[t]}\right\|_2 \le \iota_{dual}, \tag{42}$$

where $\iota_{pri} > 0$ and $\iota_{dual} > 0$ are suitably defined tolerances for the primal and dual feasibility conditions, respectively.

4) Discrete Variables Recovery: Recall that in Section IV-B, we relax the discrete variables $\boldsymbol{\delta}$ and $\boldsymbol{c}$ into continuous ones. Therefore, we need to recover the discrete variables after the proposed algorithm is converged. For simplicity, we recover the discrete variables by the rounding-off method. Based on the previous discussions, the optimal offloading scheduling and resource allocation for video transcoding with adaptive block size can be obtained while achieving the maximum average transcoding reward. Details of the proposed ADMM based algorithm are summarized in **Algorithm** 1.

Algorithm 1 ADMM-based Offloading Scheduling and Resource Allocation Algorithm with Adaptive Block Size

1: **Initialization**
a) Initialize the feasible global solution $(S_B^{(0)}, \{\boldsymbol{c}\}^{(0)}, \{\boldsymbol{X}\}^{(0)})$, and microscales (θ_c, θ_b);
b) Each SBS m determines its initial Lagrange multiplier vector $\boldsymbol{w}^S$, $\boldsymbol{w}^c$ and $\boldsymbol{w}^X$;
c) Each transcoder TC_{v_m} finds a group of nearby D2D nodes within the range of D_{ts} in small cell m.
d) Each transcoder TC_{v_m} calculates the transcoding reward as well as output size, delay and energy consumption w.r.t. *mode 0* and *mode 1* according to Section III.
$t = 0$;
2: **Repeat** To reach the global consensus on
a) *Adaptive Block Size S_B:*
Each SBS m updates the local optimal block size $\{S_B\}^{[t+1]}$ by solving $\mathcal{SP}_{1_L}$ as (29), updates the global variables $\left\{\tilde{\boldsymbol{S}}_{\boldsymbol{B}}\right\}^{[t+1]}$ as (30), and updates the the Lagrange multipliers $\left\{\boldsymbol{w}^S\right\}^{[t+1]}$ as (31).
b) *Offloading Scheduling & Resource Allocation $\{\boldsymbol{\delta}, \boldsymbol{c}, \boldsymbol{b}\}$:*
Each SBS m updates its local offloading scheduling and resource allocation decisions $\left\{\boldsymbol{F}^l\right\}^{[t+1]}$ by solving $\mathcal{SP}_{2_L}$ as (32), updates the global variables $\{\boldsymbol{F}^g\}^{[t+1]}$ as (33)-(34), and updates the the Lagrange multipliers $\left\{\boldsymbol{w}^c, \boldsymbol{w}^X\right\}^{[t+1]}$ as (35)-(36).
$t = t + 1$;
Until stopping criteria (37)-(42) are satisfied.
3: Discrete variables recovery for $\boldsymbol{\delta}$ and $\boldsymbol{c}$.
4: Obtain the solution $\{S_B, \boldsymbol{\delta}, \boldsymbol{c}, \boldsymbol{b}\}^*$.

V. Simulation Results and Discussions

In the simulation part, we set the network coverage radius of the MBS as 1000m while the SBS density and user density are set at $10^{-4}/m^2$ and $10^{-3}/m^2$, respectively. The SBSs' transmit power and idle power are $P_S^T = 0.1W$ and $P_S^I = 0.01W$ while the transmit power of users is $P_U^T = 0.02W$. Besides, the blockchain related parameters are set as follows: block size limit $\dot{S} = 0.5 \sim 2$ *MB*, block generation time $T_G = 10 \sim 20$ *s*, average transaction size $\chi_{v_m} = 200 \sim 500$ *B*, the percentage of block reward that transcoders can get $\eta_{v_m}^b = 1\%$, variable reward factor $\varepsilon = 0.1$, the marginal time needed to reach consensus $\xi = 0.01$ *s/kB*, fixed reward $\dot{R} = 5$ *token*. Other simulation parameters are listed in Table II.

