SCMA-Enabled Multi-Cell Edge Computing Networks: Design and Optimization

Pengtao Liu, Kang An, Jing Lei, Wei Liu, Yifu Sun, Gan Zheng, Fellow, IEEE and Symeon Chatzinotas, Senior Member, IEEE

Abstract—**Multi-access edge computing (MEC) is regarded as a promising approach for providing resource-constrained mobile devices with computing resources through task offloading. Sparse code multiple access (SCMA) is a code-domain non-orthogonal multiple access (NOMA) scheme that can meet the demands of multi-cell MEC networks for high data transmission rates and massive connections. In this paper, we propose an optimization framework for SCMA-enabled multi-cell MEC networks. The joint resource allocation and computation offloading problem is formulated to minimize the system utility consumption, which is defined as the weighted energy cost and latency. Due to the nonconvexity of the proposed optimization problem induced by the coupled optimization variables, we first propose an algorithm based on the block coordinate descent (BCD) method to iteratively optimize the transmit power and edge computing resources allocation, and further develop an improved low-complexity simulated annealing algorithm to solve the computation offloading and multi-cell SCMA codebook allocation problem. In addition, we extend the framework to the partial offloading case and propose an algorithm based on alternating convex search for solving the task offloading ratio. Numerical results show that the proposed multi-cell SCMA-MEC scheme achieves lower energy consumption and system latency in comparison to the orthogonal frequency division multiple access (OFDMA) and power-domain (PD) NOMA techniques.**

Index Terms—**Internet of Things, Sparse Code Multiple Access (SCMA), Multi-Access Edge Computing (MEC), binary offloading, partial offloading, resource management.**

I. Introduction

Driven by the rapid development of the Internet of Things (IoT), a large number of computation-intensive applications are emerging, such as virtual/augmented reality (VR/AR), self-driving cars, and smart home [1]. However, the limited computation capability of IoT devices remains a barrier to completing latency-critical tasks. Against this shortcoming, multi-access edge computing (MEC) can provide network-edge computing resources in base stations for resource-constrained users and offer lower latency and energy consumption for IoT devices to perform computation-intensive tasks through task offloading [2]. The research on task offloading can be divided into three aspects: task offloading decision, computation resources allocation, and mobility management [3]. Shu et al. [4] proposed a fine-grained offloading strategy to minimize the task completion time by considering dependencies between subtasks and competition among multiple edge users. In [5], dynamic voltage frequency scaling (DVFS) technology was introduced into the optimization of local computation cost. By optimizing the computational speed, transmit power, and offloading ratio of IoT devices, energy consumption and delay can be significantly reduced. The authors in [6] analyzed the joint optimization of offloading decisions and radio allocation in sliced multi-cell MEC networks. These two subproblems were solved iteratively by an alternate optimization method. The work in [7] considered MEC networks with multiple servers, where a joint task offloading and resource allocation algorithm was proposed to improve the devices' offloading benefits.

Non-orthogonal multiple access (NOMA) schemes allow plenty of devices to share the same communication resources at the same time, including time slots and frequency spectrum. It brings many advantages, such as a large number of connections, higher spectral efficiency, and lower latency [8]. Sparse code multiple access (SCMA) is a NOMA designed in the code domain, which combines multi-dimensional modulation technique with low density spread spectrum [9]. Compared to power-domain (PD) NOMA schemes, SCMA offers the benefits of coding gain and shaping gain, resulting in improved throughput and bit-error-rate (BER) [10]. Simulations demonstrated that SCMA-based systems can achieve higher throughput than PD-NOMA at the cost of more complex detections [11].

Recently, the application of NOMA into MEC for IoT scenarios has been regarded as a promising approach to provide efficient transmission and timely computation for massive mobile devices [12]–[19]. Particularly, multiple devices can simultaneously offload computation tasks to MEC servers through NOMA, further increasing the flexibility and efficiency in computation offloading. More specifically, it is shown in [12] that NOMA plays an important role in reducing the energy consumption and latency of task offloading. The latency minimization problem of NOMA assisted MEC was investigated in [13], where authors proposed an efficient layered algorithm to find the optimal offloading strategy for devices with the minimum task execution delay. The work in [14] adopted a partial offloading scheme in the hybrid

NOMA enabled MEC network, which solved the problem of minimizing energy consumption under delay restriction through power allocation, time slot scheduling, and offloading strategy. The authors in [15] proposed a NOMA scheme for edge computing awareness and designed a heuristic device clustering and resource allocation algorithm to minimize the energy consumption of MEC users. In [16], the authors exploited NOMA to maximize the energy efficiency in multi-cell MEC networks through a joint radio and computation resources allocation scheme. An SCMA-enabled MEC scheme in IoT scenarios was designed in [17], where the authors emphasize the maximization of sum rate. In [18], a joint resource allocation and task offloading optimization design in a single-cell MEC network enabled by SCMA was proposed to achieve the tradeoff between task latency and energy expenditure.

While most of the aforementioned studies have focused on the single-cell MEC network scenarios, multi-cell SCMA-based MEC systems have not yet been studied. There are several key challenges that need to be considered and addressed. First of all, in the case of multiple cells, multiple terminals can access the BS randomly by sharing the same SCMA codebook at the same time. Therefore, It is challenging to allocate SCMA codebooks to reduce inter-cell interference between devices and improve transmission rates. Secondly, considering the joint optimization of task latency and energy efficiency, we need to properly allocate local computing resources, transmit power for task offloading, and edge computing resources. Additionally, offloading too many tasks reduces the offloading benefits due to interference and competition for limited computing resources, and thus a brilliant offloading strategy is necessary. Finally, for a terminal, it needs to decide not only whether the computation task should be offloaded, but also which SCMA codebook to occupy and which MEC server to offload the task to. The coupling of SCMA codebook allocation and offloading decisions makes the problem even more challenging. Motivated by the above challenges, the contributions of this paper are presented below.

1) We design an optimization framework for SCMA-enabled multi-cell edge computing networks, where the problem of inter-cell interference and SCMA codebook reuse are analyzed. Specifically, a system utility consumption is first formulated to measure the devices' energy expenditure and task execution latency. Under the constraints of the maximum latency of tasks, the limited communication, and computation resources, an initial problem of minimizing the system utility consumption is formulated.

2) Since the formulated problem is non-convex due to the coupled variables and intractable constraints, we solve it by subdividing it into two manageable sub-problems, i.e., resource allocation and task offloading policy. As such, by using the variable substitution method, the non-convex optimization problem of resource allocation is transformed into a convex one, and the closed-form solutions under fixed power and frequency are found respectively. We further propose a joint power and computing resource allocation algorithm based on BCD to obtain the optimal power and frequency allocation. Furthermore, the task offloading and multi-cell SCMA codebook allocation based on improved simulated

annealing is proposed with low complexity.

3) We show that the system utility consumption minimization problem addressed in this paper is a unified framework of energy optimization and latency minimization, which can be achieved by changing the weighted factor. In addition, we extend to the partial offloading case and propose resource allocation algorithm and partial offloading policy based on alternating convex search.

4) Numerical simulations illustrate that the proposed joint optimization algorithm can minimize the system utility consumption (i.e. the weighted value of energy consumption and time delay), and show that the proposed multi-cell SCMA-MEC framework has significant advantages over OFDMA-MEC and PD-NOMA enabled MEC.

The rest of the paper is organized as follows. Section II designs the model of SCMA-enabled multi-cell edge computing networks and formulates the initial optimization problem. Section III proposes the joint resource allocation and task offloading scheme. Simulation results are presented in Section IV, and conclusions are made in Section V.

II. SYSTEM MODEL

The model of multi-cell SCMA-enabled MEC networks is shown in Fig. 1, where each base station (BS) is equipped with a MEC server providing task offloading services. We denote the set of IoT devices and BSs (MEC servers) in the network as $\mathcal{U} = \{1, 2, \ldots, U\}$ and $\mathcal{N} = \{1, 2, \ldots, N\}$, respectively. Multiple IoT devices can simultaneously offload computing tasks to MEC servers by SCMA. The system model consists of three parts: computing task model, task offloading, and MEC execution model. For easy reading, the key symbols are summarized in TABLE I.

