Joint Computation Offloading and Content Caching for Wireless Blockchain Networks

Mengting Liu*, F. Richard Yu†, Yinglei Teng*, Victor C. M. Leung‡, and Mei Song*
*Beijing Key Laboratory of Space-ground Interconnection and Convergence,
Beijing University of Posts and Telecommunications, Beijing, 100876, China
†Department of Systems and Computer Engineering, Carleton University, Ottawa, ON K1S 5B6, Canada
‡Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC V6T 1Z4, Canada

***Abstract*—Blockchain is widely deemed as a key enabling technology of current digital currency. However, the application of blockchain to wireless mobile networks is challenged by a computational difficult problem called proof-of-work puzzle. Accordingly, mobile edge computing (MEC) appears to be a promising solution by offloading the computation-intensive mining tasks to nearby edge computing nodes. Meanwhile, caching is also popular due to its capability in handling the ever-increasing Internet traffic. In this paper, we study computation offloading and content caching in wireless blockchain networks with MEC. Specifically, offloading mode selection (offloaded to a nearby access point (AP) or a group of device-to-device (D2D) users) and caching strategy (whether to cache the requested content and computation results or not) are jointly investigated and formulated as an optimization problem. Further, an alternating direction method of multipliers (ADMM) based algorithm is proposed to solve the optimization problem in a distributed way. Finally, the effectiveness of the proposed algorithm is demonstrated by simulation results with different system parameters.**

I. Introduction

With the rapid development of digital currency, *Bitcoin*, as a peer-to-peer (P2P) electronic cash system, has received great attention in economics, cryptography, and computer science [1]. Compared with the traditional centralized financial institutions, Bitcoin aims at achieving a complete decentralization through a public append-only ledger called *Blockchain* [2] [3]. However, blockchain has not been widely applied in wireless mobile network due to a main challenge brought by the computation-intensive mining task, i.e., the proof-of-work puzzle, which sets an extremely high demand for the computational capabilities and storage availability of the miners. To address this issue, mobile edge computing (MEC) emerges as an enabling technology that brings computing resources closer to the end users, which can significantly educe energy consumption and shorten transmission delay [4].

Recently, there have been several works focusing on mining schemes management for blockchain networks and computation offloading schemes for MEC systems. In blockchain networks, a non-cooperative game among the miners is proposed in [5], where the miner's strategy is to choose the number of transactions to be included in a block. Focusing on whether to propagate the mined block or not, the authors of [6] and [7] put forward a sequential game and a stochastic game to model the mining process, respectively. Besides, a cooperative game based blockchain mining scheme is studied to explore the optimal pool mining mechanism in [8]. And the authors of [9] adopt a new computational power splitting game to model the blockchain mining, where the miners can get the mining rewards by solving the proof-of-work puzzle.

In MEC networks, considering partial offloading, the optimal offloading ratio, computational speed and transmit power are jointly investigated in [10]. For binary offloading, various offloading scheduling schemes are discussed, such as the optimal executing order of offloading tasks in [11], the optimal amount of offloading tasks in [12], the optimal CPU time allocation in [13], and the best MEC server selection in [14]. Meanwhile, there are also several papers focusing on mode selection. [15] optimizes the individual computing mode selection (local computing or offloading) while [16] develops a distributed activation mechanism for the servers to decide whether to accept the computation tasks or not. Besides, the servers in MEC systems can realize the in-network caching function, which is able to significantly alleviate the backhaul link pressure [17].

Nevertheless, none of the above works investigate the mobile blockchain together with computation offloading and content caching in MEC systems. Therefore, we consider offloading scheduling and content caching strategy in wireless blockchain networks with MEC in this paper. The distinct features of this paper are as follows.

1) To avoid the overload of MEC service providers, two offloading modes, i.e., offloaded to a nearby access point (AP) *(mode 0)* or a group of device-to-device (D2D) users *(mode 1)*, are served as two options for computation offloading.

2) Leveraging stochastic geometry methods, the theoretical expressions of performance metrics (delay and energy consumption) are derived for each mode. Accordingly, the users only need to acquire the local channel state information (CSI), which can significantly reduce the signaling overhead.

3) Offloading mode selection (choosing *mode 0* or *mode 1* and task offloading distribution in *mode 1*) and caching strategy (whether to cache the user's requested content and computation results or not) are jointly investigated, which is formulated as an optimization problem. Further, we propose an alternating direction method of multipliers (ADMM) based algorithm to solve the problem in a distributed way.