To testify the effectiveness of our proposed algorithm (i.e., ADMM (joint)), we compare the simulation results with that of the centralized scheme and several baseline schemes including *Mode 0*-only scheme (all the transcoders offload the transcoding task to the serving SBSs), *Mode 1*-only scheme (all the transcoders offload the transcoding task to a group of nearby D2D nodes), fixed block size, uniform computing, and uniform spectrum allocation solutions.

A. The Convergence of the Proposed ADMM based Algorithm

First, the convergence performance of the proposed ADMM based algorithm with different values of microscales θ_c and θ_b is presented in Fig. 2. Several interesting observations can be obtained: **1)** In the first 10 iterations, the average transcoding profit of each transcoder in the curves increase quickly and gradually reach a stable state within the first 20 iterations, which verifies the good

TABLE II
SIMULATION PARAMETERS

Parameter	Value
The length of each video segment L_{v_m}	2-10s
The bitrate of different required versions r_{v_m}	200-5000kb/s
The pathloss exponent α	4
The available bandwidth for cellular/D2D networks B_S / B_U	20MHz/10MHZ
The power of noise σ_w^2	-174dBm/Hz
The mean value of Rayleigh fading ς	1
Computation worload/intensity X_{v_m}	18000 CPU cycles/bit
Computation energy efficiency coefficient of the processor's chip in the SBSs/users κ	10^{-26}
The number of CPU cores at SBS m c_{v_m}	10–50
Computational capability of the users $f_{k v_m}^U$	10–100 GHz CPU cycles/s
The unit price of energy ϖ_e	0.1 token/J

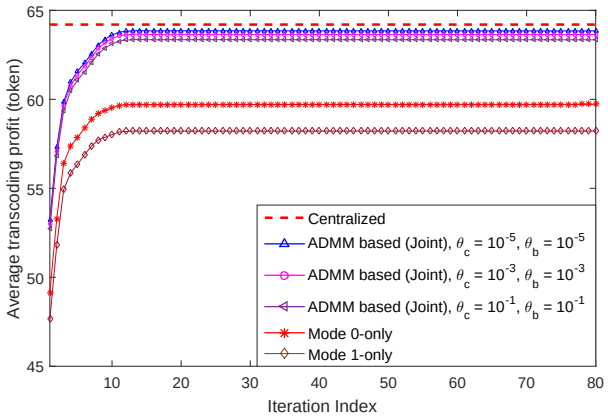
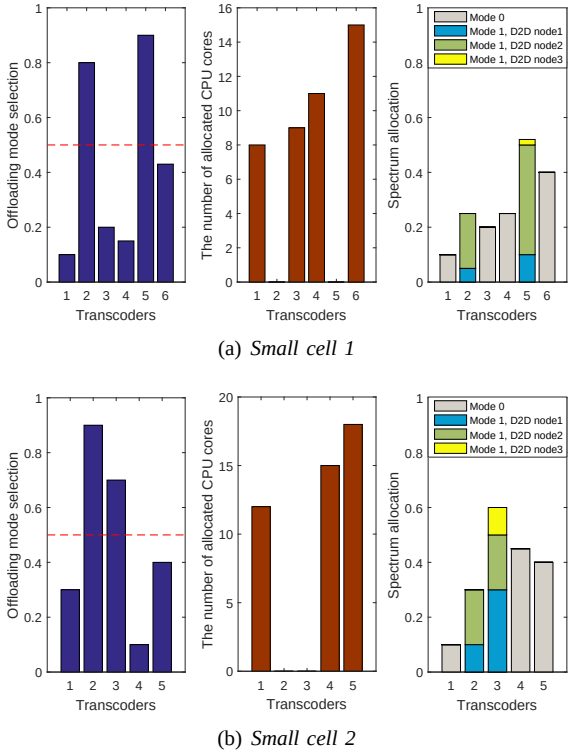


Fig. 2. Convergence performance of the proposed ADMM based algorithm with different microscales θ_c and θ_b.



