Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8858298, 11 pages
https://doi.org/10.1155/2020/8858298

Research Article

Research on Multinode Collaborative Computing Offloading Algorithm Based on Minimization of Energy Consumption

Dongsheng Han, Yu Liu, and Junhong Ni

Department of Electronic and Communication Engineering, North China Electric Power University, Baoding, 071003 Hebei, China

Correspondence should be addressed to Yu Liu; 2644270950@qq.com

Received 14 July 2020; Revised 16 October 2020; Accepted 21 October 2020; Published 6 November 2020

Academic Editor: Lisheng Fan

Mobile edge computing (MEC) nodes are deployed at positions close to users to address excessive latency and converging flows. Nevertheless, the distributed deployment of MEC nodes and offload of computational tasks among several nodes consume additional energy. Accordingly, how to reduce the energy consumption of edge computing networks while satisfying latency and quality of service (QoS) demands has become an important challenge that hinders the application of MEC. This paper built a local-edge-cloud edge computing network and proposes a multinode collaborative computing offloading algorithm. It can be applied to smart homes, realize the development of green channels, and support local users of Internet of Things (IoT) to decompose computational tasks and offload them to multiple MEC or cloud nodes. The simulation analysis reveals that the new local-edge-cloud edge computing offload method not only reduces network energy consumption more effectively compared with traditional computing offload methods but also ensures the implementation of more data samples.

1. Introduction

With the continuous development of the Internet of Things (IoT) technology in recent years, IoT network equipment has developed perception and communication abilities, and the user end of the network can extend to information exchange and communication between any goods in daily life [1]. IoT technology has also been used in various aspects of industrial production and daily life. In transportation and network performance optimization [2], IoT has been used in smart homes, smart industries, and smart cities, among others. Previous studies have largely focused on the application scene of smart homes. The local user ends of IoT in smart homes can take the form of any good. Therefore, IoT contains diversified user data, whereas intelligent electrical apparatus requires a rapid and effective processing of task data [3]. In this case, a fast, efficient, and safe task processing mode needs to be devised to meet the demands of users with a large data size or high sensitivity to latency. Given that the traditional single-cloud model cannot meet such demands, the concept of mobile edge computing (MEC) has been proposed based on cloud computing [4]. MEC is a new computing model, and MEC nodes are widely distributed in the vicinity of the client to provide intelligent services for local users. Edge nodes can be installed on the edge server (e.g., vehicles and UAV) to meet the linkage demands of different users [5]. Combined with MEC, a multinode cooperation of data tasks is realized by transmitting data between the local users of IoT and MEC nodes wherein the local user data of IoT are offloaded to nearby MEC servers, thereby addressing the limited computing capability of these users and reducing their computing task pressures. However, MEC nodes have a limited computing capacity, thereby requiring a cooperation among multiple MEC nodes to handle computing tasks with a large data size.

To solve the network energy consumption problem under a large data size at the user ends of IoT, this study initially analyzes and selects MEC nodes in a local-edge-cloud edge computing network model while considering the distances between the MEC nodes and user ends, the channel characteristics, and the CPU energy consumption.

The main contributions can be summarized as follows:

(1) The local-edge-cloud edge computing network model proposed in this paper supports the local user ends of IoT in their parallel offloading of a computing

task to multiple MEC nodes or a cloud. This study takes both network computation and transmission into account, considered from the three layers of local, edge, and cloud

(2) Latency cannot be directly accumulated due to the parallel data transmission. Instead, the time for receiving and processing data at different nodes is analyzed to determine the network latency. An integral linear programming problem that targets the optimization of network energy consumption is formulated, and single-user task offloading is analyzed by using the branch-and-bound (BB) algorithm to minimize the overall network energy consumption

(3) The simulation results show that the demands for MEC nodes increase along with the size of offload data at the local user ends of IoT. Moreover, the multinode collaborative model is significantly superior over the traditional computing offloading algorithm in terms of energy consumption and latency, especially under large offload data sizes

The rest of this study is organized as follows. In Section 2, related work is introduced. Section 3 introduces the proposed model. Section 4 discusses in detail the construction of an objective function for the multinode computing offload model and the BB algorithm used in the optimization. Section 5 analyzes the simulation results. Section 6 concludes the paper.

2. Related Work

The local user ends of IoT can offload computing tasks to MEC nodes via global and partial offloading. In global offloading, the entire computing task is offloaded to an MEC node. Liu et al. [6] used the 1D searching algorithm to reduce implementation latency to the maximum extent and gave comprehensive considerations to the queuing state in the application buffer zone and the available processing capacity. However, edge nodes have inadequate computing capacities and experience long transmission latency. To address this problem, this paper proposes a partial offload method that implements parts of the computing task at the local position and offloads the other parts to the MEC for implementation. Further details on partial offloading can be found in [7]. In a partial offload program, the distribution positions of data tasks need to be determined; tasks are successively transferred to each node to execute tasks after the user is partitioned. In [8], Yang et al. proposed the concept of task zoning, which determines offload modules and implementation methods, that is, whether the tasks are implemented at local positions or offloaded to MEC and cloud nodes. Meanwhile, Zhao et al. [9] transformed the partial offloading problem into a nonlinear constraint problem and adopted a linear programming approach to solve this problem and realize the goal of optimal processing. Given their diversity, network data of different sizes are generated. Accordingly, resource limits have become key problems in the offload process that have been discussed in [9–11]. For instance, Zhao [10] analyzed resource limits from the perspectives of network capacity and data allocation, chose an appropriate position for data processing, and guaranteed the smooth implementation of additional data tasks. In [11], a data task was segmented by employing a partial offload method, and this task was transmitted successively to MEC and cloud nodes for implementation, thereby overcoming resource limits. To address the limitations in node quantity and processing ability, You and Huang [12] proposed an optimal resource allocation strategy for a time division multiple access system to process the queuing of tasks and ensure resource processing efficiency. Aiming at the complex resource allocation problem, Ref. [13] proposed an intelligent resource allocation framework to solve the complex resource allocation problem of collaborative mobile edge computing network. The resource allocation scheme was determined according to the edge computing server's computing capacity, channel quality, resource utilization, and latency constraints.