978-1-5386-5979-3/18/$31.00 ©2018 IEEE

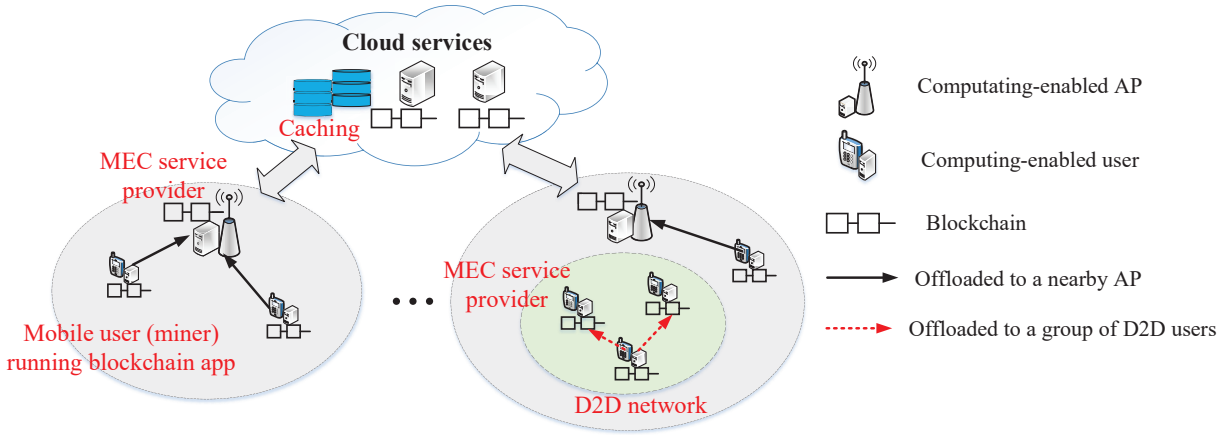

Fig. 1. An illustration of MEC enabled wireless blockchain networks.

The rest of this paper is organized as follows. The system model is presented in Section II. We conduct performance analysis in Section III. In Section IV, problem formulation and solutions are specified. Simulation results are discussed in Section V. Finally, section VI concludes the paper.

II. System Model

In this section, we present the related system model adopted in this paper, which includes application model, computation offloading model, network model and caching model.

A. Application Model

As shown in Fig. 1, we consider an MEC enabled wireless blockchain networks where M APs and N users follow two independent homogeneous Poisson point processes (HPPPs) $\Phi_a = \{AP_1, \ldots, AP_M\}$ and $\Phi_u = \{x_1, \ldots, x_N\}$ with density λ_a and λ_u, respectively. Every user is assumed to access to its nearest AP such that there are N_m users in small cell m. We denote the set of users associated with AP_m as $\Phi_u^{(m)} = \{x_{\langle m,n\rangle}\}, x_{\langle m,n\rangle} \in \Phi_u, n = 1, \ldots, N_m$. For $x_{\langle m,n\rangle}$, there are $K_{m,n}$ D2D links with a set of users $\Phi_u^{(m,n)} = \{x_{\langle m,n,k\rangle}\}, x_{\langle m,n,k\rangle} \in \Phi_u, k = 1, \ldots, K_{m,n}$. And we adopt a parameter tuple $\langle D_{m,n}, \tau_{m,n}, X_{m,n}\rangle$ to character the mining task $A_{\langle m,n\rangle}$ with input data size $D_{m,n}$ (in bits), completion deadline $\tau_{m,n}$ (in second) and computation workload/intensity $X_{m,n}$ (in CPU circles/bit). The computational capability (CPU-cycle cycles/s) of AP_m and $x_{\langle m,n,k\rangle}$ are denoted by f_m^A and $f_{m,n,k}^U$, respectively.

B. Computation Offloading Model

In this paper, we consider two offloading modes where the computation-intensive task at the miner $x_{\langle m,n\rangle}$ is:

1) ***offloaded to a nearby AP (mode 0)***: The miner $x_{\langle m,n\rangle}$ offloads the full task $A_{\langle m,n\rangle}$ to its associated AP AP_m.

2) ***offloaded to a group of D2D users (mode 1)***:

The miner $x_{\langle m,n\rangle}$ divides $A_{\langle m,n\rangle}$ into $K_{m,n}$ parts, i.e., $\{A_{\langle m,n,1\rangle}, A_{\langle m,n,2\rangle}, \ldots, A_{\langle m,n,K_{m,n}\rangle}\}$, and separately forwards them to its neighboring users through D2D links.

We denote $\delta(n_m) \in \{0,1\}, \forall m,n$ as the offloading decision of $x_{\langle m,n\rangle}$. Specifically, we have $\delta(n_m) = 0$ if $x_{\langle m,n\rangle}$ selects *mode 0* and $\delta(n_m) = 1$ if $x_{\langle m,n\rangle}$ chooses *mode 1*. And in *mode 1*, $\rho(n_m, k_{m,n}) \in [0,1], \forall m,n,k$ is defined as the ratio of offloading part $A_{\langle m,n,k_{m,n}\rangle}$ to the whole task $A_{\langle m,n\rangle}$.

C. Network Model

The channel radio propagation is assumed to comprise path loss and Rayleigh fading. Specifically, the path loss fading between $x_{\langle m,n\rangle}$ and AP_m ($x_{\langle m,n,k\rangle}$) is ${r_{m,n}}^{-\alpha}$ (${l_{m,n,k}}^{-\alpha}$) where $r_{m,n}$ ($l_{m,n,k}$) denotes the distance between $x_{\langle m,n\rangle}$ and AP_m ($x_{\langle m,n,k\rangle}$) and α is the path loss exponent. Meanwhile, Rayleigh fading between $x_{\langle m,n\rangle}$ and AP_m ($x_{\langle m,n,k\rangle}$) is represented by $h_{m,n}$ ($g_{m,n,k}$) with $h_{m,n} \sim \exp(\mu)$ ($g_{m,n,k} \sim \exp(\mu)$) where μ is the corresponding scale parameter. Assume each user transmits at the identical power P_U on the same channel with bandwidth B, and the noise power is σ^2.