(a) *Small cell 1*

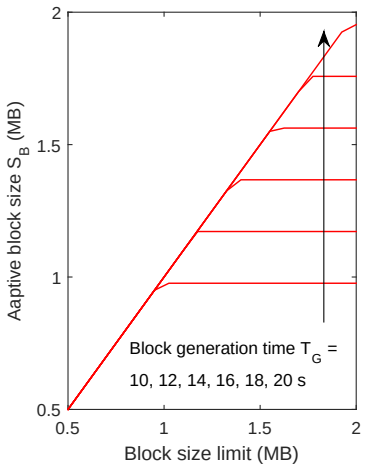
(b) *Small cell 2*

Fig. 3. An illustration of offloading mode selection and resource allocation.

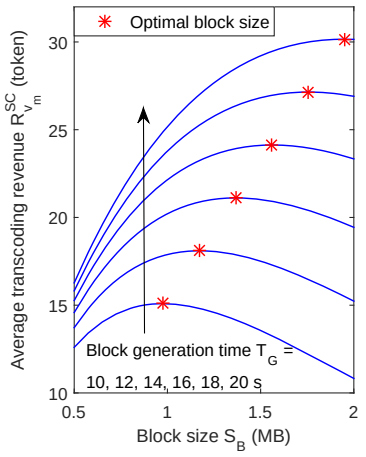
convergence of our proposed ADMM based algorithm. **2)** The gap between the proposed ADMM based algorithm and the centralized one is rather small. Besides, the average profits obtained from the single modes (*mode 0*-only or *mode 1*-only scheme) are much lower. This is because for those two single modes, the computational and spectrum resources are not fully exploited. **3)** The average transcoding profit grows with the decrease of microscales θ_c and θ_b. Note that the growth of the average transcoding profit is very small (< 5%) when θ_c and θ_b decrease from 10^{-3} to 10^{-5}, which demonstrates that our proposed ADMM based algorithm can provide a very tight lower bound solution for the original optimization problem.

B. An Illustration of Resource Allocation Profile

Next, we give an illustration of the offloading mode selection (before discrete variable recovery) and resource allocation scheme w.r.t. *small cell 1* and *small cell 2* obtained from our proposed algorithm. It can be seen that there 6 and 5 transcoders (required versions) in *small cell 1* and *2*, respectively. The left sub-figures in Fig. 3(a) and Fig. 3(b) describe the offloading decision while the middle sub-figures show the computational resource allocation. Furthermore, spectrum allocation in *mode 0* and *mode 1* are illustrated in the right sub-figures. We



(a) *An illustration of the proposed adaptive block size scheme*



(b) *Average transcoding revenue vs. block size S_B (when $\dot{S}$= 2MB)*

Fig. 4. The effect of block size limit $\dot{S}$ and block generation time T_G.

can observe that the number of D2D nodes varies from transcoder to transcoder. For example, there are 2 D2D nodes for transcoder 2 while there are 3 for transcoder 5 in *small cell 1*, which depends on the accessible D2D communication distance. Obviously, the optimal policy is not uniform among different transcoders or small cells, striving for the maximum average transcoding profit.

C. Effects of Different Parameters

In this part, the effects of several parameters including block size limit & block generation time, delay tolerance and available computational resources are investigated and the relevant simulation results are depicted in Fig. 4, Fig. 5-6 and Fig. 7-8 respectively.

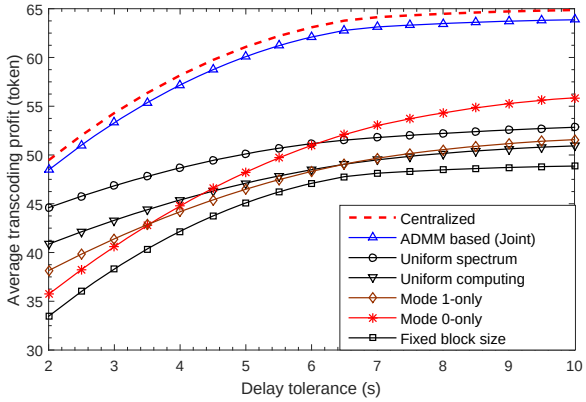
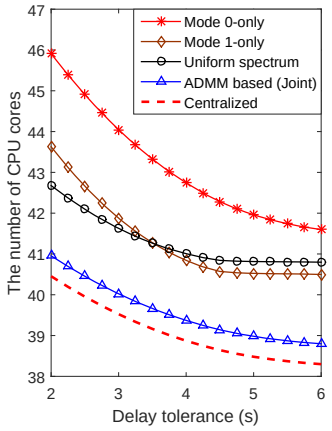


