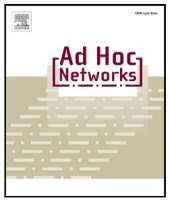
Ad Hoc Networks 153 (2024) 103333

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc



Fairness-aware task offloading and load balancing with delay constraints for Power Internet of Things

Xue Li [a], Xiaojuan Chen [a,*], Guohua Li [b]

[a] School of Electronic Information Engineering, Changchun University of Science and Technology, Changchun, 130000, Jilin, China
[b] Marketing Service Department, Changchun Power Supply Company of State Grid Jilin Electric Power Co., LTD., Changchun, 130000, Jilin, China

ARTICLE INFO

Keywords:
Power IoT
Edge computing
Load balancing
Theil index
Lyapunov optimization

ABSTRACT

The rapid development of the Power Internet of Things (PIoT) enables power smart sensing devices to offload their computation tasks to nearby edges. However, due to the increasing demand for computing and the unbalanced spatial distribution of devices, it is imperative to seek cooperative computing and task scheduling optimization schemes at the edge for PIoT. In this paper, we develop a two-tier cooperative edge network paradigm in PIoT. Then, we define a novel fairness indicator based on the Theil index to measure the allocation balance of the system. We also formulate a fairness and delay guaranteed (FDG) task offloading and load balancing optimization problem, which aims to minimize the allocation difference of the edge network while satisfying the delay constraints for multiple tasks in PIoT. Moreover, we develop a Lyapunov optimization and whale optimization algorithm (LWOA) to solve the problem. The simulation results demonstrate that for two types of typical tasks in PIoT, compared with the NonB scheme, the proposed FDG scheme decreases the time-averaged allocation difference of the system by 10% and 35%, the time-averaged allocation difference within subsystems by 5% and 6%, the time-averaged delay by approximately 5% and 7%, and the time-averaged queue backlog by approximately 30% and 40%. Research, both theoretical and experimental, has demonstrated that cooperation at the edge can significantly improve the performance of PIoT.

1. Introduction

Power Internet of Things (PIoT) connects various sensors, drives, mobile devices, machines, and the Internet in power systems through modern information and communication technology (ICT) and network technology [1–3]. Examples of applications of PIoT networks include advanced metering infrastructure (AMI) [4], high-definition surveillance, data acquisition (SCADA) [5], intelligent inspection [6], and the control system of a thermal power plant [7]. PIoT has led to increasing information interaction between generation, transmission, distribution, and substations, which has improved the reliability, observability, and efficiency of the grid [1,8]. Typically, the data collected by sensors in PIoT devices are transmitted directly to the control center or after preliminary analysis [1]. As the adoption of high proportions of clean energy, massive power sensors, and intelligent distribution terminals becomes more common, new applications are emerging in various aspects of the power system. The data volume and dimensions of the sensing information in the power IoT are experiencing exponential growth [8,9]. Network congestion [10,11], transmission delays [12], and data processing delays are becoming more prominent, demanding increased communication and computing capabilities from PIoT. A promising approach is to offload computationally intensive and latency-sensitive tasks to nearby edge nodes [13]. However, the proliferation of edge services may lead to large-scale cooperative computing and task scheduling optimization problems, which are also key research topics in PIoT [3].

There are three research hotspots for cooperative computing: end-edge cooperation, edge-edge cooperation, and edge-cloud cooperation. End-edge cooperation primarily addresses task offloading between end tasks and neighboring edge servers [14,15]. Edge-edge cooperation refers to task load balancing and resource collaboration of adjacent edge servers [16–23]. Edge-cloud cooperation refers to the collaborative scheduling of cloud and edge resources to meet the resource requirements of multiple computationally intensive tasks, especially in large-scale terminal access scenarios [24,25]. It should be noted that the edge server is the bridge between the cloud and the end devices. As such, efficient resource allocation can significantly enhance task processing, reduce the deployment expenses of edge nodes, save communication link resources, and eliminate computing and communication bottlenecks. Typically, edge resources present obvious heterogeneous [26] and distributed characteristics, and task distribution

* Corresponding author.
E-mail address: cxj001@cust.edu.cn (X. Chen).

https://doi.org/10.1016/j.adhoc.2023.103333
Received 24 July 2023; Received in revised form 14 September 2023; Accepted 20 October 2023
Available online 25 October 2023

Table 1
Comparison with existing works.

	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	[22]	[23]	This paper
Edge server cooperation	×	×	✓	✓	✓	✓	✓	✓	✓	✓	✓
Long-term constraints	✓	✓	×	×	×	✓	✓	✓	✓	×	✓
Multi-type tasks	×	×	×	×	×	×	×	✓	×	×	✓
Temporal correlation	✓	✓	×	×	×	✓	✓	✓	✓	×	✓
Fairness aware	×	✓	×	×	×	×	×	×	×	✓	✓

shows uneven features. Balancing resource and task allocation in multi-edge PIoT environments remains challenging. Thus, we propose a cooperative solution to effectively utilize the resources at the edge.

Several studies have been carried out to address issues related to cooperative computing and optimization of task scheduling at the edge. Meng et al. [16] proposed an online dispatching and scheduling algorithm in the local area networks (LAN) environment, which considers the management of network bandwidth and computing resources together to meet the maximum number of deadlines. To realize energy-aware task offloading for smart edge computing in a wireless metropolitan area network (WMAN), Xu et al. [17] proposed an energy-aware computation offloading method (EACO), which analyzed the unloading time, energy consumption, and load. Xu et al. [18] proposed a two-phase offloading optimization strategy to jointly optimize offload utility (i.e., resource utilization of the edge computing units (ECUs) and time cost of task execution) and privacy in the IoT that supports edge computing. Chen et al. [19] developed a novel online small-cell base station (SBS) peer-to-peer unloading framework considering heterogeneous task arrival patterns in both spatial and temporal domains of small-cell networks. The Lyapunov technique is utilized to maximize long-term system performance while maintaining the energy consumption of the SBSs under individual long-term constraints.

Wang et al. [20] proposed an online optimization strategy for MEC server (MECS) computation task offloading with a sleep control scheme to minimize the long-term energy consumption of the MECS network. With this strategy, the Lyapunov technique was used to solve the optimization problem without requiring future network information. Zhang et al. [21] investigated the resource optimization problem for a MEC system consisting of multiple MEC servers and multiple users, which included the optimization of communication and computation resources among multiple users and the optimization of load balancing among multiple MEC servers. To obtain the optimal system average cost under the constraints of a stable battery level, a Lyapunov-based centralized cost management algorithm (LYPCCMA) was proposed. Wang et al. [22] considered a cooperative three-tier computing network by leveraging vertical cooperation among devices, edge nodes, and cloud servers, as well as horizontal cooperation between edge nodes. The authors proposed a joint optimization offloading decision and computing resource allocation method based on alternating direction method of multipliers (ADMM) and difference of convex functions (D.C.) programming to minimize the average task duration, which was influenced by the limited battery capacity of devices. Zhou et al. [23] defined resource fairness among IoT devices from the user's perspective and formulated a joint optimization problem taking into account system efficiency and fairness. To solve this problem, Zhou et al. proposed a two-level algorithm. The upper-level algorithm combines the advantages of artificial bee colony (ABC), genetic algorithm (GA), and particle swarm optimization (PSO) to create an approximate optimal task offloading decision. The lower-level algorithm, fairness guaranteed resource allocation (FGRA), completely utilizes the server resources to generate a resource allocation scheme with fairness guarantees. To improve fairness among IoT users, Wang et al. [27] formulated a fairness resource allocation problem in an ultra-dense MEC-enabled IoT network with non-orthogonal multiple access techniques (NOMA). However, only the resource allocation of multiple IoT devices with a single MEC server was considered in [27].

Most existing studies focus on optimizing task delays, system energy consumption, and resource pricing strategies. However, only a few solutions consider the system's fairness, which is a crucial indicator of the balanced utilization of edge resources. Additionally, there is limited research on cooperative edge computing in the context of PIoT. PIoT terminals are usually sensors and smart devices distributed around power facilities, which are high in quantity but limited in computing capability. The sensor data in PIoT often require hierarchical analysis and feedback, which differ from conventional networks such as mobile networks, LANs, and MANs. Considering the privacy and sensitivity of the power system, power companies tend to deploy edge nodes and cloud centers independently. Therefore, investigating edge cooperative schemes that are specifically designed for PIoT is necessary. Table 1 compares the proposal presented in this paper with the aforementioned works.

In this paper, we first analyze the main tasks, characteristics, current status, and computing requirements of typical applications in PIoT, as shown in Table 2. Then, we propose a two-tier cooperative edge network paradigm for PIoT and develop a task offloading and load balancing scheme, referred to as the fairness and delay guaranteed (FDG) scheme, to minimize system allocation differences satisfying the delay constraints for multiple types of services in PIoT. The main contributions of this paper are summarized as follows.

(1) We develop a two-tier cooperative edge network paradigm taking into account the extensive distribution of power sensing terminals and the complicated network topology. This model includes the edge server (ES) layer and the access point (AP) layer, enabling hierarchical aggregation and processing of PIoT sensing data, which differs from research schemes under conventional networks such as mobile networks and local area networks.
(2) We investigate the fairness of resource allocation and task scheduling during task offloading and load balancing in the cooperative edge network. Further, we indicate that the system's fairness directly correlates with the allocation differences, define a novel fairness indicator based on the Theil index theory [28], and formulate the fairness-aware model to measure the allocation discrepancy of the system. The primary advantage of this indicator is that it shows both within-group and between-group allocation differences.
(3) We formulate a task offloading and load balancing optimization problem (i.e., FDG problem) taking into account the fairness, multi-task delay constraints, heterogeneity, and dynamic features of the system. The aim is to minimize the long-term unfairness of the system while ensuring that the task is executed within the delay constraint. To the best of our knowledge, no other scheme has been proposed that simultaneously considers all of the above factors in PIoT. Specifically, existing research focuses on the latency constraints of multiple tasks of the same type rather than the long-term latency constraints of various tasks; this is crucial for modeling edge collaboration problems.
(4) We design an approach to solve the proposed problem based on Lyapunov Optimization and Whale Optimization Algorithm (LWOA). Specifically, we use Lyapunov Optimization to transform the FDG problem into a queue stability problem and prove that this queue stability problem is NP-hard in each time slot. Moreover, we use the WOA algorithm with nonlinear factors to find the optimal solution and evaluate our proposed scheme.