D. Caching Model

We denote $s(n_m) \in \{0,1\}, \forall m,n$ as the caching decision for $x_{\langle m,n\rangle}$. Specifically, $s(n_m)=1$ if the MEC server manager decides to cache the requested content and computation results of $x_{\langle m,n\rangle}$ while $s(n_m) = 0$ otherwise. In this paper, we simply consider the reward (alleviated backhaul bandwidth and saved energy) of caching by $\nu(n_m) = \vartheta_c \dot{q}_{m,n} \bar{R} + \vartheta_e \bar{E}$, where $\vartheta_c (bps^{-1})$ is the unit price to lease the backhaul, $\dot{q}_{m,n}$ is the request rate of the content initially requested by $x_{\langle m,n\rangle}$, $\vartheta_e (J^{-1})$ is the unit price of energy, and $\bar{R}$ and $\bar{E}$ are the average user data rate and energy consumption, respectively.

III. Performance Analysis

In this part, we derive the performance metrics including delay and energy consumption w.r.t. *mode 0* and *mode 1*.

A. Offloaded to a Nearby AP (Mode 0)

1) Delay: The total time for completing the task includes the time spent on uploading, computing and queueing [10] as

$$T^{(A)}(n_m) = T^{(A,u)}(n_m) + T^{(A,e)}(n_m) + T^{(A,q)}(n_m), \quad (1)$$

where the separate parts in (1) are derived as follows.

- ***Uploading delay***

The time consumed for $x_{\langle m,n\rangle}$ to upload the mining task to AP_m is $T^{(A,u)}(n_m) = \frac{D_{m,n}}{R^A(n_m)}$ with upload transmission rate $R^A(n_m) \approx B\ln\left(1+\frac{P_U h_{m,n} {r_{m,n}}^{-\alpha}}{\sigma^2+\mathbb{E}(I_m)-P_U h_{m,n} {r_{m,n}}^{-\alpha}}\right)$, where $I_m = \sum_{x_i\in\Phi_u} P_U h_{i,m} {r_{i,m}}^{-\alpha}$ and $\mathbb{E}(I_m)$ is given as follows.

***Proposition** 1:* Along the same lines in [18], the expectation of the aggregate reference I_m is derived as

$$\mathbb{E}(I_m) \approx \int_{0<\hat{r}_1<\ldots\hat{r}_\beta<\infty} \int_{\xi>0} \exp\left\{-\frac{\mu\xi}{P_U\left[\sum_{i=1}^{\beta}\hat{r}_i^{-\alpha}+\sum_{c=\beta+1}^{N}(\pi\lambda_u)^{\frac{\alpha}{2}}\frac{\Gamma\left(c-\frac{\alpha}{2}\right)}{\Gamma(c)}\right]}\right\} \times f(\hat{r}_1,\ldots,\hat{r}_\beta)\, d\xi d\hat{r}_1\ldots d\hat{r}_\beta, \quad (2)$$

where $\beta = \lfloor \alpha/2 \rfloor$ ($\lfloor\bullet\rfloor$ is the floor function), $\Gamma(\bullet)$ is the gamma function, and $\{r_{m,1},\ldots,r_{m,N}\}$ is rearranged as $\{\hat{r}_1,\ldots,\hat{r}_N\}$ in ascending order with joint distance distribution $f(\hat{r}_1,\ldots,\hat{r}_N) = e^{-\lambda\pi\hat{r}_N^2}(2\lambda_u\pi)^N \hat{r}_1\ldots\hat{r}_N$.

- ***Executing time***

The execution time of the case when $x_{\langle m,n\rangle}$ chooses *mode 0* is calculated by $T^{(A,e)}(n_m) = \frac{D_{m,n}X_{m,n}}{f_m^A}$.

- ***Queuing delay***

Assuming the number of CPU cycles to be processed in the task buffer at AP_m as Q_m, the queuing delay for $x_{\langle m,n\rangle}$ is given by $T^{(A,q)}(n_m) = \frac{Q_m}{f_m^A}$.

2) Energy Consumption: The total energy consumption for task uploading, computing and CPU operation is

$$E^{(A)}(n_m) = P_U T^{(A,u)}(n_m) + \kappa_m^A \left(f_m^A\right)^3 T^{(A,e)}(n_m) + P_C T^{(A)}(n_m), \quad (3)$$

where κ_m^A is the computation energy efficiency coefficient of the processor's chip in AP_m, and P_C is the static circuit power.

B. Offloaded to a Group of D2D Users (Mode 1)

1) Delay: Different from *mode 0*, it's not until the all the single parts are offloaded and computed that the computation task is completely finished in *mode 1*, so the total delay is

$$T^{(D)}(n_m) = \max_{k=1,\ldots K_{m,n}} \rho(n_m,k_{mn})\left[T^{(D,u)}(n_m,k_{m,n}) + T^{(D,e)}(n_m,k_{m,n})\right], \quad (4)$$

- ***Uploading delay***

The time consumed for $x_{\langle m,n\rangle}$ to upload $A_{\langle m,n,K_{m,n}\rangle}$ to $x_{\langle m,n,k\rangle}$ is calculated by $T^{(D,u)}(n_m,k_{m,n}) = \frac{D_{m,n}}{R^D(n_m,k_{m,n})}$ where $R^D(n_m,k_{m,n}) \approx \ln\left(1+\frac{\frac{P_U}{K_{m,n}} g_{m,n,k} {l_{m,n,k}}^{-\alpha}}{\sigma^2+\mathbb{E}(I_m)-\frac{P_U}{K_{m,n}} g_{m,n,k} {l_{m,n,k}}^{-\alpha}}\right)$ with transmit power equally allocated among the D2D users.

- ***Executing time***

The executing time cost when $x_{\langle m,n\rangle}$ chooses *mode 1* to compute $A_{\langle m,n,K_{m,n}\rangle}$ is $T^{(D,e)}(n_m,k_{m,n}) = \frac{D_{m,n}X_{m,n}}{f^U_{m,n,k}}$.