A. Computing Task Model

We assume that each IoT device u has one computation task at a time, represented as T_u. The device can choose to perform the computing task locally or offload it to a nearby MEC server, i.e., binary offloading is adopted in the model [7]. The task T_u can be represented by three-tuple parameters, $(d_u, c_u, t_u^{\max})$, where d_u (in bits) specifies the amount of data for task description, c_u (in cycles) represents the number of CPU calculations for T_u, and $t_u^{\max}$ (in seconds) indicates the required completion time of the task. The local computing frequency of user u is defined as f_u^l (in cycles/s). When the computation task T_u is executed locally, the computation time and energy consumption can be expressed as

$$t_u^l = \frac{c_u}{f_u^l}, \quad (1)$$

$$E_u^l = \kappa (f_u^l)^2 c_u, \quad (2)$$

where κ denotes the energy factor, determined by the effective switching capacitance. The local computing rate f_u^l can be adjusted using DVFS technology [20] to optimize the execution time and energy consumption of IoT devices and the local computation policy is defined as $\mathcal{F}^l = \{f_u^l, u \in \mathcal{U}\}$.
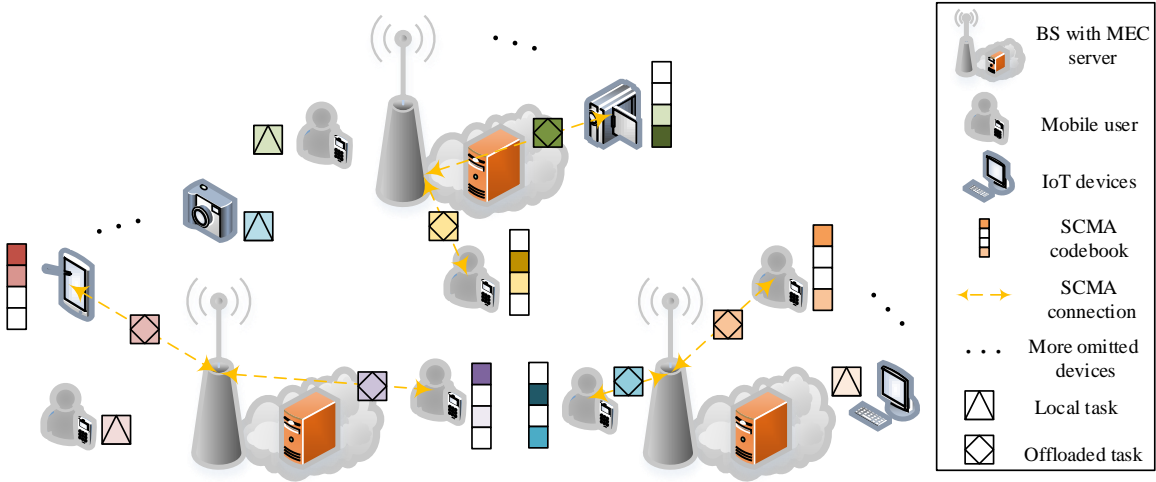
Fig. 1. A diagram of the system model, where multiple devices offload tasks to MEC servers via SCMA.

TABLE I
SUMMARY OF KEY SYMBOLS.

Symbols	Descriptions
u	Index of IoT devices
n	Index of BSs (MEC servers)
c	Index of SCMA codebooks
k	Index of subcarriers
$\mathcal{U}$	Set of IoT devices
$\mathcal{N}$	Set of BSs (MEC servers)
$\mathcal{C}$	Set of SCMA codebooks
$\mathcal{K}$	Set of subcarriers
$\mathcal{F}^l$	Local computation adjustment policy
$\mathcal{F}$	Edge computation resource allocation strategy
$\mathcal{S}$	Computation offloading strategy incorporating multi-cell SCMA codebook allocation
$\mathcal{P}$	Power allocation policy of IoT devices
T_u	Computing task of device u
d_u	The amount of data for T_u description
c_u	The number of CPU calculations for T_u
$t_u^{\max}$	The required completion time of T_u
$S_{u,c}^n$	Relationship indicator of device u, codebook c and server n
S_u^n	Offloading decision of device u
p_u	Transmission power of device u
$\mathcal{U}_{\mathrm{off}}^n$	Set of devices offloading tasks to MEC server n
f_u^l	Local computing frequency of user u
f_u^n	Allocated edge computation resource for user u at server n
f_n	Computation frequency of the MEC server n
t_u^l	Local computation time of user u
t_u^{up}	Uplink transmission time of user u
t_u^{exe}	Edge task execution time of user u
E_u^l	Local energy consumption of user u
E_u^e	Energy consumption of user u
β_t	Devices' preference to time
β_e	Devices' preference to energy
L	The number of associated pilots for each SCMA codebook
κ	Energy factor of the effective switching capacitance.
G_u^l	Local utility consumption of user u
G_u^e	Edge utility consumption of user u
G_u	Utility consumption of user u

B. Task Offloading

The devices adopt SCMA as the multiple access scheme to perform task offloading. We denote the set of codebooks and subcarriers in every single-cell SCMA system as $\mathcal{C} = \{1, 2, \ldots, C\}$ and $\mathcal{K} = \{1, 2, \ldots, K\}$, respectively. The connection between SCMA codebooks and subcarriers can be expressed by the mapping matrix $\boldsymbol{F} = (\boldsymbol{F}_1, \boldsymbol{F}_2, \cdots, \boldsymbol{F}_C)$, where $\boldsymbol{F}_c = (a_{1,c}, a_{2,c}, \cdots, a_{K,c})$ is the indicator vector for the codebook c. If $a_{k,c}^n = 1$, it means that codebook c occupies the kth subcarrier in BS n. $p_{u,c}^n$ denotes the transmit power of device u to BS n on codebook c and it is allocated to subcarrier k in a given proportion $\delta_{c,k}$, which satisfies $\sum_{\forall k \in c} \delta_{c,k}^n = 1$ [11]. Considering the characteristics of multi-cells, it should be noted that the number of codebooks is usually less than that of devices, which is different from the assumption in the single-cell SCMA network [18], [21]. The mapping between codebooks and subcarriers $a_{k,c}^n$ and the subcarrier power allocation proportion $\delta_{c,k}^n$ can be solved by the bidirectional matching principle and the water filling technique proposed in our previous work [22].

We denote $S_{u,c}^n$ as the offloading policy of users, which also incorporates multi-cell SCMA codebook allocation. If codebook c is allocated to mobile device u for offloading its computing task to the server n, then $S_{u,c}^n = 1$, otherwise, $S_{u,c}^n = 0$. Define $S_u^n = \sum_{c \in \mathcal{C}} S_{u,c}^n$ as the offloading decision of device u. And then the set of devices offloading tasks to MEC server n can be shown as $\mathcal{U}_{\mathrm{off}}^n = \{u \in \mathcal{U} \mid S_u^n = 1\}$. The offloading set of IoT devices is denoted as $\mathcal{U}_{\mathrm{off}} = \bigcup_{n \in \mathcal{N}} \mathcal{U}_{\mathrm{off}}^n$, and the computation offloading strategy can be defined as $\mathcal{S}$, a three dimensional matrix with each element $S_{u,c}^n$.

In the SCMA uplink model, we consider that multiple users in multi-cells may reuse the same codebook, and at most one codebook per user. There are L associated pilots for each codebook in a contention transmission unit (CTU) in the adopted grant-free SCMA scheme [23]. As long as the pilot sequences are different, the SCMA receiver can detect the data stream carried by the same codebook [24]. The signal to interference plus noise ratio (SINR) of user u on codebook c in BS n can be represented as [11]

$$\gamma_{u,c}^n = \frac{S_{u,c}^n \sum_{k \in \mathcal{K}} \delta_{c,k}^n p_{u,c}^n \left| h_{u,k}^n \right|^2}{I_{u,c}^n + \left(\sigma_{u,c}^n \right)^2}, \tag{3}$$

where $I_{u,c}^n = \sum_{n' \in \mathcal{N}/\{n\}} \sum_{u' \in \mathcal{U}/\{u\}} \sum_{k \in \mathcal{K}} \delta_{c,k}^{n'} p_{u',c}^{n'} \left| h_{u',k}^n \right|^2$

represents the inter-cell interference. $h_{u,k}^n$ and $(\sigma_{u,c}^n)^2$ are defined as the channel gain and noise power between the user u and BS n on the subcarrier k. From the SINR, the transmit rate of device u in BS n is expressed as

$$R_u^n = \sum_{c \in \mathcal{C}} \log_2 \left(1 + \gamma_{u,c}^n\right). \quad (4)$$

Let p_u denote the transmission power of device u, then $p_{u,c}^n = p_u S_{u,c}^n$. We define the device power allocation strategy as $\mathcal{P} = \{p_u \mid u \in \mathcal{U}_{\text{off}}\}$. Therefore, the uplink transmission time of user u when transferring the task description data d_u to MEC server n can be calculated as

$$t_u^{\text{up}} = \frac{d_u}{R_u^n}. \quad (5)$$

The energy consumption of device u can be expressed as

$$E_u^e = p_u t_u^{\text{up}}. \quad (6)$$

C. MEC Execution Model

MEC servers adopt parallel processing for multiple computation tasks. The computation frequency of the MEC server n is denoted as f_n (in cycles/s), and the computation resource allocated to every associated user by the MEC server n is quantified by f_u^n. The computation resource allocation strategy is defined as $\mathcal{F} = \{f_u^n, u \in \mathcal{U}, n \in \mathcal{N}\}$. Hence, the task execution time of user u is calculated as

$$t_u^{\text{exe}} = \frac{c_u}{f_u^n}. \quad (7)$$

D. Problem Formulation

We define the local utility consumption G_u^l as the weighted sum of task delay and energy consumption,

$$G_u^l = \beta_t t_u^l + \beta_e E_u^l = \beta_t \frac{c_u}{f_u^l} + \beta_e \kappa (f_u^l)^2 c_u, \quad (8)$$

where $\beta_e = 1 - \beta_t$. β_t and β_e represent users' preference to time and energy. The total time when offloading computation tasks to MEC servers is calculated as

$$t_u^e = t_u^{\text{up}} + t_u^{\text{exe}} = \frac{d_u}{R_u^n} + \frac{c_u}{f_u^n}. \quad (9)$$

