Joint Optimization of Computation Offloading, Data Compression, Energy Harvesting, and Application Scenarios in Fog Computing

WENLE BAI, ZIYANG MA, YULONG HAN, MENGLONG WU, ZHONGYUAN ZHAO, MENGKUN LI, AND CHENGCAI WANG

1Institute of Information, North China University of Technology, Beijing 100144, China
2Institute of Information and Communication, Beijing University of Posts and Telecommunications, Beijing 100876, China
3School of Management, Capital Normal University, Beijing 100089, China
4China Academy of Electronics and Information Technology, Beijing 100041, China

Corresponding author: Ziyang Ma (maziyang1107@163.com)

This work was supported in part by the Beijing Natural Science Foundation–Haidian Original Innovation Joint Fund Project under Grant L182039, in part by the Guangxi Key Laboratory of Cryptography and Information Security under Grant GCIS201808, in part by the Foundation of Guizhou Provincial Key Laboratory of Public Big Data under Grant 2019BDKF JJ012, and in part by the National Key Research and Development Program of China under Grant 2018YFB0803900.

ABSTRACT Fog computing is considered to be an effective method to solve the problem of high latency and high energy consumption of IoT devices. A suitable computation offloading strategy can provide a low offloading cost to the user device. Most researches on computation offloading in fog computing focus on one or two targets to improve system performance, however, the actual system needs to meet a comprehensive demand. Therefore, the joint optimization of multi-objective in multiple scenarios is a very meaningful problem. Inspired by this, the paper highlights the joint optimization research for fog computing, which proposes a Joint Computation offloading, Data compression, Energy harvesting, and Application scenarios (JCDEA) algorithm. The related mathematical model is constructed and the cost expressions of local computing, fog computing, and cloud computing are derived. Through the proposed algorithm, solving the computation offloading strategy is transformed into solving the minimum cost and is simplified by controlling strategy factors. Moreover, five simulation experiments are conducted and the meaningful conclusions are drawn, which contain that (1) the cost of fog computing is lower than that of local and cloud computing in most time slots and cloud computing can compensate for fog computing in complex environments; (2) the cost increases approximately linear with the amount of offloaded data; (3) the number of user devices and the compression ratio affect the fog-to-cloud ratio (FCR), while the FCR affects the cost; and (4) the related offloading strategy distribution and the cost are obtained for different scenarios. The JCDEA algorithm always outperforms than that of the random selection algorithm in all scenarios.

INDEX TERMS Fog computing, offloading strategy, energy harvesting, compression ratios.

I. INTRODUCTION

Nowadays, the Internet of Things (IoT) plays an important role in our lives. A large number of IoT devices generate a large amount of data. The demand for cloud service providers is increasing exponentially. However, it is difficult for cloud servers to process so much data in a short time and return the results to the user devices, so cloud computing is not friendly to latency-sensitive applications. It is necessary to find a method to deal with the data generated by IoT devices effectively. Cisco proposed the concept of fog computing in 2012 and regarded fog computing as an extension of cloud computing to meet strict latency requirements and intensive computing needs [1]–[4].

In fog computing, user devices choose to offload some or all of the data of computing-sensitive applications to the fog servers. A single fog server often does not have enough computing resources to process the offloaded task. Therefore, multiple neighboring fog servers handle the offloaded task together, or the fog server uploads the task to the cloud server,

Corresponding author: Ziyang Ma (maziyang1107@163.com)
which increases latency and energy consumption. Hence, the key issue is how to effectively choose the offloading strategy.

For the ability of applications to offload, applications that support code or data partitioning and parallelism (applications that can be partially offloaded) consist of two categories [5]. The first type of application is one that can be divided into multiple offloadable parts, all of which can be offloaded. In extreme cases, this type of application may be completely offloaded if the user device does not process any parts. The second type of application consists of some non-offloadable parts that cannot be offloaded (such as, user input, camera, or fetch locations that need to be performed by the user device) and other offloadable parts [5].

Meanwhile, the energy consumption of user devices is also a key factor. In many scenarios, mobile devices cannot be charged in time, and frequent linking, switching, and data transmission with base stations (BS) lead to excessive instantaneous discharge, which causes a significant decrease in the battery life of mobile devices [6]. Therefore, energy harvesting (EH) technology is seen as a possible solution for harvesting clean energy, including solar, wind, and kinetic energy, to provide computing energy for user devices [7], [8]. Advanced data compression (DC) technology is developing rapidly, and it can be used to reduce the amount of data transmitted on the link [9]. However, data compression requires additional computing resources to compress and decompress data [10]. Different application scenarios have different requirements, corresponding to different offloading strategies. If they are treated equally, higher costs will be incurred. For these purposes, joint optimization has always been a hot research field.

A. RELATED WORK

Computation offloading strategy has been extensively studied in mobile edge computing (MEC) [11] and mobile cloud computing (MCC) [12]. Most studies focus on one [13]–[15] or two [16]–[18] aspects of energy consumption and latency. In [19], the author designed a deadline and priority-aware task offloading (DPTO) strategy to schedule and process offloaded tasks to suitable computing devices. In [20], the author studied the computation offloading strategy in the fog-cloud system and proposed an energy-efficient computation offloading and resource allocation (ECCRA) scheme, which jointly optimized the offloading strategy, computing resource allocation, and transmission power to minimize the system cost. In [21], the author solved the content caching strategy, computation offloading strategy, and wireless resource allocation through deep reinforcement learning at the same time, and proposed a joint optimization scheme that supports fog computing networks. In [22], the author describes the joint optimization of offloading strategy, computing resource, transmission power, and wireless bandwidth allocation in mobile cloud computing to solve computation offloading problems in mixed cloud and fog systems while ensuring user fairness and maximum tolerable latency. In [23], the author proposes an energy-optimal dynamic computation offloading scheme (EDCO) to minimize energy consumption for fog computing in the IIoT. This offloading scheme outperforms local computing, full offloading, and partial offloading in terms of energy consumption and completion time with a fixed computation speed. Research [24] proposes a learning-based channel selection framework with service reliability awareness, energy awareness, backlog awareness, and conflict awareness through a combination of machine learning, Lyapunov optimization and matching theory, which aims at maximizing long-term throughput under long-term constraints on energy and service reliability. Research [25] proposed an air-ground integrated offloading architecture for vehicular edge computing tasks, called the learning-based Intent-Aware Upper Confidence Bound (IUCB) algorithm. This algorithm enables QoE-aware, uRLLC-aware and trajectory similarity-aware 3D intent awareness. In [26], the authors research the computation offloading problem for multiple tasks with different latency requirements when the user device starts one task at a time, and model the fog nodes and remote cloud selection as well as CPU computing resources and fog cloud transmission resource allocation as a Quadratically Constraint Quadratic Programming (QCQP) and provide a solution.