2) Energy Consumption: The total energy consumption for $\left\{x_{\langle m,n,1\rangle},\ldots,x_{\langle m,n,K_{m,n}\rangle}\right\}$ to complete $A_{\langle m,n\rangle}$ is given by

$$\begin{aligned} &E^{(D)}(n_m) \\ &= \sum_{k=1}^{K_{m,n}} \rho(n_m,k_{m,n})\, E^{(D)}(n_m,k_{m,n}) \\ &= \sum_{k=1}^{K_{m,n}} \rho(n_m,k_{m,n}) \left[\begin{array}{l} \frac{P_U}{K_{m,n}} T^{(D,u)}(n_m,k_{m,n}) + \\ \kappa^U_{m,n,k}\left(f^U_{m,n,k}\right)^3 T^{(D,e)}(n_m,k_{m,n}) \end{array}\right], \end{aligned} \quad (5)$$

where $\kappa^U_{m,n,k}$ is the energy efficiency coefficient of $x_{\langle m,n,k\rangle}$.

IV. Problem Formulation and Solutions

In this section, we study the optimal computation offloading scheduling and caching policy to maximize the total net revenue in terms of offloading and caching.

A. Optimization Objective

Considering the offloading decision, the net revenue $\Psi(n_m)$ for accomplishing the mining task $A_{\langle m,n\rangle}$ is given by $\Psi(n_m) = [1-\delta(n_m)]\left[\varsigma^{(A)} - \vartheta_e E^{(A)}(n_m)\right] + \delta(n_m)\left[\varsigma^{(D)} - \vartheta_e E^{(D)}(n_m)\right]$ where $\varsigma^{(A)}$ and $\varsigma^{(D)}$ are the rewards for *mode 0* and *mode 1*, respectively. Regarding content caching, the net revenue for caching is calculated by $\Lambda(n_m) = s(n_m)\left(\vartheta_c \dot{q}_{m,n}\bar{R} + \vartheta_e \bar{E} - \varpi\right)$, where ϖ represents the memory cost for caching. Therefore, the total net revenue is $\Xi = \sum_{m=1}^{M}\sum_{n=1}^{N_m}[\Psi(n_m)+\Lambda(n_m)]$.

B. Problem Formulation

We now formulate the total optimization problem with offloading scheduling and caching decision $(\boldsymbol{\delta},\boldsymbol{\rho},\boldsymbol{s})$ as follows:

$$\begin{aligned} \mathcal{P}1:\ &\max_{\boldsymbol{\delta},\boldsymbol{\rho},\boldsymbol{s}} \quad \sum_{m=1}^{M}\sum_{n=1}^{N_m}[\Psi(n_m)+\Lambda(n_m)] \\ s.t.\ &C1: [1-\delta(n_m)]+\delta(n_m)\sum_{k=1}^{K_{m,n}}\rho(n_m,k_{m,n}) = 1, \forall m,n \\ &C2: \sum_{n=1}^{N_m}[1-\delta(n_m)]R^A(n_m) \le \Omega_m, \forall m \\ &C3: [1-\delta(n_m)]T^{(A)}(n_m) + \\ &\quad \delta(n_m)\rho(n_m,k_{m,n})T^{(D)}(n_m,k_{m,n}) \le \tau_{m,n}, \forall m,n,k \\ &C4: \delta(n_m)\rho(n_m,k_{m,n})D_{n,m} \le L_{m,n,k}, \forall m,n,k \\ &C5: \sum_{m=1}^{M}\sum_{n=1}^{N_m} s(n_m)\chi_{m,n} \le C \end{aligned} \quad (6)$$

Constraints $C1$ guarantee that the offloading decision is valid. Constraints $C2$ ensure that the sum data rate of all the miners associated with AP_m doesn't exceed its backhaul capacity Ω_m. In order to meet the delay requirement, $C3$ are put forward to guarantee the task delay doesn't exceed the task deadline. Constraints $C4$ mean the data size of each offloaded part through D2D link can not exceed the link capability $L_{m,n,k}$. Besides, we use constraints $C5$ to ensure that the sum size of the cached content doesn't exceed the total storage capability C. And $\chi_{m,n}$ represents the size of caching content, which is set at 1 for convenience.

C. Problem Transformation

Observing $\mathcal{P}1$, we can find $\mathcal{P}1$ is intractable due to the non-convex feasible set, non-convex objective and high computational complexity. So we make the following transformation:

1) Binary Variable Relaxation: The binary variables $\boldsymbol{\delta}$ and $\boldsymbol{s}$ are relaxed as $0 \leq \delta(n_m) \leq 1$ and $0 \leq s(n_m) \leq 1$.

2) Substitution of the Product Term: To address the convexity brought by the product relationships between $\{\delta(n_m)\}$ and $\{\rho(n_m, k_{m,n})\}$, we define $\tilde{\rho}(n_m, k_{m,n}) = \delta(n_m)\rho(n_m, k_{m,n}), \forall m, n, k$.