Then, the edge utility consumption can be obtained as

$$G_u^e = \beta_t t_u^e + \beta_e E_u^e = \beta_t \left(\frac{d_u}{R_u^n} + \frac{c_u}{f_u^n}\right) + \beta_e \frac{p_u d_u}{R_u^n}. \quad (10)$$

Hence, the utility consumption of device u is expressed as

$$G_u = \sum_{n \in \mathcal{N}} (1 - \sum_{c \in \mathcal{C}} S_{u,c}^n) G_u^l + \sum_{n \in \mathcal{N}} \sum_{c \in \mathcal{C}} S_{u,c}^n G_u^e. \quad (11)$$

Under the constraints of the maximum latency of computation tasks, this paper considers local CPU rate adjustment, MEC server computing resource distribution, power allocation, multi-cell SCMA codebook allocation, and task offloading strategy to minimize the latency and energy consumption for

IoT users. The above optimization problem can be expressed as:

$$\min_{\mathcal{S}, \mathcal{F}^l, \mathcal{P}, \mathcal{F}} \sum_{u \in \mathcal{U}} \sum_{n \in \mathcal{N}} \sum_{c \in \mathcal{C}} S_{u,c}^n \left[\beta_t \left(\frac{d_u}{R_u^n} + \frac{c_u}{f_u^n} \right) + \beta_e \frac{p_u d_u}{R_u^n} \right]$$
$$+ \sum_{u \in \mathcal{U}} \sum_{n \in \mathcal{N}} \left(1 - \sum_{c \in \mathcal{C}} S_{u,c}^n \right) \left[\beta_t \frac{c_u}{f_u^l} + \beta_e \kappa \left(f_u^l \right)^2 c_u \right]$$
$$(12a)$$

$$s.t. \quad S_{u,c}^n \in \{0, 1\}, \forall u \in \mathcal{U}, c \in \mathcal{C}, n \in \mathcal{N} \quad (12b)$$

$$\sum_{n \in \mathcal{N}} \sum_{c \in \mathcal{C}} S_{u,c}^n \leqslant 1, \forall u \in \mathcal{U} \quad (12c)$$

$$\sum_{u \in \mathcal{U}} S_{u,c}^n \leqslant L, \forall c \in \mathcal{C}, \forall n \in \mathcal{N} \quad (12d)$$

$$\sum_{c \in \mathcal{C}} S_{u,c}^n t_u^e + \left(1 - \sum_{c \in \mathcal{C}} S_{u,c}^n \right) t_u^l \leqslant t_u^{\max}, \forall u \in \mathcal{U} \quad (12e)$$

$$0 \leqslant \sum_{n \in \mathcal{N}} \sum_{c \in \mathcal{C}} S_{u,c}^n p_{u,c}^n \leqslant p_u^{\max}, \forall u \in \mathcal{U} \quad (12f)$$

$$f_u^n > 0, \forall u \in \mathcal{U}_{\text{off}}, n \in \mathcal{N} \quad (12g)$$

$$\sum_{u \in \mathcal{U}_{\text{off}}^n} f_u^n \leqslant f_n, \forall n \in \mathcal{N} \quad (12h)$$

$$f_{u,\min}^l \leqslant f_u^l \leqslant f_{u,\max}^l, \forall u \in \mathcal{U}. \quad (12i)$$

The constraints in the above problem can be explained as follows. Constraint (12b) is the binary variable that represents the task offloading decision and multi-cell SCMA codebook allocation. Constraint (12c) indicates that each user can offload its task to one edge server using one SCMA codebook. Constraint (12d) implies that the same codebook can be reused at most L times by multiple devices. Constraint (12e) shows that the calculation time of local and edge execution cannot exceed the task tolerance latency $t_u^{\max}$. Constraint (12f) limits the transmision power for each device u. Constraints (12g) and (12h) ensure the computation frequency allocated to the associated devices is positive and does not exceed the server's computation capacity f_n. Constraint (12i) restricts the devices' computation frequency.

III. Joint Task Offloading and Resource Allocation Scheme

The offloading decision $\mathcal{S}$ is coupled among the objective function and multiple constraints, which makes the problem (12) belong to the mixed-integer nonlinear programming (MINLP) and difficult to be solved. Hence, we can fix $S_{u,c}^n$ simplifies the problem (12) into two tractable subproblems, i.e., resource allocation and task offloading policy.

A. Optimal Resource Allocation

1) Local Frequency Optimization: Firstly, the local CPU computing frequency of IoT devices can be adjusted to minimize the local utility consumption G_u^l. Considering the constraints $(12e)$ and $(12i)$, the optimization problem can be

transformed into:

$$\min_{\mathcal{F}^l} \quad \sum_{u \in \mathcal{U}} \beta_t \frac{c_u}{f_u^l} + \beta_e \kappa \left(f_u^l\right)^2 c_u \tag{13a}$$

$$s.t. \quad \frac{c_u}{f_u^l} \leqslant t_u^{\max}, \forall u \in \mathcal{U} \tag{13b}$$

$$(12i).$$

Problem (13) is a standard convex optimization problem that can be settled utilizing CVX tools or the method we proposed in [18], which is omitted for brevity.

2) Joint Power and Edge Computing Resource Allocation: In this subsection, we will investigate the transmit power allocation for devices and optimal computing resource allocation of MEC servers. Considering $S_{u,c}^n = 1$ and the constraints related to p_u and f_u^n, the optimization problem is expressed as follows,

$$\min_{\mathcal{P},\mathcal{F}} \quad \sum_{u \in \mathcal{U}_{\text{off}}} \beta_t \left(\frac{d_u}{R_u^n} + \frac{c_u}{f_u^n}\right) + \beta_e \frac{p_u d_u}{R_u^n} \tag{14a}$$

$$s.t. \quad \frac{d_u}{R_u^n} + \frac{c_u}{f_u^n} \leqslant t_u^{\max}, \forall u \in \mathcal{U}_{\text{off}} \tag{14b}$$

$$0 < p_u \leqslant p_u^{\max}, \forall u \in \mathcal{U}_{\text{off}} \tag{14c}$$

$$(12g)\ (12h).$$

When $S_{u,c}^n = 1$, the transmit rate of device u can be simplified as

$$R_u^n = \log_2 \left(1 + \frac{p_u \sum_{k \in \mathcal{K}} \delta_{c,k}^n \left|h_{u,k}^n\right|^2}{I_{u,c}^n + \left(\sigma_{u,c}^n\right)^2}\right). \tag{15}$$

An achievable upper bound on $I_{u,c}^n$ is defined as

$$\tilde{I}_{u,c}^n \triangleq \sum_{n' \in \mathcal{N}/\{n\}} \sum_{u' \in \mathcal{U}/\{u\}} \sum_{k \in \mathcal{K}} p_{u'}^{\max} \delta_{c,k}^{n'} \left|h_{u',k}^n\right|^2. \tag{16}$$

Similar to [7], [25], we consider $\tilde{I}_{u,c}^n$ to be a good approximation of $I_{u,c}^n$. Let $\zeta_{u,c}^n \triangleq \frac{\sum_{k \in \mathcal{K}} \delta_{c,k}^n \left|h_{u,k}^n\right|^2}{\tilde{I}_{u,c}^n + \left(\sigma_{u,c}^n\right)^2}$, and then $R_u^n = \log_2 \left(1 + \zeta_{u,c}^n p_u\right)$. We can use the substitution method and introduce a new variable $\xi_u^n \triangleq 1/R_u^n$, then $p_u = \left(2^{1/\xi_u^n} - 1\right)/\zeta_{u,c}^n$. The optimization problem (14) can be rewritten as

$$\min_{\xi,\mathcal{F}} \quad \sum_{u \in \mathcal{U}_{\text{off}}} \beta_t \left(d_u \xi_u^n + \frac{c_u}{f_u^n}\right) + \beta_e d_u \xi_u^n \frac{2^{1/\xi_u^n} - 1}{\zeta_{u,c}^n} \tag{17a}$$

$$s.t. \quad d_u \xi_u^n + \frac{c_u}{f_u^n} \leqslant t_u^{\max}, \forall u \in \mathcal{U}_{\text{off}} \tag{17b}$$

$$\xi_u^n \geqslant 1/\log_2 \left(1 + \zeta_{u,c}^n p_u^{\max}\right), \forall u \in \mathcal{U}_{\text{off}} \tag{17c}$$

$$(12g)\ (12h).$$

Lemma 1. *Problem (17) is a convex optimization problem.*

Proof: See Appendix. ∎

Since the constraint (17b) is coupled between ξ_u^n and f_u^n, we can use the BCD algorithm to iteratively solve the optimization problem for each variable block of ξ_u^n and f_u^n while fixing the remaining block to the last updated value. Based on the BCD method, the problem (17) can be solved by addressing two subproblems iteratively, i.e., power scheduling with given f_u^n and edge computing resource allocation with fixed ξ_u^n.