In the research of energy harvesting, the majority of energy harvesting equipment is based on solar energy and wind energy [27]. By photovoltaic cells, solar energy is converted into electrical energy [28]. Wind energy is converted into electrical energy by means of turbines, and the turbines move in two ways, horizontal and vertical turbine [29]. In the study of data compression in fog computing networks, Nguyen et al. [30] demonstrated that data compression has the potential to improve computation offloading performance in hierarchical fog-cloud systems, and discussed computation offloading strategy and resource allocation to determine the optimal compression ratio. The author proposed a non-linear model, describing the additional computing load caused by data compression and decompression. In [31], the author uses a mathematical algorithm called RACOS to jointly optimize edge server selection and energy harvesting in MEC.

Several key issues have not been resolved in the above researches. First of all, the author did not consider the contribution of data compression to computation offloading in joint optimization for fog computing. Secondly, the authors did not distinguish the difference between scenarios for fog computing. Finally, the authors did not consider the edge and cloud to work collaboratively.

B. MAIN CONTRIBUTION

First, a multi-layer architecture is established, including the local computing layer, the fog computing layer, and the cloud computing layer. The local computing layer includes mobile devices, such as mobile phones and wearable devices, as well as stationary devices, such as monitoring devices and smart
mechanical devices. The fog computing layer contains multiple fog servers and the FSM.

Secondly, an optimization algorithm is proposed, which refers to the Joint Computation offloading, Data compression, Energy harvesting, and Application scenarios (JCDEA) algorithm. The computation offloading strategies for different parameters with the JCDEA algorithm are researched, and the related math expressions are derived in theory.

Finally, the varying of cost with the amount of offloaded data, the number of users, application scenarios, and data compression ratios is obtained and compared. Moreover, it is proved that uploading to the cloud makes up for the lack of fog computing in complex environments.

II. SYSTEM ARCHITECTURE AND MATHEMATICAL MODEL

A. SYSTEM ARCHITECTURE

As shown in Figure 1, the architecture contains $K$ user devices, $J$ fog servers, a fog server manager (FSM), and a cloud server. At the fog computing layer, the fog server is deployed at the gNB. If the user device decides the computation offloading strategy by itself, it needs to communicate with multiple gNBs simultaneously, which increases the energy consumption for communication. At the same time, it also increases the calculation energy consumption of the user device. If the cloud server decides the computation offloading strategy for the user devices, the transmission latency is very high when sending the offloaded data, because the user device is far away from the cloud server. In summary, a reasonable solution is selected to set up a manager at the fog computing layer, which is called the fog server manager (FSM). The FSM with a global view determines the computation offloading strategy, which is equivalent to the MNO in [31]. If the fog servers are not able to process the offloaded task for busy work in time, gNB will upload the task to the cloud server for further processing. Each user device is equipped with energy harvesting equipment, all harvested energy is converted into electrical energy. Similar to [31], the EH process is seen as a uniform process that is independently identically distributed.

The time $T$ is divided into multiple time slots of equal length. In each time slot $t$, the user device sends offloading request information including offloaded data, application scenarios, local CPU frequency, channel status, compression ratio, and other parameters about compression to the FSM. The FSM calculates the local cost $\Psi_{k,u}(t)$, the fog cost $\Psi_{k,f}(t)$, and the cloud cost $\Psi_{k,c}(t)$, where $k$ represents the user device, $t$ represents the time slot, $u$ represents user device computing (local computing), $f$ represents fog computing, and $c$ represents cloud computing. Then, FSM selects the minimum cost according to $\Psi_{k,u}(t)$, $\Psi_{k,f}(t)$ and $\Psi_{k,c}(t)$ to determine the offloading strategy, and delivers this strategy to the user device.

If selecting fog computing, the user device compresses the offloaded task and uploads it to the fog server. If selecting cloud computing, the user device compresses the offloaded task and uploads it to the fog servers, and the fog servers further compress the offloaded data to reduce the amount of data transferred and upload it to the cloud server. Since the computation result is much smaller than the offloaded data, the transmission time of results is ignored.
Here, the data compression and decompression model are cited as in [30], which proved the relationship between the compressed data and the CPU cycles. The Gaussian movement model is used to describe the movement of the user devices as in [31].

B. MATHEMATICAL MODEL

Assuming that the raw data is $b_1$, and the compressed data is $b_o$. The data compression ratio is

$$\omega = \frac{b_1}{b_o}.$$  \hspace{1cm} (2-1)

According to [30], the CPU cycles for compression and decompression is

$$C_k = \gamma_k \left( \gamma_{k,1} \omega_k^{\gamma_{k,2}} + \gamma_{k,3} \right), \quad \omega_k \in \left[ \omega_k^{u,\text{min}}, \omega_k^{u,\text{max}} \right].$$  \hspace{1cm} (2-2)

Here, $x$ represents ’co’ or ’de’, which correspond to compression or decompression respectively. $\gamma_k$ represents the maximum CPU cycles. $\gamma_{k,1}$, $\gamma_{k,2}$ and $\gamma_{k,3}$ represent the non-negative limited parameters with GZIP, BZ2, and JPEG respectively [30]. GZIP, BZ2, and JPEG are three different compression methods. GZIP is an abbreviation for file compression programs and usually refers to implementations of the GNU Project, where GZIP stands for GNU zip. BZ2 is a data compression algorithm and program developed by Julian Seward and distributed under the Free Software/Open Source Software protocol. JPEG is a compression standard for continuous-tone still images. $\omega_k$ is the compression ratio, which is limited by the minimum $\omega_k^{u,\text{min}}$ and maximum $\omega_k^{u,\text{max}}$ due to the limited CPU frequency.