Then $\mathcal{P}1$ can be transformed into the following problem:

$$\begin{aligned}
\mathcal{P}2: \quad & \max_{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\boldsymbol{s}} \quad \sum_{m=1}^{M}\sum_{n=1}^{N_m}\left[\tilde{\Psi}(n_m)+\Lambda(n_m)\right] \\
s.t.\ & C2, C5 \\
& C1': [1-\delta(n_m)]+\sum_{k=1}^{K_{m,n}}\tilde{\rho}(n_m,k_{m,n})=1, \forall m,n \\
& C3': [1-\delta(n_m)]T^{(A)}(n_m)+ \\
& \quad \tilde{\rho}(n_m,k_{m,n})T^{(D)}(n_m,k_{m,n}) \leq \tau_{m,n}, \forall m,n,k \\
& C4': \tilde{\rho}(n_m,k_{m,n})D_{n,m} \leq L_{m,n,k}, \forall m,n,k \\
& C6: \delta(n_m) \geq \tilde{\rho}(n_m,k_{m,n}), \forall m,n,k
\end{aligned} \tag{7}$$

where $\tilde{\Psi}(n_m) = [1-\delta(n_m)]\left[\varsigma^{(A)} - \vartheta_e E^{(A)}(n_m)\right] + \sum_{k=1}^{K_{m,n}}\tilde{\rho}(n_m,k_{m,n})\left[\varsigma^{(D)} - \vartheta_e E^{(D)}(n_m,k_{m,n})\right]$.

After the above transformation, $\mathcal{P}2$ becomes a convex optimization problem, whereas, it still occurs significant computational complexity. Thus, we adopt a distributed optimization problem solving method called ADMM to solve it.

D. Problem Decomposition

It can be observed from $\mathcal{P}2$ that $\boldsymbol{s}$ is a global variable that makes the optimization problem not separable. Accordingly, we introduce a local copy of the global variable $\boldsymbol{s}$. We first denote $\widehat{\boldsymbol{s}}_m = \left\{\widehat{s}_m(n_i)\right\}, n = 1,\ldots,N_i, m, i = 1,\ldots,M$ as the local copy of $\boldsymbol{s}$. So we have $\widehat{s}_m(n_i) = s(n_i), \forall m,n,i$. Then the equivalent global consensus version of $\mathcal{P}2$ is expressed by

$$\begin{aligned}
\mathcal{P}3: \quad & \max_{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m,\boldsymbol{s}} \quad \sum_{m=1}^{M}\sum_{n=1}^{N_m}\left[\tilde{\Psi}(n_m)+\widehat{\Lambda}(n_m)\right] \\
s.t.\ & C1', C2, C3', C4', C6 \\
& C5': \sum_{i=1}^{M}\sum_{n=1}^{N_i}\widehat{s}_m(n_i)\chi_{i,n} \leq C, \forall m \\
& C7: \widehat{s}_m(n_i) = s(n_i), \forall m,n,i
\end{aligned} \tag{8}$$

where $\widehat{\Lambda}(n_m) = \widehat{s}_m(n_m)\left(\vartheta_c \dot{q}_{m,n}\bar{R} + \vartheta_e \bar{E} - \varpi\right)$.

For convenience, let Π_m be the feasible set of AP_m, i.e., $\Pi_m = \left\{\boldsymbol{\delta}_m, \tilde{\boldsymbol{\rho}}_m, \widehat{\boldsymbol{s}}_m \,|C1', C2, C3', C4', C5', C6\right\}$. Besides, we give the local utility function of each small cell by (9).

$$\Upsilon_m = \begin{cases} -\sum_{n=1}^{N_m}\left[\tilde{\Psi}(n_m)+\widehat{\Lambda}(n_m)\right], when\left\{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m\right\} \in \Pi_m \\ +\infty, \qquad otherwise \end{cases} \tag{9}$$

E. Problem Solutions Through ADMM

1) ADMM Sequential Iterations: According to [19], the augmented Lagrangian of $\mathcal{P}3$ is

$$\begin{aligned}
& \Gamma_q\left(\left\{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m\right\},\{\boldsymbol{s}\},\{\boldsymbol{\eta}_m\}\right) \\
& = \sum_{m=1}^{M}\Upsilon_m\left(\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m\right) + \sum_{m=1}^{M}\sum_{i=1}^{M}\sum_{n=1}^{N_i}\eta_m(n_i)\left[\widehat{s}_m(n_i) - s(n_i)\right] \\
& + \frac{q}{2}\sum_{m=1}^{M}\sum_{i=1}^{M}\sum_{n=1}^{N_i}\left[\widehat{s}_m(n_i) - s(n_i)\right]^2,
\end{aligned} \tag{10}$$

where $\boldsymbol{\eta}_m = \{\eta_m(n_i)\}$ are the Lagrange multipliers, and q is the penalty parameter. Denoting the local variables as $\boldsymbol{X}_m$, the whole iteration process includes the following three steps.

$$\{\boldsymbol{X}_m\}^{(t+1)} = \arg\min_{\{\boldsymbol{X}_m\}} \left\{ \begin{array}{l} \Upsilon_m(\boldsymbol{X}_m) + \\ \sum_{i=1}^{M}\sum_{n=1}^{N_i}[\eta_m(n_i)]^{(t)}\left\{\widehat{s}_m(n_i) - [s(n_i)]^{(t)}\right\} \\ + \frac{q}{2}\sum_{i=1}^{M}\sum_{n=1}^{N_i}\left\{\widehat{s}_m(n_i) - [s(n_i)]^{(t)}\right\}^2 \end{array} \right\}, \tag{11}$$

$$\{\boldsymbol{s}\}^{(t+1)} = \arg\min_{\{s(n_i)\}} \left\{ \begin{array}{l} \sum_{m=1}^{M}\sum_{i=1}^{M}\sum_{n=1}^{N_i}[\eta_m(n_i)]^{(t)}\left\{\left[\widehat{s}_m(n_i)\right]^{(t+1)} - s(n_i)\right\} \\ + \frac{q}{2}\sum_{m=1}^{M}\sum_{i=1}^{M}\sum_{n=1}^{N_i}\left\{\left[\widehat{s}_m(n_i)\right]^{(t+1)} - s(n_i)\right\}^2 \end{array} \right\}, \tag{12}$$

$$\{\boldsymbol{\eta}_m\}^{(t+1)} = \{\boldsymbol{\eta}_m\}^{(t)} + q\left\{\left[\widehat{s}_m(n_i)\right]^{(t+1)} - [s(n_i)]^{(t+1)}\right\}, \tag{13}$$

where the superscript $(t+1)$ is the iteration counter.