When users have a large number of computing tasks, a single MEC node cannot meet the demand of processing offload tasks from the user end even if the partial offload method is applied. As a result, several nodes must be selected in the collaborative processing of offload tasks. Fan et al. [14] adopted a multinode collaboration method that allows nearby MEC nodes to share the computing pressure of the target node when the computing task at the user ends is too large for a single MEC node. They also designed an algorithm for solving the optimization problem by using an interior point method and a logarithmic potential barrier function to optimize the energy consumption problem of the multinode collaboration system. This multinode collaboration method is mainly used to address the inadequate computing capacity of single nodes. Based on a dynamic and self-configuring multiequipment mobile cloud system, Habak et al. [15] implemented relevant computing tasks and expanded the range of the cloud system by using the surrounding vacant mobile equipment as MEC servers with an aim to solve the problem where the network load exceeds the computing capacity of nodes. In a multinode collaboration method, the computing task should be allocated to multiple nodes, but this action involves the allocation and deployment of nodes. Reference [16] considers link selection in collaborative networks. Based on the characteristics of two branches in the system, the buffer-assisted relay combination technology is used to provide accurate expression of interrupt probability for the common channel interference network to evaluate the transmission performance of the network. In [17], the authors selected the deployment positions of MEC nodes, such as LTE micro sites and gathering stations of multiwireless access technology communities. With the continuous popularization of MEC technology, multinode collaborative technology has been increasingly used in practice.

In the above studies, users offload the computing tasks completely or partially to one or several MEC nodes, optimize the network structure, increase the task processing capability of the network, and explore resource optimization in a multinode collaborative network structure. Nevertheless, MEC nodes are extensively distributed in ranges of local user

nodes, and several MEC nodes in a wireless network are selected to participate in computing. Involving more nodes in a network will increase its overall energy consumption. After the introduction of the green MEC philosophy, network energy consumption has become a key concern among researchers.

In the multinode task allocation model, MEC nodes that implement the computing tasks are chosen reasonably to reduce network energy consumption. Zhang et al. [18] applied a single-user mobile edge computing offload (MECO) approach to the MEC network model, where network energy consumption is treated as the optimization goal, and the appropriate offload strategy is determined by comprehensively changing the number of CPU periods and network transmission rate. However, this study only considers the single-user MECO model. Meanwhile, the authors in [19] fully considered energy consumption and latency of end users in the multiuser MECO distributed computing offload model and realized an optimal allocation of resources in the computing offload process by using game theory. Reference [20] constructs an intelligent edge computing network based on pricing. When the user is offloading data, latency and price are taken as performance indicators, stochastic game method is used to determine the user signal processing scheme, and offloading strategy is designed to reduce latency and price. In [21], to cope with energy shortage in a heterogeneous network, a shared link was established among multiple base stations (BS) and was extended to the macro and micro domains for analysis. At the same time, in the heterogeneous network, due to the complex distribution of base stations and users, multilayer switching and power distribution need to be considered. In Ref. [22], there is the switching and power distribution problem in the two-layer heterogeneous network composed of macro station and millimeter wave. A multiagent augmented learning algorithm based on the proximal policy optimization is developed to realize the interaction between multiuser devices. Ng et al. [23] proposed an offload priority function by considering quantitative equality, transmission channel, and local computing situations. By analyzing this offload priority function, the optimal network resource allocation was realized, and the overall network energy consumption was used as the measurement index.

In sum, many studies have examined multinode collaboration and data offloading. Users transmit data to multiple nodes in a step-by-step manner before their implementation. When the data size at the user ends is relatively large, then the step-by-step transmission leads to significant latency, thereby destroying the latency constraints of users and consuming a considerable amount of network energy. On this basis, the superiority of the model created in this paper is more prominent.

3. System Model

Figure 1 illustrates a local-edge-cloud edge computing network that has K local user ends of IoT served by N wireless eNodeBs. Each eNodeB is equipped with one MEC server or N MEC nodes. The computing task from the local user ends of IoT can be implemented in site, partially offloaded to the MEC nodes, or partially transmitted to the cloud server through the routers at eNodeB. Before offloading tasks, the local user ends of the IoT segment these tasks while following certain rules, and the segments choose the appropriate MEC nodes or cloud servers for task offloading based on the latency, energy consumption, computing capacity of MEC nodes, and other parameters. In Ref. [6], the sequential transmission of segmented task blocks will cause a certain latency waste. Based on the above, this paper makes improvements by transferring the segmented task block to the appropriate node to perform tasks synchronously, determining the optimal assignment location of the task at the user end, transmitting and processing the task at the same time, and processing more data under the same latency constraint. Without loss of generality, this study hypothesizes that the computing task of local user UE_1 at a moment can be segmented into N task blocks. Task block 1 can be implemented at the local user ends of IoT. Offloading to and implementing at nodes MEC_1, MEC_2, and MEC_3 are optional for task block 2, task blocks 3, 4, and 5, and task blocks 6 and 7, respectively. Given that the computing capacity of MEC nodes cannot meet the demands of residual task blocks, these blocks are transmitted to the cloud for implementation. A parallel offloading of multiple task blocks is applied to reduce the network latency and overall network energy consumption.

3.1. Network Energy Consumption. In studying the local-edge-cloud edge computing network model, the computing and transmission capacity of the network should be considered to minimize the network energy consumption because the data from the local user ends of IoT are offloaded simultaneously and implemented at multiple nodes. Therefore, "network energy consumption" in this paper includes the energy consumed for the parallel transmission of computing tasks from the local user ends to the MEC and cloud nodes and the energy consumed for transmitting a computing task from the local user ends to different nodes. The computing model of the local-edge-cloud edge computing network is defined as A_k (R_k, s_k), where R_k is the task value of user k $(k = \{1, 2, \cdots, K\})$ and s_k is the time spent by user k in executing the task. The computing energy consumption of user k can be expressed as

$$E_{\text{com}}^{k} = R_k C_k m_k, \tag{1}$$

where C_k is number of CPU turns needed to execute a computing task per bit of data and m_k is the energy consumed for each CPU turn.

