Mobile Edge Computing (MEC)-Enabled Wireless Blockchain Networks

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

***Abstract*—Blockchain technology has been applied in a variety of fields due to its capability of establishing trust in a decentralized fashion. However, the application of blockchain in wireless mobile networks is hindered by a major challenge brought by the proof-of-work (PoW) puzzle during the mining process, which sets a high demand for the computational capability and storage availability in mobile devices. To address this problem, we propose a novel mobile edge computing (MEC) enabled wireless blockchain framework where the computation-intensive mining tasks can be offloaded to nearby edge computing nodes and the cryptographic hashes of blocks can be cached in the MEC server. Particularly, two offloading modes are considered, i.e., offloaded to the nearby access point (AP) or a group of nearby users. First, we conduct the performance analysis of each mode with stochastic geometry methods. Then, the joint offloading decision and caching strategy is formulated as an optimization problem. Furthermore, an alternating direction method of multipliers (ADMM) based algorithm is utilized to solve the problem in a distributed manner. Finally, simulation results demonstrate the effectiveness of our proposed scheme.**

***Index Terms*—Mobile edge computing, blockchain, computation offloading, content caching, stochastic geometry.**

I. Introduction

WITH the thriving of Information and Communications Technologies (ICT) and the popularity of consumer devices, electronic trading with digital transactions has received great attention in e-commerce [1]. However, in the traditional centralized financial systems, the consensus is reached through trusted centralized authorities, which would bring additional cost. To address this problem, *Bitcoin* was introduced into the financial market as a peer-to-peer (P2P) electronic cash system [2]. Compared with the traditional centralized financial institutions, Bitcoin aims at achieving a complete decentralization with the help of a public append-only ledger called *Blockchain* [3], [4]. The successful application of blockchain in Bitcoin has led to significant research interests in applying blockchain in other fields, such as security, smart cities, supply chain, etc. [5], [6].

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)

Nevertheless, the application of blockchain technology is hindered by a main challenge brought by a computational process called *mining*. In order to confirm and secure the integrity and validity of transactions, the participants (a.k.a. miners) need to complete a computation-intensive task, i.e., the proof-of-work (PoW) puzzle, which sets an extremely high demand for the computational capability and storage availability of the miners.

Recently, there are several works focusing on mining schemes management for blockchain networks. The mining schemes can be divided into solo mining and pool mining according to whether the miners mine blocks individually [7]. For solo mining, a non-cooperative game among the miners is proposed in [8], where the PoW puzzle is modeled as a Poisson process and the miner's strategy is to choose the number of transactions to be included in a block. Focusing on whether or not to propagate the mined block, the authors of [9] and [10] put forward a sequential game and a stochastic game for modeling the mining process, respectively. Regarding pool mining, a cooperative game based blockchain mining scheme is proposed to explore the optimal pool mining mechanism in [11]. Besides, a new computational power splitting game for the blockchain mining is modeled in [12], where the miners can get the mining rewards by competing in solving the PoW puzzle.

Although these mining management schemes can help with finding an optimal mining policy in wired blockchain networks, it is challenging to apply these schemes to wireless mobile blockchain networks due to the limited computation and storage capacity in mobile devices. Indeed, applying mining-based blockchain to wireless mobile networks in general is challenging because of the computation-extensive mining process [13].

In this paper, we address these challenges in wireless mobile blockchain networks using recent advances in mobile edge computing (MEC). Compared to traditional cloud computing, the new MEC paradigm brings network resources (computation or storage resources) closer to the mobile users to effectively reduce the energy consumption and shorten the transmission delay [13], [14]. The contributions of this paper can be summarized as follows.

- We propose a novel MEC-enabled blockchain framework where the mobile miners can resort to the nearby edge nodes to perform the computation-intensive PoW puzzle and content caching (storing the cryptographic hashs of blocks). To the best of our knowledge, MEC-enable mobile blockchain with computation offloading and content caching has not been well studied in previous works.
- In this framework, to avoid the overload of MEC service providers, we consider two offloading modes, i.e., offloaded to the nearby access point (AP) *(mode 0)* or a group of nearby users *(mode 1)*.
- Computation offloading scheduling and caching strategy are jointly formulated as an optimization problem. Particularly, probabilistic backhaul and delay constraints are considered, which allow for a small percentage of outage that relaxes the worst case but can provide a flexible reaction for the fluctuation of the system.
- Using stochastic geometry theory [15], the theoretical expressions of related performance metrics are derived, including delay, energy consumption and orphaning probability (the probability that the total delay excesses the completion deadline in blockchain). In this way, each miner only needs to acquire the local channel state information (CSI), which can significantly reduce the signaling overhead.
- We transform the original non-convex problem into a convex one and propose an alternating direction method of multipliers (ADMM)-based algorithm, thus the problem can be efficiently solved in a distributed manner.

The rest of this paper is organized as follows. The system model is presented in Section II. We present the novel framework and conduct performance analysis in Section III. In Section IV, we formulate the computation offloading and content caching into an optimization problem and propose a distributed ADMM-based algorithm to solve it. Simulation results are discussed in Section V. Section VI concludes the paper with the future work.

II. System Model

In this section, the system model adopted in this work is presented. We first describe the application model, then we present the computation offloading model, network model and caching model in detail.

A. Application Model

As shown in Fig. 1, we consider an MEC enabled blockchain network where M APs and N users follow two independent homogeneous Poisson point processes (HPPPs) $\Phi_a = \{AP_1, AP_2, \ldots, AP_M\}$ and $\Phi_u = \{x_1, x_2, \ldots, x_N\}$ with density λ_a and λ_u, respectively. Assuming every user accesses to its nearest AP, there are N_m users in small cell m. An MEC server manager is placed in the cloud and all the APs connect to the cloud as well as the MEC server manager through the core network. We denote the set of users associated with AP_m as $\Phi_u^{(m)} = \{x_{n_m}\}, x_{n_m} \in \Phi_u, n = 1, \ldots, N_m$ and x_{n_m} stands for the nth user in small cell m. And for x_{n_m}, there are $K_{m,n}$ device-to-device (D2D) links with a set of neighboring users $\Phi_u^{(m,n)} = \left\{x_{k_{m,n}}\right\}, x_{k_{m,n}} \in \Phi_u, k = 1, \ldots, K_{m,n}$.

Every mobile user acts as a miner[1] to run a blockchain application to record the transaction performed in the network. For miner x_{n_m}, we adopt a parameter tuple $\langle D_{m,n}, \tau_{m,n}, X_{m,n}\rangle$ to character its mining task $A_{\langle m,n\rangle}$ with the input data size $D_{m,n}$ (in bits), the completion deadline $\tau_{m,n}$ (in seconds) and the computation workload/intensity $X_{m,n}$ (in CPU cycles/bit). The computational capability (CPU cycles/s) of $AP_m, m = 1, \ldots, M$ and $x_{k_{m,n}}$ are denoted by f_m^A and $f_{m,n,k}^U$, respectively.

B. Computation Offloading Model

We assume that all the APs and miners in the MEC enabled blockchain network have a computation capacity to provide task-execution services. Accordingly, the computational difficult mining task can be offloaded to the nearby APs or miners. So we consider two offloading modes where the computation-intensive task at x_{n_m} is:

1) ***offloaded to a nearby AP (mode 0)***: The miner x_{n_m} 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_{n_m} divides the whole task $A_{\langle m,n\rangle}$ into $K_{m,n}$ parts $\left\{A_{\langle m,n,1\rangle}, \ldots, A_{\langle m,n,K_{m,n}\rangle}\right\}$ and separately forwards them to its neighboring users through D2D links[2].

We denote $\delta(n_m) \in \{0,1\}$ as the task offloading decision of x_{n_m}. Specifically, we have $\delta(n_m) = 0$ if the miner x_{n_m} selects *mode 0*, i.e., offloading the mining task to its associated AP AP_m. We have $\delta(n_m) = 1$ if x_{n_m} chooses *mode 1*, i.e., offloading the task to a group of D2D users. And in the case of *mode 1*, $\rho(n_m, k_{m,n}) \in [0,1]$ is defined as the ratio of offloading part $A_{\langle m,n,k_{m,n}\rangle}$ to the whole task $A_{\langle m,n\rangle}$. So we have $\boldsymbol{\delta} = \{\delta(n_m)\}, \forall m, n$ and $\boldsymbol{\rho} = \{\rho(n_m, k_{m,n})\}, \forall m, n, k$ as the task offloading scheduling profile.

C. Network Model

The channel radio propagation between APs and users is assumed to comprise path loss and Rayleigh fading. Specifically, the path loss fading of the channel between x_{n_m}

[1]For convenience, the terms "miner" and "mobile user", "AP" and "small cell" are used interchangeably throughout this paper.

[2]The case of local execution is included in *mode 1* when a part of task is left and computed locally.