The rest of this paper is structured as follows. Section 2 presents the system model. Section 3 details the problem formulation. Section 4 proposes an approach to solve the FDG problem. Section 5 describes our simulations; and Section 6 provides the conclusions.

Table 2
Analysis of typical applications of PIoT.

Typical applications	Tasks	Task characteristics	Status analysis	Computation requirement
High-definition surveillance	Video and image monitoring, data real-time processing, image rendering, etc.	Large bandwidth, low latency sensitivity, dynamic emergencies	Widely used, mostly manual screening, lack of real-time analysis	Strong
Intelligent inspection	Transmission line drone inspection, substation robot inspection, drone inspection, etc.	Large bandwidth, latency sensitivity, dynamic emergencies	Develop rapidly, mostly manual screening, lack of real-time analysis	Strong
Intelligent operation and maintenance	On-site remote fault diagnosis, AR, VR, visual operation and guidance, etc.	Large bandwidth, high latency sensitivity, dynamic emergencies	Gradually apply to typical scenarios	Strong
Electric vehicle service	Vehicle information collection, intelligent charging scheduling, payment management, electric vehicle route planning, demand response, etc.	Small data packet, large in quantity, synthetic analysis difficulty	Develop rapidly	Strong
New energy service	Multi-dimensional analysis of energy and meteorological data, power generation prediction, fan and photovoltaic module operation status analysis, etc.	Small data packet, large in quantity, synthetic analysis difficulty	Develop rapidly	Strong
Production control	Relay protection, distribution automation, emergency load control, etc.	Ultra-reliable, ultra-low latency	Widely used	Moderate

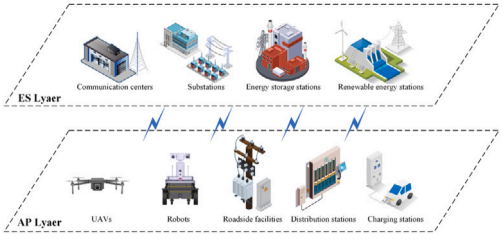


Fig. 1. The two-layer cooperative edge network for PIoT.

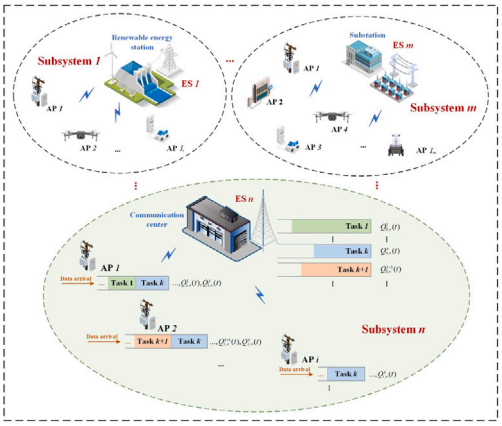


Fig. 2. System model.

2. System model

2.1. Network model

Numerous sensing terminals in PIoT, such as smart meters, inspection terminals, and monitoring terminals, are distributed throughout a city, resulting in scattered geographical locations and complex topologies. However, due to limitations like bandwidth, delay, energy consumption, and grid security, a great deal of PIoT data cannot be transmitted directly to the cloud center, such as video surveillance terminals installed on transmission towers, unmanned aerial vehicle (UAV) inspection terminals for transmission lines, and robot inspection terminals for substations. The data collected by these terminals are usually processed through a storage procedure before being manually screened. Advanced communication technology can solve data transmission delays, but improvement in these applications is required due to limited computational and bandwidth resources, as indicated in Table 2. To efficiently process the data generated by these terminals, this paper proposes a two-layer cooperative edge network model with a hierarchical structure. As shown in Fig. 1, the upper layer of the model is the edge server (ES) layer, which contains edge servers deployed in substations, communication centers, and renewable energy stations. The lower layer is the access point (AP) layer. APs can be the computing units deployed in grid roadside facilities, such as distribution transformer supervisory terminal units (TTUs), feeder terminal units (FTUs), and charging stations. APs can also be specific smart devices (e.g., drones, robots), which should be authorized in advance in the system. Additionally, wired links are used for communication and load balancing between the edge servers, while wireless channels are employed to connect the access points to the edge servers.

In the two-tier cooperative edge network, each ES can serve multiple APs, but each AP only belongs to one ES. We divide the two-tier network into multiple edge subsystems based on the number of ESs. Each edge subsystem includes an ES and its multiple APs, as shown in Fig. 2. The ES in an edge subsystem is known as the main edge server (MES), while other ESs are known as the neighbor edge servers (NESs) of that edge subsystem. Within each edge subsystem, various types of PIoT terminals with limited computing power exist . These terminals generate computational tasks randomly and require APs and ESs to handle computationally intensive and latency-sensitive tasks. In addition, the ES stores the received tasks from NES in the execution queue and does not transmit to other ESs to avoid additional system overhead and the ping-pong effect.

2.2. Task model

We consider a multi-task PIoT edge network consisting of N edge subsystems, each of which is equipped with an ES and I_n APs, as shown in Fig. 2. $\mathcal{N} = \{1, 2, \dots, n, \dots, N\}$ denotes the set of ESs. In edge subsystem n, the ES n is known as MES, and ESs of other edge subsystems are called NES. The set of APs in edge subsystem n is denoted by $\mathcal{I}_n = \{1, 2, \dots, i, \dots I_n | \forall n \in \mathcal{N}\}$. Additionally, $\mathcal{T} = \{1, 2, \dots, t, \dots, T\}$ is the time slot set. The duration of the time slot is τ (in seconds). Furthermore, k denotes the type of task. In each time slot t, a PIoT terminal randomly generates a k-type ($k \in \mathcal{K}$) task and transmits it to its AP. The generation probabilities of different tasks are independent of each other. $\mathcal{K} = \{1, 2, \dots, k, \dots, K\}$ represents the task set, which may

contain PIoT applications such as UAV inspection, equipment condition monitoring, electricity information collection, and smart homes.

In time slot t, each AP in an edge subsystem collects computation tasks from the underlying PIoT terminals. Suppose that the arrival of tasks follows a Poisson process, which is a common assumption on the computation task arrival in an edge system. We denote $Task_{i,n}^{k}(t) = \{\lambda_{i,n}^{k}(t), S_{i,n}^{k}(t), c_k, d_k\}$ as the k-type task received by AP i of edge subsystem n in time slot t, where $\lambda_{i,n}^{k}(t)$ (in tasks) is the arrival rate of $Task_{i,n}^{k}(t)$, which is randomly drawn from $\lambda_{i,n}^{k}(t) \in [\lambda_{i,n}^{k_{\min}}, \lambda_{i,n}^{k_{\max}}]$; $S_{i,n}^{k}(t) = \lambda_{i,n}^{k}(t)s_k(t)$ (in bits) is the data size of $Task_{i,n}^{k}(t)$, where $s_k(t)$ (bits/task) denotes the average data size of the k-type task; and c_k (cycles/bit) and d_k (seconds) denote the required service capability (i.e., the workload) and the maximum execution delay of the k-type task, respectively.

2.3. Computing model

It is assumed that data received by an AP can be split into non-overlapping subsets. These subsets are processed in parallel by the AP and ES. In this paper, we divide $S_{i,n}^{k}(t)$ $(\forall n \in \mathcal{N}, i \in \mathcal{I}_n, k \in \mathcal{K})$ into two subsets proportionally, the *AP-subset* (i.e., $S_{i,n}^{k,A}(t)$) and the *ES-subset* (i.e., $S_{i,n}^{k,E}(t)$). The *AP-subset* can only be executed by AP i and the *ES-subset* can be computed by MES or balanced to NES for execution. Let $\alpha_{i,n}^{k}(t) \in [0, 1]$ represent the partition ratio of $S_{i,n}^{k}(t)$, then

$$S_{i,n}^{k,A}(t) = S_{i,n}^{k}(t)\alpha_{i,n}^{k}(t), \tag{1}$$

$$S_{i,n}^{k,E}(t) = S_{i,n}^{k}(t)(1 - \alpha_{i,n}^{k}(t)). \tag{2}$$

2.3.1. AP-computing

In time slot t, *AP-subsets* arrives at the computation queue of AP waiting for execution. Let $Q_{i,n}^{k}(t)$ denote the computation queue of the k-type task of AP i in the edge subsystem n in time slot t. The evolution of $Q_{i,n}^{k}(t)$ is determined by:

$$Q_{i,n}^{k}(t+1) = \max\left\{Q_{i,n}^{k}(t) + S_{i,n}^{k}(t)\alpha_{i,n}^{k}(t)c_k - f_{i,n}^{k}(t)\tau, 0\right\}, \tag{3}$$

where $S_{i,n}^{k}(t)\alpha_{i,n}^{k}(t)c_k$ is the workload added to the queue in time slot t (i.e., the CPU cycles required by $S_{i,n}^{k,A}(t)$), and $f_{i,n}^{k}(t)$ (in cycles/second) is the computation capacity allocated by AP i to the k-type task; this satisfies $\sum_{k\in\mathcal{K}} f_{i,n}^{k}(t) \le f_{i,n}^{\max}$, where $f_{i,n}^{\max}$ is the maximum available computing resources of AP i in the edge subsystem n.

Here we focus on task offloading within the edge subsystem and load balancing between edge subsystems. We have not considered the transmission delay from the PIoT terminal to the AP in our analysis. The total service delay of $S_{i,n}^{k,A}(t)$ includes queuing delay and computation delay. Queuing delay refers to the time between $S_{i,n}^{k,A}(t)$ joining the queue and AP i starting $S_{i,n}^{k,A}(t)$ execution. Assuming that the system is stable, we can estimate the queuing delay by dividing the queue length by the long-term average computing capacity when the subset arrives.