2) Local Variables Update: To conclude, AP_m needs to solve the following optimization problem at iteration $(t+1)$.

$$\begin{aligned}
\mathcal{P}_L: \quad & \min_{\{X_m\}} \Upsilon_m(\boldsymbol{X}_m) + \\
& \sum_{i=1}^{M}\sum_{n=1}^{N_i}\left\{[\eta_m(n_i)]^{(t)}\widehat{s}_m(n_i) + \frac{q}{2}\left\{\widehat{s}_m(n_i) - [s(n_i)]^{(t)}\right\}^2\right\} \\
& s.t. \left\{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m\right\} \in \Pi_m.
\end{aligned} \tag{14}$$

Then $\mathcal{P}_L$ can be further decoupled into two sub-problems, i.e., $\mathcal{P}_{L1}$ and $\mathcal{P}_{L2}$.

$$\begin{aligned}
\mathcal{P}_{L1}: \quad & \min_{\{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}}\}} -\sum_{n=1}^{N_m}\tilde{\Psi}(n_m) \\
& s.t. \quad C1', C2, C3', C4', C6.
\end{aligned} \tag{15}$$

$$\begin{aligned}
\mathcal{P}_{L2}: \quad & \min_{\{\widehat{s}_m\}} -\sum_{n=1}^{N_m}\widehat{s}_m(n_m)\left(\vartheta_c \dot{q}_{m,n}\bar{R} - \varpi\right) + \\
& \sum_{i=1}^{M}\sum_{n=1}^{N_i}\left\{[\eta_m(n_i)]^{(t)}\widehat{s}_m(n_i) + \frac{q}{2}\left\{\widehat{s}_m(n_i) - [s(n_i)]^{(t)}\right\}^2\right\} \\
& s.t. \quad C5'.
\end{aligned} \tag{16}$$

Note that both $\mathcal{P}_{L1}$ and $\mathcal{P}_{L2}$ are convex problems, so the corresponding optimal solutions can be obtained using standard software, such as CVX, SeDuMi, etc.

3) Global Variables and Lagrange Multipliers Update:

Observing that (12) is a unconstrained quadratic problem and strictly convex, the global solution w.r.t. $\boldsymbol{s}$ is obtained by

$$[s(n_i)]^{(t+1)} = \frac{1}{qM}\sum_{m=1}^{M}[\eta_m(n_i)]^{(t)} + \frac{1}{M}\sum_{m=1}^{M}\left[\widehat{s}_m(n_i)\right]^{(t+1)}, \forall n, i. \tag{17}$$

Besides, the Lagrange multipliers $\boldsymbol{\eta}$ can be updated by (13).

4) Algorithm Stopping Criterion: In the implementation process, we adopt the rational stopping criteria proposed in [19] as $\left\|\widehat{\boldsymbol{s}}_m^{(t+1)} - \boldsymbol{s}^{(t+1)}\right\|_2 \le \iota_{pri}$ and $\left\|\boldsymbol{s}^{(t+1)} - \boldsymbol{s}^{(t)}\right\|_2 \le \iota_{dual}$, where $\iota_{pri} > 0$ and $\iota_{dual} > 0$ are the feasibility tolerances for the primal and dual feasibility conditions, respectively.

5) Binary Variables Recovery: For simplicity, we recover the binary variables $\boldsymbol{\delta}$ and $\boldsymbol{s}$ by the rounding principle, i.e., $\dot{\delta}(n_m) = round(\delta(n_m))$ and $\dot{s}(n_m) = round(s(n_m))$. Finally, the proposed algorithm is summarized as follows.

Algorithm 1 Distributed ADMM-based Computation Offloading and Content Caching Algorithm

1: Initialization
   1) The MEC server manager determines the stopping criterion threshold ι_{pri} and ι_{dual};
   2) The MEC server manager initializes the feasible solution $\{\boldsymbol{s}\}^{(0)}$ and sends it to all the APs;
   3) Each AP AP_m determines its initial Lagrange multiplier vector $\{\boldsymbol{\eta}_m\}^{(0)}$ and sends it to the MEC server manager;
2: Solving the optimization problem distributedly.
   Repeat
   1) For computation offloading, each AP AP_m obtains the optimal offloading decision $\{\boldsymbol{\delta}_m, \tilde{\boldsymbol{\rho}}_m\}$ by solving $\mathcal{P}_{L1}$.
   2) For content caching, each AP AP_m updates its local variables $\left\{\widehat{\boldsymbol{s}}_m\right\}^{(t+1)}$ by solving $\mathcal{P}_{L2}$ and sends the results to the MEC server manager;
   3) The MEC server manager updates the Lagrange multiplier $\{\boldsymbol{\eta}_m\}^{(t+1)}$ and sends them to every AP;
   Until $\left\|\widehat{\boldsymbol{s}}_m^{(t+1)} - \boldsymbol{s}^{(t+1)}\right\|_2 \le \iota_{pri}$ and $\left\|\boldsymbol{s}^{(t+1)} - \boldsymbol{s}^{(t)}\right\|_2 \le \iota_{dual}$.
3: Output the optimal solution $\left\{\boldsymbol{\delta}^*, \tilde{\boldsymbol{\rho}}^*, \widehat{\boldsymbol{s}}_m^*\right\}$.
4: Binary variables recovery for $\boldsymbol{\delta}$ and $\boldsymbol{s}$.