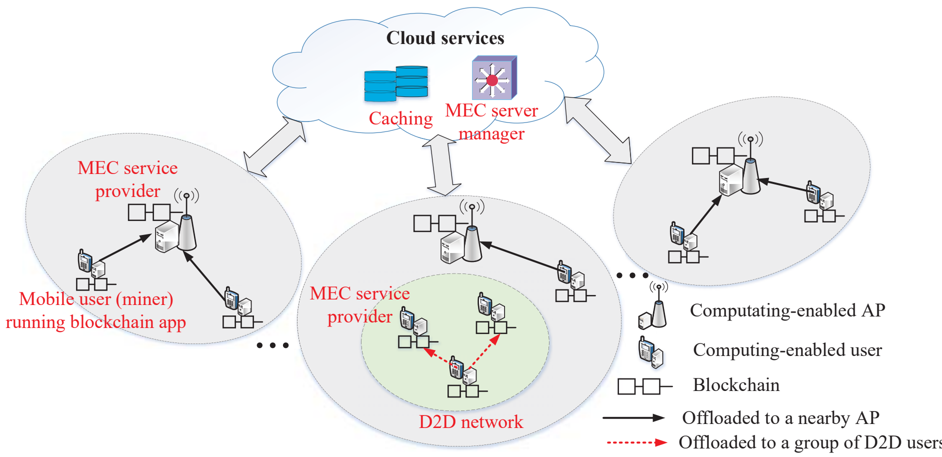

Fig. 1. An illustration of MEC enabled wireless blockchain networks with two offloading modes: *mode 0* (the mining task is offloaded to a nearby AP) and *mode 1* (the computation task is offloaded to a group of D2D users).

and AP_m ($x_{k_{m,n}}$) is ${r_{m,n}}^{-\alpha}$ (${l_{m,n,k}}^{-\alpha}$) where $r_{m,n}$ ($l_{m,n,k}$) denotes the distance between x_{n_m} and AP_m ($x_{k_{m,n}}$) and α is the path loss exponent. Meanwhile, Rayleigh fading of the link between x_{n_m} and AP_m ($x_{k_{m,n}}$) 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. Note that $h_{m,n}$, $g_{m,n,k}, \forall m,n,k$ are mutually independent with each other. Assume each user transmits at the identical power P_U on the same channel with bandwidth B, and the noise power is σ^2.

Therefore, for x_{n_m}, the maximum upload transmission rate (in bit/s) to offload task $A_{\langle m,n\rangle}$ to AP_m is expressed by

$$R^A(n_m) = B\ln\left(1+\frac{P_U h_{m,n} {r_{m,n}}^{-\alpha}}{\sigma^2+\sum\limits_{x_i\in\Phi_u\backslash x_{n_m}} P_U h_{m,i} {r_{m,i}}^{-\alpha}}\right). \quad (1)$$

Similarly, the maximum upload transmission rate (in bit/s) for x_{n_m} to offload a single task part $A_{\langle m,n,k\rangle}$ to $x_{k_{m,n}}$ is

$$R^D(n_m,k_{m,n}) = B\ln\left(1+\frac{\frac{P_U}{K_{m,n}} g_{m,n,k} {l_{m,n,k}}^{-\alpha}}{\sigma^2+\sum\limits_{x_j\in\Phi_u\backslash x_{n_m}} \frac{P_U}{K_{m,n}} g_{m,j,k} {l_{m,j,k}}^{-\alpha}}\right). \quad (2)$$

where the transmit power P_U is uniformly distributed across all the D2D links.

D. Caching Model

We denote $s(n_m) \in \{0,1\}$ as the caching decision for x_{n_m}. Specifically, $s(n_m) = 1$ if the MEC server manager decides to cache the cryptographic hashs of blocks w.r.t. x_{n_m} while $s(n_m) = 0$ otherwise. Hence, $\boldsymbol{s} = \{s(n_m)\}, \forall m,n$ can be regarded as the caching decision profile. In this paper, we simply consider the reward of caching the cryptographic hashs of blocks w.r.t. x_{n_m} as

$$v(n_m) = \dot{q}_{m,n} R^c_{m,n}, \quad (3)$$

where $\dot{q}_{m,n}$ is the request rate of the cached content and $R^c_{m,n}$ (in *token*) is the caching reward.

It's worth noting that the storage capability of the MEC server is limited, thus the sum size of all the cached contents shouldn't exceed the storage capability C, i.e. $\sum\limits_{m=1}^{M}\sum\limits_{n=1}^{N_m} s(n_m)\chi_{m,n} \le C$ where $\chi_{m,n}$ is the size of the cryptographic hashs of blocks w.r.t. x_{n_m}.

III. Framework and Performance Analysis

In this section, we derive the theoretical expressions of performance metrics including delay, energy consumption and orphaning probability w.r.t. two offloading modes.

A. Offloaded to a Nearby AP (Mode 0)

In this part, we analyze the performance of *mode 0* when the miner offloads the task to its associated AP. Assume that every miner is only aware of the local CSI, i.e., the reference signal received power (RSRP) $RSRP^{(A)}(n_m) = P_U h_{m,n} {r_{m,n}}^{-\alpha}$.

1) Delay:

For $x_{n_m}, \forall m,n$, the total time to accomplish its mining task $A_{\langle m,n\rangle}$ includes the time spent on uploading, computing and queueing[3] [16], which can be expressed by

$$T^{(A)}(n_m) = T^{(A,u)}(n_m) + T^{(A,e)}(n_m) + T^{(A,q)}(n_m), \quad (4)$$

[3]Since the output data size is much smaller than the input one and the transmit power of the AP is much stronger compared with that of mobile users, the time spent on result feedback by AP can be neglected. And the same assumption is considered for *mode 1*.

where the separate parts in (4) including uploading delay $T^{(A,u)}(n_m)$, executing time $T^{(A,e)}(n_m)$ and queueing delay $T^{(A,q)}(n_m)$ are derived as follows.

- ***Uploading delay***

The time consumed for miner $x_{n_m}, \forall m, n$ to upload the mining task to the nearby AP AP_m is formulated as

$$T^{(A,u)}(n_m) = \frac{D_{m,n}}{R^A(n_m)}. \quad (5)$$

Note that to calculate the denominator part of (5), i.e., upload transmission rate $R^A(n_m)$, full CSI is needed, which brings significant signaling overhead, especially in dense networks. To address this problem, we provide an approximation for $R^A(n_m)$ with stochastic geometry methods. Averaging the aggregate interference, $R^A(n_m)$ can be approximated by

$$R^A(n_m) \approx B \ln\left(1 + \frac{P_U h_{m,n} {r_{m,n}}^{-\alpha}}{\sigma^2 + \mathbb{E}(I_m) - P_U h_{m,n} {r_{m,n}}^{-\alpha}}\right), \quad (6)$$

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 in the following proposition.

Proposition 1. *The cumulative distribution function (CDF) of the aggregate interference I_m and its expectation is*

$$\begin{aligned} &F_{I_m}(\xi) = \mathbb{P}[I_m \le \xi] \\ &\approx \int_{0<\hat{r}_1<\ldots<\hat{r}_\beta<\infty} \left[1 - \exp\left\{-\frac{\mu\xi}{P_U\left[\sum_{i=1}^{\beta}\hat{r}_i^{-\alpha} + \sum_{c=\beta+1}^{N}(\pi\lambda_u)^{\frac{\alpha}{2}}\frac{\Gamma\left(c-\frac{\alpha}{2}\right)}{\Gamma(c)}\right]}\right\}\right] \\ &\times f\left(\hat{r}_1, \ldots, \hat{r}_\beta\right) d\hat{r}_1 \ldots d\hat{r}_\beta \end{aligned} \quad (7)$$

and

$$\begin{aligned} &\mathbb{E}(I_m) \\ &= \int_{0<\hat{r}_1<\ldots<\hat{r}_\beta<\infty} \int_{\xi>0} \left[1 - F_{I_m}(\xi)\right] f\left(\hat{r}_1, \ldots, \hat{r}_\beta\right) d\xi d\hat{r}_1 \ldots d\hat{r}_\beta \\ &\approx \int_{0<\hat{r}_1<\ldots<\hat{r}_\beta<\infty} \int_{\xi>0} \exp\left\{\frac{\mu\xi}{P_U\left[\sum_{i=1}^{\beta}\hat{r}_i^{-\alpha} + \sum_{c=\beta+1}^{N}(\pi\lambda_u)^{\frac{\alpha}{2}}\frac{\Gamma\left(c-\frac{\alpha}{2}\right)}{\Gamma(c)}\right]}\right\} \\ &\times f\left(\hat{r}_1, \ldots, \hat{r}_\beta\right) d\xi d\hat{r}_1 \ldots d\hat{r}_\beta. \end{aligned} \quad (8)$$

where $\beta = \lfloor \alpha/2 \rfloor$ ($\lfloor\bullet\rfloor$ is the floor function), $\Gamma(\bullet)$ is the gamma function, and $\{r_{m,1}, r_{m,2}, \ldots, r_{m,N}\}$ is rearranged as $\{\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_N\}$ in ascending order with joint distance distribution $f(\hat{r}_1, \ldots, \hat{r}_N) = e^{-\lambda_u \pi \hat{r}_N^2}(2\lambda_u\pi)^N \hat{r}_1 \ldots \hat{r}_N$,

Proof. See Appendix 1. □

In a special case when $\alpha = 4$, $\mathbb{E}(I_m)$ is simplified as

$$\begin{aligned} &\mathbb{E}(I_m) \approx \int_0^\infty \int_0^{\hat{r}_2} \int_0^\infty \exp\left\{-\frac{\mu\xi}{P_U\left[\hat{r}_1^{-4} + \hat{r}_2^{-4} + (\pi\lambda_u)^2 G(N)\right]}\right\} \\ &\times f(\hat{r}_1, \hat{r}_2)\, d\xi d\hat{r}_1 d\hat{r}_2. \end{aligned} \quad (9)$$

where $G(N) = \frac{N-2}{N-1} \approx 1$ when N is large.

- ***Executing time***

The time used when miner x_{n_m} chooses *mode 0* to compute its mining task $A_{\langle m,n\rangle}$ is

$$T^{(A,e)}(n_m) = \frac{D_{m,n} X_{m,n}}{f_m^A}. \quad (10)$$

- ***Queuing delay***

In this paper, we only consider the queuing delay brought by the waiting tasks in the task buffer at the time when the miners make decisions. Assuming the total number of CPU cycles to be processed in the task buffer of AP_m as Q_m. Then the queuing delay for x_{n_m} can be calculated by

$$T^{(A,q)}(n_m) = \frac{Q_m}{f_m^A}. \quad (11)$$

2) Energy Consumption:

From the above discussion, we neglect the energy consumed by the miner to receive the computation results, and only consider the energy cost for task uploading, computing and CPU operation. Accordingly, the total energy consumption for x_{n_m} to complete task $A_{\langle m,n\rangle}$ is

$$\begin{aligned} &E^{(A)}(n_m) \\ &= P_U T^{(A,u)}(n_m) + \kappa_m^A \left(f_m^A\right)^3 T^{(A,e)}(n_m) + P_C T^{(A,q)}(n_m), \end{aligned} \quad (12)$$

where κ_m^A denotes the computation energy efficiency coefficient of AP_m [17], [18] and P_C is the static circuit power.