The power scheduling problem given f_u^n can be written as

$$\min_{\xi} \quad \sum_{u \in \mathcal{U}_{\text{off}}} \beta_t d_u \xi_u^n + \beta_e d_u \xi_u^n \frac{2^{1/\xi_u^n} - 1}{\zeta_{u,c}^n} \tag{18a}$$

$$s.t. \quad \xi_u^n \leqslant \frac{t_u^{\max} - c_u/f_u^n}{d_u}, \forall u \in \mathcal{U}_{\text{off}} \tag{18b}$$

$$(17c).$$

The root of equation $\beta_t d_u + \frac{\beta_e d_u}{\zeta_{u,c}^n}[2^{1/\xi_u^n}(1 - \ln 2/\xi_u^n) - 1] = 0$ is denoted as $\xi_u^{n,0}$, which can be solved by the bisection method. From the Appendix, we know that the objective function is convex and it decreases monotonically with the increase of ξ_u^n when $\xi_u^n < \xi_u^{n,0}$, otherwise it increases monotonically. We define $\xi_u^{n,l} = 1/\log_2 \left(1 + \zeta_{u,c}^n p_u^{\max}\right)$ and $\xi_u^{n,h} = \left(t_u^{\max} - c_u/f_u^n\right)/d_u$. Hence, the optimal solution $\xi_u^{n^*}$ is

$$\xi_u^{n^*} = \begin{cases} \xi_u^{n,l}, & \xi_u^{n,0} \leqslant \xi_u^{n,l} \\ \xi_u^{n,0}, & \xi_u^{n,l} \leqslant \xi_u^{n,0} \leqslant \xi_u^{n,h}. \\ \xi_u^{n,h}, & \xi_u^{n,0} \geqslant \xi_u^{n,h} \end{cases} \tag{19}$$

The optimal power scheduling can be obtained by $p_u^* = \left(2^{1/\xi_u^{n^*}} - 1\right)/\zeta_{u,c}^n$.

The edge computing resource allocation with fixed ξ_u^n is rephrased as

$$\min_{\mathcal{F}} \quad \sum_{u \in \mathcal{U}_{\text{off}}} \beta_t c_u/f_u^n \tag{20a}$$

$$s.t. \quad f_u^n \geqslant \frac{c_u}{t_u^{\max} - d_u \xi_u^n} \tag{20b}$$

$$(12g)\ (12h).$$

Since the problem (20) is convex, it can be solved by the Karush-Kuhn-Tucker (KKT) condition. Specifically, the Lagrangian of the problem (20) is calculated as

$$\mathcal{L}\left(f_u^n, \lambda, \mu_u\right) = \sum_{u \in \mathcal{U}_{\text{off}}} \beta_t c_u/f_u^n + \lambda \left(\sum_{u \in \mathcal{U}_{\text{off}}} f_u^n - f_n\right) + \sum_{u \in \mathcal{U}_{\text{off}}} \mu_u \left(\frac{c_u}{t_u^{\max} - d_u \xi_u^n} - f_u^n\right), \tag{21}$$

where λ and μ_u are non-negative Lagrange multipliers. The optimal $f_u^{n^*}$, λ^* and μ_u^* should satisfy the following equations,

$$\frac{\partial \mathcal{L}}{\partial f_u^n} = -\frac{\beta_t c_u}{(f_u^{n^*})^2} + \lambda^* - \mu_u^* = 0, \tag{22}$$

$$\mu_u^* \left(\frac{c_u}{t_u^{\max} - d_u \xi_u^n} - f_u^{n^*}\right) = 0, \tag{23}$$

$$\sum_{u \in \mathcal{U}_{\text{off}}} \sqrt{\frac{\beta_t c_u}{\lambda^* - \mu_u^*}} = f_n. \tag{24}$$

Define the set of $\mu_u = 0$ is $\mathcal{U}_0$, and the optimal edge computing resource allocation solution $f_u^{n^*}$ can be written as

$$f_u^{n^*} = \begin{cases} \dfrac{c_u}{t_u^{\max} - d_u \xi_u^n}, & u \in \mathcal{U}_{\text{off}}^n \backslash \mathcal{U}_0 \\ \dfrac{(f_n - \sum_{u \in \mathcal{U}_{\text{off}}^n \backslash \mathcal{U}_0} f_u^{n^*})\sqrt{c_u}}{\sum_{u \in \mathcal{U}_0} \sqrt{c_u}}, & u \in \mathcal{U}_0 \end{cases} \tag{25}$$

The optimal allocation strategy of edge computing resources can be obtained from (25). In each MEC server, the computation resources are evenly distributed with the square root of the workload $\sqrt{c_u}$. For those devices which can not satisfy the delay demand, f_u^{n*} is set to $c_u/(t_u^{\max} - d_u\xi_u^n)$. The rest of computational frequency is equally allocated w.r.t $\sqrt{c_u}$. The above process is repeated until all the devices meet the latency requirement, and then the optimal solution f_u^{n*} is obtained.

In the joint power and computing resource allocation algorithm based on BCD, the transmit power and edge computing resource allocation vectors are updated iteratively, as summarized in Algorithm 1. By performing Algorithm 1 at each base station, the optimal transmission power $\mathcal{P}^*$ and edge resource allocation strategy $\mathcal{F}^*$ could be obtained.

Algorithm 1: Joint Power and Computing Resource Allocation Algorithm Based on Block Coordinate Descent

---

Input: β_t, β_e, $p_u^{\max}$, f_n, $\zeta_{u,c}^n$, T_u, $\mathcal{U}_{\text{off}}^n$

Output: Optimal transmit power $\mathcal{P}_n^*$ and edge computing resource allocation strategy $\mathcal{F}_n^*$ at BS n.

1 **Initialize:** Set $k = 0$, convergence tolerance $\epsilon_1, \epsilon_2 > 0$, use the bisection method to get $\xi_u^{n,0}$ and find initial feasible solution $(\xi_u^n)^0 = \xi_u^{n,0}$.

2 **repeat**

3   Compute $(f_u^n)^k$ by (25) with given $(\xi_u^n)^k$;

4   Compute $(\xi_u^n)^{k+1}$ by (19) with given $(f_u^n)^k$;

5     $k = k + 1$;

6 **until** $\left\|(f_u^n)^k - (f_u^n)^{k-1}\right\| \leq \epsilon_1$, $\left\|(\xi_u^n)^k - (\xi_u^n)^{k-1}\right\| \leq \epsilon_2$;

7 Compute optimal transmit power
$p_u^k = \left(2^{1/(\xi_u^n)^k} - 1\right)/\zeta_{u,c}^n$.

8 Then, set p_u^k and $(f_u^n)^k$, $\forall u \in \mathcal{U}_{\text{off}}^n$ as the optimal transmit power $\mathcal{P}_n^*$ and edge computing resource allocation strategy $\mathcal{F}_n^*$ at BS n.

B. Task Offloading and Multi-cell SCMA Codebook Allocation

With a fixed task offloading and multi-cell SCMA codebook allocation decision $S_{u,c}^n$, we get the optimal solutions for the power and computation resources allocation $\mathcal{P}^*$, $\mathcal{F}^*$ $\mathcal{F}^{l^*}$. Therefore, the optimization problem is formulated as

$$\min_{\mathcal{S}} \sum_{u\in\mathcal{U}} \sum_{n\in\mathcal{N}} \sum_{c\in\mathcal{C}} S_{u,c}^n G_u^e\left(\mathcal{P}^*, \mathcal{F}^*\right)$$
$$+ \sum_{u\in\mathcal{U}} \sum_{n\in\mathcal{N}} \left(1 - \sum_{c\in\mathcal{C}} S_{u,c}^n\right) G_u^l\left(\mathcal{F}^{l^*}\right) \quad (26)$$
$$s.t. \quad (12b)\ (12c)\ (12d).$$

Using the exhaustive search method on all possible task offloading strategies and multi-cell SCMA codebook allocation is a direct and brute approach for problem (26). However, the total number of candidate solutions is $2^{U \times N \times C}$, and the exponential complexity makes the exhaustive search method impractical. Therefore, a low complexity algorithm based on improved simulated annealing is proposed.

The simulated annealing (SA) algorithm is derived from the process of crystal cooling. When the solid is heated and then cooled, as the temperature decreases slowly, the atoms are arranged into a certain shape, forming regular crystals with high density and low energy, which corresponds to the global optimal solution in the algorithm. The SA algorithm consists of two parts: the metropolis criterion [26] and the annealing process. Instead of using fully determined rules, the metropolis criterion accepts new states with probability. Specifically, the improved simulated annealing algorithm would update new solutions from the solutions of the last iteration in the following four ways:

(1) Change task offloading decision for device u.

(2) Randomly select another MEC server of the three closest MEC servers and find an SCMA codebook that has been allocated less than L devices.

(3) Select another SCMA codebook that has been allocated less than L devices.

(4) Exchange the MEC server and SCMA codebook policy with another device.

If the value of the new objective function (G^{new}) is less than the previous value (G^{old}), then the new solution ($\mathcal{S}^{\text{new}}$) is accepted. Otherwise, the algorithm accepts $\mathcal{S}^{\text{new}}$ as the new solution with probability $e^{-\Delta G/T}$. Finally, as the temperature decreases, the algorithm gradually converges to the global optimal solution. At each iteration, it needs to perform Algorithm 1 with $\mathcal{S}^{\text{new}}$ in each base station, and obtain the optimal transmission power $\mathcal{P}^*$ and edge resource allocation strategy $\mathcal{F}^*$. The specific steps for task offloading and multi-cell SCMA codebook allocation are described in Algorithm 2.