Let $\bar{f}_{i,n}^{k}$ denote the long-term average of the computation capacity allocated by AP i to compute the k-type task.

$$\bar{f}_{i,n}^{k} \triangleq \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} f_{i,n}^{k}(t) \tag{4}$$

Then the queuing delay of $S_{i,n}^{k,A}(t)$ is:

$$d_{i,n}^{k,A_{que}}(t) = \frac{Q_{i,n}^{k}(t)}{\bar{f}_{i,n}^{k}}, \tag{5}$$

and the computation delay of $S_{i,n}^{k,A}(t)$ is:

$$d_{i,n}^{k,A_{cpu}}(t) = \frac{S_{i,n}^{k,A}(t)c_k}{f_{i,n}^{k}(t)} = \frac{S_{i,n}^{k}(t)\alpha_{i,n}^{k}(t)c_k}{f_{i,n}^{k}(t)}. \tag{6}$$

Accordingly, the service delay of $S_{i,n}^{k,A}(t)$ is:

$$d_{i,n}^{k,A}(t) = d_{i,n}^{k,A_{cpu}}(t) + d_{i,n}^{k,A_{que}}(t). \tag{7}$$

2.3.2. ES-computing

Unlike AP-computing, the computation queue of ES includes the tasks offloaded by the APs that it serves in the edge subsystem and the tasks balanced by NES. To analyze the ES-computing process, we first select two edge subsystems that are neighbors to each other (i.e., $\forall m, n \in \mathcal{N}, i \in \mathcal{I}_n, j \in \mathcal{I}_m$). Let $\gamma_{i,n}^{m} \in \{0, 1\}$ be the balancing decision variable of $S_{i,n}^{k,E}(t)$. If $S_{i,n}^{k,E}(t)$ is balanced to NES m by MES n, $\gamma_{i,n}^{m} = 1$; otherwise, $\gamma_{i,n}^{m} = 0$. Since $\gamma_{i,n}^{m}$ satisfies $\sum_{m\in\mathcal{N}} \gamma_{i,n}^{m} = 1$, the *ES-subset* can be computed by a unique ES. Similarly, $\gamma_{j,m}^{n} \in \{0, 1\}$ denotes whether the *ES-subset* is balanced from MES m to NES n. If $\gamma_{i,n}^{n}$=1, then $S_{i,n}^{k,E}(t)$ is computed by MES n. We define $Q_{0,n}^{k}(t)$ as the computation queue of the k-type task of ES n in time slot t. The evolution of $Q_{0,n}^{k}(t)$ is determined by:

$$Q_{0,n}^{k}(t+1) = \max\left\{Q_{0,n}^{k}(t) + \sum_{m\in\mathcal{N}}\sum_{j\in\mathcal{I}_m} \gamma_{j,m}^{n} S_{j,m}^{k,E}(t)c_k - f_{0,n}^{k}(t)\tau, 0\right\}. \tag{8}$$

If *ES-subset* is executed by MES, the service delay should include the transmission delay from AP to MES, the MES queuing delay, and the MES computing delay. However, if executed by NES, the service delay should include the transmission delay from AP to MES, the transmission delay between MES and NES, the NES queuing delay, and the NES computing delay. Then, we take ES n balancing $S_{i,n}^{k,E}(t)$ to ES m as an example to analyze the service delay.

(1) Transmission delay: Suppose that the wireless channel quality obeys the independent and equally distributed block fading distribution in time. That is, in each time slot, the channel power gain is fixed, and the channel power gain of the same user is independently distributed in different time slots. According to Shannon's formula, the data transmission rate is related to the transmission power, the channel power gain, and the channel bandwidth. The uplink transmission rate between AP i and ES n is:

$$r_{i,n}^{k}(t) = w_{i,n}^{k}(t)\log_2\left(1 + \frac{p_{i,n}(t)h_{i,n}(t)}{\sigma_{i,n}^{2}}\right), \tag{9}$$

where $w_{i,n}^{k}(t)$ is the channel bandwidth, $h_{i,n}(t)$ represents the channel gain from AP i to ES n, $\sigma_{i,n}^{2}$ denotes the receiver noise power, and $p_{i,n}(t)$ characterizes the transmission power of AP i, which meets $p_{i,n}(t) \le p_{i,n}^{\max}$ (i.e., $p_{i,n}(t)$ is no more than the maximum transmission power $p_{i,n}^{\max}$). Then the transmission delay from AP i to ES n is denoted by:

$$d_{i,n}^{k,trA}(t) = \frac{S_{i,n}^{k,E}(t)}{r_{i,n}^{k}(t)}. \tag{10}$$

Let $B_{n,m}$ be the bandwidth from ES n to ES m. We adopt (11) to allocate the bandwidth resources.

$$B_{n,m}^{k}(t) = \frac{S_{i,n}^{k,E}(t)B_{n,m}}{\sum_{i\in\mathcal{I}_n} \gamma_{i,n}^{m} S_{i,n}^{k,E}}. \tag{11}$$

Without loss of generality, we introduce the variable:

$$\Gamma_{n,m}^{k} = \begin{cases} 1, m \ne n \\ 0, m = n, \end{cases} \tag{12}$$

and then the transmission delay between ES n and ES m is:

$$d_{i,n}^{k,trE}(t) = \frac{\Gamma_{n,m}^{k} S_{i,n}^{k,E}(t)}{B_{n,m}^{k}(t)}. \tag{13}$$

(2) The queuing delay: Similar to (4), the long-term average of the computation capacity allocated by ES n to compute the k-type task is denoted as follows:

$$\bar{f}_{0,m}^{k} \triangleq \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} f_{0,m}^{k}(t). \tag{14}$$

Then the queuing delay of ES m is:

$$d_{i,n}^{k,E_{que}}(t) = \frac{Q_{0,m}^{k}(t)}{\bar{f}_{0,m}^{k}}. \tag{15}$$

(3) The computation delay is:

$$d_{i,n}^{k,E_{cpu}}(t) = \frac{S_{i,n}^{k,E}(t)c_k}{f_{0,m}^k(t)} = \frac{S_{i,n}^k(t)\left(1-\alpha_{i,n}^k(t)\right)c_k}{f_{0,m}^k(t)}. \tag{16}$$

According to the above discussion, the service delay of ES-computing is:

$$d_{i,n}^{k,E}(t) = d_{i,n}^{k,trA}(t) + d_{i,n}^{k,trE}(t) + d_{i,n}^{k,E_{que}}(t) + d_{i,n}^{k,E_{cpu}}(t). \tag{17}$$

Since AP and ES provide parallel computing services for the *AP-subset* and the *ES-subset*, the total service delay of $S_{i,n}^{k,E}(t)$ is as follows:

$$d_{i,n}^k(t) = \max\left\{d_{i,n}^{k,A}(t), d_{i,n}^{k,E}(t)\right\}. \tag{18}$$

In addition, considering that the transmission delay between the PIoT terminal and the AP is neglected in this paper, and the time-averaged approximation is used for the queuing delay analysis, the compensation factor $\beta_k \in [0,1], k \in \mathcal{K}$ is introduced to ensure that the long-term service delay maximally meets the task's delay constraint. That is:

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T} d_{i,n}^k(t) \le \beta_k d_k. \tag{19}$$

2.4. Fairness-aware model

Here, we define the concept of fairness and provide a mathematical representation of the long-term fairness of the system.

Definition 1. Fairness indicates that the computing resources allocated to a workload unit for a certain type of task in the system should be equalized.

Notably, various tasks and networks may have distinct metrics, such as latency and overhead. Thus, for example, a smaller difference in resource allocation for a unit of workload ensures the fairness thereof, provided that the conditions of specific indicators are met. In terms of workload, we consider that a fair system should have a workload that is balanced evenly to the computing entities with rich computational resources.

In the proposed two-tier cooperative edge network, the fairness includes allocation differences within the edge subsystem and between multiple edge subsystems, which correspond to task offloading and load balancing processes. In this paper, we develop a time average difference model of the proposed two-tier cooperative edge network based on the Theil index, which is a measure of fairness. The Theil index is a dependable approach for evaluating spatial inequalities and is commonly used to measure the level of disparity in the regional distribution of income [28], healthcare [29], and carbon emissions [30,31]. The Theil index is decomposable, measuring the inequality within and between groups, which correspond to the fairness within and between edge subsystems, respectively. Therefore, in time slot t, the allocation difference of the k-type task can be defined as follows:

$$F_{Th}^k(t) = F_{Tb}^k(t) + F_{Tw}^k(t), \tag{20}$$

where $F_{Tb}^k(t)$ and $F_{Tw}^k(t)$ represent the allocation difference between groups and within groups, respectively.