Suppose that $s_k^u$, $s_k^d$, $s_k^e$ is the strategy factors for local computing, fog computing, and cloud computing, respectively. $s_k^u = 1$, $s_k^d = 1$, $s_k^e = 1$ mean the offloaded task is processed at the user device, the fog servers, and the cloud server, respectively. Otherwise, the factors are zero. Since the offloaded task is processed at one location, $s_k^u + s_k^d + s_k^e = 1$ [30].

Local CPU cycles consist of three aspects: the CPU cycles that must be executed locally, the CPU cycles that the offloaded task executes locally, and the CPU cycles for compression. The CPU cycles that must be executed locally are the CPU cycles consumed by non-offloadable applications, including cameras, voice recorders, etc. The CPU cycle that must be executed locally are related to the users’ usage habits and are very random [30]. If the user device chooses the local computing, it means that the offloading failed and the offloaded task was performed locally. The CPU cycles that the offloaded task executes locally are the CPU cycles required for an offloadable application to execute locally if the user device selects local computing. If the user device does not choose the local computing, it means that the user device needs to compress the offloaded task. The CPU cycles for compression are the CPU cycles that the user device needs to compress the offloaded task if the user device does not select the local computing. The applications in the article are partially offloadable applications and are divided into two parts. The first part is that must be executed locally, and the second part is that can be offloaded, and the data amount of the first part is much smaller than that of the second part. Suppose the data size to be processed for essential local execution and offloading is $b_{k,u}$ and $b_{k,o}$, respectively. The CPU cycles processing 1bit data are $\eta$. The CPU cycles for local execution are $C_{k,u} = \eta b_{k,u}$. Assuming that the CPU cycles for compression are $C_{k,o}$, the local CPU cycles are

$$C_k^u = C_{k,u} + s_k^d C_{k,o} + \left( 1 - s_k^d \right) C_k^{de}.$$  \hspace{1cm} (2-3)

Here, $C_{k,o}$ are the CPU cycles of the offloaded task processed locally. If local computing ($s_k^d = 1$), the local CPU cycles include the CPU cycles of essential local execution and offloading. If not local computing ($s_k^d = 0$), the local CPU cycles include the CPU cycles of essential local execution and compression. The local CPU frequency is $f_k^u$, and the computing time is

$$t_{k,1} = \frac{C_k^u}{f_k^u}.$$  \hspace{1cm} (2-4)

If the energy coefficient specified by the CPU model is $\lambda$, the local computing energy consumption is

$$\varepsilon_{k,1} = \lambda C_k^u \left( \frac{f_k^u}{p_k} \right)^2.$$  \hspace{1cm} (2-5)

Let $I_{k,j}(t)$ be the connection indicator. $I_{k,j}(t) = 1$ means that the user device $k$ is connected to the fog server $j$. Otherwise, $I_{k,j}(t) = 0$. The channel power gain is $h_{k,j}(t)$. The data transmission rate is

$$R_{k,j}(t) = B \log_2 \left( 1 + \frac{h_{k,j}(t) \cdot p_k}{N} \right).$$  \hspace{1cm} (2-6)

where $B$ represents the transmission bandwidth, $p_k$ represents the transmission power. $N$ represents the noise. When the compression ratio is $\omega_k^{u,de}$ and the offloaded data is $b_{k,i}$, the transmission time is

$$t_{k,2} = \frac{b_{k,i}}{\omega_k^{u,de} R_{k,j}(t) \sum_j I_{k,j}(t)}.$$  \hspace{1cm} (2-7)

The formula (2-7) is composed of two parts. One is to divide the compressed data equally to $J$ fog servers, and the other is the data transmission time between user $k$ and fog server $j$. The transmission energy consumption is $\varepsilon_{k,2} = p_k t_{k,2}$. The fog server decomposition and computing time is

$$t_{k,3} = \frac{C_{k,o} + C_k^{de}}{f_k^f \sum_j I_{k,j}(t)},$$  \hspace{1cm} (2-8)

where $f_k^f$ is the CPU frequency of the fog servers, and $C_k^{de}$ is the CPU cycles for decompression. If the time coordinating a fog server is $\varphi$, the fog server coordination time is

$$t_{k,4} = \varphi \sum_j I_{k,j}(t).$$  \hspace{1cm} (2-9)
optimal energy harvesting is

The transmission rate between the fog and cloud is $d_k$. The transmission time is

$$ t_{k,5} = \frac{b_{k,i}}{\omega_{k,a} \omega_{k,f} d_k}. \quad (2-10) $$

The computing time of the cloud server is fixed $T_C$. The optimal energy harvesting is $E_{k,h}(t)$. The latency factor is $\alpha$. The energy consumption factor is $\beta$. And $\alpha + \beta = 1$. $\alpha < \beta$ means the energy consumption requirement is more stringent than the latency requirement, and the application scenario is an energy-sensitive scenario. $\alpha > \beta$ means this application scenario is a latency-sensitive scenario. $\alpha = \beta$ means this application scenario is a general scenario. Combined Formula (2-1) - (2-10), the overall cost of the user device $k$ is

$$ \Psi_k(t) = \alpha \left[ t_{k,1} + (1 - s^u_k) t_{k,2} + s^f_k t_{k,3} + s^f_k t_{k,4} ight] $$

$$ + \beta \left[ E_{k,1} + (1 - s^u_k) E_{k,2} - E_{k,h}(t) \right] $$

$$ \left[ C_{k,a} s^u_k C_{k,o} + (1 - s^u_k) C_{k}^{co} \right] \frac{b_{k,i}}{f_k^u} $$

$$ + \left( 1 - s^u_k \right) \frac{b_{k,i}}{\omega_{k,a} R_{k,j}(t)} \sum_j I_{k,j}(t) \right) $$

$$ + \frac{\omega_{k,a} R_{k,j}(t)}{\omega_{k,a} \omega_{k,f} d_k} + s^f_k T_C \right] $$