3) Orphaning Probability:

The probability of successful mining depends on if the whole process including offloading and computing is successfully finished before the task expires. In other words, the total delay should not excess the completion deadline, otherwise, the task would be discarded, which is called orphaning [8]. To offer a measurement for the orphaning probability of the mining task $A_{\langle m,n\rangle}$, we define $p_O^{(A)}(n_m)$ as

$$p_O^{(A)}(n_m) = \mathbb{P}\left[T^{(A)}(n_m) \ge \tau_{m,n}\right]. \quad (13)$$

Following Proposition 1, we can derive $p_O^{(A)}(n_m)$ as

$$\begin{aligned} &p_O^{(A)}(n_m) = \mathbb{P}\left[T^{(A)}(n_m) \ge \tau_{m,n}\right] \\ &= \mathbb{P}\left[T^{(A,u)}(n_m) + T^{(A,e)}(n_m) + T^{(A,q)}(n_m) \ge \tau_{m,n}\right] \\ &= \mathbb{P}\left[\frac{D_{m,n}}{R^A(n_m)} \ge \tau_{m,n} - T^{(A,e)}(n_m) - T^{(A,q)}(n_m)\right] \\ &\overset{(a)}{=} \mathbb{P}\left[R_{A,i,\varepsilon^A(i)} \le \frac{D_{m,n}}{\tau_{m,n} - T^{(A,e)}(n_m) - T^{(A,q)}(n_m)}\right] \\ &= 1 - F_{I_m}\left(\iota_{m,n}^A\right), \end{aligned} \quad (14)$$

where (a) holds conditioned on $T^{(A,e)}(n_m) + T^{(A,q)}(n_m) \le \tau_{m,n}$, $\iota_{m,n}^A = \frac{P_U h_{m,n} {r_{m,n}}^{-\alpha}}{1 - \exp\left[-\frac{D_{m,n}/B}{\tau_{m,n} - T^{(A,e)}(n_m) - T^{(A,q)}(n_m)}\right]} - \sigma^2$ and $F_{I_m}(\bullet)$ is given in Proposition 1.

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

Similarly, we analyze the performance of *mode 1*, i.e., the mining task is offloaded to a group of D2D users. Recall that every miner $x_{n_m}, \forall m, n$ in *mode 1* would separate $A_{\langle m,n\rangle}$ into $K_{m,n}$ parts $\left\{A_{\langle m,n,1\rangle}, \ldots, A_{\langle m,n,K_{m,n}\rangle}\right\}$ and uploads them to $K_{m,n}$ available D2D users. And each user in *mode 1* is assumed to be aware of the local CSI, i.e., $RSRP^{(D)}\left(n_m, k_{m,n}\right) = P_U g_{m,n,k} {l_{m,n,k}}^{-\alpha}/K_{m,n}$. Since Φ_u is a HPPP with density λ_u and x_{n_m} is randomly chosen, $\Phi_u \backslash x_{n_m}$ is also a HPPP with density λ_u according to Slivnyak's Theorem [15].

1) Delay:

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

$$\begin{aligned} T^{(D)}\left(n_m\right) &= \max_{k=1,\ldots,K_{m,n}} \rho\left(n_m, k_{m,n}\right) T^{(D)}\left(n_m, k_{m,n}\right) \\ &= \max_{k=1,\ldots,K_{m,n}} \rho\left(n_m, k_{m,n}\right) \left[T^{(D,u)}\left(n_m, k_{m,n}\right) + T^{(D,e)}\left(n_m, k_{m,n}\right)\right], \end{aligned} \tag{15}$$

where $T^{(D)}\left(n_m, k_{m,n}\right)$ is the time required for the case if the whole mining task $A_{\langle m,n\rangle}$ is offloaded to $x_{k_{m,n}}$. Note that the queueing delay in *mode 1* is ignored since each miner only has a small computational capacity.

- ***Uploading delay***

The time consumed for miner x_{n_m} to upload the mining task part $A_{\langle m,n,K_{m,n}\rangle}$ to $x_{k_{m,n}}$ is formulated as

$$T^{(D,u)}\left(n_m, k_{m,n}\right) = \frac{D_{m,n}}{R^D\left(n_m, k_{m,n}\right)}. \tag{16}$$

Along the same lines, the maximum upload transmission rate $R^D\left(n_m, k_{m,n}\right)$ can be approximated as

$$R^D\left(n_m, k_{m,n}\right) \approx B \ln\left(1 + \frac{\frac{P_U}{K_{m,n}} g_{m,n,k} {l_{m,n,k}}^{-\alpha}}{\sigma^2 + \mathbb{E}\left(I_m\right) - \frac{P_U}{K_{m,n}} g_{m,n,k} {l_{m,n,k}}^{-\alpha}}\right), \tag{17}$$

where $\mathbb{E}\left(I_m\right)$ is given in Proposition 1.

- ***Executing time***

The executing time cost when miner x_{n_m} chooses *mode 1* to compute the mining task part $A_{\langle m,n,K_{m,n}\rangle}$ is

$$T^{(D,e)}\left(n_m, k_{m,n}\right) = \frac{D_{m,n} X_{m,n}}{f^U_{m,n,k}}. \tag{18}$$

2) Energy Consumption:

The total energy consumption for the D2D users to complete $A_{\langle m,n\rangle}$ is

$$\begin{aligned} E^{(D)}\left(n_m\right) &= \sum_{k=1}^{K_{m,n}} \rho\left(n_m, k_{m,n}\right) E^{(D)}\left(n_m, k_{m,n}\right) \\ &= \sum_{k=1}^{K_{m,n}} \rho\left(n_m, k_{m,n}\right) \left[\begin{array}{l} \frac{P_U}{K_{m,n}} T^{(D,u)}\left(n_m, k_{m,n}\right) + \\ \kappa^U_{m,n,k} \left(f^U_{m,n,k}\right)^3 T^{(D,e)}\left(n_m, k_{m,n}\right) \end{array}\right], \end{aligned} \tag{19}$$

where $\kappa^U_{m,n,k}$ denotes the computation energy efficiency coefficient of the processor's chip in $x_{k_{m,n}}$.

3) Orphaning Probability:

Similar to *mode 0*, the orphaning probability of the mining task $A_{\langle m,n\rangle}$ when the miner chooses *mode 1* is

$$\begin{aligned} p_O^{(D)}\left(n_m, k_{m,n}\right) &= \mathbb{P}\left(T^{(D)}\left(n_m\right) \geq \tau_{m,n}\right) \\ &= \max_{k=1,\ldots,K_{m,n}} \mathbb{P}\left(\rho\left(n_m, k_{m,n}\right) T^{(D)}\left(n_m, k_{m,n}\right) \geq \tau_{m,n}\right) \\ &= 1 - \prod_{k=1}^{K_{m,n}} \mathbb{P}\left\{\begin{array}{l} \rho\left(n_m, k_{m,n}\right)\left[T^{(D,u)}\left(n_m, k_{m,n}\right)\right. \\ \left. + T^{(D,e)}\left(n_m, k_{m,n}\right)\right] \leq \tau_{m,n} \end{array}\right\} \\ &= 1 - \prod_{k=1}^{K_{m,n}} F_{I_m}\left(\iota^D_{m,n,k}\right), \end{aligned} \tag{20}$$

where $\iota^D_{m,n,k} = \dfrac{\frac{P_U}{K_{m,n}} g_{m,n,k} {l_{m,n,k}}^{-\alpha}}{1 - \exp\left[-\dfrac{D_{m,n}/B}{\frac{\tau_{m,n}}{\rho\left(n_m, k_{m,n}\right)} - T^{(D,e)}\left(n_m, k_{m,n}\right)}\right]} - \sigma^2$.

IV. Computation Offloading and Content Caching Optimization

In this section, we study the optimal computation offloading scheduling and caching decision $(\boldsymbol{\delta}, \boldsymbol{\rho}, \boldsymbol{s})$ in order to maximize the total net revenue in terms of offloading and caching, which is formulated as an optimization problem. Then the original non-convex optimization problem is transformed into a convex one and we propose an ADMM-based algorithm to solve it.

A. Optimization Objective

Considering computation offloading, the net revenue of accomplishing the mining task $A_{\langle m,n\rangle}$ is specified by

$$\begin{aligned} \Psi\left(n_m\right) = &\left[1 - \delta\left(n_m\right)\right]\left[\varsigma^{(A)} - \vartheta_e E^{(A)}\left(n_m\right)\right] \\ &+ \delta\left(n_m\right)\left[\varsigma^{(D)} - \vartheta_e E^{(D)}\left(n_m\right)\right], \end{aligned} \tag{21}$$

where ϑ_e (in *token/J*) is the unit price of energy, $\varsigma^{(A)}$ (in *token*) and $\varsigma^{(D)}$ (in *token*) are the rewards of *mode 0* and *mode 1*[4], respectively. To differentiate the rewards between offloading and local execution in *mode 1*, we set $\varsigma^{(D)} = \varsigma^{(D_O)}$ and $\varsigma^{(D)} = \varsigma^{(D_L)}$ ($\varsigma^{(D_O)} > \varsigma^{(D_L)}$), respectively.