For simplicity, we refer to the APs and the ES collectively as computational entities. In the edge subsystem m ($\forall m \in \mathcal{N}$), $j = 0$ indicates that the computational entity is the ES, and the other cases indicate that the computational entities are APs. In time slot t, suppose that $x_{j,m}^k(t)$ ($\forall m \in \mathcal{N}, j = 0,1,\dots,I_m, \forall k \in \mathcal{K}$) is the workload of a k-type task allocated to computational entities in the edge subsystem m. The expression of $x_{j,m}^k(t)$ is:

$$x_{j,m}^k(t) = \begin{cases} \sum_{n=1}^{N}\sum_{i=1}^{I_n}\gamma_{i,n}^m S_{i,n}^{k,E}(t)\,c_k, j = 0 \\ S_{j,m}^{k,A}(t)c_k, j = 1,2,\dots,I_m. \end{cases} \tag{21}$$

Let $y_{j,m}^k(t)$ represent the computing resources allocated by the computational entity j for executing the k-type task workload. The expression of $y_{j,m}^k(t)$ is shown as follows:

$$y_{j,m}^k(t) = \begin{cases} f_{0,m}^k(t), j = 0, \\ f_{j,m}^k(t), j = 1,2,\dots,I_m. \end{cases} \tag{22}$$

Let $x_m^k(t)$ denote the total workload of the k-type task assigned to edge subsystem m, and $y_m^k(t)$ represent the total computing resources allocated to the k-type task by the edge subsystem m.

$$x_m^k(t) = \sum_{n=1}^{N}\sum_{i=1}^{I_n}\gamma_{i,n}^m S_{i,n}^{k,E}(t)c_k + \sum_{j=1}^{I_m} S_{j,m}^{k,A}(t)c_k, \tag{23}$$

$$y_m^k(t) = \sum_{j=0}^{I_m} f_{j,m}^k(t). \tag{24}$$

In addition, $x_k(t)$ denotes the total workload of the k-type task in the system, and $y_k(t)$ denotes the total computing resources allocated to the k-type task in the system.

$$x_k(t) = \sum_{m=1}^{N} x_m^k(t) = \sum_{n=1}^{N}\sum_{i=1}^{I_n} S_{i,n}^k(t)c_k. \tag{25}$$

$$y_k(t) = \sum_{n=1}^{N}\sum_{i=0}^{I_n} f_{i,n}^k(t). \tag{26}$$

Then the allocation difference within the edge subsystem is determined by

$$F_{Tw}^k(t) = \sum_{m=1}^{N}\frac{x_m^k(t)}{x_k(t)}\left[\sum_{j=0}^{I_m}\frac{x_{j,m}^k(t)}{x_m^k(t)}\ln\left[\frac{x_{j,m}^k(t)/x_m^k(t)}{y_{j,m}^k(t)/y_m^k(t)}\right]\right], \tag{27}$$

and the allocation difference between edge subsystems is determined by

$$F_{Tb}^k(t) = \sum_{m=1}^{N}\frac{x_m^k(t)}{x_k(t)}\ln\left[\frac{x_m^k(t)/x_k(t)}{y_m^k(t)/y_k}\right]. \tag{28}$$

Thus, the allocation difference of time slot t with respect to the k-type task is expressed as follows:

$$\begin{aligned} F_{Th}^k(t) = &\sum_{m=1}^{N}\frac{x_m^k(t)}{x_k(t)}\ln\left[\frac{x_m^k(t)/x_k(t)}{y_m^k(t)/y_k}\right] \\ &+\sum_{m=1}^{N}\frac{x_m^k(t)}{x_k(t)}\left[\sum_{j=0}^{I_m}\frac{x_{j,m}^k(t)}{x_m^k(t)}\ln\left[\frac{x_{j,m}^k(t)/x_m^k(t)}{y_{j,m}^k(t)/y_m^k(t)}\right]\right]. \end{aligned} \tag{29}$$

According to the analysis above, we define the allocation difference of the system as (30), which measures the fairness of task offloading and load balancing at the edge in PIoT. As such, the system is considered fairer when $F(t)$ is smaller.

$$F(t) = \frac{1}{K}\sum_{k=1}^{K} F_{Th}^k(t). \tag{30}$$

Additionally, we represent the time average allocation difference as (31) to measure the long-term fairness of the system.

$$\bar{F}(t) = \lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T} F(t). \tag{31}$$

3. Problem formulation

We define $\varphi_{i,n}^k(t) = \{\alpha_{i,n}^k(t), f_{i,n}^k(t), \gamma_{i,n}^m\}$ as the decision variables. The time-averaged optimization problem $\mathcal{P}1$ is formulated as (32), which aims to achieve a long-term balanced allocation of system tasks and computing resources under multi-task delay constraints. We use $\mathcal{P}1$ to

denote the FDG optimization problem.

$$
\begin{aligned}
\mathcal{P}1: \min_{\varphi_{i,n}^k(t)} \quad & \bar{F}(t) = \lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} F(t) \\
s.t. \quad & C1: \limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbf{E}\left\{\left|Q_{i,n}^k(t)\right|\right\} < \infty \\
& \quad \forall n \in \mathcal{N}, i = 0, 1, \ldots, I_n, k \in \mathcal{K} \\
& C2: \lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbf{E}\left\{d_{i,n}^k(t)\right\} \le \beta_k d_k, \\
& \quad \forall n \in \mathcal{N}, i \in \mathcal{I}_n, k \in \mathcal{K} \\
& C3: \gamma_{i,n}^m(t) \in \{0,1\}, \forall m, n \in \mathcal{N}, i \in \mathcal{I}_n \\
& C4: \sum_{m\in\mathcal{N}} \gamma_{i,n}^m = 1, \forall m, n \in \mathcal{N}, i \in \mathcal{I}_n \\
& C5: \sum_{k=1}^{K} f_{i,n}^k \le f_{i,n}^{\max}, \forall n \in \mathcal{N}, i \in \mathcal{I}_n, k \in \mathcal{K} \\
& C6: \sum_{k=1}^{K} f_{0,n}^k \le f_{0,n}^{\max}, \forall n \in \mathcal{N}, k \in \mathcal{K}
\end{aligned}
\tag{32}
$$

In $\mathcal{P}1$, $C1$ denotes the constraint for queue stability, $C2$ refers to the limit on task delays, and $C3$ and $C4$ pertain to the restrictions on decision variables for achieving balance. Additionally, $C5$ and $C6$ relate to the constraints on computation capacity. Importantly, the problem, $\mathcal{P}1$, involves stochastic programming. The decisions regarding offloading and balance are interdependent and affect the system's fairness. Additionally, given the tight coupling of variables, existing static methods experience difficulties in solving this problem.

4. Algorithm design

In this paper, we leverage Lyapunov optimization to transform $\mathcal{P}1$ into a queue stability problem. Lyapunov optimization [32], an optimization method for dynamic systems, combines Lyapunov control theory with stochastic network optimization. This approach balances stability and performance when developing a control policy.

Let $Q(t)=[Q_{0,1}^1(t), Q_{1,1}^1(t), \ldots, Q_{i,n}^k(t), \ldots, Q_{I_N,N}^K(t)]$ and $H(t)=[H_{1,1}^1(t), \ldots, H_{i,n}^k(t), \ldots, H_{I_N,N}^K(t)]$ respectively denote the actual queue vector and the virtual queue vector of the system in time slot t. The expressions for $Q_{i,n}^k(t)$ and $Q_{0,n}^k(t)$ are given in (3) and (8). The evolution of $H_{i,n}^k(t)$ is described as follows:

$$
H_{i,n}^k(t+1) = \max\left\{H_{i,n}^k(t) + d_{i,n}^k(t) - \beta_k d_k, 0\right\} \tag{33}
$$

Suppose that $H_{i,n}^k(t)$ is mean rate stable, i.e.,

$$
\lim_{t\to\infty} \frac{\mathbf{E}\left\{\left|H_{i,n}^k(t)\right|\right\}}{t} = 0, \tag{34}
$$

and $H_{i,n}^k(t)$ meets $\mathbf{E}\{|H_{i,n}^k(t)|\} < \infty$. Then according to the necessary condition for rate stability [32], the constraint $C2$ in (32) holds. That is, we can use the average rate stability of the virtual queue to replace $C2$.

We define $\Theta(t) \triangleq [Q(t), H(t)]$ as the queue of the system in time slot t. $L(\Theta(t))$ denotes the Lyapunov function of $\Theta(t)$, which is used to measure the queue backlog of the system. The system's queue stability can be guaranteed if the Lyapunov function is maintained at a low level.

$$
L(\Theta(t)) \triangleq \frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=0}^{I_n}\left(Q_{i,n}^k(t)\right)^2 + \frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}\left(H_{i,n}^k(t)\right)^2. \tag{35}
$$

The fluctuation of queue backlogs can be measured using Lyapunov drift at different times. The one-step conditional Lyapunov drift function of $\Theta(t)$ is defined as follows.

$$
\Delta(\Theta(t)) \triangleq \mathbf{E}\{L(\Theta(t+1)) - L(\Theta(t)) | \Theta(t)\}. \tag{36}
$$

Theorem 1. *Supposing there are constants $B \ge 0$, the Lyapunov drift function $\Delta(\Theta(t))$ has the following upper bound for all t, and all possible values of $\Theta(t)$:*

$$
\begin{aligned}
&\Delta(\Theta(t)) \\
&\le B + \mathbf{E}\left\{\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n} Q_{i,n}^k(t)(S_{i,n}^{k,A}(t)c_k - f_{i,n}^k(t)\tau) | \Theta(t)\right\} \\
&+ \mathbf{E}\left\{\sum_{k=1}^{K}\sum_{n=1}^{N} Q_{0,n}^k(t)\left[\sum_{m\in\mathcal{N}}\sum_{j\in\mathcal{I}_m}\gamma_{j,m}^n S_{j,m}^{k,E}(t)c_k - f_{0,n}^k(t)\tau\right] | \Theta(t)\right\} \\
&+ \mathbf{E}\left\{\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n} H_{i,n}^k(t)(d_{i,n}^k(t) - \beta_k d_k) | \Theta(t)\right\}.
\end{aligned}
\tag{37}
$$

Proof. See Appendix.

Then, we define the Lyapunov *drift-plus-penalty* expression as follows:

$$
\Delta(\Theta(t)) + V\mathbf{E}\{F(t)|\Theta(t)\}, \tag{38}
$$

where $V \ge 0$ is a control parameter, which denotes the trade-off between queue stability and allocation difference.