C. Analysis of Special Cases

The minimization problem of the weighted energy consumption and latency is studied in Problem 12. In this subsection, we analyze two special cases of the above problem. The analysis explains the generality and applicability of the studied problem and proposed algorithms. When the weighting factor β_t is equal to 1, the latency minimization problem can be formulated as follows,

$$\min_{\mathcal{S},\mathcal{F}^l,\mathcal{P},\mathcal{F}} \sum_{u\in\mathcal{U}} \sum_{n\in\mathcal{N}} \sum_{c\in\mathcal{C}} S_{u,c}^n \left(\frac{d_u}{R_u^n} + \frac{c_u}{f_u^n}\right)$$
$$+ \sum_{u\in\mathcal{U}} \sum_{n\in\mathcal{N}} \left(1 - \sum_{c\in\mathcal{C}} S_{u,c}^n\right) \frac{c_u}{f_u^l}$$
$$s.t. \quad (12b)\ (12c)\ (12d)\ (12e)\ (12f)\ (12g)\ (12h)\ (12i).$$
$$(27)$$

Similarly, when the parameter $\beta_e = 1$, the energy minimization problem can be rewritten as,

$$\min_{\mathcal{S},\mathcal{F}^l,\mathcal{P}} \sum_{u\in\mathcal{U}} \sum_{n\in\mathcal{N}} \sum_{c\in\mathcal{C}} S_{u,c}^n \frac{p_u d_u}{R_u^n}$$
$$+ \sum_{u\in\mathcal{U}} \sum_{n\in\mathcal{N}} \left(1 - \sum_{c\in\mathcal{C}} S_{u,c}^n\right) \kappa \left(f_u^l\right)^2 c_u \quad (28)$$
$$s.t. \quad (12b)\ (12c)\ (12d)\ (12e)\ (12f)\ (12i).$$

These two problems are special cases of the problem studied in this paper. Obviously, the algorithms proposed can be directly used to solve the problems of minimizing task execution delay and energy consumption.

Algorithm 2: Task Offloading and Multi-cell SCMA Codebook Allocation Based on Simulated Annealing

---

Input: $\mathcal{U}$, $\mathcal{C}$, $\mathcal{N}$, $\mathcal{K}$, β_t, $p_u^{\max}$, f_n, T_u, G_u^{l*}, T, Minimum temperature $T_{\min}$, Reduction factor α

Output: The optimal task offloading and multi-cell SCMA codebook allocation decision $\mathcal{S}^*$

1 Initialize: Randomly generate an initial feasible solution $\mathcal{S}^{\text{old}}$ and calculate $G^{\text{old}} = \sum_{u \in \mathcal{U}} G_u$.

2 while $T > T_{\min}$ **do**

3   Generate new solutions $\mathcal{S}^{\text{new}}$ from $\mathcal{S}^{\text{old}}$ in one of several following ways based on the probability $rand \in (0, 1)$.

4   **if** $rand < 0.2$ **then**

5   Change task offloading decision for device u

6   **else**

7   **if** $rand < 0.4$ **then**

8   Randomly select another MEC server of the three closest servers and find an SCMA codebook that has been allocated less than L devices.

9   **else**

10   **if** $rand < 0.8$ **then**

11   Select another SCMA codebook that has been allocated less than L devices.

12   **else**

13   Exchange the MEC server and SCMA codebook policy with another device.

14   **end**

15   **end**

16   **end**

17   Perform Algorithm 1 with $\mathcal{S}^{\text{new}}$ in each base station, and obtain the optimal transmission power $\mathcal{P}^*$ and edge resource allocation strategy $\mathcal{F}^*$.

18   Calculate G^{new}, and $\Delta G = G^{\text{new}} - G^{\text{old}}$.

19   **if** $\Delta G \leqslant 0$ **then**

20 $\mathcal{S}^{\text{old}} = \mathcal{S}^{\text{new}}$, $G^{\text{old}} = G^{\text{new}}$

21   **else**

22   **if** $e^{-\Delta G/T} > rand(0, 1)$ **then**

23 $\mathcal{S}^{\text{old}} = \mathcal{S}^{\text{new}}$, $G^{\text{old}} = G^{\text{new}}$

24   **end**

25   **end**

26 $T = \alpha * T$

27 end

D. Extension to Partial Offloading

This subsection extends the proposed algorithms to partial offloading. In this case, we assume that task T_u can be divided into two arbitrarily sized blocks, one of which is executed locally by the UE itself and the other (δ_u) can be offloaded to the edge server.

When the number of CPU cycles processed locally is $(1 - \delta_u)c_u$, the execution time and energy consumption can be expressed as $t_u^l = (1 - \delta_u)c_u/f_u^l$ and $E_u^l = \kappa(f_u^l)^2(1 - \delta_u)c_u$, respectively. When computation offloading is performed, the total time spent can be given as $t_u^c = \delta_u d_u/R_u^n + \delta_u c_u/f_u^n$, and the energy consumption is expressed as $E_u^c = \delta_u p_u d_u/R_u^n$.

Since parallel computing is performed simultaneously on the device and the MEC server, the delay of the overall task should be the maximum value of the two, i.e., $\max\left\{(1 - \delta_u)c_u/f_u^l, \delta_u d_u/R_u^n + \delta_u c_u/f_u^n\right\}$. In addition, the overall energy consumption can be expressed as $\kappa(f_u^l)^2(1 - \delta_u)c_u + \delta_u p_u d_u/R_u^n$.

Therefore, similar to binary offloading, the tradeoff problem between latency and energy consumption can be formulated as

$$\min_{\mathcal{S}, \delta, \mathcal{F}^l, \mathcal{P}, \mathcal{F}} \sum_{u \in \mathcal{U}} \beta_t \max\left\{\sum_{n \in \mathcal{N}} \sum_{c \in \mathcal{C}} S_{u,c}^n \delta_u \left(\frac{d_u}{R_u^n} + \frac{c_u}{f_u^n}\right), (1 - \delta_u)\frac{c_u}{f_u^l}\right\}$$
$$+ \sum_{u \in \mathcal{U}} \beta_e \left[\sum_{n \in \mathcal{N}} \sum_{c \in \mathcal{C}} S_{u,c}^n \delta_u \frac{p_u d_u}{R_u^n} + (1 - \delta_u)\kappa \left(f_u^l\right)^2 c_u\right]$$
$$\tag{29a}$$

$$s.t. \quad \delta_u \in [0, 1], \forall u \in \mathcal{U} \tag{29b}$$

$$\max\left\{\delta_u t_u^e, (1 - \delta_u) t_u^l\right\} \leqslant t_u^{\max}, \forall u \in \mathcal{U} \tag{29c}$$

$$(12b) \ (12c) \ (12d) \ (12f) \ (12g) \ (12h) \ (12i).$$

The constraint (29b) represents the offloading policy δ_u is a continuous variable within $[0, 1]$. Constraint (29c) implies that the task execution time cannot exceed the maximum task execution latency $t_u^{\max}$. Note that $S_{u,c}^n$ in this section only represents the relationship between IoT devices, BS, and multi-cell SCMA codebook allocation. δ_u stands for offloading policy.

When fixing $S_{u,c}^n$, the optimization problem is transformed as

$$\min_{\delta, \mathcal{F}^l, \mathcal{P}, \mathcal{F}} \sum_{u \in \mathcal{U}} \beta_t \max\left\{\delta_u \left(\frac{d_u}{R_u^n} + \frac{c_u}{f_u^n}\right), (1 - \delta_u)\frac{c_u}{f_u^l}\right\}$$
$$+ \sum_{u \in \mathcal{U}} \beta_e \left[\delta_u \frac{p_u d_u}{R_u^n} + (1 - \delta_u)\kappa \left(f_u^l\right)^2 c_u\right] \tag{30a}$$

$$s.t. \quad 0 \leqslant p_{u,c}^n \leqslant p_u^{\max}, \forall u \in \mathcal{U} \tag{30b}$$

$$(12g) \ (12h) \ (12i) \ (29b) \ (29c).$$

Using the same idea in subsection III-A, let $\xi_u^n = 1/R_u^n$, the optimization problem (30) can be rewritten as

$$\min_{\delta, \mathcal{F}^l, \xi, \mathcal{F}} \sum_{u \in \mathcal{U}} \beta_t \max\left\{\delta_u \left(d_u \xi_u^n + \frac{c_u}{f_u^n}\right), (1 - \delta_u)\frac{c_u}{f_u^l}\right\}$$
$$+ \sum_{u \in \mathcal{U}} \beta_e \left[\delta_u d_u \xi_u^n \frac{(2^{1/\xi_u^n} - 1)}{\zeta_{u,c}^n} + (1 - \delta_u)\kappa \left(f_u^l\right)^2 c_u\right]$$
$$\tag{31a}$$

$$s.t. \quad \delta_u \left(d_u \xi_u^n + \frac{c_u}{f_u^n}\right) \leqslant t_u^{\max}, \forall u \in \mathcal{U} \tag{31b}$$

$$(1 - \delta_u)c_u/f_u^l \leqslant t_u^{\max}, \forall u \in \mathcal{U} \tag{31c}$$

$$\xi_u^n \geqslant 1/\log_2\left(1 + \zeta_{u,c}^n p_u^{\max}\right), \forall u \in \mathcal{U} \tag{31d}$$

$$(12g) \ (12h) \ (12i) \ (29b).$$

Lemma 2. *[27] If $f_1(x)$ and $f_2(x)$ are convex, then their pointwise maximum with $\textbf{dom} f = \textbf{dom} f_1 \cap \textbf{dom} f_2$, defined by $f(x) = \max\{f_1(x), f_2(x)\}$ is also convex.*

Even if the transformed problem (31) is still non-convex, it is noted that when δ is fixed, the problem is a standard convex optimization problem, which can be proved by using *Lemma 2*.