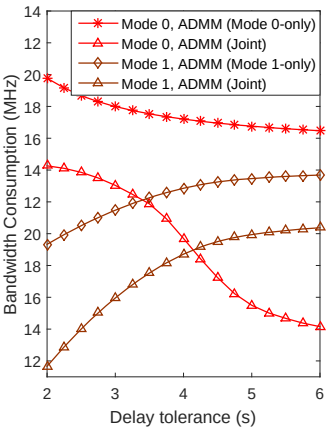
Fig. 5. Average transcoding profit vs. delay tolerance.

1) Effects of Block Size Limit & Block Generation Time: Fig. 4 illustrates our proposed adaptive block size scheme and discusses the effects of block size limit $\dot{S}$ as well as block generation time T_G. Observing Fig. 4(a), we can find that the block size S_B obtained from our proposed scheme first increases and then reaches stable, and the stable point varies with different $\dot{S}$ and T_G, which shows the obtained block size is adaptive to the variation of block size limit and block generation time. The reasons behind can be found in Fig. 4(b). For one thing, larger block size brings higher block reward as well as induces higher orphaning probability, thus the average transcoding revenue first grows and then declines, where the highest revenue is achieved at the optimal block size. Besides, it can be seen that the increase of block generation time contributes to higher average transcoding revenue due to the less chance of block orphaning. However, in practice, large block generation interval would also decrease the transactional rate, which introduces another tradeoff left for future works.

2) Effects of Delay Tolerance: In Fig. 5, the average transcoding profit obtained from our proposed ADMM based algorithm with varying delay tolerance is compared with the centralized algorithm and the other baseline schemes. We can find that the average transcoding profit of our proposed algorithm is very close to that achieved by the centralized algorithm. Besides, all the schemes obtain higher average profit with the increasing delay tolerance. The reason lies in: as the delay constraint becomes more relaxed, more transcoders incline to select *mode 1* which would cause longer delay but can save the energy cost for transcoding, thus higher average transcoding profit can be achieved. Additionally, it's no wonder to find that the solution with uniform computing and uniform spectrum get lower average profits because uniform resource allocation usually cannot reach the optimal policy. Compared with the proposed adaptive block size scheme, the transcoding



(a) *Computational resource consumption*



(b) *Bandwidth consumption*

Fig. 6. Resource consumption vs. delay tolerance.

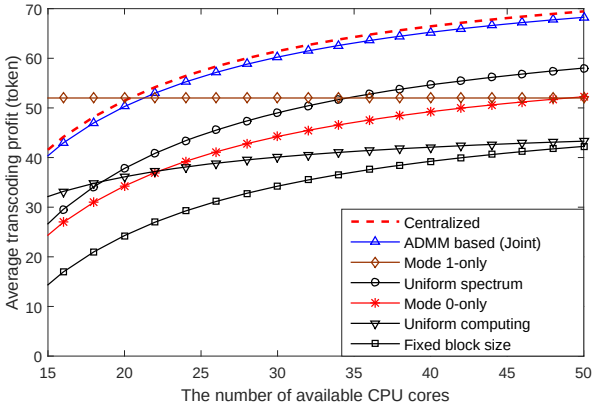


Fig. 7. Average transcoding profit vs. available CPU cores.