According to Theorem 1, we can derive that the Lyapunov *drift-plus-penalty* expression satisfies the following:

$$
\begin{aligned}
&\Delta(\Theta(t)) + V\mathbf{E}\{F(t)|\Theta(t)\} \\
&\le B + V\mathbf{E}\{F(t)|\Theta(t)\} \\
&+ \sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n} Q_{i,n}^k(t)\,\mathbf{E}\left\{S_{i,n}^{k,A}(t)\,c_k - f_{i,n}^k(t)\tau | \Theta(t)\right\} \\
&+ \sum_{k=1}^{K}\sum_{n=1}^{N} Q_{0,n}^k(t)\mathbf{E}\left\{\sum_{m\in\mathcal{N}}\sum_{j\in\mathcal{I}_m}\gamma_{j,m}^n S_{j,m}^{k,E}(t)\,c_k - f_{0,n}^k(t)\tau | \Theta(t)\right\} \\
&+ \sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n} H_{i,n}^k(t)\mathbf{E}\left\{\left(d_{i,n}^k(t) - \beta_k d_k\right) | \Theta(t)\right\}.
\end{aligned}
\tag{39}
$$

The expression on the right side of (39) provides an upper bound for the Lyapunov *drift-plus-penalty* expression. This means that $\mathcal{P}1$ is transformed into a problem of minimizing the Lyapunov *drift-plus-penalty* expression, which is then turned into a problem of minimizing the upper bound of the Lyapunov *drift-plus-penalty* expression. For ease of formulation, we introduce the auxiliary functions $G(t)$:

$$
\begin{aligned}
G(t) = VF(t) &+ \sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n} Q_{i,n}^k(t)\left[S_{i,n}^{k,A}(t)c_k - f_{i,n}^k(t)\tau\right] \\
&+ \sum_{k=1}^{K}\sum_{n=1}^{N} Q_{0,n}^k(t)\left[\sum_{m\in\mathcal{N}}\sum_{j\in\mathcal{I}_m}\gamma_{j,m}^n S_{j,m}^{k,E}(t)c_k - f_{0,n}^k(t)\tau\right] \\
&+ \sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n} H_{i,n}^k(t)\left[d_{i,n}^k(t) - \beta_k d_k\right].
\end{aligned}
\tag{40}
$$

Therefore, $\mathcal{P}1$ is rewritten as $\mathcal{P}2$:

$$
\begin{aligned}
\mathcal{P}2: \quad & \min \quad G(t) \\
& s.t. \quad C3, C4, C5, C6
\end{aligned}
\tag{41}
$$

Theorem 2. *$\mathcal{P}2$ is NP-hard for all time slots t.*

Proof. We prove the theorem by reducing the multiple knapsack problem (MKP), which is a typical NP-complete problem, to $\mathcal{P}2$. One common example of the MKP can be stated as follows. Assign n' items $\{x_1^{MKP}, \ldots, x_{n'}^{MKP}\}$ to m' differentiated backpacks $\{y_1^{MKP}, \ldots, y_{n'}^{MKP}\}$, where each item $x_{i'}^{MKP}$ has an associated profit $v_{i'} > 0$ and weight $w_{i'} > 0$, and the capacity of the backpack $y_{i'}^{MKP}$ is $W_{j'}$. MKP refers to the selection of m' disjoint subsets of items that maximize the total profit of the selected items without exceeding the capacity of each backpack [23]. In each time slot t,

(1) The set of n' items in MKP maps to the task workload $S_{i,n}^{k}(t)$ in $\mathcal{P}2$, including the $S_{i,n}^{k,A}(t)$ and $S_{i,n}^{k,E}(t)$.
(2) The weight $w_{i'} > 0$ in MKP corresponds to the computing resources needed for $S_{i,n}^{k}(t)$ in $\mathcal{P}2$, that is, $S_{i,n}^{k}(t)\alpha_{i,n}^{k}(t)c_k$ and $S_{i,n}^{k}(t)(1-\alpha_{i,n}^{k}(t))c_k$.
(3) The associated profit $v_{i'} > 0$ in MKP maps to the components of the auxiliary functions $G_{i,n}^{k}(t)$ in $\mathcal{P}2$, which satisfies $G(t) = \sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=0}^{I_n} G_{i,n}^{k}(t)$. Due to the competition of resources, $G_{i,n}^{k}(t)$ is variable.
(4) The set of m' differentiated backpacks $\{y_1^{MKP}, \ldots, y_{n'}^{MKP}\}$ in MKP is mapped to the distinct edge subsystems in $\mathcal{P}2$. The capacities $W_{j'}$ of differentiated backpacks are mapped to computing resources for distinct edge subsystems.
(5) The aim of $\mathcal{P}2$ is to minimize $G(t)$, which can be transformed into maximizing $-G(t)$. Then the total profit maximization in MKP corresponds to maximizing $-G(t)$ in $\mathcal{P}2$.

Following the mapping above, $\mathcal{P}2$ is NP-hard for all time slots t.

Population-based optimization algorithms are widely used to solve NP-hard problems, such as the whale optimization algorithm (WOA) [33], grey wolf optimization (GWO), dingo optimization algorithm (DOA), and archimedes optimization algorithm (AOA). WOA is a heuristic optimization algorithm proposed to imitate the hunting behavior of humpback whales. Compared with other intelligent algorithms, WOA has the advantages of fewer parameters, fast convergence, and more effortless execution, so it has been used to solve many engineering problems. Therefore, based on the Lyapunov optimization, we utilize the WOA algorithm to solve the FDG problem in this paper. The representations of the target prey and search agents are shown as follows.

(1) *Search agent*: A search agent in a multi-dimensional search space is denoted as $\varphi_{i,n}^{k}(t) = \{\alpha_{i,n}^{k}(t), f_{i,n}^{k}(t), \gamma_{i,n}^{m}\}$. The position of each humpback whale (i.e., search agent) represents a feasible solution.
(2) *Target prey*: Suppose that the best candidate solution in the search space is either the target prey or close to the optimal solution, as the exact position of the target prey is unknown in advance.
(3) *Encircling prey*: The remaining search agents adjust their positions to move closer to the target prey.
(4) *Bubble-net attacking*: This phase is divided into shrinking encircling and spiral position updating. Humpback whales swim in a spiral path while encircling their target within a shrinking circle during optimization. We assign a 50% probability to choosing shrinking encircling or spiral position updating. In this phase, search agents update their positions with respect to the best solution obtained so far.
(5) *Search for prey*: In this process, the search agent updates its position according to the randomly selected search agent to perform a global search.

We refer to the combination of Lyapunov optimization and the WOA joint algorithm as the LWOA algorithm, and its pseudo-code is shown in Algorithm 1.

Algorithm 1: LWOA

Input: System parameters: the location parameters, the transmission parameters, the task model parameters, and the queue model parameters
Output: FDG Scheme (i.e., the best solution of the FDG problem)

```
Initialize the population parameters and Lyapunov parameter V.
for each time slot do
    for iter < maximum number of iterations do
        for each search agent do
            Calculate the fitness (i.e., G(t));
            Update WOA exploitation and exploration
             parameters (i.e., a, A, C), and probability (i.e., p);
            if p < 0.5 then
                if |A| < 1 then
                    Encircling prey: location updating;
                else
                    Bubble-net attacking: location updating;
                end
            else
                Search for prey: location updating;
            end
        end
        Check if any search agent goes beyond the search space
         and amend it;
        Update the best search agent if there is a better solution;
    end
    Return the best search agent (i.e., the best solution of the
     FDG problem in this time slot);
    Update the queue model;
end
```

5. Simulation

5.1. Simulation setup

We select an area of approximately 1000×1500 m^2 in a city of Jilin province as the experimental area, which contains seven power communication stations, each serving ten sub-communication sites. The ESs are deployed at communication stations and are neighbor nodes to each other. The APs are set up at the sub-communication sites. The simulation scenario is shown in Fig. 3. The computing capability of APs is randomly selected within $\{1.2, 2.4, 3.6\}$ GHz, and the computing capability of ESs is randomly selected within $\{10, 20, 30, 50\}$ GHz. The channel bandwidth for wireless communication between AP and ES is 180 kHz, and the number of channels occupied by each AP is randomly chosen within the range [3, 15]. The transmission power of APs is 0.2 W. The noise power spectral density is 3.98×10^{-21}. The path loss model is $140.7 + 36.7log_{10}(dis)$ dB, where dis is the distance between AP and ES. Fiber optic communication is used among ESs. The wired link capacity is in the range [500, 1000] Mbps. Moreover, we consider power intelligent inspection and monitoring data processing (i.e., *Task-1* and *Task-2*) to establish the task model. The arrival of tasks follows a Poisson process, and the reach rate is randomly taken at [5, 10] tasks/slot. Let the data sizes of *Task-1* and *Task-2* be 0.2 Mb/task and 0.5 Mb/task, respectively. The average workloads are 2000 cycles/bit and 1500 cycles/bit, and the maximum allowable delays are 0.5 s and 1 s, respectively. The control parameter is $V = 50$.

We compare our proposed FDG scheme with the following schemes:

(1) NonB (none of the balance): The NonB scheme does not consider load balancing between edge subsystems, which means that the APs or ES perform the computing tasks within their respective edge subsystems.
(2) ExhS (exhaustive search): This scheme adopts the exhaustive search method to find the optimal execution positions for each AP's *ES-subset* based on pre-defined task partitioning ratios. Furthermore, this approach is a brute force method, and the number of feasible solutions grows exponentially with the number of nodes. Therefore, we only conduct experiments with a few node networks.

We compare our proposed LWOA algorithm with the following baseline algorithms:

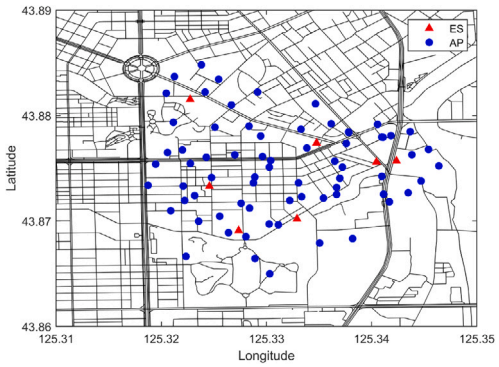
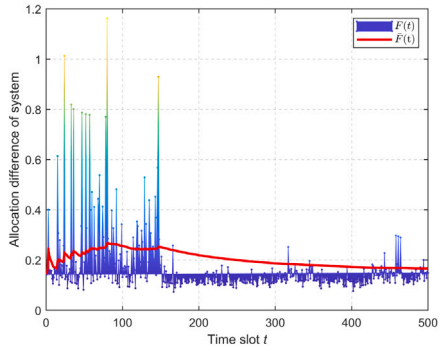
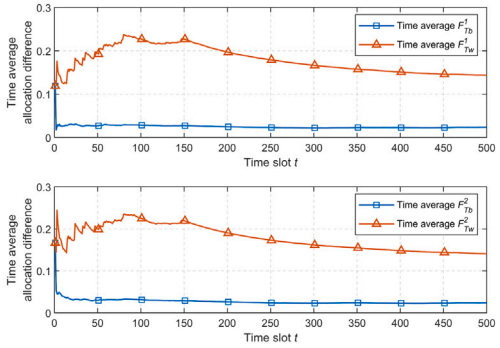


Fig. 3. The simulation scenario.