When the computing task cannot be executed completely at the local user ends of IoT, this task must be offloaded to the appropriate nodes, which will consume a certain amount of transmission energy. Transmission energy consumption is related to both the transmission time and transmission power of the task. The transmission energy can be formulated as

$$E_{\text{trans}}^{k} = t_k p_k, \tag{2}$$

where t_k is the transmission time of the computing task of

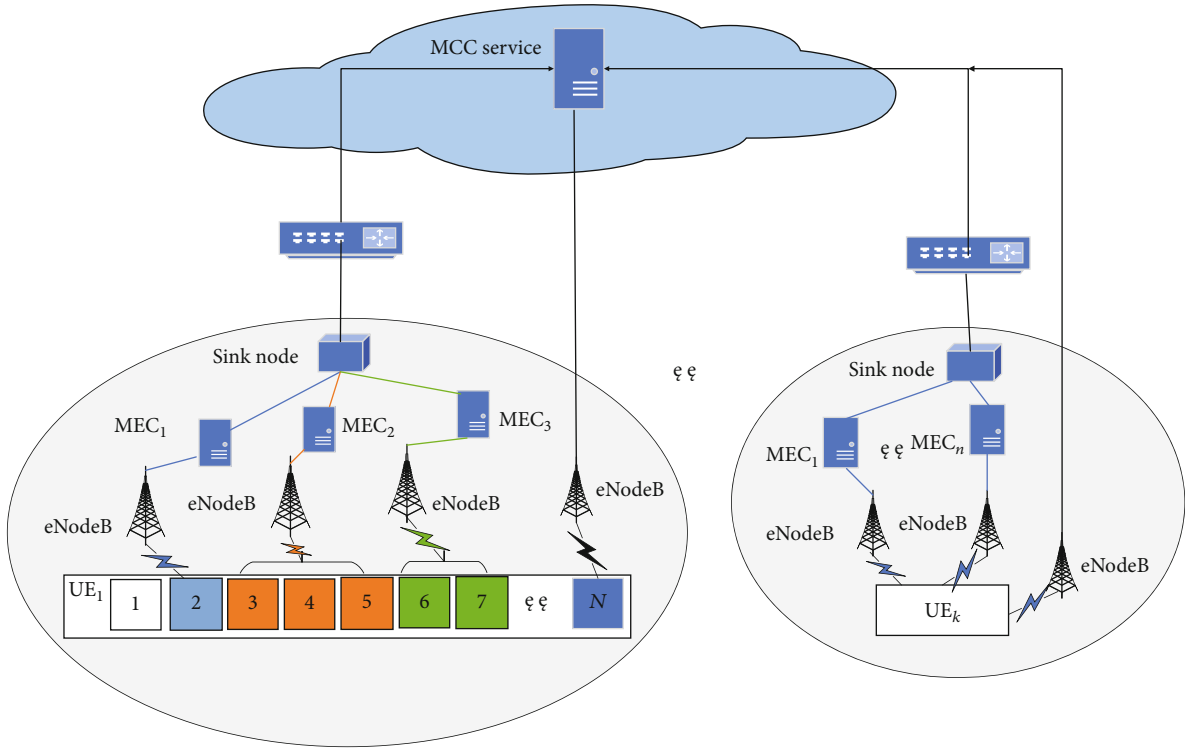

Figure 1: Local-edge-cloud edge computing network model.

user k and p_k is the transmission power between user k and the offload nodes. The overall energy consumed by user k to execute a task is computed as the sum of transmission energy consumption and computing energy consumption:

$$E^k_{\text{total}} = E^k_{\text{com}} + E^k_{\text{trans}}. \tag{3}$$

3.2. Computing Capacity. The number of CPU turns needed for user k to implement 1 bit of task at local users, MEC nodes, and cloud nodes is denoted by C^L_k, C^E_k, C^C_k, respectively. Meanwhile, the energy consumed for each CPU turn in implementing the computing task of user k at local users, MEC nodes, and cloud nodes is denoted by m^L_k, m^E_k, m^C_k. Under the multinode collaboration mode, the data are segmented at the local user ends of IoT, and the segmented data are transmitted to the MEC or MCC nodes for computing. To easily observe the offload condition of segmented tasks, one data unit ϕ (kbit) is set, and the data at the local user ends of IoT are expressed as data units. The data of user k are divided into M_k data units as $R_k = M_k\phi$. For all nodes, parameter ρ is set, where $\rho_{k\to 0}$ denotes the number of data units in the local computing of user k. The network has n MEC nodes, where $n = \{1, 2, \cdots, N\}$. $\rho_{k\to n}$ and $\rho_{k\to N+1}$ refer to the number of data units that local user k offloads to MEC_n and the cloud nodes for task execution, respectively. With respect to the selection problem between the local user ends of IoT and MEC nodes, parameter $\beta_{k,m,n}$ indicates that the computing task block m of local user k is offloaded and implemented at node n. In this model, the local user ends of IoT segment the computing task into several blocks and offload them to multiple MEC and cloud nodes. A data unit can only be offloaded to a single node ($\sum_{n=1}^{N}\beta_{k,m,n} = 1$), while one MEC node can receive several data units ($\sum_{m=1}^{M}\beta_{k,m,n} = \rho_{k\to n}$). When $n = 0$, the computing task is implemented at the local user ends of IoT, but when $n = N + 1$, the computing task is implemented at cloud nodes.

Given that the data are segmented at the local user ends of IoT and transmitted to several nodes simultaneously, the data allocated to different nodes should meet the computing capacities of different nodes. The data of user k are analyzed as

$$C^n_k R^n_k \le F. \tag{4}$$

Let F_0, F_n, F_{N+1} be the computing capacities of the local user ends, MEC nodes, and cloud nodes, that is, the number of CPU turns needed to implement the computing task. In equation (4), $R^n_k = \rho_{k\to n}\phi$, where $n = 0$ denotes the size of the task implemented at the local user ends of IoT, $n = \{1, 2, \cdots, N\}$ refers to the size of the task implemented at the MEC nodes, and $n = N + 1$ refers to the size of the task implemented at the cloud nodes.

3.3. Computing Latency. Computing latency is determined by computing the number of nodes, number of CPU turns, and node computing capacity. When the computing task is executed at the local user ends of IoT, the data computing latency of user k can be expressed as

$$s^L_k = \frac{C^L_k R^0_k}{F_0}. \tag{5}$$

Given that the data are segmented at the local user ends of IoT and are transmitted to several MEC nodes simultaneously for implementation, the computing latency is computed as the maximum computing latency of different nodes. The computing latency of one node can be formulated as

$$s_{k,n}^{E} = \frac{R_k^n C_{k,n}^{E}}{F_n}. \tag{6}$$

When the computing task cannot be implemented at the local user ends of IoT and MEC nodes, this task should be transmitted to cloud servers. The computing latency at the cloud nodes can be formulated as

$$s_k^{C} = \frac{R_k^{N+1} C_k^{C}}{F_{N+1}}. \tag{7}$$

3.4. Transmission. The transmission links in a network refer to the wireless communication links between the MEC server and UE, the transmission VLAN among MEC servers, and the transmission links between the MEC and cloud servers. In the network transmission process, the relationship between network computing capacity and transmission capacity should be considered. If the computing capacity is too high, then the channel resources in the network cannot be allocated to the local user ends of IoT, thereby congesting the channels and increasing network latency. Let R_k (bit) be the data size that local user k of IoT needs to process. Specifically, R_k^0 refers to the size of the computing task implemented at the local user ends of IoT, R_k^n is the size of the computing task implemented at the MEC nodes, and R_k^{N+1} is the size of the computing task implemented at the cloud nodes. When the computing task can be implemented at the local user ends of IoT and does not need to be transmitted, no transmission energy is consumed. Transmission energy is only consumed when the computing task is offloaded to the MEC and cloud nodes.

