PAPER

# Cloud-Edge-Device Collaborative High Concurrency Access Management for Massive IoT Devices in Distribution Grid

**Shuai LI**[†,††], **Xinhong YOU**[†,††a)], **Shidong ZHANG**[†,††], **Mu FANG**[†††], *and* **Pengping ZHANG**[†,††], *Nonmembers*

**SUMMARY** Emerging data-intensive services in distribution grid impose requirements of high-concurrency access for massive internet of things (IoT) devices. However, the lack of effective high-concurrency access management results in severe performance degradation. To address this challenge, we propose a cloud-edge-device collaborative high-concurrency access management algorithm based on multi-timescale joint optimization of channel pre-allocation and load balancing degree. We formulate an optimization problem to minimize the weighted sum of edge-cloud load balancing degree and queuing delay under the constraint of access success rate. The problem is decomposed into a large-timescale channel pre-allocation subproblem solved by the device-edge collaborative access priority scoring mechanism, and a small-timescale data access control subproblem solved by the discounted empirical matching mechanism (DEM) with the perception of high-concurrency number and queue backlog. Particularly, information uncertainty caused by externalities is tackled by exploiting discounted empirical performance which accurately captures the performance influence of historical time points on present preference value. Simulation results demonstrate the effectiveness of the proposed algorithm in reducing edge-cloud load balancing degree and queuing delay.

***key words:*** *cloud-edge-device collaboration, high-concurrency access management, multi-timescale optimization, channel pre-allocation, data access control, discounted empirical matching*

## 1. Introduction

The large-scale integration of renewable energy into distribution grid has spurred new types of services such as distributed renewable energy control, intelligent load demand response, and high-frequency electric information acquisition [1]–[5]. These services involve thousands of internet of things (IoT) devices for data collection, transmission, and computation, which impose high requirements on high-concurrency access of massive data [6]–[9]. Cloud-edge-device collaboration has provided a feasible solution by combining advantages of edge computing and cloud computing [10]–[14]. Edge computing sinks computing at the edge of the network, which can process and analyze data faster and improve network delay [15], [16]. Cloud computing has strong computing power and big data analysis capability. The data collected by devices are pre-processed and stored on edge gateway to address the device-cloud communication bottleneck. However, the lack of an effective high-concurrency access management mechanism results in severe performance degradation of queuing delay, computation load balancing, and access success rate when the limited capacity of communication and computing is overwhelmed by the simultaneous access of massive data [17]–[21]. Therefore, how to achieve cloud-edge-device collaborative high-concurrency access management for distribution grid remains an open issue.

The core of cloud-edge-device collaborative high-concurrency access management for massive IoT devices in distribution grid lies in deep-level coordination between device-edge channel pre-allocation and edge-cloud data access control [22], [23]. On the one hand, edge-device channel pre-allocation is important to ensure access success rate of high-priority device and reduce performance deviation of edge-side data queue backlog [24], [25]. On the other hand, edge-cloud data access control improves performance of edge-cloud load balancing and queuing delay through dynamic matching between device data queues with cloud computing resources. It is intuitive to jointly optimize device-edge channel pre-allocation and edge-cloud data access control to achieve high-concurrency access management for massive IoT devices in distribution grid. However, the following technical challenges need to be addressed.

First, load balancing improvement and queuing delay reduction are not necessarily consistent. The mere optimization of load balancing may cause data queue backlog to increase, and vice versa. Second, device-edge channel pre-allocation and edge-cloud data access control are not implemented in the same timescale. Device-edge channel pre-allocation should be optimized in large timescale to reduce communication overhead caused by frequent channel switching [26]–[28]. Conversely, edge-cloud data access control needs to adapt with time-varying data arrival and computing resource fluctuation, which requires small-timescale optimization. Moreover, the tight coupling between them significantly increases the complexing of high-concurrency access management. Last but not least, the data access control decision for one edge gateway has an unneglectable impact on the performance of queuing delay and load balancing of the other edge gateways, resulting in externalities for high-concurrency access management. This impact is uncertain and cannot be predicted in advance, thereby leading to the information uncertainty.

Some works have investigated issues of load balancing

Manuscript received August 8, 2023.
Manuscript revised September 4, 2023.
Manuscript publicized October 26, 2023.