Regarding content caching, the net revenue for caching the cryptographic hashs of blocks w.r.t. x_{n_m} is

$$\Lambda\left(n_m\right) = s\left(n_m\right)\left[\nu\left(n_m\right) - \varpi\right] = s\left(n_m\right)\left(\dot{q}_{m,n} R^c_{m,n} - \varpi\right), \tag{22}$$

where $\nu\left(n_m\right)$ (in *token*) is the reward of caching given in (3) and ϖ (in *token*) represents the memory cost for caching.

[4]The reward of *mode 0/mode 1* is defined as the net revenue for the miner who chooses *mode 0/mode 1*, which can be calculated by the difference between the income and the cost. Specifically, the income refers to the block reward of successfully mining, and the cost varies according to different modes. For *mode 0*, the cost includes the admission fee to join in the blockchain system, the fee for leasing the bandwidth, etc. For *mode 1*, the cost also includes the reward that the miner needs to share with its D2D users.

Considering the total net revenue of computation offloading and caching, and the cost of decision making Z (in *token*), the optimization objective is formulated as

$$\begin{aligned}\Xi &= \sum_{m=1}^{M}\sum_{n=1}^{N_m}\left[\Psi\left(n_m\right)+\Lambda\left(n_m\right)-\mathrm{Z}\right]\\ &= \sum_{m=1}^{M}\sum_{n=1}^{N_m}\left\{\begin{array}{l}\left[1-\delta\left(n_m\right)\right]\left[\varsigma^{(A)}-\vartheta_e E^{(A)}\left(n_m\right)\right]+\\ \delta(n_m)\sum\limits_{k=1}^{K_{m,n}}\rho(n_m,k_{m,n})\left[\varsigma^{(D)}-\vartheta_e E^{(D)}(n_m,k_{m,n})\right]\\ +s\left(n_m\right)\left(\dot{q}_{m,n}R_{m,n}^{c}-\varpi\right)-\mathrm{Z}\end{array}\right\}.\end{aligned} \tag{23}$$

B. Problem Formulation

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

$$\begin{aligned}\mathcal{P}1:\ &\max_{\boldsymbol{\delta},\boldsymbol{\rho},\boldsymbol{s}} \quad \sum_{m=1}^{M}\sum_{n=1}^{N_m}\left[\Psi\left(n_m\right)+\Lambda\left(n_m\right)-\mathrm{Z}\right]\\ s.t.\ &C1:[1-\delta(n_m)]+\delta(n_m)\sum_{k=1}^{K_{m,n}}\rho\left(n_m,k_{m,n}\right)=1,\forall m,n\\ &C2:\mathbb{P}\left\{\sum_{n=1}^{N_m}\left[1-\delta\left(n_m\right)\right]R^{A}\left(n_m\right)\geq\Omega_m\right\}\leq\gamma_m^{BH},\forall m\\ &C3:[1-\delta(n_m)]p_O^{(A)}(n_m)+\\ &\qquad\delta(n_m)p_O^{(D)}(n_m,k_{m,n})\leq\gamma_m^{O},\forall m,n\\ &C4:\delta\left(n_m\right)\rho\left(n_m,k_{m,n}\right)D_{n,m}\leq L_{m,n,k},\forall m,n,k\\ &C5:\sum_{m=1}^{M}\sum_{n=1}^{N_m}s\left(n_m\right)\chi_{m,n}\leq C\end{aligned} \tag{24}$$

The first set of constraint $C1$ guarantees the task offloading scheduling decision is valid. Constraint $C2$ is probabilistic backhaul constraint which ensure the probability that the sum data rate of all offloading miners associated with AP_m exceeds its backhaul capacity Ω_m isn't greater than a threshold γ_m^{BH}. In order to meet the delay requirement, $C3$ is put forward to guarantee the task delay doesn't exceed the deadline. Note that $C3$ is also probabilistic constraints to enforce the orphaning probability of each mining task does not exceed a threshold γ_m^{O}. Constraint $C4$ means the data size of each offloaded part through D2D link can not exceed the link capability $L_{m,n,k}$. Besides, we use constraint $C5$ to ensure that the sum size of all the cached content doesn't exceed the total storage capability C of the MEC server.

C. Problem Transformation

Problem $\mathcal{P}1$ is far from easy to handle due to the following three aspects:

- Since $\boldsymbol{\delta}$ and $\boldsymbol{s}$ are binary variables, and $C2$, $C3$ and $C4$ are non-convex functions of variable $(\boldsymbol{\delta},\boldsymbol{\rho})$, the feasible set of $\mathcal{P}1$ is not convex.
- There exist product relationships between $\{\delta\left(n_m\right)\}$ and $\{\rho\left(n_m,k_{m,n}\right)\}$, so the objective function in $\mathcal{P}1$ is not a convex function.
- The problem has a quite large size. If we assume that the average number of users and its D2D users in each small cell are $\dot{m}$ and $\dot{k}$, respectively, the number of variables in this problem could reach $M\dot{m}\left(\dot{k}+2\right)$, and the complexity for a centralized algorithm to find a globally optimal solution will be $O\left(M\dot{m}\left(\dot{k}+2\right)\right)^{x}$ ($x>0$, $x=1$ implies a linear algorithm while $x>1$ implies a polynomial time algorithm) even if we simply consider all the variables as binary variables. Additionally, as the network becomes denser, the computational complexity will increase dynamically in our problem.

In brief, $\mathcal{P}1$ is a mixed discrete and non-convex optimization problem, which is also well known as NP-hard problem. So a transformation and simplification of the original problem are necessary. To address the difficulty of the original problem, we can make the following transformation and relaxation:

1) ***Binary Variable Relaxation:***

We first relax the binary variables $\boldsymbol{\delta}$ and $\boldsymbol{s}$ into real value variables as $0\leq\delta\left(n_m\right)\leq1$ and $0\leq s\left(n_m\right)\leq1$, respectively. It can also be interpreted as the probability of the miner x_{n_m} choosing *mode 1* and the probability to cache the cryptographic hashs of blocks w.r.t. x_{n_m}.

2) ***Constraint Relaxation:***

To tackle the non-convexity of the probabilistic constraints $C2$, we make a relaxation as follows:

$$\begin{aligned}&\mathbb{P}\left\{\sum_{n=1}^{N_m}\left[1-\delta\left(n_m\right)\right]R^{A}\left(n_m\right)\geq\Omega_m\right\}\\ &\leq\sum_{n=1}^{N_m}\left[1-\delta\left(n_m\right)\right]\mathbb{P}\left[R^{A}\left(n_m\right)\geq\Omega_m/N_m\right],\end{aligned} \tag{25}$$

where the inequality still holds even in the worst case when all the users offload their tasks to the associated AP in each cell. Accordingly, this relaxation makes the constraint more conservative. And for convenience, we denote $p_B\left(n_m\right)=\mathbb{P}\left[R^{A}\left(n_m\right)\geq\Omega_m/N_m\right]$, which can be calculated by

$$\begin{aligned}p_B\left(n_m\right)&=\mathbb{P}\left[R^{A}\left(n_m\right)\geq\Omega_m/N_m\right]\\ &=F_{I_m}\left[\frac{p_U h_{m,n}r_{m,n}{}^{-\alpha}}{1-\exp\left(-\frac{\Omega_m}{BN_m}\right)}-\sigma^2\right].\end{aligned} \tag{26}$$

Similarly, we make a relaxation for $C3$ as

$$\begin{aligned}&p_O^{(D)}(n_m,k_{m,n})\\ &=\max_{k=1,\ldots,K_{m,n}}\mathbb{P}\left[\rho\left(n_m,k_{m,n}\right)T^{(D)}\left(n_m,k_{m,n}\right)\geq\tau_{m,n}\right]\\ &\leq\sum_{k=1}^{K_{m,n}}\rho\left(n_m,k_{m,n}\right)\mathbb{P}\left[T^{(D)}\left(n_m,k_{m,n}\right)\geq\tau_{m,n}\right].\end{aligned} \tag{27}$$

Also, let $\bar{p}_O^{(D)}\left(n_m,k_{m,n}\right)=\mathbb{P}\left[T^{(D)}\left(n_m,k_{m,n}\right)\geq\tau_{m,n}\right]$

for convenience. Following the similar steps, we can get

$$\begin{aligned}&\bar{p}_O^{(D)}\left(n_m,k_{m,n}\right)=\mathbb{P}\left(T^{(D)}\left(n_m,k_{m,n}\right)\geq\tau_{m,n}\right)\\&=\mathbb{P}\left[T^{(D,u)}\left(n_m,k_{m,n}\right)+T^{(D,e)}\left(n_m,k_{m,n}\right)\geq\tau_{m,n}\right]\\&=\mathbb{P}\left[R^D\left(n_m,k_{m,n}\right)\leq\frac{D_{m,n}}{\tau_{m,n}-T^{(D,e)}\left(n_m,k_{m,n}\right)}\right]\\&=1-F_{I_m}\left(\bar{\iota}_{m,n,k}^D\right),\end{aligned}\tag{28}$$

where $\bar{\iota}_{m,n,k}^D=\frac{\frac{p_U}{K_{m,n}}g_{m,n,k}l_{m,n,k}{}^{-\alpha}}{1-\exp\left[-\frac{D_{m,n}/B}{\tau_{m,n}-T^{(D,e)}\left(n_m,k_{m,n}\right)}\right]}-\sigma^2.$

3) ***Substitution of the Product Term:***

Although after the relaxation of the binary variables and constraints, $\mathcal{P}1$ is still intractable due to the product relationships between $\{\delta\left(n_m\right)\}$ and $\{\rho\left(n_m,k_{m,n}\right)\}$. To make the problem solvable, we make a definition:

$$\tilde{\rho}\left(n_m,k_{m,n}\right)=\delta\left(n_m\right)\rho\left(n_m,k_{m,n}\right),\forall m,n,k.\tag{29}$$