Let t_k denote the transmission time for one data unit ϕ (kbit), where $t_k > 0$. Therefore,

$$t_k = \frac{R_k}{r_k}, \tag{8}$$

where r_k refers to the data transmission rate from user k to the chosen nodes. The total transmission time in the computing offload process is calculated by the number of bit units that the user offloads to nodes $\rho_{k\to n}$. Suppose that n MEC servers receive data from the user end. These data are segmented at the local user ends of IoT, and data transmission is performed simultaneously. However, $\rho_{k\to n}$ computing tasks will experience $\rho_{k\to n} t_k$ transmission time in the task transmission process of each part. Given that each node has unique basic parameters, the size of the offloaded data also varies. The transmission time from the local user ends of IoT to the nodes shall be taken as the transmission time from the local user ends to the node with the largest offloaded task. This node should meet

$$\sum_{n=1}^{N+1} \rho_{k\to n} t_k \le T, \tag{9}$$

where T represents the latency in meeting the QoS demands of users.

3.5. Transmission Power. In the transmission from local user k to the chosen nodes, the transmission rate can be expressed as

$$r_k = W \log_2\left(1 + \frac{p_{k,n}\mathrm{PL}_{k,n}}{\sigma^2}\right), \tag{10}$$

where W is the channel bandwidth, $p_{k,n}$ is the transmission power between local user k and node n, and h_k is the channel characteristics between local user k and node n. The differences in the channel characteristics can be ascribed to the variances in the distances of each node from the local user k. The value of PL_k meets the large-scaled attenuation characteristic and is related to transmission distance. PL_k is expressed as $\mathrm{PL} = \mathrm{PL}_{\mathrm{FS}}(d_0) + 10n \lg (d/d_0) + X_\sigma$, where d is the transmission distance, d_0 is the reference distance, n is the route loss index, and X_σ is a Gaussian random variable with a 0 mean and σ^2 standard deviation. Meanwhile, W_E, W_C represent the bandwidths between the local users of IoT and edge nodes and those between the users and cloud nodes. When $n = \{1, 2, \cdots, N\}$, $p_{k,n}$ and $\mathrm{PL}_{k,n}$ represent the transmission power and loss between user k and MEC_n. When $n = N + 1$, these parameters represent the transmission power and loss between local user k and the cloud nodes.

According to equations (1) and (3), data transmission rate (r_k) can be expressed in two ways. The transmission power from the local user k to node n can be expressed as

$$P_{k,n} = \frac{\left(2^{r_{k,n}/W} - 1\right)\sigma^2}{\mathrm{PL}_{k,n}}. \tag{11}$$

In sum, to analyze the transmission in the local-edge-cloud edge computing network and computing situations, network energy consumption can be computed as total energy consumption = computing energy consumption + transmission energy consumption. Computing energy consumption includes the computing energy consumed by the local user ends of IoT and by the collaboration between MEC nodes and cloud servers. Meanwhile, transmission energy consumption includes the wireless transmission energy consumption between the local user ends of IoT and MEC nodes and that between the local user ends and cloud servers. Network latency, which includes computing latency and transmission latency, is considered in computing network energy consumption given that the network energy consumption should be minimized under the premise of meeting network latency requirements.

4. Multinode Collaborative Computing Offloading Algorithm

In the local-edge-cloud edge computing network model, one part of the computing task is implemented at the local user ends of IoT, whereas the other parts are offloaded to the appropriate nodes. The data of the user are segmented following certain rules and are offloaded simultaneously to several nodes. Given that MEC nodes are close to the local user ends of IoT, a short data transmission time is achieved. However, the offload positions should be chosen reasonably based on the user demand given the limited computing capacity of MEC nodes.

4.1. Establishment of an Objective Function. According to equation (3), the overall network energy consumption includes computing and transmission energy consumption. In the local-edge-cloud edge computing network model, certain tasks are distributed to all levels. In other words, network energy consumption includes the computing and transmission energy consumption of the local user ends of IoT, MEC nodes, and cloud nodes.

Implementing the computing task at the local user ends of IoT only consumes computing energy. The energy consumed can be formulated as

$$E_L = \sum_{k=1}^{K} \rho_{k\to 0}\phi C_k^L m_k^L. \tag{12}$$

When the computing task is offloaded to edge nodes, several MEC nodes surround the local user ends of IoT. Therefore, the appropriate MEC nodes should be selected. Let the selection parameter be $\rho_{k\to n}$, $n = \{1, 2, \cdots, N\}$, which reflects the selection of MEC nodes. The overall energy consumption of the MEC node includes both computing and transmission energy consumption and can be expressed as

$$E_E = \sum_{k=1}^{K}\sum_{n=1}^{N}\left(\rho_{k\to n}\phi C_{k,n}^E m_{k,n}^E + \rho_{k\to n} t_{k,n}\frac{\sigma^2\left(2^{1/2^{t_{k,n}}W_E} - 1\right)}{\mathrm{PL}_{k,n}^E}\right). \tag{13}$$

When the computing task is partially offloaded to the cloud servers, the overall energy consumption of cloud nodes can be expressed as

$$E_C = \sum_{k=1}^{K}\rho_{k\to N+1}\phi C_k^C m_k^C + \rho_{k\to N+1} t_{k,N+1}\frac{\sigma^2}{PL_{k,N+1}^C}\left(2^{1/t_{k,N+1}W_C} - 1\right). \tag{14}$$