V. Simulation Results and Discussions

In the simulation, the network coverage radius is set at 200m, where the AP and user density are assumed to be $10^{-4}/m^2$ and $10^{-3}/m^2$. The users transmit at power 0.1W on the channel with bandwidth 20MHz, and the noise power is -174dBm/Hz. The static circuit power is set at 0.05W. The input data size is 5 kbits, the completion deadline is 10ms and the computation workload/intensity is 18000 CPU cycles/bit.

First, we present the convergence performance of our proposed ADMM-based algorithm. As shown in Fig. 2, the total revenues increase dramatically in the first 10 iterations and then reach a stable status within the first 30 iterations, which illustrate that our proposed ADMM-based algorithm can converge quickly. Although the convergence progress with $q = 1.5$ reaches the stable status fastest while $q = 0.5$ slowest, there are subtle differences. And it's noticed that the gap between the proposed ADMM-based algorithm and the centralized one is rather small. Another observation from Fig. 2 is that the total net revenue grows with the increase of storage capacity.

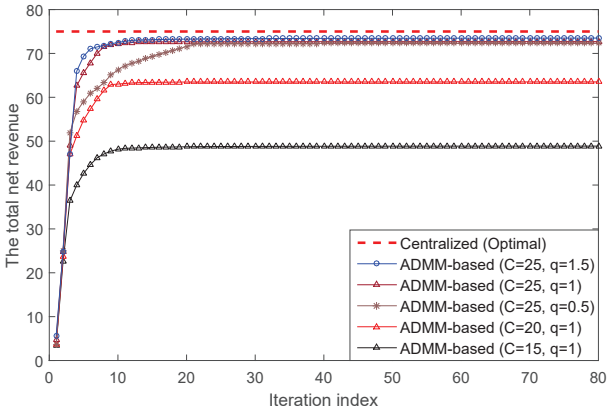

Fig. 2. Convergence performance of distributed ADMM-based algorithm.

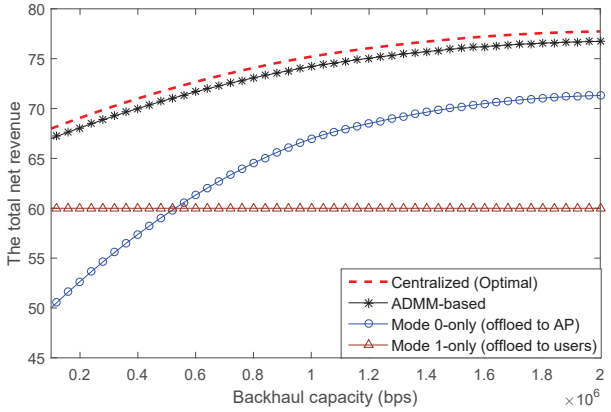

Fig. 3. The total net revenue vs. backhaul capacity.

Fig. 3 depicts the total net revenue with respect to varying backhaul capacity. We can see that the centralized algorithm achieves the highest total net revenue, but it is obvious that the gap between our proposed algorithm and the centralized one is not wide. Additionally, all of the schemes get higher net revenue with the increase of backhaul capacity while the revenue received from *mode 1*-only scheme remains constant. This can be explained as: when the miners choose to offload their mining tasks through D2D links, it wouldn't cost any pressure for backhaul links.

In Fig. 4, the total net revenue with varying completion deadlines is depicted. It can be seen that all the four schemes obtain higher net revenue with the increasing completion deadlines. This is because when the completion deadline increases, more miners can complete the mining task successfully. Additionally, as expected, *mode 1*-only scheme can receive higher revenue than *mode 0*-only scheme. In *mode 1*-only scheme,

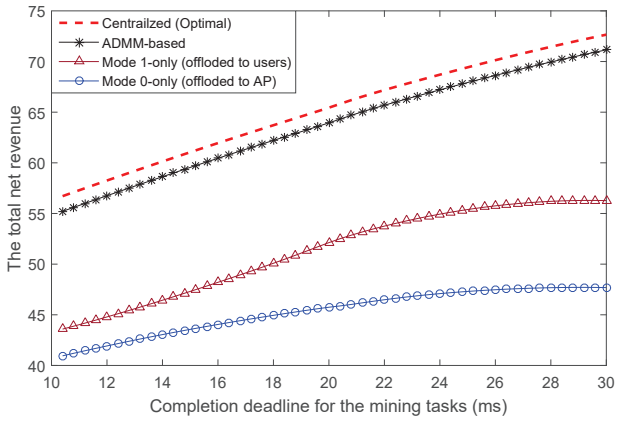

Fig. 4. The total net revenue vs. completion deadline of the mining tasks.