Combining the above binary variable relaxation, constraint relaxation and substitution of the product term, the original optimization problem can be transformed into (the constant part Z is removed):

$$\begin{aligned}\mathcal{P}2:\ &\max_{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\boldsymbol{s}}\quad\sum_{m=1}^{M}\sum_{n=1}^{N_m}\left[\tilde{\Psi}\left(n_m\right)+\Lambda\left(n_m\right)\right]\\ s.t.\ &C5\\ &C1':\left[1-\delta(n_m)\right]+\sum_{k=1}^{K_{m,n}}\tilde{\rho}\left(n_m,k_{m,n}\right)=1,\forall m,n\\ &C2':\sum_{n_m=1}^{N_m}\left[1-\delta\left(n_m\right)\right]p_B\left(n_m\right)\leq\gamma_m^{BH},\forall m\\ &C3':\left[1-\delta\left(n_m\right)\right]p_O^{(A)}\left(n_m\right)+\\ &\quad\sum_{k=1}^{K_{m,n}}\tilde{\rho}\left(n_m,k_{m,n}\right)\bar{p}_O^{(D)}\left(n_m,k_{m,n}\right)\leq\gamma_m^O,\forall m,n\\ &C4':\tilde{\rho}\left(n_m,k_{m,n}\right)D_{n,m}\leq L_{m,n,k},\forall m,n,k\\ &C6:\delta\left(n_m\right)\geq\tilde{\rho}\left(n_m,k_{m,n}\right),\forall m,n,k\end{aligned}\tag{30}$$

where $\tilde{\Psi}\left(n_m\right)=\left[1-\delta\left(n_m\right)\right]\left[\varsigma^{(A)}-\vartheta_eE^{(A)}\left(n_m\right)\right]+\sum_{k=1}^{K_{m,n}}\tilde{\rho}\left(n_m,k_{m,n}\right)\left[\varsigma^{(D)}-\vartheta_eE^{(D)}\left(n_m,k_{m,n}\right)\right]$ and $C6$ ensures that $\tilde{\rho}\left(n_m,k_{m,n}\right)$ doesn't exceed $\delta\left(n_m\right)$.

It's noted that $\rho\left(n_m,k_{m,n}\right)=0$ holds when $\delta\left(n_m\right)=0$. Obviously, if the miner chooses to offload its task to the nearby AP, no part of the task would be offloaded to the nearby miners. Therefore, there is a complete mapping between $\{\rho\left(n_m,k_{m,n}\right)\}$ and $\{\tilde{\rho}\left(n_m,k_{m,n}\right)\}$, which is shown in (31).

$$\rho\left(n_m,k_{m,n}\right)=\begin{cases}\tilde{\rho}\left(n_m,k_{m,n}\right)/\delta\left(n_m\right),\delta\left(n_m\right)>0\\0,\qquad otherwise\end{cases}\tag{31}$$

Remark 1. *If $\mathcal{P}2$ is feasible, it is jointly convex with respect to all the optimization variables $\boldsymbol{\delta}$, $\tilde{\boldsymbol{\rho}}$ and $\boldsymbol{s}$.*

Proof. After the above transformation and relaxation, $\left[\tilde{\Psi}\left(n_m\right)+\Lambda\left(n_m\right)\right]$ in the optimization objective becomes a linear combination of variables $\boldsymbol{\delta}$, $\tilde{\boldsymbol{\rho}}$ and $\boldsymbol{s}$, then it's obvious that the objective function, i.e., $\tilde{\Xi}=\sum_{m=1}^{M}\sum_{n=1}^{N_m}\left[\tilde{\Psi}\left(n_m\right)+\Lambda\left(n_m\right)\right]$, is a convex function. Meanwhile, in $\mathcal{P}2$, $C1'$ is a closed set and all the other constraints are linear, thus $\mathcal{P}2$ is a convex optimization problem. □

Although $\mathcal{P}2$ turns out to be a convex optimization problem that can be solved by various methods, it will incur significant computational complexity. To handle this problem, ADMM has gained great popularity as a useful tool for decomposing difficult convex optimization problems into a set of simpler sub-problems [19]. Therefore, we adopt the ADMM technique to decouple $\mathcal{P}2$, such that the computational complexity of $\mathcal{P}2$ can be significantly reduced.

D. Problem Decomposition

It can be observed from $\mathcal{P}2$ that $\boldsymbol{s}$ in $C5$ is a global variable that makes the optimization problem not separable. Here, we introduce a local copy of the global variable $\boldsymbol{s}$, thus $\mathcal{P}2$ can be further separated and solved independently at each small cell. For AP_m, we first denote $\widehat{\boldsymbol{s}}_m=\left\{\widehat{s}_m\left(n_i\right)\right\},n=1,\ldots,N_i,m,i=1,\ldots,M$ as the local copy of $\boldsymbol{s}$. So we have

$$\widehat{s}_m\left(n_i\right)=s\left(n_i\right),\forall m,n,i.\tag{32}$$

Then the equivalent global consensus version of $\mathcal{P}2$ is

$$\begin{aligned}\mathcal{P}3:\ &\max_{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m;\boldsymbol{s}}\sum_{m=1}^{M}\sum_{n=1}^{N_m}\left[\tilde{\Psi}\left(n_m\right)+\widehat{\Lambda}\left(n_m\right)\right]\\ s.t.\ &C1',C2',C3',C4',C6\\ &C5':\sum_{i=1}^{M}\sum_{n=1}^{N_i}\widehat{s}_m\left(n_i\right)\chi_{i,n}\leq C,\forall m\\ &C7:\widehat{s}_m\left(n_i\right)=s\left(n_i\right),\forall m,n,i\end{aligned}\tag{33}$$

where the consensus constraint $C7$ in $\mathcal{P}3$ ensures the consistency between the local variable $\left\{\widehat{s}_m\left(n_i\right)\right\}$ and the corresponding global one $\{s\left(n_i\right)\}$, and $\widehat{\Lambda}\left(n_m\right)=\widehat{s}_m\left(n_m\right)\left(\vartheta_c\dot{q}_{m,n}\bar{R}+\vartheta_e\bar{E}-\varpi\right)$.

For convenience, let Π_m be the feasible set of small cell m, i.e., $\Pi_m=\left\{\boldsymbol{\delta}_m,\tilde{\boldsymbol{\rho}}_m,\widehat{\boldsymbol{s}}_m|C1',C2',C3',C4',C5',C6\right\}$. Note that Π_m can be completely decoupled from other small cells. Besides, we give the local utility function of each small cell as follows:

$$\Upsilon_m=\begin{cases}-\sum_{n=1}^{N_m}\left[\tilde{\Psi}(n_m)+\widehat{\Lambda}(n_m)\right],when\left\{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m\right\}\in\Pi_m\\ \infty,\qquad otherwise\end{cases}\tag{34}$$

Then $\mathcal{P}3$ can be rewritten as

$$\begin{aligned}\mathcal{P}3':\ &\min_{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m;\boldsymbol{s}}\sum_{m=1}^{M}\Upsilon_m\left(\left\{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m\right\}\right)\\ s.t.\ &C7:\widehat{s}_m\left(n_i\right)=s\left(n_i\right),\forall m,n,i.\end{aligned}\tag{35}$$

E. Problem Solutions Through ADMM

1) ADMM Sequential Iterations:

According to [19], [20], the augmented Lagrangian of $\mathcal{P}3'$ can be expressed by

$$\begin{aligned}&\Gamma_q\left(\left\{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m\right\},\{\boldsymbol{s}\},\{\boldsymbol{\eta}_m\}\right)\\&=\sum_{m=1}^{M}\Upsilon_m\left(\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{\boldsymbol{s}}_m\right)+\sum_{m=1}^{M}\sum_{i=1}^{M}\sum_{n=1}^{N_i}\eta_m(n_i)\left[\widehat{s}_m(n_i)-s(n_i)\right]\\&+\frac{q}{2}\sum_{m=1}^{M}\sum_{i=1}^{M}\sum_{n=1}^{N_i}\left[\widehat{s}_m(n_i)-s(n_i)\right]^2,\end{aligned}\tag{36}$$

where $\boldsymbol{\eta}=\{\boldsymbol{\eta}_m\}=\{\eta_m(n_i)\}$, $m,i=1,\ldots,M$ are the Lagrange multipliers w.r.t. $C7$ in $\mathcal{P}3'$, and q is the so called penalty parameter, which is a constant parameter intended for adjusting the convergence speed. We denote the local variables as $\boldsymbol{X}=\{\boldsymbol{X}_m\}=\left\{\delta(n_m),\tilde{\rho}(n_m,k_{m,n}),\widehat{s}_m(n_i)\right\},m=1,2,\ldots,M$. The whole iteration process of the local variable $\boldsymbol{X}$, the global variables $\boldsymbol{s}$ and the Lagrange multipliers $\boldsymbol{\eta}$ includes the following three steps.

$$\{\boldsymbol{X}_m\}^{(t+1)}=\underset{\{\boldsymbol{X}_m\}}{\arg\min}\left\{\begin{array}{l}\Upsilon_m(\boldsymbol{X}_m)+\\\sum\limits_{i=1}^{M}\sum\limits_{n=1}^{N_i}[\eta_m(n_i)]^{(t)}\left\{\widehat{s}_m(n_i)-[s(n_i)]^{(t)}\right\}\\+\frac{q}{2}\sum\limits_{i=1}^{M}\sum\limits_{n=1}^{N_i}\left\{\widehat{s}_m(n_i)-[s(n_i)]^{(t)}\right\}^2\end{array}\right\},\tag{37}$$

$$\{\boldsymbol{s}\}^{(t+1)}=\underset{\{s(n_i)\}}{\arg\min}\left\{\begin{array}{l}\sum\limits_{m=1}^{M}\sum\limits_{i=1}^{M}\sum\limits_{n=1}^{N_i}[\eta_m(n_i)]^{(t)}\left\{\left[\widehat{s}_m(n_i)\right]^{(t+1)}-s(n_i)\right\}\\+\frac{q}{2}\sum\limits_{m=1}^{M}\sum\limits_{i=1}^{M}\sum\limits_{n=1}^{N_i}\left\{\left[\widehat{s}_m(n_i)\right]^{(t+1)}-s(n_i)\right\}^2\end{array}\right\},\tag{38}$$

$$\{\boldsymbol{\eta}_m\}^{(t+1)}=\{\boldsymbol{\eta}_m\}^{(t)}+q\left\{\left[\widehat{s}_m(n_i)\right]^{(t+1)}-[s(n_i)]^{(t+1)}\right\},\tag{39}$$