(a) The allocation difference of system



(b) The decomposition of time average allocation difference

Fig. 4. The allocation difference versus time slot t.

(1) GWO: The algorithm treats feasible decisions as grey wolves within a wolf group. By simulating the predation behavior of the group, the algorithm determines the best offloading and balancing scheme for each AP's task in each time slot.
(2) AOA: The algorithm views potential decisions as objects that are either partially or completely submerged in a fluid. These objects move and collide to achieve equilibrium. The best position of the objects is identified as the optimal solution, i.e., the best offloading and balancing plan for each task.
(3) DOA: The DOA treats possible decisions as Australian dingo dogs within a population and simulates the persecution, group attack, and scavenger behavior of the dingo dog to search for the optimal individual, which represents the optimal offloading and balancing scheme for each AP's task.

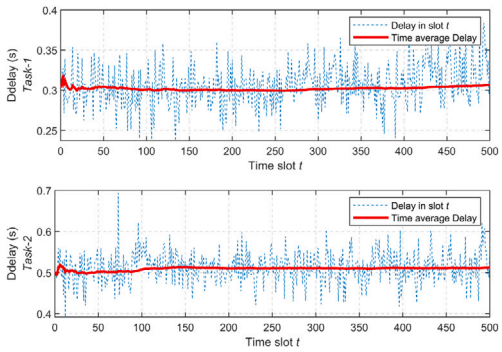


Fig. 5. The delay of different tasks versus time slot t.

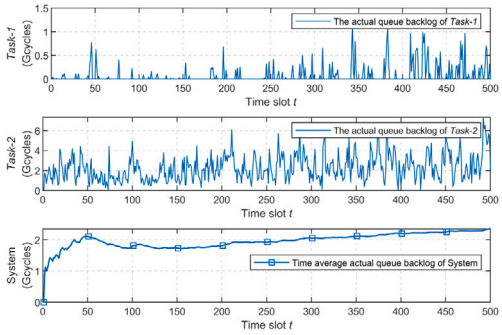


Fig. 6. The actual queue backlog versus time slot t.

5.2. Performance analysis

(1) Fairness: Fig. 4(a) shows the variation of the system's allocation difference (i.e., $F(t)$) and time average allocation difference (i.e., $\bar{F}(t)$) for each time slot. Overall, $F(t)$ tends to be stable with minor fluctuations from the previous period. When $t \leq 148$, $\bar{F}(t)$ experiences significant swings, indicating that the system is in the regulation phase. When $148 < t < 500$, $F(t)$ is between 0.07 and 0.29 for each time slot, and $\bar{F}(t)$ shows a steady and gradual decrease. After that, $\bar{F}(t)$ stabilizes around 0.17, a value considerably lower than 1. According to the analysis theory of the economic development imbalance measure, the Theil index usually belongs in the range [0, 1]. Similarly, $F(t)$ and $\bar{F}(t)$ should generally also belong to this range. Furthermore, smaller $F(t)$ and $\bar{F}(t)$ indicate that the allocation difference is smaller and the fairness is better. Once the information is analyzed, it can be determined that implementing the FDG scheme would improve the system's fairness performance. Correspondingly, when $148 < t < 500$, $F(t)$ of specific time slots fluctuates occasionally, but there is no continuous deviation, which indicates that FDG has regulation ability and stability. In Fig. 4(b), the allocation difference between groups and within groups eventually stabilizes for all tasks. Due to the sizable inherent gap of the computational resources between APs and ESs within the edge subsystem, the time average allocation difference within groups is more significant than the time average allocation difference between groups. Improving the resource configuration of APs can effectively reduce the time average allocation difference within groups, leading to a decrease in the allocation difference of the system.
(2) Delay: Fig. 5 shows the delay trends of *Task-1* and *Task-2* in different time slots. The average time delay of *Task-1* is approximately 0.3 s, and that of *Task-2* is approximately 0.5 s. Both these values adhere to the delay constraint standards. Note that

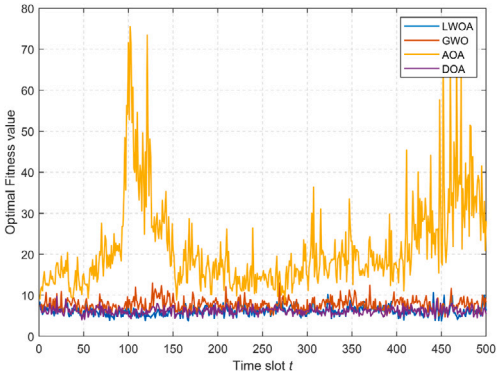


Fig. 7. The comparison of optimization performance among different algorithms (comparison of the optimal fitness values of each algorithm for each time slot).

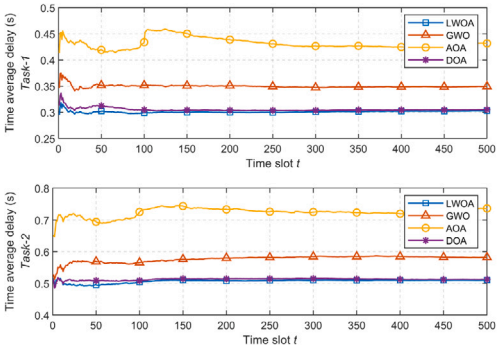


Fig. 8. The comparison of time average delay performance among different algorithms.

Table 3
The runtimes of different algorithms.

Algorithm	Runtime (s)
LWOA	0.69
GWO	0.86
AOA	0.81
DOA	0.71

the maximum allowable delays for *Task-1* and *Task-2* are 0.5 s and 1 s, respectively. Furthermore, according to the delay fluctuation curves of each type of task, it can be seen that the delays of *Task-1* and *Task-2* in each time slot satisfy the maximum allowable delay requirement. These numerical results indicate that the FDG scheme can effectively fulfill the delay constraints of various tasks concerning time slot granularity while ensuring a steady long-term delay of the tasks.

(3) Queue backlog: Fig. 6 illustrates the queue backlog of the system and different task types. The queue backlog for each time slot of *Task-1* and *Task-2* fluctuates within a specific range, and the total system queue backlog tends to increase initially and then stabilize. This pattern aligns with the queue stability described in Lyapunov optimization theory.

5.3. Comparison of algorithms

Fig. 7 through Fig. 9 show a comparison of LWOA with GWO, AOA, and DOA under the same experimental conditions. First, the optimization performance of each algorithm is shown in Fig. 7. We compare the optimal fitness values of the four algorithms in each time slot, which reflects the superiority of the optimal solution obtained by the algorithms. AOA's optimal fitness value fluctuates significantly

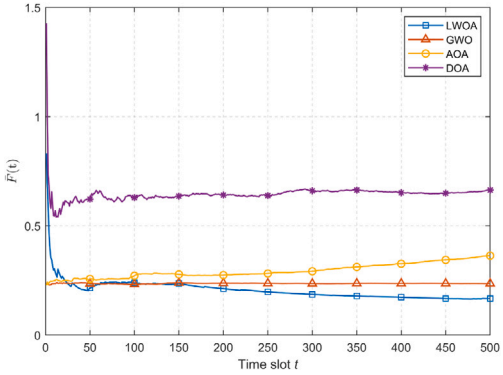


Fig. 9. The comparison of time average allocation difference performance among different algorithms.

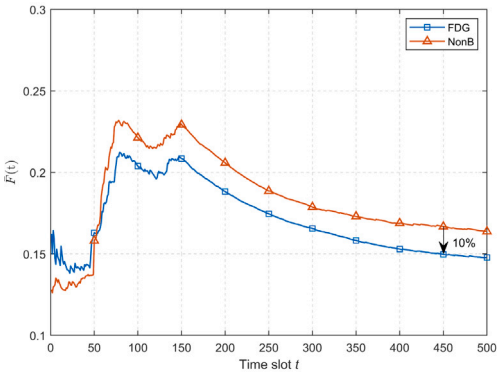
with time compared to the other three algorithms, indicating that the solution of AOA is less stable. The remaining three algorithms have similar performance. Fig. 8 compares the delay performances of the four algorithms for *Task-1* and *Task-2*. The results show that LWOA and DOA have similar delay performance, which is better than the delay performance of GWO and AOA. Fig. 9 compares the system fairness performance captured by $\bar{F}(t)$ for the four algorithms. After $t = 200$, $\bar{F}(t)$ of the AOA increases slightly. The lack of stability of the AOA is confirmed by the later period of the curve. $\bar{F}(t)$ of LWOA and GWO is able to maintain a stable, low level, which is significantly better than that of DOA and AOA. Furthermore, we conduct 1000 runs of the four algorithms on a laptop with an Intel Core i5-8250U 1.8 GHz processor and 16 GB of RAM. Table 3 compares the average running times of the four algorithms, demonstrating that LWOA has the shortest running time. In summary, LWOA achieves the best overall performance.

5.4. Comparison of schemes

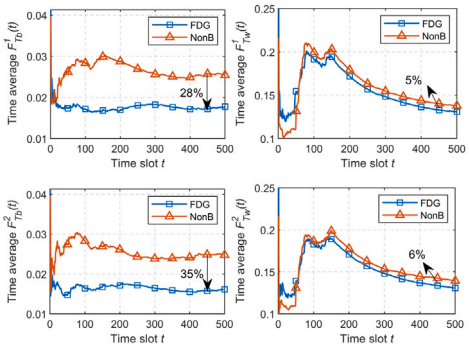
We compare the proposed FDG scheme with the NonB (none of the balance) scheme and the ExhS (exhaustive search) scheme.