[†]The authors are with Distribution Technology Center, State Grid Shandong Electric Power Research Institute, Jinan 250002, China.
[††]The authors are with Shandong Smart Grid Technology Innovation Center, Jinan 250002, China.
[†††]The author is with State Grid Shandong Electric Power Company, Jinan, 250001, China.
a) E-mail: yxh_1234567892023@163.com

DOI: 10.1587/transfun.2023EAP1094

and queuing delay in cloud-edge-device collaboration. In [29], Zhang et al. attempted to achieve load balancing by optimizing the maximum throughput of high-concurrency access based on device-to-device (D2D) communication resource allocation. In [30], Li et al. proposed a dynamic distributed queuing-based random access approach to reduce the queuing delay by flexibly adjusting the number of device queues. However, these works do not consider the inconsistency between load balancing and queuing delay. The unilateral optimization of one performance metric inevitably causes performance deterioration of the other one. Femto-cloud was applied in the edge to jointly optimize load balancing and queuing delay [31] by addressing the communication bottleneck between device and remote cloud. In [32], Bui et al. divided accessed devices into certain groups and studied grouping-based access control optimization to solve device access conflicts, reduce cloud load and queuing delay, and improve device access efficiency. These works have ignored the optimization of device-edge channel pre-allocation. They are not suitable to the high-concurrency access scenario in distribution grid where communication resources are limited to meet the access needs of massive devices. In [33], Tom et al. proposed an analysis and prediction model based on cloud-edge computing resource attributes. Communication resources are dynamically allocated based on the number of access devices to maximize access channel resource utilization. In [34], Xu et al. investigated channel resource pre-allocation based on multi-agent deep deterministic policy gradient to reduce queuing delay of safety-critical information. However, the multi-timescale joint optimization of device-edge channel pre-allocation and edge-cloud data access control has not been investigated.

Matching theory offers an efficient and flexible solution to combinatorial optimization problems. With its low computational complexity, it can quickly find deterministic solutions within a reasonable computing time. Additionally, matching theory considers the interdependencies and relationships among elements in the combinatorial optimization problem. By taking into account these connections, it can identify optimal matches that maximize overall efficiency or meet specific criteria. It has been widely applied in edge-cloud collaboration [35], data offloading [36], data access control [37], and channel allocation [38]. In [39], Rahim et al. designed a matching theory-based idle channel allocation scheme to satisfy the users' quality of service (QoS) requirement. In [40], Agoun et al. proposed an entity matching-oriented and policy-oriented method to achieve secure access control. However, these works rely on iterative comparison of preferences to derive a stable matching, and have not considered the issue of uncertain information caused by externalities. Since the optimization performance of one entity is affected by matching decision of other entities, the key information required for calculating matching preference, e.g., potential performance of load balancing and queuing delay after matching, becomes uncertain, and conventional preference-based matching approaches are infeasible. Although several studies addressed the problem of externalities by using the swap matching methodology [41], [42], the high computation complexity of swap matching makes it difficult for real-world implementation.

In this paper, we propose a high-concurrency access management algorithm based on multi-timescale joint optimization of channel pre-allocation and load balancing degree to address the above challenges. We formulate a cloud-edge-device collaborative high-concurrency access management model. The optimization objective is to jointly minimize edge-cloud load balancing degree and queuing delay under the constraints of pre-allocation channel numbers, device access success rate, data queue numbers, and backlogs of high-concurrency access. Then, we decompose the formulated problem into a large-timescale device-edge channel pre-allocation subproblem and a small-timescale edge-cloud data access control subproblem, which are solved alternatingly by the proposed algorithm. We summarize three contributions as follows.

- **Joint guarantee of edge-cloud load balancing degree and queuing delay:** We define the optimization objective as the weighted sum of edge-cloud load balancing degree and edge-cloud queuing delay. Particularly, the load balancing degree considers the deviations between the data queue backlog and the average data queue backlog in both the edge gateway and cloud server. The weighted sum is also utilized to construct the matching preference of matching options and cloud servers. This enforces higher priority for matching option with better load balancing degree and lower queuing delay.
- **Large-timescale device-edge channel pre-allocation based on device-edge collaborative access priority:** We construct a device-edge collaborative access priority scoring mechanism to solve the large-timescale device-edge channel pre-allocation subproblem. Through service classification and comparison, the edge-side empirical data queue backlog performance deviation and device-side access success rate deviation are calculated based on the historical average queue backlog, average queue input, average queue output, and average access success rate. Then, they are combined to derive the device-edge collaborative access priority score. Access channel pre-allocation optimization is realized based on the access priority score under the constraint of pre-allocation channel numbers.
- **Small-timescale edge-cloud data access control based on discounted empirical matching:** We propose a discounted empirical matching mechanism (DEM) to solve the small-timescale edge-cloud data access control subproblem. DEM exploits the historical weighted sum performance of edge-cloud load balancing degree and queuing delay to construct the two-sided matching preference, which effectively overcomes information uncertainty caused by externalities. We adopt a discount factor to describe the influence weight of historical performance at different points in time for the calculation of the present preference value, ensuring the accuracy of the present matching relationship.