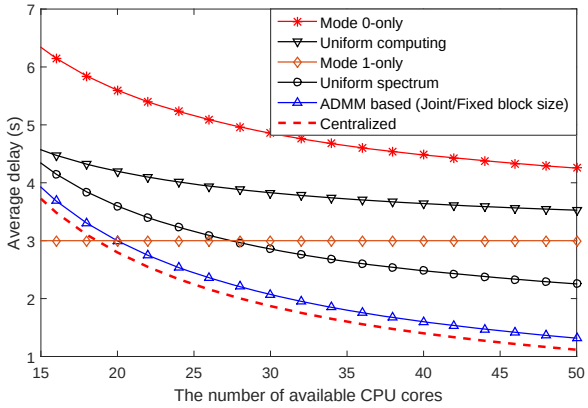


Fig. 8. Average delay vs. available CPU cores.

profit achieved by the fixed block size scheme is much lower because it doesn't get the highest block reward. For the same reason, the computational resource and bandwidth consumption in *mode 0* decrease with the increasing delay tolerance while *mode 1* shows an increase in bandwidth consumption in Fig. 6. Additionally, Fig. 6 also illustrates that the proposed ADMM based algorithm has an advantage over one single mode (*mode 0*-only scheme or *mode 1*-only scheme) in the terms of saving resources.

3) Effects of Available Computational Resources: The effects of available computational resource on average transcoding profit and average delay are investigated in Fig. 7 and Fig. 8, respectively. We can see that all the schemes can achieve higher transcoding profit and takes shorter time to complete the transcoding job with more available CPU cores, except the *mode 1*-only solution that remains unchanged. It is obvious that the gap between our proposed algorithm and the centralized algorithm is not wide. In addition, the reaction of the solution with uniform computing is the least distinct with the increase of available CPU cores among all the schemes.

VI. Conclusion and Future Work

In this paper, we proposed a novel blockchain-based framework for video streaming systems with MEC. We formulated the joint design of offloading scheduling, resource allocation and adaptive block size scheme as an optimization problem to maximize the average transcoding profit for the transcoders. First, we presented the offloading framework with two offloading modes and introduced an incentive mechanism to facilitate the cooperations among content creators, transcoders and consumers. Then the original optimization problem was reformulated using the RLT. Further, a low-complexity ADMM-based algorithm was put forward to solve the problem in a distributed and efficient way. Finally, simulation results verified the effectiveness of the adaptive block size scheme and manifested that the proposed ADMM-based algorithm can provide a tight

lower bound for the original problem and outperforms other baseline schemes. In our future works, we will study the transcoders selection scheme considering a wide range of factors, such as the held stake, reputation value, communication and computing capability. Another interesting direction is to regard the smart contract on the blockchain as the ADMM coordinator, which is an effective way to enable distributed optimization among non-trusting nodes.

REFERENCES

[1] C. Inc., "Cisco visual networking index: Global mobile data traffic forecast update, 2016-2021 white paper," Tech. Rep., Feb. 2017.

[2] SLIVER.tv, "Theta: A decentralized video streaming network, powered by a new blockchain and token v1.6," *Available: https://s3.us-east-2.amazonaws.com/assets.thetatoken.org/Theta+white+paper+11.07.17.pdf*, Jan. 2018.

[3] V. Inc., "Viewly: Whitepaper," *Available: https: //view.ly /downloads /whitepaper. pdf*, Jan. 2018.

[4] Virtu.media, "Virtutv/token platform whitepaper v1.0," *Available: https://virtu.media/whitepaper.pdf*, Jan. 2018.

[5] D. Inc., "White papers for video streaming and hosting," *Available: https://www.dacast.com/white-papers-for-video-streaming/*, Oct. 2017.

[6] Flixxo, "Flixxo: Community based video distribution white paper v0.5," *Available: https://www.flixxo.com*, Oct. 2017.

[7] D. Petkanic and E. Tang, "Livepeer whitepaper: Protocol and economic incentives for a decentralized live video streaming network," *Available: https://github.com/livepeer/wiki/blob/master/WHITEPAPER.md*, Dec. 2017.

[8] C. Wright and A. Serguieva, "Blockchain-based payment collection supervision system using pervasive bitcoin digital wallet," in *Proc. IEEE Int'l Conf. on Big Data*. Boston, MA, Dec. 2017, pp. 4255–4264.

[9] D. Magazzeni, P. McBurney, and W. Nash, "Validation and verification of smart contracts: A research agenda," *Computer*, vol. 50, no. 9, pp. 50–57, Spet. 2017.