$$ + \beta \left[ E_{k,j}(t) + s^f_k C_{k,o} \right] f_k^{u2} $$

$$ + \left( 1 - s^u_k \right) p_k \frac{b_{k,i}}{\omega_{k,a} R_{k,j}(t)} \sum_j I_{k,j}(t) \right) $$

$$ = \alpha \left[ t_{k,1} + (1 - s^u_k) t_{k,2} + s^f_k t_{k,3} + s^f_k t_{k,4} \right] $$

$$ + \beta \left[ E_{k,1} + (1 - s^u_k) E_{k,2} - E_{k,h}(t) \right] $$

$$ \left[ C_{k,a} s^u_k C_{k,o} + (1 - s^u_k) C_{k}^{co} \right] \frac{b_{k,i}}{f_k^u} $$

$$ + \left( 1 - s^u_k \right) \frac{b_{k,i}}{\omega_{k,a} R_{k,j}(t)} \sum_j I_{k,j}(t) \right) $$

$$ + \frac{\omega_{k,a} R_{k,j}(t)}{\omega_{k,a} \omega_{k,f} d_k} + s^f_k T_C \right] $$

$$ + \beta \left[ \lambda \left[ C_{k,u} + s^u_k C_{k,o} \right] f_k^{u2} + \left( 1 - s^u_k \right) p_k \right. $$

$$ \left. \frac{b_{k,i}}{\omega_{k,a} R_{k,j}(t)} \sum_j I_{k,j}(t) \right) $$

$$ + \left( 1 - s^u_k \right) p_k \frac{b_{k,i}}{\omega_{k,a} R_{k,j}(t)} \sum_j I_{k,j}(t) \right] $$

(2-11)

It can be seen from the Formula (2-11) that the cost of the user device $k$ consists of two parts, the latency cost and the energy consumption cost. $\alpha$ and $\beta$ determine the application scenarios. The latency cost includes 6 parts, which are local computing and compression latency, fog and cloud transmission latency, and cloud server coordination latency. The energy consumption cost consists of three parts: local computing and compression energy consumption, wireless transmission energy consumption, and harvested energy. After the local computing cost, fog computing cost, and cloud computing cost are respectively determined, the computation offloading strategy corresponding to the smallest cost among the three is taken as the computation offloading strategy for the user device $k$ in the time slot $t$. Thus, solving suitable computation offloading strategies transforms into solving the minimum of (2-11), which is

$$ P_1(\text{Problem1}) : \text{minimize} \Psi_k(t) \quad (2-12) $$

subject to $s^u_k$, $s^f_k$, $s^o_k \in [0, 1]$ (2-13)

$$ s^u_k + s^f_k + s^o_k = 1 \quad (2-14) $$

Here, $\tau_d$ represents the offloading deadline which is equal to the length of the time slot. $\psi_{\max}$ is the maximum instantaneous discharge threshold. $E_{k,h}\left( t \right)$ is the maximum harvesting energy. (2-17) indicates the computing time and transmission time cannot exceed a time slot. (2-18) indicates the offloading energy consumption cannot exceed the maximum instantaneous discharge threshold.

III. PROPOSED ALGORITHM

The variables in the cost function from $t$ to $t+1$ are $I_{k,j}(t)$, $R_{k,j}(t)$ and $E_{k,h}(t)$. These variables are affected by location, so that the system state of $t+1$ is related to that of $t$, but not to the system state of $t-1$. This is because the state before $t-1$ only affects the state of $t$ and not the state of $t+1$. It is obvious that the next state of the system is only related to the current state, not to history. Hence, the joint optimization problem $P_1$ is a Markov process. If using traditional methods to solve the problem, we first need to discretize it. However, there are too many feasible solutions after discretizing. It is impractical to traverse all states [31]. So, we propose to simplify the $P_1$ by controlling strategy factors.

A. LOCAL COMPUTING

In local computing, it is seen from Section II that the strategy factors are $s^u_k = 1, s^f_k = s^o_k = 0$, and $P_1$ is simplified to

$$ P_2 : \text{minimize} \Psi_{k,a}(t) \quad (3-1) $$

subject to $0 \leq \frac{C_{k,u} + C_{k,o}}{f_k^{u2}} \leq \tau_d \quad (3-2)$
If the penalty $Q_k$ is generated, $D_k(t) = 1$, which means the offloaded task is delayed a time slot. Otherwise, $D_k(t) = 0$. Therefore, $P_3$ can be transformed into

$$P_4 : \text{minimize } \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{k} \mathbb{E}(I_k(t))$$

subject to $0 \leq C_{k,u} + \frac{\omega_{k,u}}{f_k^u} \leq \omega_{k,u}$

$$0 \leq \frac{C_{k,o} + \frac{\omega_{k,o}}{f_k^o} + \frac{b_{k,i}}{\omega_{k,o} R_{k,j}(t)}}{f_k^o} \leq \omega_{k,o} R_{k,j}(t) \sum_j I_k,j(t)$$

$$0 \leq \frac{C_{k,o} + \frac{\omega_{k,o}}{f_k^o} + \frac{b_{k,i}}{\omega_{k,o} R_{k,j}(t)}}{f_k^o} \leq \psi_{max}$$

$$0 \leq E_{k,h}(t) \leq E_{k,h}^{max}(t)$$

$$I_k,j(t) \in [0, 1]$$

$$\alpha + \beta = 1$$

Assume that

$$\Theta(t) = [\psi_1(t), \ldots, \psi_K(t)]$$

which is the energy of the system. Given a set of parameters

$$\theta = [\theta_1, \ldots, \theta_K]$$

and $\theta_k \geq 0$, a Lyapunov function is constructed,

$$L(\Theta(t)) = \frac{1}{2} \sum_{k=1}^{K} (\psi_k(t) - \theta_k)^2 \leq \sum_{k=1}^{K} \psi_k(t)^2$$

$$\theta_k \geq V \frac{B \log_2(1 + \frac{b_{k,i}}{k_{th}}} (\frac{p_k b_{k,i}}{N}) - \frac{b_{k,i}}{b_{k,i}} \right)$$

$$+ M_{th} \frac{C_{k,o}}{f_k^o} + \min \left\{ E_{k,all}^{max}, \psi_{max} \right\}$$