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

2) Local Variables Update:

As mentioned above, the local variables can be updated separately in each small cell according to (37). In other words, AP_m needs to solve the following optimization problem (after eliminating the constant terms) at iteration $(t+1)$.

$$\begin{aligned}\mathcal{P}_L:\quad&\min_{\{X_m\}}\Upsilon_m(X_m)+\\&\sum_{i=1}^{M}\sum_{n=1}^{N_i}\left\{[\eta_m(n_i)]^{(t)}\widehat{s}_m(n_i)+\frac{q}{2}\left\{\widehat{s}_m(n_i)-[s(n_i)]^{(t)}\right\}^2\right\}\\&s.t.\left\{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}},\widehat{s}_m\right\}\in\Pi_m.\end{aligned}\tag{40}$$

Observing $\mathcal{P}_L$, we can find that it is a convex problem comprising the quadratic objective function and affine inequality constraints, which can be further divided into two sub-problems $\mathcal{P}_{L1}$ and $\mathcal{P}_{L2}$ as follows.

$$\begin{aligned}\mathcal{P}_{L1}:\quad&\min_{\{\boldsymbol{\delta},\tilde{\boldsymbol{\rho}}\}}-\sum_{n=1}^{N_m}\tilde{\Psi}(n_m)\\&s.t.\quad C1',C2',C3',C4',C6.\end{aligned}\tag{41}$$

$$\mathcal{P}_{L2}:\begin{array}{l}\min\limits_{\{\widehat{s}_m\}}-\sum\limits_{n=1}^{N_m}\widehat{s}_m(n_m)\left(\vartheta_c\dot{q}_{m,n}\bar{R}-\varpi\right)+\\\sum\limits_{i=1}^{M}\sum\limits_{n=1}^{N_i}\left\{[\eta_m(n_i)]^{(t)}\widehat{s}_m(n_i)+\frac{q}{2}\left\{\widehat{s}_m(n_i)-[s(n_i)]^{(t)}\right\}^2\right\}\\s.t.\quad C5'.\end{array}\tag{42}$$

Note that the optimization problem $\mathcal{P}_{L1}$ is a Linear Programming (LP) problem while $\mathcal{P}_{L2}$ is a Quadratic Programming (QP) problem [21]. Thus, the corresponding optimal solutions can be obtained efficiently in polynomial time using standard software, such as CVX, SeDuMi, etc. [22]. As for $\mathcal{P}_{L2}$, we need to provide an initial feasible solution $\{\boldsymbol{s}\}^{(0)}$. We consider a special case where only one miner's cryptographic hash is stored in the MEC server. And we denote this miner as $x_{\bar{n}_{\bar{m}}}$, so we set $s(\bar{n}_{\bar{m}})^{(0)}=1$ and $s(n_m)^{(0)}=0,\forall n_m\neq\bar{n}_{\bar{m}}$. In this way, $C5'$ is naturally satisfied and the proposed algorithm can be well performed.

3) Global Variables and Lagrange Multipliers Update:

Now we move on to the global variables. Observe that (38) is an unconstrained quadratic problem and strictly convex due to the added quadratic regularization term in augmented Lagrangian in (36), the global solution w.r.t. $\boldsymbol{s}$ can be obtained by simply setting the gradient of $\boldsymbol{s}$ to zero as

$$\sum_{m=1}^{M}[\eta_m(n_i)]^{(t)}+q\sum_{m=1}^{M}\left\{\left[\widehat{s}_m(n_i)\right]^{(t+1)}-s(n_i)\right\}=0,\forall n,i,\tag{43}$$

and we can get

$$[s(n_i)]^{(t+1)}=\frac{1}{qM}\sum_{m=1}^{M}[\eta_m(n_i)]^{(t)}+\frac{1}{M}\sum_{m=1}^{M}\left[\widehat{s}_m(n_i)\right]^{(t+1)},\forall n,i.\tag{44}$$

Through initializing the Lagrange multipliers as zeros at iteration (t), the equation can be reduced to

$$[s(n_i)]^{(t+1)}=\frac{1}{M}\sum_{m=1}^{M}\left[\widehat{s}_m(n_i)\right]^{(t+1)},\forall n,i,\tag{45}$$

where the global variable $\boldsymbol{s}$ in each iteration is calculated by averaging out all the corresponding local copies in all the APs, and the Lagrange multipliers $\boldsymbol{\eta}$ can be obtained by (39).

4) Algorithm Stopping Criterion and Convergence:

In this part, we prove that the strong duality holds for problem $\mathcal{P}3'$ due to the following reasons: (1) According to **Remark 1**, the primal problem $\mathcal{P}2$ is convex. After the introduction of the local copy of global variables, the equivalent global consensus version of $\mathcal{P}2$, i.e., problem $\mathcal{P}3'$, is still a **convex** problem (the objective

Algorithm 1 Binary Variables Recovery

Find the variables $\dot{s}(n_m)$ which satisfy $\dot{s}(n_m) > 0.5$.
Compute the first partial derivations of augmented Lagrangian $D(n_m) = \partial\Gamma_q \big/ \dot{s}(n_m)$.
Sort all the partial derivations $D(n_m), \forall n, m$ as $D_1, D_2, \ldots, D_N$ according to the descending order, i.e., from largest to smallest. Correspondingly, the caching strategy variables are denoted as $s_1, s_2, \ldots, s_N$.
for $n = 1, 2, \ldots, N$ **do**
  Set $s_n = 1$ and $s_{n+1}, s_{n+2}, \ldots, s_N = 0$;
  If the constraint $C5$ in $\mathcal{P}1$ doesn't hold,
  Then Break.
end for
Output the recovered binary variables $\{\dot{s}(n_m)\}, \forall n, m$.

is linear to the variables, $C1'$ is a closed set, constraints $C2', C3', C4', C5', C6, C7$ are linear). (2) Please note that the objective function and all the variables of problem $\mathcal{P}3'$ are **bounded**. Therefore, when the optimal solution reaches, the inequality $\left\{\boldsymbol{\delta}^*, \tilde{\boldsymbol{\rho}}^*, \widehat{\boldsymbol{s}}_m{}^*\right\}$ reaches, the inequality $\sum_{n=1}^{N_m}\left[\tilde{\Psi}(n_m) + \widehat{\Lambda}(n_m)\right] < \infty$ holds. (3) All the constraints including $C1', C2', C3', C4', C5', C6$ and $C7$ are affine w.r.t. the optimization variables, and a feasible solution for $\mathcal{P}2$ always exists according to our assumption in **Remark 1**, thus the **strong duality holds** for problem $\mathcal{P}3'$ according to the refined Slater's condition for affine inequality constraints [21, Ch. 5.2.3].

According to [21], the objective function of $\mathcal{P}3'$ is convex, closed and proper due to its convexity. Thus, there exists saddle points for the Lagrangian (36) and the proposed ADMM-based algorithm meets the objective convergence, residual convergence as well as dual variable convergence when $t \to \infty$. In the implementation process, we adopt the rational stopping criteria proposed in [19]. Specifically, the residuals for both the primal and dual feasibility conditions of small cell m in iteration $(t+1)$ should be small enough such that $\left\|\widehat{\boldsymbol{s}}_m{}^{(t+1)} - \boldsymbol{s}^{(t+1)}\right\|_2 \le \iota_{pri}, \forall m$, and $\left\|\boldsymbol{s}^{(t+1)} - \boldsymbol{s}^{(t)}\right\|_2 \le \iota_{dual}, \forall m$, where $\iota_{pri} > 0$ and $\iota_{dual} > 0$ are the feasibility tolerances for the primal and dual feasibility conditions, respectively.

5) Binary Variables Recovery:

Recall that in Section IV-C, we relax the binary variables $\boldsymbol{\delta}$ and $\boldsymbol{s}$ into continuous ones in order to transform the original problem $\mathcal{P}1$ into a convex problem. Therefore, we need to recover the binary variables after the proposed algorithm has converged. For the purpose of maximizing the total net revenue in terms of offloading and caching, we attempt to obtain the optimal offloading scheduling scheme and cache as much content as possible. Accordingly, **Algorithm** 1 is proposed to recover the binary variables $\boldsymbol{\delta}$ and $\boldsymbol{s}$ to deal with the marginal benefit of each mining task. It should be noticed that we only take the binary variables $\boldsymbol{s}$ as an example and the same method can be applied to $\boldsymbol{\delta}$. Based on the previous discussions, the optimal offloading decision and caching strategy can be obtained while achieving the maximum net revenue.