[10] Q. He, C. Zhang, X. Ma, and J. Liu, "Fog-based transcoding for crowdsourced video livecast," *IEEE Comm. Mag.*, vol. 55, no. 4, pp. 28–33, April 2017.

[11] L. Wei, J. Cai, C. H. Foh, and B. He, "Qos-aware resource allocation for video transcoding in clouds," *IEEE Trans. Circ. Sys. Video Tech.*, vol. 27, no. 1, pp. 49–61, Jan. 2017.

[12] G. Y. Gao, Y. Wen, and C. Westphal, "Resource provisioning and profit maximization for transcoding in information centric networking," in *Proc. IEEE Conf. on Computer Commun. Workshops (INFOCOM WKSHPS)*. San Francisco, CA, April 2016, pp. 97–102.

[13] S. Andrew, "An examination of single-transaction blocks and their effect on network throughput and block size," *Available: https://www.bitcoinunlimited.info/resources/1txn.pdf*, 2015.

[14] P. R. Rizun, "A transaction fee market exists without a block size limit," *BLOCK SIZE LIMIT DEBATE WORKING PAPER*, Aug. 2015.

[15] B. Group, "Block size increase v1.1," *Available: http://bitfury.com/content/5-white-papers-research/block-size-1.1.1.pdf*, Sept. 2015.

[16] W. T. Tsai, X. Bai, and L. Yu, "Design issues in permissioned blockchains for trusted computin," in *Proc. IEEE Symposium on Service-Oriented System Engineering (SOSE)*. San Francisco, CA, April 2017, pp. 153–15.

[17] M. T. Liu, F. R. Yu, Y. L. Teng, V. C. M. Leung, and M. Song, "Joint computation offloading and content caching for wireless blockchain networks," in *Proc. IEEE Conf. on Computer Commun. Workshops (INFOCOM WKSHPS)*. Honolulu, HI, April 2018, pp. 1–6.

[18] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Joint computation offloading, resource allocation and content caching in cellular networks with mobile edge computing," in *Proc. IEEE International Conf. on Commun. (ICC)*. Paris, France, May 2017, pp. 1–6.

[19] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, Sept. 2017.

[20] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1784–1797, Mar. 2018.

[21] Y. Zhou, F. R. Yu, J. Chen, and Y. Kuo, "Video transcoding, caching, and multicast for heterogeneous networks over wireless network virtualization," *IEEE Commun. Letters*, vol. 22, no. 1, pp. 141–144, Jan. 2018.

[22] J. G. Andrews, F. Baccelli, and R. K. Ganti, "A tractable approach to coverage and rate in cellular networks," *IEEE Trans. Commun.*, vol. 59, no. 11, pp. 3122–3134, Nov. 2011.

[23] P. Zhao, H. Tian, C. Qin, and G. Nie, "Energy-saving offloading by jointly allocating radio and computational resources for mobile edge computing," *IEEE Access*, vol. 5, pp. 11 255–11 268, June 2017.

[24] L. Muche, "Contact and chord length distribution functions of the poisson-voronoi tessellation in high dimensions," *Advances in Applied Probability*, vol. 42, no. 1, pp. 48–68, Oct. 2010.

[25] Z. H. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, "Edge computing resource management and pricing for mobile blockchain," *[online] arXiv:1710.01567v1 [cs.CR]*, 2017.

[26] N. Houy, "The bitcoin mining game," *Available: https://halshs.archives-ouvertes.fr/file/index/docid/958224/filename/1412.pdf*, Mar. 2014.

[27] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[28] E. Dalkiran and H. D. Sherali, "Rlt-pos: Reformulation-linearization technique-based optimization software for solving polynomial programming problems," *Mathematical Programming Computation*, vol. 8, no. 3, pp. 337–375, Feb. 2016.

[29] H. Tabrizi, B. Peleato, G. Farhadi, J. M. Cioffi, and G. Aldabbagh, "Spatial reuse in dense wireless areas: A cross-layer optimization approach via admm," *IEEE Trans. Wireless Commun.*, vol. 14, no. 12, pp. 7083–7095, Dec. 2015.

[30] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.