$$E_{k,all}^{max} = \varepsilon_{k,1} + \sum_{j=1}^{N} \varepsilon_{k,j}^{max}$$

$$\varepsilon_{k,j}^{max} = \frac{p_k}{\tau_d - t_{k,j}}$$

$$h_{k,1}^{max} = \max h_{k,j}(t)$$

The Formula (3.23) represents the power of the user device $k$ is always stable near a non-negative $\theta_k$. The Lyapunov drift function is

$$\Delta(\Theta(t)) = E[L(\Theta(t+1)) - L(\Theta(t))]$$

The evolution equation of energy is

$$\psi_k(t+1) = \psi_k(t) - \varepsilon_{k,1} - \varepsilon_{k,2} + E_{k,h}(t)$$

According to the Formula (3.28) and (3.29), we obtain that

$$\psi_k(t+1) - \varepsilon_{k,1} - \varepsilon_{k,2} + E_{k,h}(t)$$

$$\leq \text{min}(\frac{E_{k,all}^{max}}{\psi_{max}} - 1, 1)$$

$$+ (E_{k,all}^{max} + \tau_d - t_{k,4})$$
Then, we get that
\[
\Delta \left( \Theta(t) \right) \leq \sum_{k=1}^{K} \psi_k^2 \left( E_{k,h}(t) - \varepsilon_{k,1} - \varepsilon_{k,2} \right) + C. \quad (3-31)
\]

Here, \( C = \frac{1}{2} \sum_{k=1}^{K} \left[ \left( \varepsilon_{max}^{k,h} \right)^2 + \left( \varepsilon_{max}^{k,all} \right)^2 \right] \). Define the upper boundary of \( \tilde{P}_4 \) as
\[
\Delta_{V}^{up} \left( \Theta(t) \right) = \sum_{k=1}^{K} \psi_k^2 \left( E_{k,h}(t) - \varepsilon_{k,1} - \varepsilon_{k,2} \right) + V \sum_{k=1}^{K} \Xi \left( I_k(t) \right) + C. \quad (3-32)
\]

This can convert \( P_4 \) to
\[
P_5 : \text{minimize } \Delta_{V}^{up} \left( \Theta(t) \right)
\]
subject to \( 0 \leq \frac{C_{k,u} + C_{k}^{co}}{f_k^{u}} + \frac{b_{k,i}}{\omega_{k,u} R_{k,j}(t)} \sum_{j} I_{k,j}(t) \)
\[
+ \frac{C_{k,o} + C_{k}^{de}}{f_k^{u}} \sum_{j} I_{k,j}(t) + \phi \sum_{j} I_{k,j}(t) \leq \tau_d \quad (3-33)
\]
\[
0 \leq \lambda \left( C_{k,u} + C_{k}^{co} \right) f_k^{u2}
+ p_k \frac{b_{k,i}}{\omega_{k,u} R_{k,j}(t)} \sum_{j} I_{k,j}(t) \leq \psi_{max} \quad (3-34)
\]
\[
0 \leq E_{k,h}(t) \leq E_{max}^{k,h}(t) \quad (3-35)
\]

Therefore, our goal is to find an asymptotically optimal solution for \( P_5 \). \( P_5 \) can be decomposed into two parts, optimal energy harvesting and optimal fog server selection.

\[
P_6 \text{ (optimal energy harvesting) :}
\]
minimize \( \sum_{k=1}^{K} \psi_k^2 \left( E_{k,h}(t) \right) \)
subject to \( 0 \leq E_{k,h}(t) \leq E_{max}^{k,h}(t) \)
\[ \quad \text{(3-37)} \]

Then, the optimal energy harvesting is
\[
E_{k,h}^*(t) = E_{max}^{k,h}(t) \cdot \left\{ \psi_k(t) \leq 0 \right\} \quad (3-38)
\]

\[
P_7 \text{ (optimal fog server selection) :}
\]
minimize \( \sum_{k=1}^{K} \psi_k^2 \left( E_{k,h}(t) \right) + V \sum_{k=1}^{K} \Xi \left( I_k(t) \right) \)
subject to \( 0 \leq \frac{C_{k,u} + C_{k}^{co}}{f_k^{u}} + \frac{b_{k,i}}{\omega_{k,u} R_{k,j}(t)} \sum_{j} I_{k,j}(t) \)
\[
+ \frac{C_{k,o} + C_{k}^{de}}{f_k^{u}} \sum_{j} I_{k,j}(t) + \phi \sum_{j} I_{k,j}(t) \leq \tau_d \quad (3-40)
\]
\[
0 \leq \lambda \left( C_{k,u} + C_{k}^{co} \right) f_k^{u2}
+ p_k \frac{b_{k,i}}{\omega_{k,u} R_{k,j}(t)} \sum_{j} I_{k,j}(t) \leq \psi_{max} \quad (3-41)
\]
\[
0 \leq E_{k,h}(t) \leq E_{max}^{k,h}(t) \quad (3-39)
\]

Then, the RACOS algorithm is used to calculate \( P_7 \) to get the fog server selection [31]. We substitute the fog server selection and the energy harvesting into \( P_5 \) to get the minimum cost of fog computing.

\[ \]
To the end, the problem decomposition of the JCDEA optimization algorithm in fog computing is shown in Figure 2. And the JCDEA algorithm is shown below. It is assumed that there are \( K \) user devices and \( T \) time slots.

**The JCDEA Optimization Algorithm in Fog Computing**

For \( t = 1 \) to \( T \) do

At the beginning of the time slot \( t \), the channel state is obtained

for \( k = 1 \) to \( K \) do

Calculate the local cost and harvesting energy \( P_2 \)

end for

for \( k = 1 \) to \( K \) do

Obtain the fog harvesting energy by solving \( P_6 \)

Obtain the fog server election by solving \( P_7 \)

if fog server selections == 0

fog cost = local cost

else

Obtain fog cost by solving \( P_3 \)

end for

end for

The offloading method that is the smallest cost is used as the offloading strategy for the user device in the time slot.