The rest of this paper is organized as follows. The high-concurrency access management model based on cloud-edge-device collaboration is introduced in Sect. 2. Section 3 describes the high-concurrency access management algorithm based on multi-timescale joint optimization of channel pre-allocation and load balancing degree. Simulation results are given in Sect. 4. Section 5 provides the conclusion.

## 2. High-Concurrency Access Management Model Based on Cloud-Edge-Device Collaboration

### 2.1 High-Concurrency Access Management Architecture Based on Cloud-Edge-Device Collaboration

The cloud-edge-device collaborative high-concurrency access management architecture for distribution grid is shown in Fig. 1, including cloud layer, edge layer, and device layer.

The device layer contains $M$ IoT devices deployed on smart meters, distributed generators, and charging piles, which upload the collected data to the edge gateway through power line communication.

The edge layer contains $N$ edge gateways for storing and uploading data, the set of which is $\mathcal{G} = \{g_1, \cdots, g_n, \cdots, g_N\}$. The set of $M_n$ devices within the communication range of the edge gateway $g_n$ is defined as $\mathcal{D}_n = \{d_1^n, \cdots, d_m^n, \cdots, d_{M_n}^n\}, \forall \mathcal{D}_n \cap \mathcal{D}_{n'} = \emptyset, n \neq n'$, $\sum_{n=1}^{N} M_n = M$. $g_n$ pre-allocates access channels for $M_n$ devices within its communication range to upload collected data. The uploaded data of $M_n$ devices are stored in $M_n$ data queues on $g_n$. Then, edge gateway accesses to the cloud layer and uploads data to cloud server for processing via 5G communication networks.

The cloud layer contains a high-concurrency access platform and $K$ cloud servers. The cloud server processes the access data of edge gateway to support power services such as distributed renewable control, load demand response control, panoramic perception, and electric information acquisition. The set of cloud servers is $\mathcal{S} = \{s_1, \cdots, s_k, \cdots, s_K\}$. The high-concurrency access platform dynamically adjusts the edge-device channel pre-allocation strategy according to the edge-side empirical data queue backlog performance deviation and access success rate deviation. It also performs edge-cloud data access control to realize joint minimization of load balancing degree and queuing delay.

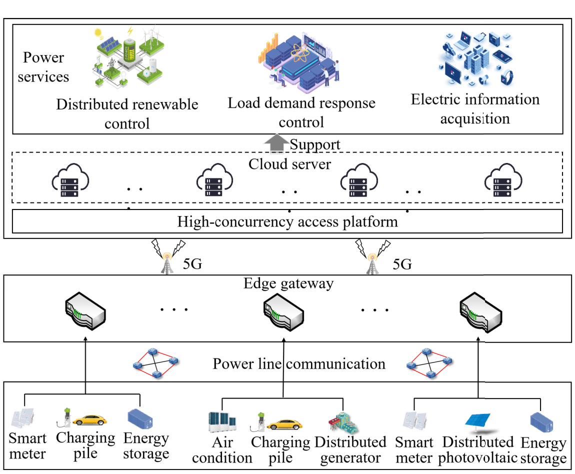


**Fig. 1** Cloud-edge-device collaborative high-concurrency access management architecture for IoT devices in distribution grid.

The total optimization duration is divided into $I$ periods. The set is $\mathcal{I} = \{1, \cdots, i, \cdots, I\}$. Each period contains $T_0$ slots with length $\tau$. The set is $\mathcal{T} = \{1, \cdots, t, \cdots, T\}$, i.e., $T = IT_0$. The device-edge channel pre-allocation is optimized in large timescale to reduce communication overhead caused by frequent switching of access channels, and the edge-cloud data access control is optimized in small timescale to adapt with rapid changes of channel states and fluctuations of computing resources. Define $x_m^n(i)$ as the variable of large-timescale channel pre-allocation in the $i$-th period. $x_m^n(i) = 1$ indicates that channel is pre-allocated to the device $d_m^n$ for data uploading, and otherwise $x_m^n(i) = 0$. Define $y_{m,k}^n(t)$ as the indicator variable of small-timescale edge-cloud data access in the $t$-th slot. $y_{m,k}^n(t) = 1$ indicates that the edge gateway $g_n$ uploads the data of $d_m^n$ to the cloud server $s_k$ for processing, and otherwise $y_{m,k}^n(t) = 0$.