(1) The NonB scheme involves each edge subsystem handling its workload independently. From Fig. 10(a), the time average allocation difference $\bar{F}(t)$ of the FDG scheme decreases by approximately 10% compared to the NonB scheme. Fig. 10(b) shows that with balance enabled, the allocation difference between edge subsystems (i.e., the time averaged $F^1_{Tb}(t)$ and $F^2_{Tb}(t)$) drops by 28% and 35% for *Task-1* and *Task-2*, respectively, while the allocation difference between edge subsystems (i.e., the time averaged $F^1_{Tw}(t)$ and $F^2_{Tw}(t)$) only drops by 5% and 6%. This is because $F^1_{Tb}(t)$ and $F^2_{Tb}(t)$ measure the allocation difference between groups, whereas $F^1_{Tw}(t)$ and $F^2_{Tw}(t)$ denote the allocation difference within an edge subsystem. Load balancing impacts the allocation difference between groups more, which aligns with the theory of Theil index decomposition. In Fig. 11, regarding the delay performance, the FDG scheme can reduce the time-averaged delay by 5% for *Task-1* and approximately 7% for *Task-2*. For Fig. 12, the FDG scheme with load balancing enabled can maintain the systems time-averaged queue backlog at a lower level, with a reduction of approximately 30% for *Task-1* and approximately 40% for *Task-2* compared to the NonB scheme.

(2) We compare the search results obtained by ExhS and the proposed FDG scheme to characterize the approximate optimal performance of the FDG scheme. The exhaustive method searches all possible solutions, which is time-consuming. Because the solution space grows exponentially with APs, ESs, and task types, we have reduced the network scale during the simulation. First,



(a) The time average allocation difference



(b) The decomposition of time average allocation difference

Fig. 10. The comparison of allocation difference performance between FDG and NonB schemes.

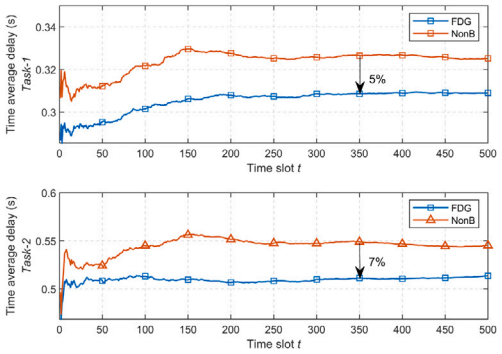


Fig. 11. The comparison of time average delay performance between FDG and NonB schemes.

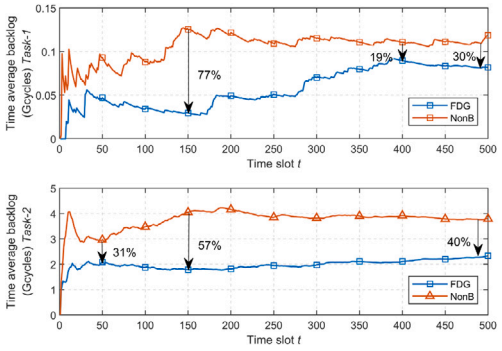


Fig. 12. The comparison of time average backlog performance between FDG and NonB schemes.

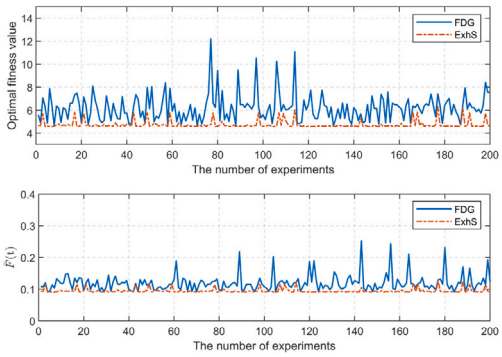


Fig. 13. The comparison of FDG and ExhS schemes.

there are three ESs in the system, two APs with each ES, and only one kind of task in the system. Second, the partition ratio ranges from 0 to 1 with an interval of 0.05. It should be noted that the assignment ratio is preset due to the impossibility of exhaustively enumerating the entire real number range. Finally, we use the enumeration method to search the optimal computation scheme of tasks under the preset proportions. We conduct 100 consecutive comparison experiments between the FDG and ExhS schemes. The comparison of optimal fitness values and fairness indicators $\bar{F}(t)$ of the FDG scheme and the ExhS scheme is shown in Fig. 13. The average relative error of optimal fitness for 100 experiments is approximately 29%. Meanwhile, the average relative error of $\bar{F}(t)$ for 100 repeated experiments is approximately 22%. Regarding running time, FDG takes approximately 0.6 s per solution under the above experimental parameters. In contrast, ExhS takes 77.5 s, which is 129 times longer than the FDG scheme.

6. Conclusion

This paper focuses on cooperative computing and task scheduling optimization in PIoT. First, we propose a two-layer edge network paradigm for PIoT. Second, we design the FDG to achieve a fairness-aware task offloading and load balancing scheme. This scheme considers the edge subsystem's fairness, delay constraints, resource heterogeneity, and dynamic characteristics. Third, we conduct simulation experiments to analyze the trends of the indicators and the system's performance. Extensive simulation results show that optimizing collaborative computing and task scheduling among edge subsystems can significantly improve the performance of edge computing in PIoT.

For the simulation model presented in this paper, we focus on a limited experimental area consisting of seven electric power communication sites to explore two common types of PIoT. In the future, it will be necessary to expand the analysis to more complex and practical PIoT scenarios.

Funding

This work was supported in part by Jilin Provincial Science and Technology Development under grant number 20210203044SF, and in part by Jilin Provincial Development and Reform Commission (the Capital Construction Fund of Jilin Provincial Budget in 2022) under grant number 2022C045-8.

CRediT authorship contribution statement

Xue Li: Conceptualization, Methodology, Software, Investigation, Writing – original draft. **Xiaojuan Chen:** Conceptualization, Supervision, Writing – review & editing. **Guohua Li:** Resources.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Xiaojuan Chen reports financial support was provided by Jilin Provincial Department of Science and Technology. Xiaojuan Chen reports financial support was provided by Jilin Provincial Development and Reform Commission.

Data availability

Data will be made available on request.

Appendix

The proof of Theorem 1 proceeds as follows:

First, according to (35), we have

$$\begin{aligned}
&L(\Theta(t+1))-L(\Theta(t))\\
&=\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=0}^{I_n}\left(Q_{i,n}^{k}(t+1)\right)^2-\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=0}^{I_n}\left(Q_{i,n}^{k}(t)\right)^2\\
&\quad+\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}\left(H_{i,n}^{k}(t+1)\right)^2-\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}\left(H_{i,n}^{k}(t)\right)^2\\
&=\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}\left[\left(Q_{i,n}^{k}(t+1)\right)^2-\left(Q_{i,n}^{k}(t)\right)^2\right]\\
&\quad+\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\left[\left(Q_{0,n}^{k}(t+1)\right)^2-\left(Q_{0,n}^{k}(t)\right)^2\right]\\
&\quad+\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}\left[\left(H_{i,n}^{k}(t+1)\right)^2-\left(H_{i,n}^{k}(t)\right)^2\right]
\end{aligned}\tag{42}$$

For the first component of (42), we substitute (35) into $Q_{i,n}^{k}(t+1)^2$. Also, $\max\{q-b,0\}^2\le(q-b)^2$, then

$$\begin{aligned}
\left(Q_{i,n}^{k}(t+1)\right)^2&=\left[\max\left\{Q_{i,n}^{k}(t)+S_{i,n}^{k,A}(t)c_k-f_{i,n}^{k}(t)\tau,0\right\}\right]^2\\
&\le\left[Q_{i,n}^{k}(t)-\left(f_{i,n}^{k}(t)\tau-S_{i,n}^{k,A}(t)c_k\right)\right]^2\\
&=\left(Q_{i,n}^{k}(t)\right)^2-2Q_{i,n}^{k}(t)\left(f_{i,n}^{k}(t)\tau-S_{i,n}^{k,A}(t)c_k\right)\\
&\quad+\left(f_{i,n}^{k}(t)\tau-S_{i,n}^{k,A}(t)c_k\right)^2
\end{aligned}\tag{43}$$

$$\begin{aligned}
\left(Q_{i,n}^{k}(t+1)\right)^2-\left(Q_{i,n}^{k}(t)\right)^2&\le-2Q_{i,n}^{k}(t)\left(f_{i,n}^{k}(t)\tau-S_{i,n}^{k,A}(t)c_k\right)\\
&\quad+\left(f_{i,n}^{k}(t)\tau-S_{i,n}^{k,A}(t)c_k\right)^2
\end{aligned}\tag{44}$$

Suppose that the system satisfies the boundedness assumption, then

$$\mathbf{E}\{S_{i,n}^{k}(t)^2\}\le\delta^2,\forall n\in\mathcal{N},i\in\mathcal{I}_n,k\in\mathcal{K}\tag{45}$$

where δ^2 is a bounded constant. According to (1) and $\alpha_{i,n}^{k}(t)\in[0,1]$, we have

$$\mathbf{E}\left\{S_{i,n}^{k,A}(t)^2\right\}\le\mathbf{E}\left\{S_{i,n}^{k}(t)^2\right\}\le\delta^2,\tag{46}$$

$$\begin{aligned}
\left(f_{i,n}^{k}(t)\tau-S_{i,n}^{k,A}(t)c_k\right)^2&\le\left(f_{i,n}^{k}(t)\tau\right)^2+\left(S_{i,n}^{k,A}(t)c_k\right)^2\\
&\le\left(f_{i,n}^{\max}\tau\right)^2+\left(\delta c_k\right)^2.
\end{aligned}\tag{47}$$

Let

$$B_1=\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}\left[\left(f_{i,n}^{\max}\tau\right)^2+\left(\delta c_k\right)^2\right],\tag{48}$$

then the first component of (42) satisfies:

$$\begin{aligned}
&\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}\left[\left(Q_{i,n}^{k}(t+1)\right)^2-\left(Q_{i,n}^{k}(t)\right)^2\right]\\
&\le B_1+\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}Q_{i,n}^{k}(t)\left(S_{i,n}^{k,A}(t)c_k-f_{i,n}^{k}(t)\tau\right)
\end{aligned}\tag{49}$$