Update the battery energy and user devices’ location.

end for

**IV. SIMULATING RESULTS**

**A. PARAMETER SETTINGS**

The length of the time slot is set to 2ms. There are 50 time slots. The longest transmission distance of gNB is 400 meters. The shortest distance is 150 meters. The maximum channel power gain is \( 1.02 \times 10^{-13} \). The maximum CPU cycle is \( \gamma_{k,0} = \mu b_k \). The distribution of base stations in Melbourne is used for experiments [32]. There are 80 user devices in the system, and their distribution and movement are random. The type of application offloaded in the experiment is a partially offloaded application, consisting of the parts that must be executed locally and the parts that can be offloaded. The remaining parameters are shown in Table1. The parameters related to compression in Table1 refer to [30], and the parameters related to transmission and calculation refer to [31].

**B. SIMULATION RESULTS**

Figure 3 shows the cost of local computing, fog computing, and cloud computing of the 5th user device. The cost of local computing remains constant as it is not affected by wireless communication. The cost of fog computing is lower than that of local and cloud computing in most time slots. Only in the two time slots, the cost of fog computing is equal to that of local computing and higher than that of cloud computing. This is due to the fact that in both time slots, the user device does not find suitable fog servers to offload the task, and it uploads to the cloud server. It can be seen that cloud computing makes up for the lack of fog computing in the complex environment. The reason for the small proportion of local computing is that local computing is suitable for applications that must be executed locally, including cameras, voice recorders, etc. So for applications that can be offloaded, local computing is not practical.

As shown in Figure 4 shows the cost and connection distribution on the different amount of offloaded data.
When offloading less than 9000 bits, the proportion of fog computing is stable at about 61.18%. When offloading more than 12000 bits, the proportion of fog computing dropped by 7.18%. When the offloaded data increased to 15000 bits, the proportion of fog computing dropped by 13.95%. At the same time, the proportion of cloud computing increased by 13.95% and exceeded the proportion of fog computing. This is because as the amount of offloaded data increases, the CPU cycles for compression and decompression increase, resulting in increasing the cost and the proportion of cloud computing.

When offloading data of 3000bit to 15000bit, the percentage of local computing is 0%. This is due to the fact that the processed data within a time slot is so large that local computing is not applicable. And the offloaded tasks are offloadable tasks in the experiment, not including tasks that must be executed locally. Besides, the random selection algorithm is a benchmarking algorithm in which a fog server is randomly selected for computation offloading or transmission [31]. The cost of the JCDEA algorithm is obviously lower than that of the random selection algorithm for different amounts of offloaded data, with the smallest difference of 0.554E-4 for 3000 bits and the largest difference of 1.048E-4 for 12,000 bits.

Figure 5 shows the cost and connection distribution on the different number of user devices. When the number of user devices increases from 100 to 300 and when the number of user devices is increased by 50 at a time, the cost first increases rapidly and then slowly, increasing by 43.85%, 1.29%, 16.72%, and 5.08% respectively. When the number of user devices is 250, the proportion of cloud computing exceeded that of fog computing for the first time. With the increase of the number of user devices, the proportion of local computing also increases. This is because the increase in the number of user devices intensifies the competition between user devices, resulting in the increase of the coordination cost of fog servers. Therefore, a small part of user devices choose local computing, while another part of user devices choose cloud computing. The user devices of local computing are ignored since there are few user devices that choose local computing. And the Fog-to-Cloud Ratio (FCR) is defined, which is equal to the ratio of the number of user devices which choose fog computing to the number of user devices which choose cloud computing. Combining the cost and the FCR, we can see that the FCR is closely related to the cost, and as the FCR increases, the cost decreases. As the number of user devices increases, the cost of the random selection algorithm increases, but is higher than the cost of the JCDEA algorithm. The largest gap is 0.772E-4 when there are 200 user devices in the system, while the smallest gap is 0.635E-4 when there are 300 user devices in the system.

Figure 6 provides the cost and connection distribution on different application scenarios. Latency factor $\alpha = 0.3$, $\alpha = 0.5$, and $\alpha = 0.7$ represent latency-sensitive scenarios, general scenarios, and energy-sensitive scenarios, respectively. The proportion of fog computing in latency-sensitive and energy-sensitive scenarios is higher than that in general scenarios. It shows that fog computing is effective in
latency-sensitive and energy-sensitive scenarios. The cost of energy-sensitive scenarios is the lowest, however, that of latency-sensitive scenarios is the highest. This is because, in energy-sensitive scenarios, it is permissible that user devices delay one or more time slots to offload the task. However, in a latency-sensitive scenario, user devices try to offload tasks in a short time, so multiple fog servers are required to jointly provide computing at the same time, and the user device spends some additional costs to select the fog servers. It can be seen that the cost of the random selection algorithm increases with increasing latency factor and is higher than the cost of the JCDEA algorithm in all three scenarios, with a maximum difference of 1.136E-4 in the general scenario and the smallest difference of 0.554E-4 in the latency-sensitive scenario.

Figure 7 shows the cost and connection distribution on the different compression ratios. It can be seen from Figure 7 that under the condition of offloaded 3000 bits, 80 user devices, Melbourne base station distribution, and the energy-sensitive scenario, as the compression ratio increases, the cost increases on the overall trend, but it is not monotonous. The proportion of local computing is ignored as the proportion is small. The FCR is 2.06898, 1.56102, 1.37140, 2.24412, 1.47948, and 1.18575 for different compression ratios. So the average cost is negatively related to the FCR in the system. An interesting discovery is that the cost reaches the minimum when the compression ratio is 2.7, and the FCR reaches the maximum, which is 2.24412. It provides a reference application scenario in fog computing. Comparing the random selection algorithm to the JCDEA algorithm, we find that the cost trend of the random selection algorithm is similar to the cost trend of the JCDEA algorithm, but is still higher than the JCDEA algorithm at different compression ratios, with a maximum difference of 1.063E-4 at a compression ratio of 2.7 and a minimum difference of 0.413E-4 at a compression ratio of 2.8.