The interior-point method and CVX tools can be used to solve it. When $\mathcal{F}^l, \xi, \mathcal{F}$ are fixed, that problem can be transformed to be a linear programming problem w.r.t δ, which is easy to be settled. Then, the optimal solutions can be solved by an alternating convex search algorithm, shown in Algorithm 3. When the optimal $\delta, \mathcal{F}^l, \mathcal{P}, \mathcal{F}$ are solved, the improved simulated annealing algorithm proposed in Algorithm 2 can be used to solve $\mathcal{S}$.

Fig. 2. Trend of the system utility consumption versus the number of iterations and cells.

Algorithm 3: Resource Allocation Algorithm and Partial Offloading Policy Based on Alternate Convex Search

---

Input: β_t, β_e, T_u, $p_u^{\max}$, f_n, $\zeta_{u,c}^n$, $f_{u,\min}^l$, $f_{u,\max}^l$, convergence tolerance ϵ.

Output: Optimal resource allocation $\mathcal{F}^{l^*}, \mathcal{P}^*, \mathcal{F}^*$ and partial task offloading strategy δ^*.

1 **Initialize:** Set iterxation index $k = 1$ and the initial value of δ^1 equals 0.5.

2 **repeat**

3   Substitute δ^k into the problem (31) to obtain $\left((f^l)^{k+1}, (\xi_u^n)^{k+1}, (f_u^n)^{k+1}\right)$ by solving the convex optimization problem using the interior point method.

4   Updata δ^{k+1} based on $\left((f^l)^{k+1}, (\xi_u^n)^{k+1}, (f_u^n)^{k+1}\right)$ by solving the linear programming problem.

5    $k = k + 1$;

6    $G(k-1) = G_u\left(\delta_u^{k-1}, (f^l)^{k-1}, (\xi_u^n)^{k-1}, (f_u^n)^{k-1}\right)$;

7    $G(k) = G_u\left(\delta_u^k, (f^l)^k, (\xi_u^n)^k, (f_u^n)^k\right)$;

8 **until** $\|G(k) - G(k-1)\| \leq \epsilon$;

9 Compute optimal transmit power $p_u^* = \left(2^{1/(\xi_u^n)^k} - 1\right)/\zeta_{u,c}^n.$

IV. SIMULATION RESULTS AND ANALYSIS

In this section, representative simulation results are shown to evaluate the performance of the multi-cell SCMA-enabled MEC design. Specifically, N BSs and U users are randomly distributed in an area, and the adjacent BSs are set 100m apart from each other. The number of SCMA codebooks is set as $C = 6$. For each codebook, the number of associated pilot sequences is $L = 2$ [24]. The channels between the users and BSs are generated by a distance dependent path loss, modeled as $PL(\text{dB}) = 140.7 + 36.7 log_{10}(d)_{[km]}$ [7] with 8dB log-normal multipath shadowing. The maximum transmit power of users $p_u^{\max}$ is 23dBm. We set the system bandwidth B as 20MHz. Every BS is equipped with a 20GHz MEC server. The task description data d_u and the workload c_u are randomly distributed in $[300, 1200]$KB and $[0.2, 1]$Gcycles, respectively [28]. The maximum latency $t_u^{\max}$ is between 0.5s and 2s. The terminal device can adjust its computation rate as 0.2GHz~1GHz and $\kappa = 5 \times 10^{-27}$ [29]. The termination temperature and cooling speed of the simulated annealing algorithm are set as $T_{\min} = 0.0001$ and $\alpha = 0.98$. The simulation results were obtained on a laptop with equipped with i7-8550u and 16GB RAM.

The convergence performance of the proposed task offloading and multi-cell SCMA codebook allocation based on improved simulated annealing algorithm is illustrated in Fig. 2.
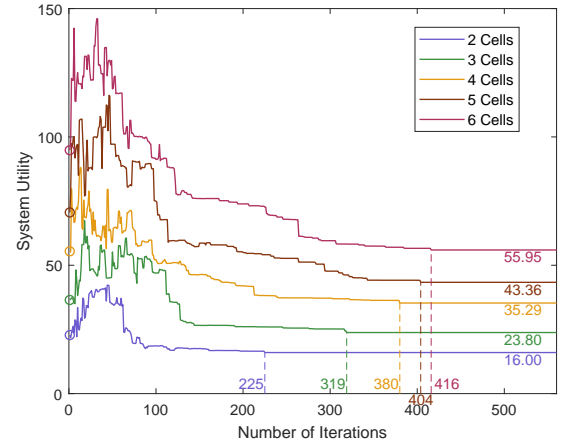
We consider that each BS can serve an average of 6 devices, and when the cell number increases from 2 to 6, the number of devices grows from 12 to 36. In all cases, the utility value experiences a period of fluctuation due to label 13 in Algorithm 2, then decreases, and finally converges with the increase of iterations. It can be seen from the figure that the convergence can be achieved within 500 iterations in all cells. When the number of cells grows from 2 to 6, the number of iterations reaching equilibrium goes from 225 to 416. Compared with the exhaustive search method, which requires $2^{12 \times 2 \times 6} \sim 2^{36 \times 6 \times 6}$ iterations [7], the complexity of proposed algorithm is significantly reduced.

To demonstrate the superiority of the proposed SCMA-MEC scheme, we introduce two reference schemes, which are the same as the proposed scheme except that the offloading process adopts OFDMA [28] and PD-NOMA [11] techniques, respectively. Fig. 3 shows the energy consumption and latency comparison between the proposed SCMA-MEC scheme and the multi-cell edge computing networks enabled by PD-NOMA and OFDMA. In PD-NOMA, each subcarrier can be assigned to at most two users simultaneously. It can be seen that the overall execution delay and energy consumption increase with the number of cells in all schemes. Compared with OFDMA and PD-NOMA, the energy cost and delay of the multi-cell SCMA-MEC network are significantly reduced, and the gap increases with the number of cells.

In the case of two cells, the average energy consumption of SCMA is 0.064J, which is reduced by 0.072J and 0.215J in comparison to PD-NOMA and OFDMA. The average latency equals 0.478s, reduced by 0.229s and 0.347s, respectively. When the cell number is six, the average energy expenditure of SCMA is 0.267J, which is 36.6% and 47.0% lower than that of PD-NOMA and OFDMA, and the time delay comes to 0.589s, with a relative reduction of 22.4% and 34.6%.

Fig. 4 depicts the impact of multi-cell SCMA codebook allocation on average system latency and offloading efficiency. The random SCMA codebook allocation uses a fixed user order algorithm [21]. We can see that the proposed algorithm has an advantage in reducing the latency compared with the random SCMA codebook allocation. In two cells, the
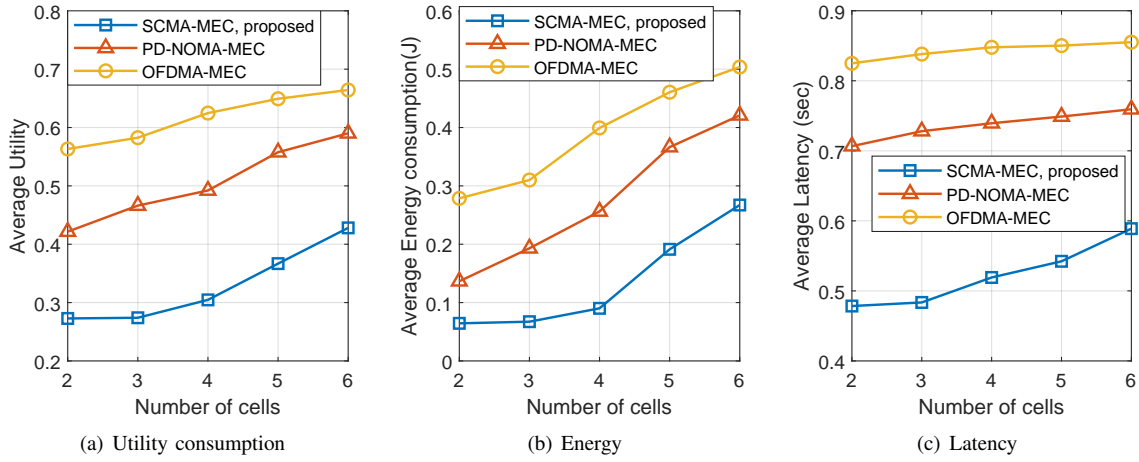
Fig. 3.   Comparison of average (a) utility consumption, (b) energy, and (c) latency while varying the number of cells using different access schemes.
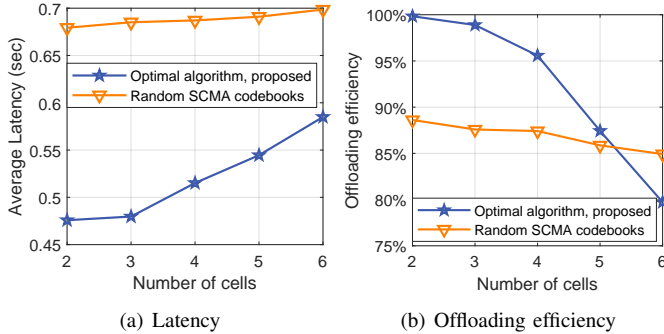