Similarly, we can prove that

$$\begin{aligned}
&\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\left[\left(Q_{0,n}^{k}(t+1)\right)^2-\left(Q_{0,n}^{k}(t)\right)^2\right]\\
&\le B_2+\sum_{k=1}^{K}\sum_{n=1}^{N}Q_{0,n}^{k}(t)\left[\sum_{m\in\mathcal{N}}\sum_{j\in\mathcal{I}_m}\gamma_{j,m}^{n}S_{j,m}^{k,E}(t)c_k-f_{0,n}^{k}(t)\tau\right]
\end{aligned}\tag{50}$$

$$\begin{aligned}
&\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}\left[\left(H_{i,n}^{k}(t+1)\right)^2-\left(H_{i,n}^{k}(t)\right)^2\right]\\
&\le B_3+\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}H_{i,n}^{k}(t)\left(d_{i,n}^{k}(t)-\beta_k d_k\right),
\end{aligned}\tag{51}$$

where B_2 and B_3 are the bounded constants. Thus,

$$\begin{aligned}
&\Delta(\Theta(t))=\mathbf{E}\{L(\Theta(t+1))-L(\Theta(t))\,|\Theta(t)\}\\
&\le B+\mathbf{E}\left\{\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}Q_{i,n}^{k}(t)\left(S_{i,n}^{k,A}(t)c_k-f_{i,n}^{k}(t)\tau\right)|\Theta(t)\right\}\\
&+\mathbf{E}\left\{\sum_{k=1}^{K}\sum_{n=1}^{N}Q_{0,n}^{k}(t)\left[\sum_{m\in\mathcal{N}}\sum_{j\in\mathcal{I}_m}\gamma_{j,m}^{n}S_{j,m}^{k,E}(t)c_k-f_{0,n}^{k}(t)\tau\right]|\Theta(t)\right\}\\
&+\mathbf{E}\left\{\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{i=1}^{I_n}H_{i,n}^{k}(t)\left(d_{i,n}^{k}(t)-\beta_k d_k\right)|\Theta(t)\right\}
\end{aligned}\tag{52}$$

where $B=B_1+B_2+B_3$. This concludes the proof.

References

[1] Z. Alavikia, M. Shabro, A comprehensive layered approach for implementing internet of things-enabled smart grid: A survey, Digit. Commun. Netw. 8 (3) (2022) 388–410.

[2] G. Bedi, G.K. Venayagamoorthy, R. Singh, R.R. Brooks, K.-C. Wang, Review of Internet of Things (IoT) in electric power and energy systems, IEEE Internet Things J. 5 (2) (2018) 847–870.

[3] Y. Liang, T. Li, Ubiquitous power Internet of Things-oriented low-latency edge task scheduling optimization strategy, Front. Energy Res. 10 (2022).

[4] S.M.A.A. Abir, A. Anwar, J. Choi, A.S.M. Kayes, IoT-enabled smart energy grid: Applications and challenges, IEEE Access 9 (2021) 50961–50981.

[5] M.A. Ferrag, M. Babaghayou, M.A. Yazici, Cyber security for fog-based smart grid SCADA systems: Solutions and challenges, J. Inf. Secur. Appl. 52 (2020) 102500.

[6] A. Lekidis, A.G. Anastasiadis, G.A. Vokas, Electricity infrastructure inspection using AI and edge platform-based UAVs, Technologies and Materials for Renewable Energy, Environment and Sustainability, Energy Reports 8 (2022) 1394–1411.

[7] S.D. Milić, v. Đurović, M.D. Stojanović, Data science and machine learning in the IIoT concepts of power plants, Int. J. Elec. Power 145 (2023) 108711.

[8] J. Zhou, B. Cen, Z. Cai, Y. Chen, Y. Sun, H. Xue, W.O. Tan, Workload modeling for microservice-based edge computing in power Internet of Things, IEEE Access 9 (2021) 76205–76212.

[9] X. Shen, L. Zhu, C. Xu, K. Sharif, R. Lu, A privacy-preserving data aggregation scheme for dynamic groups in fog computing, Inform. Sci. 514 (2020) 118–130.

[10] W. Hou, H. Wen, N. Zhang, J. Wu, W. Lei, R. Zhao, Incentive-driven task allocation for collaborative edge computing in industrial Internet of Things, IEEE Internet Things J. 9 (1) (2022) 706–718.

[11] S. Chen, H. Wen, J. Wu, W. Lei, W. Hou, W. Liu, A. Xu, Y. Jiang, Internet of Things based smart grids supported by intelligent edge computing, IEEE Access 7 (2019) 74089–74102.

[12] X. Li, D. Li, J. Wan, C. Liu, M. Imran, Adaptive transmission optimization in SDN-based industrial Internet of Things with edge computing, IEEE Internet Things J. 5 (3) (2018) 1351–1360.

[13] T.K. Rodrigues, K. Suto, H. Nishiyama, J. Liu, N. Kato, Machine learning meets computation and communication control in evolving edge and cloud: Challenges and future perspective, IEEE Commun. Surv. Tutorials 22 (1) (2020) 38–67.

[14] J. Peng, H. Qiu, J. Cai, W. Xu, J. Wang, D2D-Assisted multi-user cooperative partial offloading, transmission scheduling and computation allocating for MEC, IEEE Trans. Wirel. Commun. 20 (8) (2021) 4858–4873.

[15] M. Hua, H. Tian, X. Lyu, W. Ni, G. Nie, Online offloading scheduling for NOMA-Aided MEC under partial device knowledge, IEEE Internet Things J. 9 (3) (2022) 2227–2241.
[16] J. Meng, H. Tan, C. Xu, W. Cao, L. Liu, B. Li, Dedas: Online task dispatching and scheduling with bandwidth constraint in edge computing, in: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, 2019, pp. 2287–2295.
[17] X. Xu, Y. Li, T. Huang, Y. Xue, K. Peng, L. Qi, W. Dou, An energy-aware computation offloading method for smart edge computing in wireless metropolitan area networks, J. Netw. Comput. Appl. 133 (2019) 75–85.
[18] X. Xu, C. He, Z. Xu, L. Qi, S. Wan, M.Z.A. Bhuiyan, Joint optimization of offloading utility and privacy for edge computing enabled IoT, IEEE Internet Things J. 7 (4) (2020) 2622–2629.
[19] L. Chen, S. Zhou, J. Xu, Computation peer offloading for energy-constrained mobile edge computing in small-cell networks, IEEE ACM Trans. Netw. PP (4) (2018) 1–14.
[20] S. Wang, X. Zhang, Z. Yan, W. Wenbo, Cooperative edge computing with sleep control under nonuniform traffic in mobile edge networks, IEEE Internet Things J. 6 (3) (2019) 4295–4306.
[21] W. Zhang, G. Zhang, S. Mao, Joint parallel offloading and load balancing for cooperative-MEC systems with delay constraints, IEEE Trans. Veh. Technol. 71 (4) (2022) 4249–4263.
[22] Y. Wang, X. Tao, X. Zhang, P. Zhang, Y.T. Hou, Cooperative task offloading in three-tier mobile computing networks: An ADMM framework, IEEE Trans. Veh. Technol. 68 (3) (2019) 2763–2776.
[23] J. Zhou, X. Zhang, Fairness-aware task offloading and resource allocation in cooperative mobile-edge computing, IEEE Internet Things J. 9 (5) (2022) 3812–3824.
[24] R. Li, Z. Zhou, X. Chen, Q. Ling, Resource price-aware offloading for edge-cloud collaboration: A Two-timescale online control approach, IEEE Trans. Cloud Comput. 10 (1) (2022) 648–661.
[25] J. Gao, R. Chang, Z. Yang, Q. Huang, Y. Zhao, Y. Wu, A task offloading algorithm for cloud-edge collaborative system based on Lyapunov optimization, Cluster Comput. 26 (1) (2023) 337–348.
[26] Y. Keshtkarjahromi, Y. Xing, H. Seferoglu, Adaptive and heterogeneity-aware coded cooperative computation at the edge, IEEE Trans. Mob. Comput. 22 (3) (2023) 1301–1312.
[27] Q. Wang, F. Zhou, Fair resource allocation in an MEC-enabled ultra-dense IoT Network with NOMA, in: ICC 2019, 2019.
[28] X. Yan, S. Mohd, Trends and causes of regional income inequality in China, Sustainability 15 (9) (2023) 7673.
[29] J. Rój, Inequity in the access to ehealth and its decomposition case of Poland, Int. J. Environ. Res. Public. Health 19 (4) (2022).
[30] K. Fang, M. Mao, C. Tian, J. Chen, W. Wang, R. Tan, Exploring the impact of emissions trading schemes on income inequality between urban and rural areas, J. Environ. Manag. 329 (2023) 117067.
[31] G. Luo, T. Balezentis, S. Zeng, Per capita CO2 emission inequality of China's urban and rural residential energy consumption: A Kaya-Theil decomposition, J. Environ. Manag. 331 (2023) 117265, Publisher: Academic Press.
[32] M.J. Neely, Stochastic Network Optimization with Application to Communication and Queueing Systems, Synthesis Lectures on Learning, Networks, and Algorithms, Springer International Publishing, Cham, 2010.
[33] S. Mirjalili, A. Lewis, The whale optimization algorithm, Adv. Eng. Softw. 95 (2016) 51–67.

Xue Li is currently pursuing a Ph.D. degree in Information and Communication Engineering at Changchun University of Science and Technology, Changchun, China. Her primary research interests include power Internet of Things, multi-access edge computing, cooperative computing, and stochastic network optimization.

Xiaojuan Chen received a Ph.D. in communication and information systems from Jilin University, China 2006. She was a Visiting Researcher at the University of British Columbia, UK, Canada, in 2012. She is a Professor at the School of Electronic Information Engineering, Changchun University of Science and Technology, Changchun, China. Her primary research interests include power Internet of Things, multi-access edge computing, cooperative computing, and stochastic network optimization.

Guohua Li received a master's degree from Northeast Electric Power University. She works for Changchun Power Supply Company of State Grid Jilin Electric Power Co., LTD. Her primary research interests include new power system construction, power IoT, power marketing, and quality service.