V. CONCLUSION
In this paper, a joint computation offloading, data compression, energy harvesting, and application scenarios optimization problem is researched in the distributed fog computing wireless network. The three-layer architecture is established, containing multiple user devices and fog servers. The JCDEA algorithm is proposed and the related math expressions are derived in theory, which is applied to several scenarios in fog computing. The JCDEA algorithm transforms solving the offloading strategy into solving the minimum cost of local computing, fog computing, and cloud computing. The experiment results suggested that when the offloaded data in each time slot exceeds 12000 bits and the number of user devices exceeds 200, the proportion of cloud computing is higher than that of fog computing. In addition, the cost of energy-sensitive scenarios is the lowest one of the three, and the proportion of fog computing in latency-sensitive scenarios is the highest. Moreover, when the number of user devices and the compression ratio increases independently, a surprising thing is that the average cost is negatively related to the FCR, and when the compression ratio is 2.7, the cost reaches the lowest. In summary, local computing is suitable for applications that cannot be offloaded. Fog computing is suitable for latency-sensitive scenarios and energy-sensitive scenarios and is suitable for the application of offloading less than 12000 bits within a time slot. Cloud computing is suitable for applications that have no requirements on latency and energy consumption, and offload more than 15000 bits in a time slot. The cost of the JCDEA algorithm is lower than that of the random selection algorithm for different offloading data quantities, the number of user devices, scenarios, and compression rates. In the next work, we will apply more fog server managers to the fog computing layer to obtain more general conclusions.

REFERENCES
[1] C. Arivazhagan and V. Natarajan, “A survey on fog computing paradigms, challenges and opportunities in IoT,” in Proc. Int. Conf. Commun. Signal Process. (ICCSSP), Jul. 2020, pp. 385–389.
A. Younis, B. Qiu, and D. Pompili, “Latency-aware hybrid edge cloud computing,” IEEE Commun. Surveys Tuts., vol. 22, no. 4, pp. 2489–2520, Jul. 2020.

[2] H. Wadhwa and R. Aron, “Fog computing with the integration of Internet of Things Architecture, applications and future directions,” in Proc. IEEE Int'l Conf Parallel Distrib. Process. Appl., Ubiquitous Comput. Commun., Big Data Cloud Comput., Social Comput. Netw., Sustain. Comput. Commun. (ISP/AUCC/BDC/Cloud/SocialCom/SustainCom), Dec. 2018, pp. 987–994.

[3] A. Seal and A. Mukherjee, “On the emerging coexistence of edge, fog and cloud computing paradigms in real-time Internets-of-Everything which operates in the big-squared data space,” in Proc. SoutheastCon, Apr. 2018, pp. 1–9.

[4] P. Mach and Z. Becvar, “Mobile edge computing: A survey on architecture and computation offloading,” IEEE Commun. Surveys Tuts., vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.

[5] Y. Shinn, H. Park, and W. Shin, “Joint time allocation for wireless energy harvesting decode-and-forward relay-based IoT networks with rechargeable and nonrechargeable batteries,” IEEE Internet Things J., vol. 8, no. 4, pp. 2792–2801, Feb. 2021.

[6] H. Sharma, A. Haque, and Z. A. Jaffery, “An efficient solar energy harvesting system for wireless sensor nodes,” in Proc. 2nd IEEE Int. Conf. Power Electron., Intell. Control Energy Syst. (ICEPECIES), Oct. 2018, pp. 461–464.

[7] T. Ruhan, Z. J. Chew, and M. Zhu, “Energy-aware approaches for energy harvesting powered wireless sensor nodes,” IEEE Sensors J., vol. 17, no. 7, pp. 2165–2173, Jul. 2017.

[8] S. A. Abdulzahra, A. K. M. Al-Qurabat, and A. K. Idrees, “Data reduction based on compression technique for big data in IoT,” in Proc. Int. Conf. Emerg. Smart Comput. Inform., (ESCI), Mar. 2020, pp. 103–108.

[9] S. Hamdani, A. Awaisan, and S. Almajali, “Compression techniques used in IoT: A comparative study,” in Proc. 2nd Int. Conf. Trends Comput. Sci. (ICTCS), Oct. 2019, pp. 1–5.

[10] S. Dubey and J. Meena, “Computation offloading techniques in mobile edge computing environment: A review,” in Proc. Int. Conf. Comm. Signal Process. (ICSSP), Jul. 2020, pp. 1217–1223.

[11] S. Guo, M. Chen, K. Liu, X. Liao, and B. Xiao, “Robust computation offloading and resource scheduling in cloudlet-based mobile cloud computing,” IEEE Trans. Mobile Comput., early access, Feb. 14, 2020, doi: 10.1109/TMC.2020.2973993.

[12] S. Yu, B. Dab, Z. Movahedi, R. Langar, and L. Wang, “A socially-aware hybrid computation offloading framework for multi-access edge computing,” IEEE Trans. Mobile Comput., vol. 19, no. 6, pp. 1247–1259, Jun. 2020.

[13] Z. Li, Y. Sun, Z. Hao, and Y. Zhang, “Energy efficient spectrum resource allocation in mobile edge computing,” in Proc. IEEE 5th Int. Conf. Comput. Commun. (ICCC), Dec. 2019, pp. 1114–1119.

[14] A. Younis, B. Qiu, and D. Pompili, “Latency-aware hybrid edge cloud framework for mobile augmented reality applications,” in Proc. 17th Annu. IEEE Int. Conf. Sens., Commun., Netw. (SECON), Jun. 2020, pp. 1–9.

[15] M. Qin, N. Cheng, Z. Jing, T. Yang, W. Xu, Q. Yang, and R. R. Rao, “Service-oriented energy-latency tradeoff for IoT task partial offloading in MEC-enhanced multi-RAT networks,” IEEE Internet Things J., vol. 8, no. 3, pp. 1896–1907, Feb. 2020.

[16] A. Younis, T. X. Tran, and D. Pompili, “Energy-latency-aware task offloading and approximate computing at the mobile edge,” in Proc. IEEE 16th Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS), Nov. 2019, pp. 299–307.

[17] A. H. Ismail, T. A. Soliman, G. M. Salama, N. A. El-Bahnasawy, and H. F. A. Hameed, “Congestion-aware and energy-efficient MEC model with low latency for 5G,” in Proc. 7th Int. Jpn.-Afr. Conf. Electron., Commun., Comput., (JAC-ECC), Dec. 2019, pp. 156–159.