(a) Latency

(b) Offloading efficiency

Fig. 4.   Impact of multi-cell SCMA codebook allocation on the (a) average latency and (b) offloading efficiency, compared to random SCMA codebooks.

average delay of the two schemes is 0.476s and 0.679s, respectively, and the reduction is 0.203s. In the case of six cells, the average system latency of the proposed algorithm is 0.585s, which is 0.114s lower than that of random SCMA codebook allocation. It can be deduced from the data that the proposed algorithm can reduce the latency by about 16.3%. Offloading efficiency is defined as the ratio of the number of offloaded devices to the total number of devices. Due to the inter-cell interference of multiple cells, the transmit rate decreases with the growth of cell numbers, which leads to less "willingness" of devices to offload tasks. Therefore, the offloading efficiency drops from 99.8% to 79.7%. Since the random codebook allocation ignores the interference between different cells and only considers the competition of multiple devices for computing resources, the offloading efficiency decreases slightly. Therefore, compared with random codebook allocation, our algorithm can achieve more channel-adapted offloading efficiency and lower system latency.

The comparison of full offloading, local computing, and optimal offloading is shown in Fig. 5. Overall, in terms of average utility consumption and latency, the optimal offloading is smaller than that of full offloading and local computing. With a small number of cells (i.e., 2, 3, 4), the average utility consumption of optimal offloading is similar to that of full offloading, suggesting that edge computing plays an important role when interference between devices is relatively low. As

the number of cells gradually increases, the performance of the proposed scheme improves more significantly. With a cell count of 6, the average utility consumption of optimal offloading is 0.401, which is 0.641 (61.5%) and 0.241 (37.5%) lower than that of local computing and full offloading, respectively, demonstrating the advantages of the proposed task offloading algorithm. In terms of energy consumption, the optimal offloading is similar to full offloading and much smaller than that of local computing. As the number of cells increases, the energy consumption and latency of local computing remain almost constant. This is due to the fact that devices' computing is independent of the cell count. When considering full offloading, the interference between devices increases as the number of cells increases, resulting in longer average latency. At a cell count of 2, the average local computing delay is $1.02s$, full offloading latency and optimal offloading delay are close to each other, which are $0.465s$ and $0.452s$ respectively. In this case, almost all of the optimal offloading options are to be fully offloaded to MEC servers for computation. When the number of cells is 6, the full offloading delay rises to $1.118s$ due to interference, which exceeds the local calculation delay. The optimal offload decision reduces the latency to $0.552s$, which is 45.9% and 50.6% lower than the local and full offload approaches respectively.
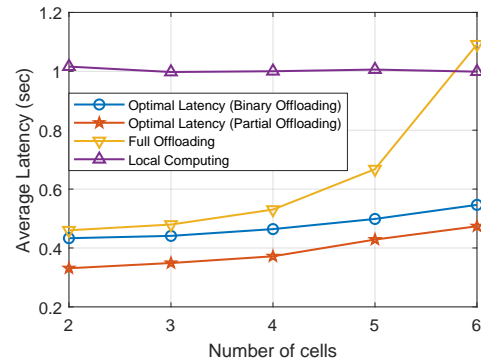

Fig. 6.   Comparison of average latency versus different offloading schemes.

Fig.6 compares the differences between four computing schemes (binary offloading, partial offloading, full offloading,

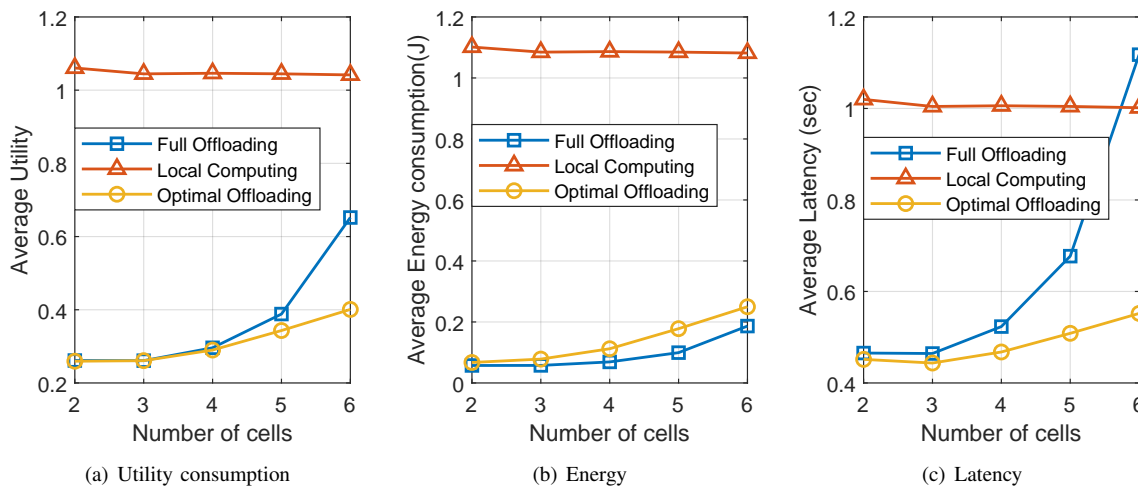(a) Utility consumption   (b) Energy   (c) Latency

Fig. 5.   Comparison of average (a) utility, (b) energy consumption, and (c) latency when considering full offloading, local computing and optimal offloading.

local computing) with the minimum latency as the optimization objective. It can be seen that the partial offloading case outperforms the other three schemes in general. When the cell number equals 2, the optimal delay of binary offloading is 0.433s, which is close to full offloading. The average latency of partial offloading is 0.331s, which is 0.102s (23.6%) lower than that of binary offloading. This is because the partial offloading scheme can upload a portion of the task, which can utilize the computing power of both local devices and MEC servers to reduce task execution latency. It is also noted that the advantage of partial offloading over binary offloading decreases as the number of cells increases, with a difference of 0.073s for a cell count of 6. This is due to the fact that interference between devices reduces the percentage of devices offloaded, resulting in a lower advantage of partial offloading.

the figure that as β_t increases, the average delay is reduced at the cost of gradually increasing the energy consumed by devices. Furthermore, as the number of cells in the system grows, the average energy consumption and latency of the devices increases. When the optimization objective is energy minimization, the average energy consumption of the device is 0.06J in the case of three cells, and 0.167J when cells count up to 6, at an increase of 0.107J. When minimizing the latency, the average delay of the task is 0.441s when the cell number is 3, and 0.547s when the cell number equals 6, increasing by 24%. This illustrates that the device interference between multi-cells and the competition for limited resources will lead to the reduction of computation offloading benefits, resulting in an increase in energy consumption and latency.
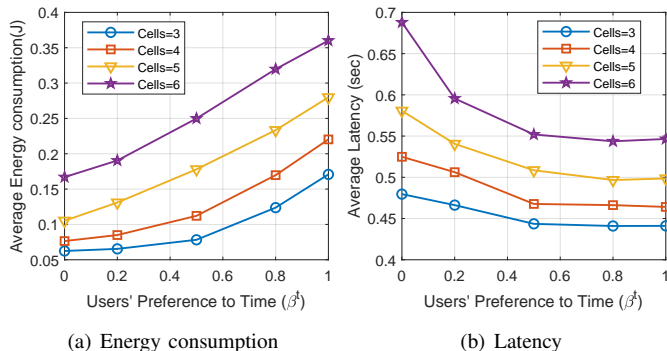

(a) Energy consumption   (b) Latency

Fig. 7.   Relationship between average (a) energy consumption and (b) latency with user time preference (β_t).

Fig. 8.   Comparison of average utility consumption while varying different task parameters using different schemes.