The overall network energy consumption is then computed as the total energy consumed by the local user ends of IoT, MEC nodes, and cloud nodes:

$$E_{\text{total}} = E_L + E_E + E_C. \tag{15}$$

Given that the network model considers the computing and transmission of data from the local user ends of IoT, network latency includes both computing and transmission latencies. The computing latency of the local user ends of IoT, MEC nodes, and cloud nodes should be considered when applying a local-edge-cloud edge computing network model. The computing latency of local user k can be expressed as

$$s_k^L = \frac{C_k^L \rho_{k\to 0}\phi}{F_0}. \tag{16}$$

Unlike in the mutual transmission computing offload model, the computing task is segmented at the local user ends of IoT and are transmitted simultaneously to multiple nodes for processing. Therefore, the computing latency is taken as the maximum computing latency of MEC and cloud nodes:

$$s_k' = \max\left\{\frac{\rho_{k\to n} C_k^n \phi}{F_n},\ n = 0, 1, \cdots, N, N+1\right\}. \tag{17}$$

The overall computing latency of the network is then formulated as

$$s_k = s_k^L + s_k'. \tag{18}$$

Meanwhile, transmission latency mainly involves the wireless transmission links from the local user ends of IoT to the MEC nodes and the VLAN transmission network from the local user ends of IoT to the cloud nodes. Given that the data are segmented at the local user ends of IoT, the appropriate nodes should be selected for the simultaneous transmission of segmented data. When parallel data transmission is applied, the overall transmission latency of the network can be expressed as

$$t_k = \max\left\{\rho_{k\to n} t_{k,n},\ n = 0, 1, \cdots, N, N+1\right\}. \tag{19}$$

The network latency is then computed as the sum of computing latency and transmission latency:

$$D_k = s_k + t_k. \tag{20}$$

The goal of this multinode collaborative computing offload model is to minimize the overall network energy consumption while meeting the time constraints. The optimization system of the multinode collaborative computing offload network is

$$\min \quad E_{\text{total}} \tag{21}$$

$$\text{s.t.} \quad D_k \le T, \tag{22}$$

$$t_{k,n} > 0, \tag{23}$$

$$\sum_{n=1}^{N}\beta_{k,m,n} = 1, \tag{24}$$

$$\sum_{m=1}^{M} \beta_{k,m,n} = \rho_{k\to n}, \tag{25}$$

$$\rho_{k\to n}\phi C_{k,n}^{E} \le F_n, \quad n = \{1, 2, \cdots, N\}, \tag{26}$$

$$C_k^L \rho_{k\to 0}\phi \le F_0, \tag{27}$$

$$C_k^C \rho_{k\to N}\phi \le F_{N+1}, \tag{28}$$

where (22) and (23) are the limiting conditions of transmission time (with (22) indicating that the network latency is smaller than the latency limit of user ends), (24) and (25) denote the data allocation after the segmentation at the local user ends (with (24) indicating that only one data unit can be offloaded to one MEC node and (25) indicating the number of unit tasks that can be processed by a single MEC node), and (26) to (28) denote the computing capacity limitations of MEC nodes, local user ends of IoT, MEC nodes, and cloud nodes.

To address the above conditions, the size of the computing task blocks offloaded to different nodes in equation (21) is denoted by $\rho_{k\to n}$. The task allocation of nodes under optimal energy consumption is evaluated by analyzing the value of $\rho_{k\to n}$. Given that $\rho_{k\to n}$ determines the number of data units, its value can only be expressed as an integer. Therefore, the optimization problem becomes an integer programming problem.

4.2. Optimization Based on the BB Algorithm. The resource allocation scheme for MEC nodes is determined by using the BB algorithm, which searches all feasible solution spaces for the optimization problem with constraints. During the implementation of this algorithm, all feasible solution spaces are continuously divided into smaller subsets, and a lower or upper bound is calculated as a solution for each subset. With respect to the integer programming problem, the BB algorithm solves the ordinary linear programming problem through simplex and divides the nonintegral decision variables into two proximate integers. The conditions are then listed and added into the original problem. Meanwhile, the constraint vector after updating is solved, from which the upper or lower bound of the numerical value is identified.

In using the BB algorithm to solve the energy consumption optimization problem, equation (21) is taken as the objective function with $\rho_{k\to 0}, \rho_{k\to 1}, \cdots, \rho_{k\to N}, \rho_{k\to N+1}$ as the independent variable. This objective function can be viewed as a linear programming problem that is expressed by its independent variable. The independent variable ρ meets

$$\rho_{k\to 0} + \rho_{k\to 1} + \cdots + \rho_{k\to N} + \rho_{k\to N+1} = M_k. \tag{29}$$

Equation (21) can then be expressed as

$$E_{\text{total}} = \nu_0\rho_{k\to 0} + \nu_1\rho_{k\to 1} + \cdots + \nu_N\rho_{k\to N} + \nu_{N+1}\rho_{k\to N+1}, \tag{30}$$

where $\nu_0, \nu_1, \cdots, \nu_N, \nu_{N+1}$ is the coefficient before ρ, and the coefficient vector of independent variables in the objective function can be expressed by $f = [\nu_0\ \nu_1\ \cdots\ \nu_N\ \nu_{N+1}]^T$. The constraint condition (1) for latency in equation (21) is then transformed as

$$D_k = D_k^0\rho_{k\to 0} + D_k^1\rho_{k\to 1} + \cdots + D_k^N\rho_{k\to N} + D_k^{N+1}\rho_{k\to N+1} \le T, \tag{31}$$

where $D_k^0, D_k^1, \cdots, D_k^N, D_k^{N+1}$ is the coefficient before ρ in the constraint condition equation (22). Constraints (5) to (7) in equation (21) can then be transformed into

$$\begin{aligned} a_{10}\rho_{k\to 0} + a_{11}\rho_{k\to 1} + \cdots + a_{1N}\rho_{k\to N} + a_{1N+1}\rho_{k\to N+1} &\le F_n, \\ a_{20}\rho_{k\to 0} + a_{21}\rho_{k\to 1} + \cdots + a_{2N}\rho_{k\to N} + a_{2N+1}\rho_{k\to N+1} &\le F_0, \\ a_{30}\rho_{k\to 0} + a_{31}\rho_{k\to 1} + \cdots + a_{3N}\rho_{k\to N} + a_{3N+1}\rho_{k\to N+1} &\le F_{N+1}. \end{aligned} \tag{32}$$

These equations transform the constraints in equation (21) into a standard form of the independent variable ρ. Let

$$A = \begin{bmatrix} D_k^0 & D_k^1 & \cdots & D_k^N & D_k^{N+1} \\ a_{10} & a_{11} & \cdots & a_{1N} & a_{1N+1} \\ a_{20} & a_{21} & \cdots & a_{2N} & a_{2N+1} \\ a_{30} & a_{31} & \cdots & a_{3N} & a_{3N+1} \\ 1 & 1 & \cdots & 1 & 1 \end{bmatrix}, \tag{33}$$

where A refers to the constraint matrix formed by this constraint equation set. Let $b = [\,T\ \ F_n\ \ F_0\ \ F_{N+1}\ \ M_k\,]^T$, where b refers to the right vector of this constraint equation set. The value ranges of independent variable ρ can be expressed by constraints (6) to (8) of the objective function, which are denoted by lb and ub.