TABLE I
SIMULATION PARAMETERS

Parameter	**Value**
The transmit power of the users P_U	0.1W
The static circuit power P_C	0.05W
The pathloss exponent α	4
The bandwidth B	20MHz
The power of noise σ^2	-174dBm/Hz
The mean value of Rayleigh fading μ	1
Input data size $D_{m,n}$	5 kbits
Delay threshold $\tau_{m,n}$	15 s
Computation worload/intensity $X_{m,n}$	18000 CPU cycles/bit
Computation energy efficiency coefficient of the processor's chip in the APs/users $\kappa_m^A/\kappa_{m,n,k}^U$	10^{-26}
Computational capability of the APs f_m^A	10–100 GHz CPU cycles/s
Computational capability of the users $f_{m,n,k}^U$	0.1–1 GHz CPU cycles/s
The unit price of energy ϑ_e	0.1 $Token/J$

V. SIMULATION RESULTS AND DISCUSSIONS

In this section, we compare the performance of the proposed distributed ADMM-based algorithm with that of the centralized scheme, *mode 0*-only scheme (all the miners offload their tasks to the nearby APs), *mode 1*-only scheme (all the miners offload their tasks to a group of D2D users) and random selection scheme (the miners randomly selects an offloading mode). Note that the simulation results obtained in this section are based on an average over a number of Monte Carlo simulations for various parameters. In the simulation part, the network coverage radius is set at 500m, where the AP density and user density are assumed to be $10^{-5}, 10^{-4}/m^2$ and $10^{-3}/m^2$, respectively. Other main simulation parameters are listed in Table I.

A. The Convergence of Proposed ADMM-based Algorithm

First, we present the convergence performance of the proposed distributed ADMM-based algorithm with different AP density and storage capability. 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. Besides, with the increase of AP density (from $\lambda_a = 10^{-5}\big/m^2$ to $\lambda_a = 10^{-4}\big/m^2$ and $\lambda_a = 10^{-3}\big/m^2$), the total revenue increases since the miners who choose *mode 0* can access to a closer AP and shorten the total delay, which would save energy consumption and

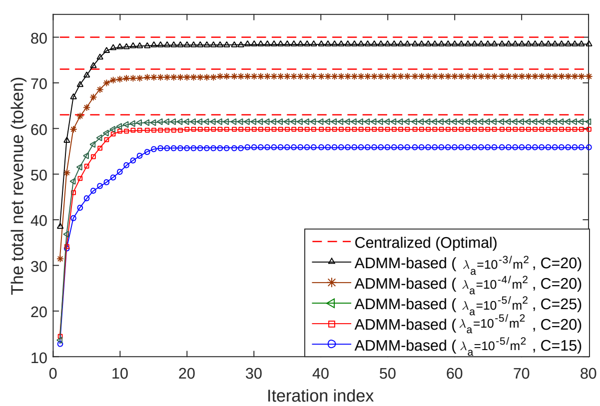

Fig. 2. Convergence performance of distributed ADMM-based algorithm with different AP density and storage capability C

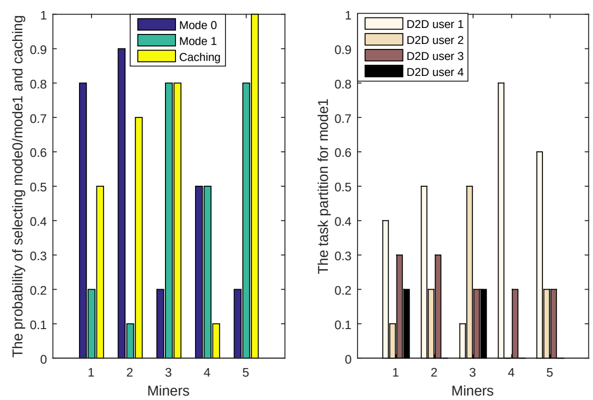

Fig. 3. Offloading/Caching decision among different miners in *small cell 1*

increase the total net revenue. 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 C. Note that the growth of the total revenue is more distinct from $C = 15$ to $C = 20$ than that from $C = 20$ to $C = 25$. This is because as the storage capacity becomes larger, there is more room to cache the cryptographic hashs of blocks for the miners, thus the total net revenue increases. And considering an extreme case when the storage capacity is large enough to accommodate all the contents of users, the net revenue curve would reach its peak and keep flat.

B. Optimal Computation Offloading and Content Caching Policy Obtained from the Proposed Algorithm

What does the optimal policy look like? Fig. 3 gives an illustration of the optimal computation offloading and

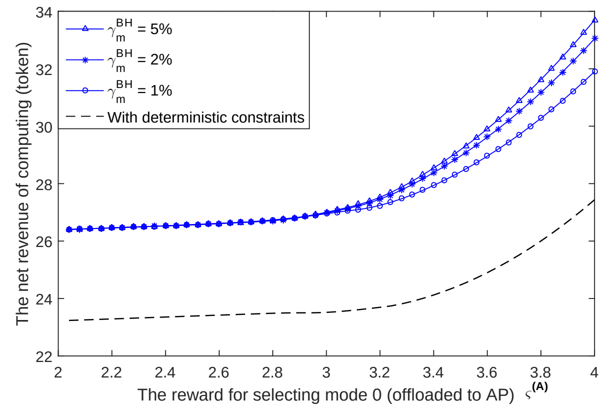

Fig. 4. The net revenue of computing vs. the reward for selecting *mode 0* with probabilistic/deterministic backhaul constraints ($\varsigma^{(D_O)} = 3$, $\varsigma^{(D_L)} = 2$, $\gamma_m^O = 1\%$)

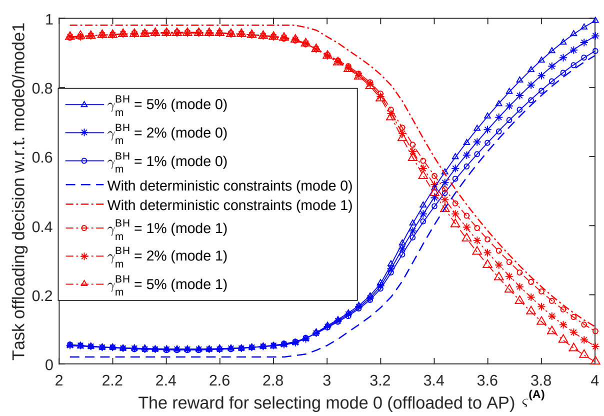

Fig. 5. The ratio of miners selecting *mode 0*/*mode 1* vs. the reward for selecting *mode 0* with probabilistic/deterministic backhaul constraints ($\varsigma^{(D_O)} = 3$, $\varsigma^{(D_L)} = 2$, $\gamma_m^O = 1\%$)

content caching scheme w.r.t. *small cell 1* obtained from our proposed algorithm. There are 5 miners in *small cell 1* and the number of D2D users varies from miner to miner. For example, there are 4 D2D users who can help computing for miner 1 while there are only 2 for miner 4, which depends on the restriction of D2D communication distance D_{ts}. It can be seen from Fig. 3 (left) that the optimal policy is a random policy before binary recovery. Like miner 1, the probabilities to select *mode 0* and *mode 1* are 0.8 and 0.2, respectively, and the probability for its cryptographic hash to be cached is 0.5. Meanwhile, Fig. 3 (right) describes the task offloading ratio among different D2D links. When miner 1 chooses *mode 1*, the task offloading ratio would be 0.4, 0.1, 0.3 and 0.2 for different D2D users. As we can see, due to different channel conditions and sizes of computation tasks of different

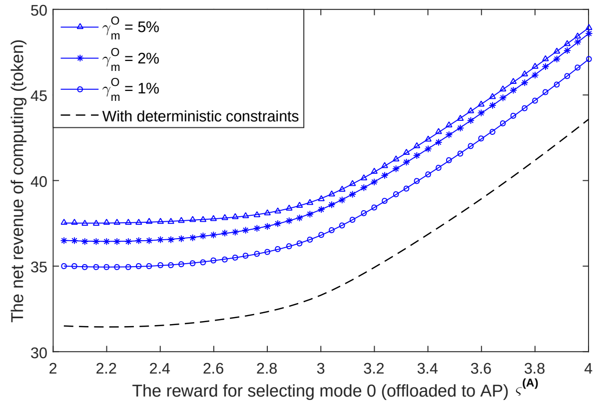

Fig. 6. The net revenue of computing vs. the reward for selecting *mode 0* with probabilistic/deterministic delay constraints ($\varsigma^{(D_O)}=3$, $\varsigma^{(D_L)}=2$, $\gamma_m^{BH}=1\%$)

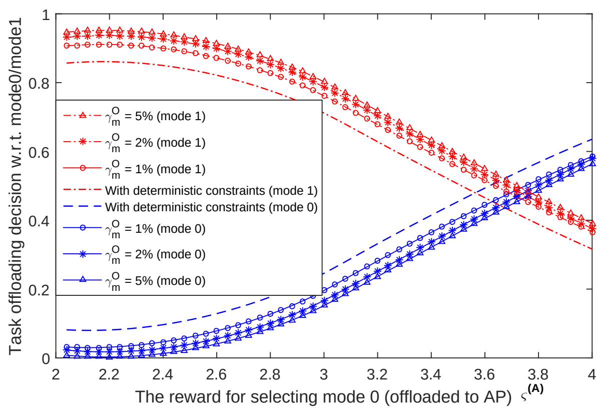

Fig. 7. The ratio of miners selecting *mode 0*/*mode 1* vs. the reward for selecting *mode 0* with probabilistic/deterministic delay constraints ($\varsigma^{(D_O)}=3$, $\varsigma^{(D_L)}=2$, $\gamma_m^{BH}=1\%$)

miners, the optimal policy among miners is not uniform, in order to reach the maximum total net revenue.

C. The Comparison Between the Results with Probabilistic and Deterministic Constraints

We compare the net revenue of computing with varying reward $\varsigma^{(A)}$ for miners to offload their tasks to APs (selecting *mode 0*) under probabilistic/deterministic backhaul and delay constraints in Fig. 4 and Fig. 6, respectively. Meanwhile, the corresponding offloading decisions are demonstrated in Fig. 5 and Fig. 7. Due to the similarity assumptions and observations between Fig. 4/Fig. 5 and Fig. 6/Fig. 7, we only make explanations for Fig. 4 and Fig. 5, and similar conclusions can be obtained from Fig. 6 and Fig. 7.