Fig. 7 shows the average energy consumption and time cost when changing the user's preference to time (β_t), and it is important to note that the user's preference to energy β_e equals $1 - \beta_t$. In addition, it is worth noting that when β_t is equal to 0, the solution corresponds to the energy minimization problem, while $\beta_t = 1$, the algorithms proposed could find the optimal solution for the latency minimization problem. That is, this paper achieves a unified framework of energy minimization and delay minimization. It can be seen from
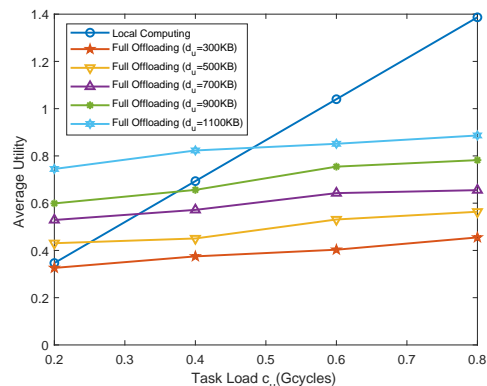
A comparison of the average utility consumption of local computing and full offloading against the variation of task parameters is shown in Fig. 8. It can be seen from the graph that the system utility consumption, i.e., the weighted value of energy consumption and latency increases with the growth of the required CPU computing cycles c_u. In addition, local computing increases by a larger proportion than full offloading, due to the fact that the delay and energy consumption of local computing are both related to c_u and the limited

local computing capacity leads to a significant increase in latency and energy consumption when c_u increases. The local computing does not involve the data d_u uploaded by the task, so the local utility is independent of d_u. The utility value of full offloading increases as d_u grows. This is because the larger task input data leads to an increase in task transmit time. Specifically, when $c_u = 0.2$Gcycles and $d_u = 1100$kB, the average utility of local computing is 0.347, which owns a 53.3% gain, compared with that of full offloading (0.745). In this case, the devices prefer local computing. When c_u equals 0.8Gcycles and d_u equals 300kB, the average utility of full offloading equals 0.506, a 70.8% reduction compared to local computing. Therefore, it is more advantageous to assign tasks with small task description data and large computation to the MEC server.

V. Conclusions

In this paper, we investigated the joint resource allocation and task offloading for multi-cell SCMA-MEC systems. Aiming to minimize the system latency and devices' energy consumption, we proposed an iterative optimization scheme based on BCD for the transmit power of users and computing resources allocation of MEC servers, as well as an algorithm based on the improved simulated annealing for task offloading decision and multi-cell SCMA codebook allocation. Then we explained two special cases of the problem studied in this paper, namely latency minimization and energy optimization. Finally, the framework was extended to the partial offloading case, and an algorithm for solving the partial offloading ratio based on alternating convex search was proposed. Numerical results showed that the multi-cell SCMA-MEC scheme achieves lower energy consumption and system delay in comparison to the OFDMA and PD-NOMA techniques. Compared to the random codebook assignment, the multi-cell SCMA codebook allocation algorithm can achieve improved channel-adapted offloading efficiency and reduce the system latency by approximately 16.3%.

Appendix

A. The Proof of Lemma 1

Proof: Let $G_u^e(\xi_u^n, f_u^n)$ denote the objective function of problem (17). The derivatives and second-order derivatives of G_u^e with respect to (w.r.t) ξ_u^n and f_u^n can be calculated as

$$\frac{\partial G_u^e}{\partial \xi_u^n} = \beta_t d_u + \frac{\beta_e d_u}{\zeta_{u,c}^n}\left[2^{\frac{1}{\xi_u^n}}\left(1 - \frac{\ln 2}{\xi_u^n}\right) - 1\right] \quad (32)$$

$$\frac{\partial G_u^e}{\partial f_u^n} = \frac{-\beta_t c_u}{(f_u^n)^2} \quad (33)$$

$$\frac{\partial^2 G_u^e}{\partial (\xi_u^n)^2} = \frac{\beta_e d_u 2^{\frac{1}{\xi_u^n}}(\ln 2)^2}{\zeta_{u,c}^n (\xi_u^n)^3} \quad (34)$$

$$\frac{\partial^2 G_u^e}{\partial (f_u^l)^2} = \frac{2\beta_t c_u}{(f_u^n)^3} \quad (35)$$

$$\frac{\partial^2 G_u^e}{\partial f_u^l \xi_u^n} = \frac{\partial^2 G_u^e}{\partial \xi_u^n f_u^l} = 0 \quad (36)$$

The values β_t, β_e, d_u, $\zeta_{u,c}^n$ and the variable f_u^l and ξ_u^n are all positive. It is easy to deduce the Hessian matrix of G_u^e is a diagonal matrix with positive elements, i.e. positive-definite. Hence, the objective function is convex w.r.t ξ_u^n and f_u^n. Additionally, the constraints (17b), (17c), (12g) and (12h) are all affine functions. Therefore, the problem (17) is proved to be a convex optimization problem. ∎

References

[1] Q. Pham, F. Fang, V. N. Ha, M. J. Piran, M. Le, L. B. Le, W. Hwang, and Z. Ding, "A survey of multi-access edge computing in 5G and beyond: Fundamentals, technology integration, and state-of-the-art," *IEEE Access*, vol. 8, pp. 116 974–117 017, 2020.

[2] 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, 2017.

[3] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[4] C. Shu, Z. Zhao, Y. Han, G. Min, and H. Duan, "Multi-user offloading for edge computing networks: A dependency-aware and latency-optimal approach," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 1678–1689, 2020.

[5] 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, 2016.

[6] S. Zarandi and H. Tabassum, "Delay minimization in sliced multi-cell mobile edge computing (MEC) systems," *IEEE Communications Letters*, vol. 25, no. 6, pp. 1964–1968, 2021.

[7] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 856–868, 2019.

[8] Z. Ding, Y. Liu, J. Choi, Q. Sun, M. Elkashlan, I. Chih-Lin, and H. V. Poor, "Application of non-orthogonal multiple access in LTE and 5G networks," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 185–191, 2017.

[9] H. Nikopour and H. Baligh, "Sparse code multiple access," in *IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2013, pp. 332–336.

[10] J. V. C. Evangelista, Z. Sattar, G. Kaddoum, and A. Chaaban, "Fairness and sum-rate maximization via joint subcarrier and power allocation in uplink SCMA transmission," *IEEE Transactions on Wireless Communications*, vol. 18, no. 12, pp. 5855–5867, 2019.

[11] M. Moltafet, N. M. Yamchi, M. R. Javan, and P. Azmi, "Comparison study between PD-NOMA and SCMA," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 2, pp. 1830–1834, 2018.

[12] Z. Ding, P. Fan, and H. V. Poor, "Impact of non-orthogonal multiple access on the offloading of mobile edge computing," *IEEE Transactions on Communications*, vol. 67, no. 1, pp. 375–390, 2019.

[13] Y. Wu, K. Ni, C. Zhang, L. P. Qian, and D. H. K. Tsang, "Noma-assisted multi-access mobile edge computing: A joint optimization of computation offloading and time allocation," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 12, pp. 12 244–12 258, 2018.

[14] H. Li, F. Fang, and Z. Ding, "Joint resource allocation for hybrid NOMA-assisted MEC in 6G networks," *Digital Communications and Networks*, vol. 6, no. 3, pp. 241 – 252, 2020.

[15] A. Kiani and N. Ansari, "Edge computing aware NOMA for 5G networks," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1299–1306, 2018.

[16] B. Liu, C. Liu, and M. Peng, "Joint radio and computation resource allocation for noma-enabled mec in multi-cell networks," in *IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[17] A. Alnoman, S. Erkucuk, and A. Anpalagan, "Sparse code multiple access-based edge computing for iot systems," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 7152–7161, 2019.

[18] P. Liu, J. Lei, and W. Liu, "An optimization scheme for SCMA-based multi-access edge computing," in *IEEE 93rd Vehicular Technology Conference (VTC-Spring)*, 2021.

[19] J. Du, W. Liu, G. Lu, J. Jiang, D. Zhai, F. R. Yu, and Z. Ding, "When mobile-edge computing (mec) meets nonorthogonal multiple access (noma) for the internet of things (IoT): System design and optimization," *IEEE Internet of Things Journal*, vol. 8, no. 10, pp. 7849–7862, 2021.

[20] W. Kim, M. S. Gupta, G.-Y. Wei, and D. Brooks, "System level analysis of fast, per-core dvfs using on-chip switching regulators," in *2008 IEEE 14th International Symposium on High Performance Computer Architecture*, 2008, pp. 123–134.

[21] M. Dabiri and H. Saeedi, "Dynamic SCMA codebook assignment methods: A comparative study," *IEEE Communications Letters*, vol. 22, no. 2, pp. 364–367, 2018.

[22] P. Liu, K. An, J. Lei, G. Zheng, Y. Sun, and W. Liu, "Scma-based multi-access edge computing in iot systems: An energy-efficiency and latency tradeoff," *IEEE Internet of Things Journal*, pp. 1–1, 2021.

[23] K. Au, L. Zhang, H. Nikopour, E. Yi, A. Bayesteh, U. Vilaipornsawai, J. Ma, and P. Zhu, "Uplink contention based SCMA for 5G radio access," in *IEEE Globecom Workshops (GC Wkshps)*, 2014, pp. 900–905.

[24] M. B. Shahab, R. Abbas, M. Shirvanimoghaddam, and S. J. Johnson, "Grant-free non-orthogonal multiple access for IoT: A survey," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 1805–1838, 2020.

[25] Y. Du and G. de Veciana, "wireless networks without edges: Dynamic radio resource clustering and user scheduling," in *IEEE Conference on Computer Communications (INFOCOM)*, 2014, pp. 1321–1329.

[26] P. J. Van Laarhoven and E. H. Aarts, "Simulated annealing," in *Simulated annealing: Theory and applications*. Springer, 1987, pp. 7–15.

[27] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[28] J. Zhang, X. Hu, Z. Ning, E. C.-H. Ngai, L. Zhou, J. Wei, J. Cheng, and B. Hu, "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2633–2645, 2018.

[29] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, pp. 4569–4581, 2013.