The basic process of the BB algorithm is shown in Table 1.

Since the BB algorithm searches the solution space in a breadth-first way, the original problem is divided into multiple branches to search for the optimal solution at the same time, eliminating a large number of nodes that have no chance to become the best value.

A local-edge-cloud edge computing network has K UE and N MEC nodes, the data of each UE is divided into M task blocks, and it is necessary to determine the allocation strategy of the UE task blocks and the offloading node of the partitioned data. The time complexity of the UE task block allocation process is determined to be $O(K2^M)$. Since multitask blocks are transmitted at the same time, there is no need for sorting by the new allocation strategy, and the optimal solution can be directly searched for the data offloading node. The computational complexity of this process is $O(K(N+2)3^{M-1})$, and the sum of the two is the overall computational complexity of the BB algorithm $O(K(2^M + (N+2)3^{M-1}))$.

Table 1: Basic process of the BB algorithm.

BB algorithm
Input: coefficient vector f of the objective function, inequality constraint matrix A, right vector of inequality constraint b, upper and lower bounds of independent variables lb and ub
Output: minimize network energy consumption (E_{total}) and task allocation to different nodes ($\rho_{k\to 0}, \rho_{k\to 1}, \cdots, \rho_{k\to N}, \rho_{k\to N+1}$)
(1) Set the optimal solution $\rho = \Phi$, and the optimal upper bound of the function is $F = +\infty$.
(2) Calculate E_{total} for the initial task allocation strategy. Whether the test coefficient under this allocation strategy is nonpositive is determined by simplex.
(3) If the test coefficient is nonpositive, then the independent variable is the optimal value (ρ^*); otherwise, no optimal value is obtained, and $E_{\text{total}} = +\infty$.
(4) The value of the independent variableρ is adjusted by simplex to make all test coefficients nonpositive and meet $E^*_{\text{total}} < F$. All components of ρ^* are integers. Therefore, E^*_{total} and ρ^* are the outputs of the objective function.
(5) Let $E^*_{\text{total}} < F$. Some components of ρ^* are not integers, and the noninteger components of option ρ^* are denoted by ρ^*_k. A dichotomous approach is applied to divide the original lower constraint into two incompatible constraints.
(6) The optimal solutions to the newly formed constraints are solved by simplex. The above steps are repeated until the output x value is an integer.

5. Simulation Results

The multinode computing offloading algorithm proposed in Section 3 is compared with the traditional single cloud offload and multinode mutual transmission computing offloading algorithms. These algorithms are compared under different data sizes with overall network energy consumption as the measurement standard. For the multinode collaborative offload model, three MEC nodes are set, and $\rho_0, \rho_1, \rho_2, \rho_3, \rho_4$ represent the number of user data units at the local user ends of IoT, the three MEC nodes, and the cloud nodes, respectively. The three MEC nodes correspond to different CPU parameters, and their distances from the local user ends of IoT are denoted by d_1, d_2, d_3, respectively, assuming that the network transmission bandwidth meets user demand. The data processing situation at one local user end of IoT is initially analyzed to compare the network energy consumption of the models.

The assumption is that the network bandwidth is large enough to meet user needs; regardless of the limitation of transmission bandwidth, the effect of data transmission rate on network energy consumption is considered. Following $t_{k,i} = L_{E,i}/r_{E,i}$, the data transmission rates in the three cases are shown in Table 2.

The basic parameters used in the simulation are listed in Table 3.

5.1. Energy Consumption. The computing data size is set to $M_k \sim (1000, 2500)$ to analyze the network energy consumption of the three computing offload models. Figure 2 presents the results.

Figure 2 shows that the network energy consumption of the multinode collaborative computing offload model is lower than that of the other two models. Specifically, when the offload data size at the local user ends of IoT is smaller than 1500 kbit, the network energy consumption of the multinode collaborative computing offload model, which involves parallel data transmission, is equal to that of the multinode mutual transmission computing offload model. Otherwise, the network energy consumption of the multinode collaborative computing offload model is lower than that of the multinode mutual transmission computing offload model. The network optimization effect of the proposed model is similar to that of the multinode mutual transmission computing offload model when the offload data size is small. However, the proposed model shows some advantages in network latency that can be attributed to its parallel transmission of computing tasks. Meanwhile, when the offload data size of the network is large, the proposed model significantly outperforms the other two models in terms of network energy consumption and network latency.

The allocations of offload data size among nodes within the range of 1000 kbit to 5000 kbit are shown in Figure 3. The number of nodes for resource allocation gradually increases along with the computing offload data size. When the data size at the local user ends of IoT is not too large, the data can be processed between the local end users and MEC servers and do not need to be offloaded to cloud nodes for execution. A higher number of tasks for processing correspond to higher node number requirements. The proposed algorithm outperforms the other two models when the task data size at the local user ends of IoT is larger and is thereby conducive to optimizing the network.

When the network bandwidth is changed, the effects of information transmission rate on network energy consumption should be considered.

Figure 4 shows that the lowest network energy consumption is achieved under case 3, whereas the lowest and highest transmission rates are observed under cases 1 and 3, respectively. The overall network energy consumption is negatively correlated with network transmission rate. Given that all computing tasks are transmitted simultaneously in the proposed multinode collaborative computing offload model, a higher transmission rate leads to a larger data size for simultaneous transmission and a higher offload quantity at the local user ends of IoT. The simulation results reveal that the total data sizes under cases 1 to 3 are 5000, 7500, and 17500 kbit, respectively. In sum, the overall data size that the network can process increases along with the network transmission rate. At the same computing data size, the network energy consumption decreases along with an increasing transmission rate.

When the data size for processing at the local user ends of IoT is very large, the overall data transmission rate in the network should be increased. Specifically, when the task data size in the computing network ranges from 10000 kbit to 50000 kbit, the data transmission rate should be increased to 2 Gbit/s. The task allocation among nodes is shown in Figure 5.

Table 2: Transmission rates under the three conditions.