[18] M. Adhikari, M. Mukherjee, and S. N. Srima, “DPTO: A deadline and priority-aware task offloading in fog computing framework leveraging multilevel feedback queueing,” IEEE Internet Things J., vol. 7, no. 7, pp. 5773–5782, Jul. 2020.

[19] Q. Li, J. Zhao, Y. Gong, and Q. Zhang, “Energy-efficient computation offloading and resource allocation in fog computing for Internet of everything,” China Commun., vol. 16, no. 3, pp. 32–41, Mar. 2019.

[20] Y. Wei, F. R. Yu, M. Song, and Z. Han, “Joint optimization of caching, computing, and radio resources for fog-enabled IoT using natural actor-critic deep reinforcement learning,” IEEE Internet Things J., vol. 6, no. 2, pp. 2061–2073, Apr. 2019.

[21] J. Du, L. Zhao, J. Feng, and X. Chu, “Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee,” IEEE Trans. Comput., vol. 66, no. 4, pp. 1594–1608, Apr. 2018.

[22] S. Chen, Y. Zheng, W. Lu, V. Varadarajan, and K. Wang, “Energy-optimal dynamic computation offloading for industrial IoT in fog computing,” IEEE Trans. Green Commun. Netw., vol. 4, no. 2, pp. 566–576, Jun. 2020.

[23] H. Liao, Z. Zhou, X. Zhao, L. Zhang, S. Muntaaz, A. Jolfaei, S. H. Ahmed, and A. K. Bashir, “Learning-based context-aware resource allocation for edge-computing-empowered industrial IoT,” IEEE Internet Things J., vol. 7, no. 5, pp. 4260–4277, May 2020.

[24] H. Liao, Z. Zhou, W. Kong, Y. Chen, X. Wang, Z. Wang, and S. A. Otaibi, “Learning-based intent-aware task offloading for air-ground integrated vehicular edge computing,” IEEE Trans. Intell. Transp. Syst., early access, Oct. 29, 2020, 10.1109/TITS.2020.3027437.

[25] M. Mukherjee, S. Kumar, Q. Zhang, R. Matam, C. X. Mavromoustakis, Y. Lv, and G. Mastorakis, “Task data offloading and resource allocation in fog computing with multi-task delay guarantee,” IEEE Access, vol. 7, pp. 152911–152918, 2019.

[26] D. K. Sah and T. Amgoth, “Renewable energy harvesting schemes in wireless sensor networks: A survey,” Inf. Fusion, vol. 63, pp. 223–247, Nov. 2020.

[27] H. Yu and Q. Yue, “Indoor light energy harvesting system for energy-aware wireless sensor node,” Energy Procedia, vol. 16, pp. 1027–1032, Jan. 2012.

[28] D. Ramasur and G. P. Hancke, “A wind energy harvester for low power wireless sensor networks,” in Proc. IEEE Int. Instrum. Meas. Technol. Conf., May 2012, pp. 2623–2627.

[29] T. T. Nguyen, V. N. Ha, L. B. Le, and R. Scholer, “Joint data compression and computation offloading in hierarchical fog-cloud systems,” IEEE Trans. Wireless Commun., vol. 19, no. 1, pp. 293–309, Jan. 2020.

[30] H. Zhao, S. Deng, C. Zhang, W. Du, Q. He, and J. Yin, “A mobility-aware cross-edge computation offloading framework for partitionable applications,” in Proc. IEEE Int. Conf. Web Services (iCWS), Jul. 2019, pp. 193–200.

[31] P. Lai, Q. He, M. Abdelrazef, F. Chen, J. Hosking, J. Grundy, and Y. Yang, “Optimal edge user allocation in edge computing with variable sized vector bin packing,” in Proc. Int. Conf. Service-Oriented Comput., Nov. 2019, pp. 230–245.

WENLEI BAI was born in Shanxi, China, in 1967. He received the Ph.D. degree in communication engineering from the Beijing University of Posts and Telecommunication of China, in 2006. He is currently working as a Professor with the North China University of Technology. He has published about 20 articles in the related areas. His research interests include wireless communications, statistical signal processing, and multi-user communication.

ZIYANG MA was born in Hebei, China. He received the B.E. degree in EE from the College of Science and Technology, North China Electric Power University, in 2018. He is currently pursuing the M.E. degree in electronics and communication engineering with the North China University of Technology. His research interests include wireless networks, wireless communications, fog computing, and edge intelligence in wireless networks.

YULONG HAN received the M.S. degree in information and communication engineering from the North China University of Technology, Beijing, China, in 2015. He is currently pursuing the Ph.D. degree with the Beijing University of Posts and Telecommunications. He is also a Lecturer with the School of Information Science and Technology, NCUT. His main research interests include wireless communication, including non-orthogonal multiple access, intelligent reflecting surface aided communications, and cooperative communications.
MENGLONG WU received the Ph.D. degree in communications and information systems from the Beijing University of Posts and Telecommunications. He is currently an Associate Professor of information science and technology with the North China University of Technology. He mainly engaged in optical communication.

ZHONGYUAN ZHAO received the B.S. degree in applied mathematics and the Ph.D. degree in communication and information systems from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2009 and 2014, respectively. He is currently an Associate Professor with BUPT. His research interests include fog/edge computing, content caching, and edge intelligence in wireless networks. He received Exemplary Editors Award twice, in 2017 and 2018. He was a recipient of the Best Paper Awards at the IEEE CIT 2014 and WASA 2015, and the Exemplary Reviewer Awards of IEEE Transaction on Communications 2017 and the IEEE Wireless Communication Letters 2019. Since 2016, he has been serving as an Editor for the IEEE COMMUNICATIONS LETTERS.

MENKUN LI received the Ph.D. degree from the China University of Petroleum, Beijing, China, in 2011. He was a Master Supervisor. He held a postdoctoral position with Tsinghua University. He is currently a Lecturer with the School of Management, Capital Normal University, China. His research interests include network security, cloud computing, and techno-economics.

CHENGCAI WANG received the bachelor’s degree in electronic and information engineering from Hangzhou Dianzi University, Hangzhou, China, in 2013, and the master’s degree in control theory and control engineering from Peking University, Beijing, China, in 2016. He is currently a Researcher with the China Academy of Electronics and Information Technology, Beijing. His current research interests include robotics and artificial intelligence.