Fig. 4 shows the comparison of the net revenue of computing under probabilistic backhaul constraints $\gamma_m^{BH}=1\%, 2\%, 5\%$ and the deterministic backhaul constraints. Several interesting observations can be obtained: 1) Compared to the net revenues with deterministic backhaul constraints, the net revenues received under probabilistic constraints are obviously larger (12% higher) even with a small threshold $\gamma_m^{BH}=1\%$. The reason behind this observation is that instead of ensuring the one-hundred percent backhaul constraints (the deterministic constraints), the probabilistic ones allow for a small percentage of outage that relaxes the individual worst case, which can significantly improve the utilization of computation resources of the APs and further enhance the net revenue. 2) With a more relaxed threshold of probabilistic backhaul constraints, from $\gamma_m^{BH}=1\%$ to 2% and 5%, the increase of the gap with the deterministic constraints becomes less distinct. This is because even with more relaxed backhaul constraints, the net revenue of computing would be restricted by delay constraints. And in practice, the threshold γ_m^{BH} is always set below 5% to guarantee system performance. 3) As expected, the curves ascend with the increase of the reward $\varsigma^{(A)}$ for *mode 0* and detailed explanations are illustrated by the observations from Fig. 5.

In Fig. 5, the details of task offloading decision, i.e., the ratio of miners choosing *mode 0* and *mode 1* are presented. First, due to the more relaxed backhaul constraints, more miners prefer to select *mode 0* under probabilistic constraints while the ratio of miners choosing *mode 1* becomes less. Second, as the reward $\varsigma^{(A)}$ of *mode 0* increases, more and more miners incline to offload their mining tasks to nearby APs in order to get higher net revenue of computing, thus the ratio of miners choosing *mode 0* increases. On the contrary, less miners decide to offload their tasks through D2D links and the ratio of miners selecting *mode 1* gets smaller.

D. Investigation on the Effects of Different Parameters

In this part, the effects of several parameters including backhaul capacity, completion deadline of the mining tasks and the size of mining tasks are investigated and the relevant simulation results are depicted in Fig. 8, and Fig. 9, respectively. The revenue of our proposed ADMM-based algorithm is shown by the blue line, in comparison with revenue of centralized algorithm (in red line) and other benchmarks (*mode 0*-only scheme, *mode 1*-only scheme and random selection scheme).

Fig. 8 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 centralized algorithm is not wide. And it's no wonder that the total net revenues obtained from one single mode (*mode*

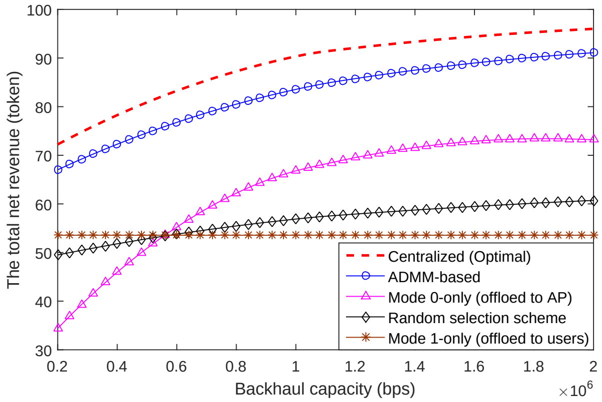

Fig. 8. The total net revenue vs. backhaul capacity ($\varsigma^{(A)}=3$, $\varsigma^{(D_O)}=3$, $\varsigma^{(D_L)}=2$, $\gamma_m^{BH}=1\%$, $\gamma_m^{O}=1\%$)

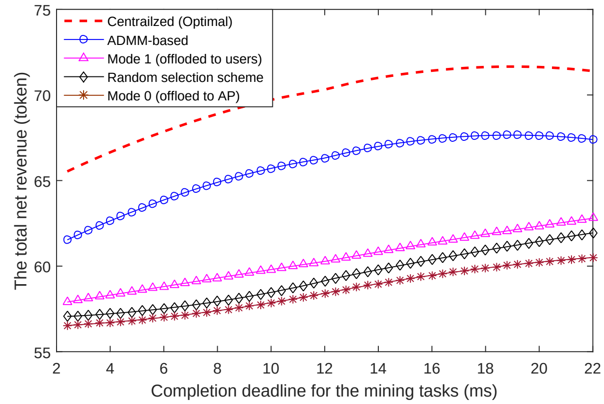

Fig. 9. The total net revenue vs. completion deadline of the mining tasks ($\varsigma^{(A)}=3$, $\varsigma^{(D_O)}=3$, $\varsigma^{(D_L)}=2$, $\gamma_m^{BH}=1\%$, $\gamma_m^{O}=1\%$)

0-only or *mode 1*-only) or random selection scheme is much lower. Because in these three schemes, the computation resources are not fully exploited and the optimal policy cannot be reached. Moreover, the total revenue received by the random selection scheme lies between that of two single modes. This is because when the miners select the mode randomly, the total revenue is made up of a part of miners in *mode 0* and the rest in *mode 1*. Additionally, all of the proposed ADMM-based algorithm, centralized algorithm, *mode 0*-only scheme and random selection scheme 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 by: when the miners choose to offload their mining tasks through D2D links, it wouldn't cost any pressure for backhaul links.

In Fig. 9, the total net revenue of our proposed ADMM-based algorithm with varying completion deadlines of mining tasks is shown compared with the centralized algorithm and the other three schemes. It can be seen that the revenue of our proposed algorithm is very close to the revenue achieved by the centralized algorithm with various completion deadlines. Besides, all the five schemes obtain higher net revenue with the increasing completion deadlines. This is because when the completion deadline increases, the orphaning probabilities of both *mode 0*-only scheme and *mode 1*-only scheme get smaller and more miners can complete the mining task successfully, thus the total net revenue increases. Additionally, it's worth noticed that *mode 1*-only scheme can receive higher revenue than *mode 0*-only and random selection scheme. In *mode 1*-only scheme, a group of D2D users can share computation resources and balance the uneven distribution of the computation workloads over users. Therefore, by reasonably assigning task offloading ratios among different D2D users, the total delay can be much smaller than that of *mode 0*-only scheme.

VI. Conclusion and Future Work

In this paper, we proposed a novel MEC-enabled wireless blockchain framework to address the challenges of PoW puzzle. In this framework, we studied the joint computation offloading decision and content caching issue, which is formulated as an optimization problem. In particular, the probabilistic backhaul and delay constraints were considered. First, we conducted the performance analysis for each offloading mode with stochastic geometry methods. Then, a distributed ADMM-based algorithm was put forward to solve the problem in an efficient and practical way. Finally, the performance evaluation of the proposed scheme was presented in comparison with the centralized solution and the other three baselines. Simulation results indicate that our proposed scheme with probabilistic constraints can achieve better performance than that with deterministic constraints, which can exploit the network resources more efficiently. Additionally, the proposed scheme is able to address the PoW puzzle issue in an effective way, which can provide some useful insights to design the computation offloading and caching schemes in MEC-enabled wireless blockchain networks. In the future work, it is interesting to consider other QoS constraints in wireless blockchain networks.

References

[1] L. Van Der Horst, K. K. R. Choo, and N. A. Le-Khac, "Process memory investigation of the bitcoin clients electrum and bitcoin core," *IEEE Access*, vol. 5, no. 99, pp. 22 385–22 398, Oct. 2017.

[2] 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 Int'l Conf. on Wireless and Mobile Comput., Netw. and Commun. (WiMob)*. Rome, Italy, Oct. 2017, pp. 139–146.

[3] A. Lei, H. Cruickshank, Y. Cao, P. Asuquo, C. P. A. Ogah, and Z. Sun, "Blockchain based dynamic key management for heterogeneous intelligent transportation systems," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1832–1843, Dec. 2017.

[4] M. 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.

[5] U. Government, "Distributed ledger technology beyond block chain," Tech. Rep., 2016.

[6] F. R. Yu, J. M. Liu, Y. He, P. B. Si, and Y. H. Zhang, "Virtualization for distributed ledger technology (vdlt)," *IEEE Access*, 2018.

[7] L. Wang and Y. Liu, "Exploring miner evolution in bitcoin network," in *Proc. Int'l Conf. on Passive and Active Netw. Measurement*. New York, NY, Mar. 2015, pp. 290–302.

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

[9] J. Beccuti, "The bitcoin mining game: On the optimality of honesty in proof-of-work consensus mechanism," Tech. Rep., Aug. 2017.

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

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

[12] 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 28th Computer Security Foundations Symposium (CSF)*. Verona, Italy, Jul. 2015, pp. 397–411.

[13] Z. H. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, "When mobile blockchain meets edge computing: Challenges and applications," *arXiv:1711.05938v1 [cs.DC]*, 2017.

[14] A. Stanciu, "Blockchain based distributed control system for edge computing," in *Proc. 21st Int'l Conf. Control Sys. and Computer Science (CSCS)*. Bucharest, May. 2013, pp. 667–671.

[15] D. Stoyan, W. S. Kendall, and J. Mecke, *Stochastic geometry and its application*. NY: John Wiley & Sons Ltd., 1995.

[16] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commun.*, vol. 64, no. 10, pp. 4268–4282, Oct. 2016.

[17] 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. 5994–600, Sept. 2017.

[18] 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.

[19] 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.

[20] L. Chen, F. R. Yu, H. Ji, G. Liu, and V. C. M. Leung, "Distributed virtual resource allocation in small-cell networks with full-duplex self-backhauls and virtualization," *IEEE Trans. Veh. Tech.*, vol. 65, no. 7, pp. 5410C–5423, Aug. 2016.

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

[22] M. Grant, S. Boyd, and Y. Ye, *Grant M, Boyd S. CVX: Matlab Software for Disciplined Convex Programming, version 1.21*. Global Optimization, Jan. 2008, pp. 155–210.