	Transmission rate of MEC_1 $r_{E,1}$ (Mbit/s)	Transmission rate of MEC_2 $r_{E,2}$ (Mbit/s)	Transmission rate of MEC_3 $r_{E,3}$ (Mbit/s)	Transmission rate of MCC r_c (Mbit/s)
Case 1	40	80	20	10
Case 2	80	200	40	20
Case 3	200	400	100	40

Table 3: Simulation parameter settings.

Parameter	Symbols	Value
Data volume of user K	R_k	500 kbit+ϕM_k
Data unloading unit	ϕ	1 kbit
Computing capacities of local user ends	F_0	1.26 GHz
Computing capacities of MEC nodes	F_n	16 GHz
Computing capacities of cloud nodes	F_{N+1}	64 GHz
Number of CPU turns required by the local user to calculate 1 bit of data	C_k^L	500 turn/bit
Number of CPU turns required by theMEC_1 node to calculate 1 bit of data	$C_{k,1}^E$	200 turn/bit
Number of CPU turns required by theMEC_2 node to calculate 1 bit of data	$C_{k,2}^E$	300 turn/bit
Number of CPU turns required by theMEC_3 node to calculate 1 bit of data	$C_{k,3}^E$	100 turn/bit
Number of CPU turns required by the cloud node to calculate 1 bit of data	C_k^C	50 turn/bit
CPU energy consumption of the local user per turn	m_k^L	10 W
CPU energy consumption of the MEC_1 node per turn	$m_{k,1}^E$	80 W
CPU energy consumption of the MEC_2 node per turn	$m_{k,2}^E$	150 W
CPU consumption of the MEC_3 node per turn	$m_{k,3}^E$	200 W
CPU consumption of the cloud node per turn	m_k^C	1000 W
Latency constraint of the local user	T	100 ms
Distance between the local user and MEC_1 node	d_1	120 m
Distance between the local user and MEC_2 node	d_2	100 m
Distance between the local user and MEC_3 node	d_3	150 m

The data transmission rate in the network increases when the data size at the local user ends of IoT is very large. Figure 5 shows that the size of data offloaded to node ρ_3 significantly increases along with the size of data at the local user ends given the low CPU energy consumption recorded at ρ_3. When the computing task is very large, the CPU energy consumption becomes a main influencing factor for network energy consumption. The total data size offloaded to the cloud nodes continuously increases along with offload data size, thereby highlighting the superiority of the edge-cloud cooperation mechanism under a large data size.

6. Conclusions

To realize green communication in smart homes, a multinode collaborative computing offload model is proposed in this paper. In this model, the local user ends of IoT segment the computing task following certain rules. Afterward, the segmented data are reasonably distributed and simultaneously transmitted to multiple nodes for implementation. The traditional single-cloud computing offload model and multinode mutual transmission computing offload model are analyzed on this basis. By treating the overall network energy consumption as the optimization goal and latency as the optimization condition, the allocation of resources among MEC nodes is determined by using a BB algorithm. The proposed model is also compared with the two aforementioned traditional models. Under a large offload task size, the proposed multinode collaborative computing offload model achieves the lowest network energy consumption and the best latency characteristics among all models. The CPU parameters of the

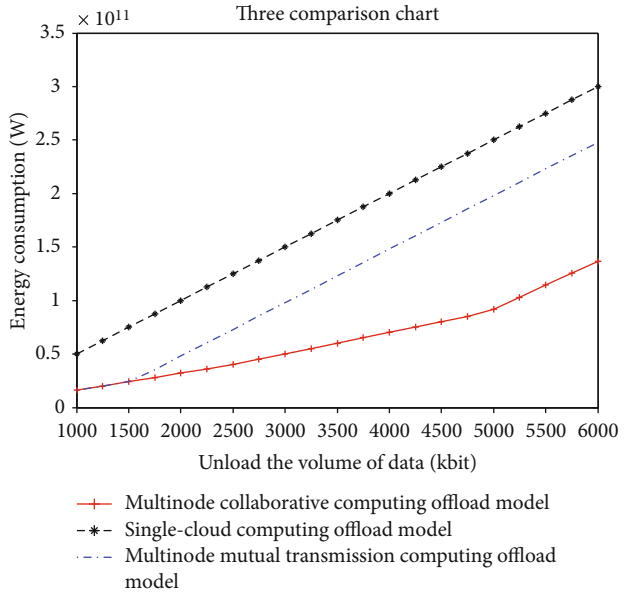

Figure 2: Energy consumption of the three offload models.

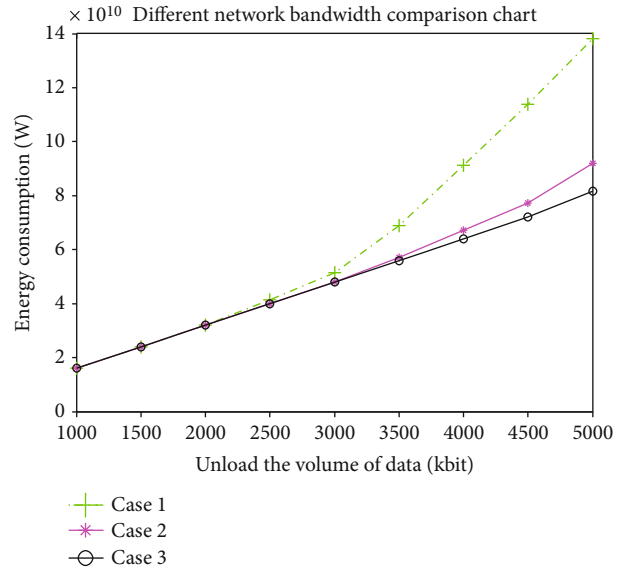

Figure 4: Energy consumption under different network bandwidths.

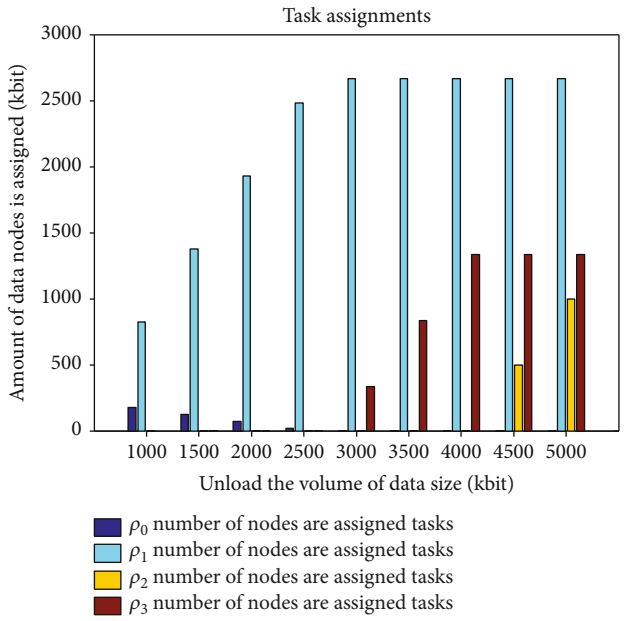

Figure 3: Task allocation among nodes in the multinode collaborative computing offload model.