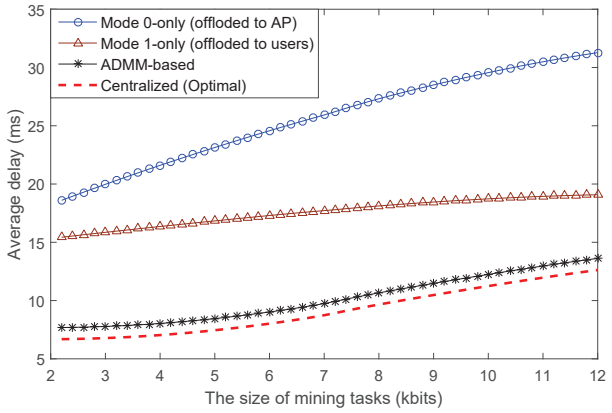

Fig. 5. The average delay vs. the size of mining tasks.

a group of D2D users can share computation resources and balance the uneven distribution of the computation workloads among users. Fig. 5 discusses the average delay w.r.t. the size of mining tasks. The largest delay is achieved by *mode 0*-only scheme, because it costs more time for a single resource-limited AP to handle a large amount of requests at a time. Meanwhile, the *mode 1*-only scheme where a group of D2D users share computation resources can reduce the total delay.

VI. Conclusion and Future Work

In this paper, we studied computation offloading and content caching in wireless blockchain networks with MEC. To maximize the total net revenue, offloading scheduling and content caching strategy were formulated as an optimization problem and a distributed ADMM-based algorithm was proposed to solve the problem. Simulation results indicated that our proposed scheme is able to address the proof of work puzzle issue effectively. In future work, probabilistic QoS constraints will be considered to further exploit the network resources.

Acknowledgment

This work was supported in part by the scholarship from China Scholarship Council (No. 201706470059), in part by the National Natural Science Foundation of China under Grant No. 61771072, and in part by Beijing Natural Science Foundation under Grant No. L171011.

References

[1] P. W. Chen, B. S. Jiang, and C. H. Wang, "Blockchain-based payment collection supervision system using pervasive Bitcoin digital wallet," in *Proc. IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob),* Rome, Italy, Oct. 2017, pp. 139-146.

[2] H. Jang and J. Lee, "An Empirical Study on Modeling and Prediction of Bitcoin Prices with Bayesian Neural Networks Based on Blockchain Information," *IEEE Access,* vol. PP, no. 99, pp. 1-1, Dec. 2017.

[3] A. Stanciu, "Blockchain Based Distributed Control System for Edge Computing," in *Proc. 21st International Conference on Control Systems and Computer Science (CSCS),* Bucharest, May 2017, pp. 667-671.

[4] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, Fourthquarter 2017.

[5] N. Houy, "The bitcoin mining game," *Browser Download This Paper,* Available: https://ssrn.com/abstract=2407834, Mar. 2014.

[6] J. Beccuti *et al.*,"The bitcoin mining game: On the optimality of honesty in proof-of-work consensus mechanism," *Swiss Economics Working Paper 0060*, Aug. 2017.

[7] A. Kiayias, E. Koutsoupias, M. Kyropoulou, and Y. Tselekounis, "Blockchain mining games," in *Proc. the ACM Conference on Economics and Computation,* Maastricht, Netherlands, Jul. 2016, pp. 365–382.

[8] B. A. Fisch, R. Pass, and A. Shelat, Socially optimal mining pools, [online] arXiv:1703.03846v1 [cs.GT], 2017.

[9] L. Luu, R. Saha, I. Parameshwaran, P. Saxena, and A. Hobor, "On power splitting games in distributed computation: The case of bitcoin pooled mining," in *Proc. IEEE the 28th Computer Security Foundations Symposium (CSF),* Verona, Italy, Jul. 2015, pp. 397–411.

[10] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-Edge Computing: Partial Computation Offloading Using Dynamic Voltage Scaling, *IEEE Transactions on Communications*, vol. 64, no. 10, pp. 4268–4282, Oct. 2016.

[11] Y. Mao, J. Zhang, and K. B. Letaief, "Joint Task Offloading Scheduling and Transmit Power Allocation for Mobile-Edge Computing Systems," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC),* San Francisco, CA, Mar. 2017, pp. 1-6.

[12] 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 Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, Sept. 2017.

[13] Y. H. Yu, J. Zhang, and K. B. Letaief, "Joint Subcarrier and CPU Time Allocation for Mobile Edge Computing," in *Proc. IEEE Global Communications Conference,* Washington, DC, Dec. 2016, pp. 1-6.

[14] Y. Ge *et al.*, "A game theoretic resource allocation for overall energy minimization in mobile cloud computing system," in *Proc. ACM/IEEE International Symposium on Low Power Electronics and Design.*, Redondo Beach, California, Jul. 2012, pp. 279-284.

[15] S. Z. Bi and Y. T. Zhang, "Computation Rate Maximization for Wireless Powered Mobile-Edge Computing with Binary Computation Offloading," [online] arXiv:1708.08810v2 [cs.DC], 2017.

[16] S. Ranadheera, S. Maghsudi, and E. Hossain, "Computation Offloading and Activation of Mobile Edge Computing Servers: A Minority Game," [online] arXiv:1710.05499v1 [cs.GT], 2017.

[17] J. Liu, B. Bai, J. Zhang, and K. B. Letaief, "Content caching at the wireless network edge: A distributed algorithm via belief propagation," in *Proc. IEEE International Conference on Communications (ICC),* Kuala Lumpur, May 2016, pp. 1-6.

[18] M. T. Liu, Y. Teng, and M. Song, "Performance analysis of coordinated multipoint joint transmission in ultra-dense networks with limited backhaul capacity," *Electronics Letters*, vol. 51, no. 25, pp. 2111–2113, Dec. 2015.

[19] S. Boyd *et al.*, "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.