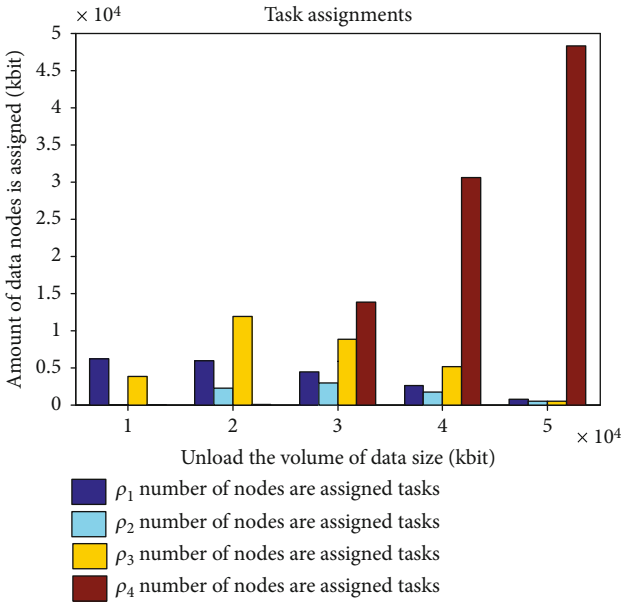

Figure 5: Comparison of offload schemes under a large task data size.

MEC nodes greatly influence the network energy consumption. Under a large data size, the multi-MEC node and edge-cloud collaborative model show improved network characteristics. Meanwhile, both network bandwidth and information transmission rate can influence the data offload performance of the network to some extent. In a multinode collaborative computing offload model, a parallel transmission of segmented data tasks is applied to process large computing tasks at a low overall network energy consumption and high data transmission rate.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

We declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61771195), the Natural Science Foundation of Hebei Province (No. F2018502047), and the Fundamental Research Funds for the Central Universities (No. 2020MS098).

References

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] H. Hui, C. Zhou, S. Xu, and F. Lin, "A novel secure data transmission scheme in industrial Internet of things," *China Communications*, vol. 17, no. 1, pp. 73–88, 2020.

[3] J. Y. Kim, H. Lee, J. Son, and J. Park, "Smart home web of objects-based IoT management model and methods for home data mining," in *2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pp. 327–331, Busan, 2015.

[4] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, *Mobile edge computing—a key technology towards 5G*, Eur. Telecommun. Stand, Inst, White Paper, Sophia Antipolis, France, 2015.

[5] Z. Wang, "An adaptive deep learning-based UAV receiver design for coded MIMO with correlated noise," *Physical Communication*, vol. 99, pp. 1–10, 2020.

[6] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Latency-optimal computation task scheduling for mobile-edge computing systems," in *Proc. IEEE Int.Symp. Inf. Theory (ISIT)*, pp. 1451–1455, Barcelona, Spain, 2016.

[7] Z. Ning, P. Dong, X. Kong, and F. Xia, "A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4804–4814, 2019.

[8] L. Yang, J. Cao, H. Cheng, and Y. Ji, "Multi-user computation partitioning for latency sensitive mobile cloud applications," *IEEE Transactions on Computers*, vol. 64, no. 8, pp. 2253–2266, 2015.

[9] Y. Zhao, S. Zhou, T. Zhao, and Z. Niu, "Energy-efficient task offloading for multiuser mobile cloud computing," in *Proc. IEEE/CIC Int. Conf.Commun. China (ICCC)*, pp. 1–5, Shenzhen, China, November 2015.

[10] H. Zhao, *Research on computational unloading in resource-limited mobile edge computing system*, Beijing university of posts and telecommunications, 2019.

[11] O. Muñoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 10, pp. 4738–4755, 2015.

[12] C. You and K. Huang, "Multiuser resource allocation for mobileedge computation offloading," in *Proc. IEEE Glob. Commun. Conf.(GLOBECOM)*, pp. 1–6, Washington, DC, USA, December 2016.

[13] J. Chen, S. Chen, Q. Wang, B. Cao, G. Feng, and J. Hu, "iRAF: a deep reinforcement learning approach for collaborative mobile edge computing IoT networks," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 7011–7024, 2019.

[14] W. Fan, Y. Liu, B. Tang, F. Wu, and Z. Wang, "Computation offloading based on cooperations of mobile edge computing-enabled base stations," *IEEE Access*, vol. 6, pp. 22622–22633, 2018.

[15] K. Habak, M. Ammar, K. A. Harras, and E. Zegura, "Femto clouds: leveraging mobile devices to provide cloud service at the edge," in *Proc.IEEE 8th Int. Conf. Cloud Comput.*, pp. 9–16, New York, NY, USA, June 2015.

[16] D. Deng, J. Xia, L. Fan, and X. Li, "Link selection in buffer-aided cooperative networks for green IoT," *IEEE Access*, vol. 8, pp. 30763–30771, 2020.

[17] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5G," in *Eur. Telecommun. Standards Inst.*, vol. 11, pp. 1–16, ETSI White Paper, Sophia Antipolis, France, 2015.

[18] 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.

[19] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2016.

[20] Z. Zhao, W. Zhou, D. Deng, J. Xia, and L. Fan, "Intelligent mobile edge computing with pricing in Internet of things," *IEEE Access*, vol. 8, pp. 37727–37735, 2020.

[21] D. Han, S. Li, Y. Peng, and Z. Chen, "Energy sharing-based energy and user joint allocation method in heterogeneous network," *IEEE Access*, vol. 8, pp. 37077–37086, 2020.

[22] D. Guo, L. Tang, X. Zhang, and Y. Liang, "Joint optimization of handover control and power allocation based on multi-agent deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, 2020.

[23] D. W. K. Ng, E. S. Lo, and R. Schober, "Energy-efficient resource allocation in multi-cell OFDMA systems with limited backhaul capacity," *IEEE Transactions on Wireless Communications*, vol. 11, no. 10, pp. 3618–3631, 2012.