### 2.2 Data Queue Backlog Model of Edge Gateway and Cloud Server

The evolution of data queue backlog of edge gateway and cloud server is shown in Fig. 2. Define $Q_m^n(t)$ as the data queue backlog of the device $d_m^n$ in the edge gateway $g_n$, which is

$$Q_m^n(t+1) = Q_m^n(t) - U_m^n(t) + x_m^n(i) A_m^n(t), \tag{1}$$

where $A_m^n(t)$ is the amount of data uploaded to $g_n$ by $d_m^n$ in the $t$-th slot, and $U_m^n(t)$ is the amount of data of $d_m^n$ which are uploaded to the cloud server by $g_n$. $A_m^n(t)$ depends on the device-edge data throughput and the maximum amount of data collected by $d_m^n$, which can be expressed as

$$A_m^n(t) = \min\{\tau R_m^{n,PLC}(t), A_m^{n,thr}(t)\}, \tag{2}$$

where $R_m^{n,PLC}(t)$ is the transmission rate from $d_m^n$ to $g_n$ based on power line communication. $A_m^{n,thr}(t)$ is the maximum amount of data collected by $d_m^n$. Similarly, $U_m^n(t)$ depends on the edge-cloud data throughput and the data queue backlog $Q_m^n(t)$ of $g_n$, which can be expressed as

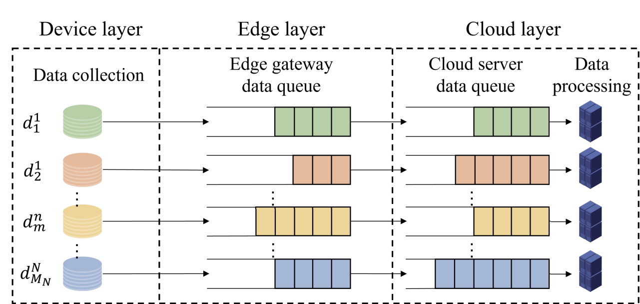


**Fig. 2** The evolution of data queue backlog of edge gateway and cloud server.

$$L_m^{n,Q}(t) = \frac{Q_m^n(t)}{\frac{1}{t-1}\left(\sum_{e=1}^{i-1} x_m^n(e) \sum_{z=(e-1)T_0+1}^{eT_0} A_m^n(z) + \sum_{z=(i-1)T_0+1}^{t-1} x_m^n(i) A_m^n(z)\right)}. \tag{8}$$

$$\begin{aligned} U_m^n(t) &= \sum_{s_k\in\mathcal{S}} y_{m,k}^n(t) u_{m,k}^n(t) \\ &= \sum_{s_k\in\mathcal{S}} y_{m,k}^n(t)\min\{\tau R_{m,k}^{n,5G}(t), Q_m^n(t)\}, \end{aligned} \tag{3}$$

where $R_{m,k}^{n,5G}(t)$ is the transmission rate from $g_n$ to the cloud server $s_k$ based on 5G, and $u_{m,k}^n(t)$ is the amount of data of $d_m^n$ which are uploaded to the cloud server $s_k$ by $g_n$. $y_{m,k}^n(t)$ represents the indicator variable of small-timescale edge-cloud data access in the $t$-th slot.

The queue backlog of data of $d_m^n$ in $s_k$ which are uploaded by $g_n$ is

$$W_{m,k}^n(t+1) = W_{m,k}^n(t) - E_{m,k}^n(t) + y_{m,k}^n(t) u_{m,k}^n(t), \tag{4}$$

where $E_{m,k}^n(t)$ is the amount of data of $d_m^n$ processed by $s_k$ in the $t$-th slot, i.e.,

$$E_{m,k}^n(t) = \min\{\frac{\tau f_{m,k}^n(t)}{\varpi_{m,k}^n(t)}, W_{m,k}^n(t)\}, \tag{5}$$

where $f_{m,k}^n(t)$ is the amount of computing resources of $s_k$ used to process the data of $d_m^n$ in the $t$-th slot. $\varpi_{m,k}^n(t)$ is the amount of computing resources required to process single bit data of $d_m^n$.

### 2.3 Load Balancing Degree Model between Edge Gateway and Cloud Server

Based on [24], we use queue backlog deviation to quantify the load balancing degree. Defined the load balancing degree of device $d_m^n$ in edge gateway $g_n$ as $\phi_n^Q(t)$, which is quantified as the difference between the data queue backlog $Q_m^n(t)$ and the average queue backlog. $\phi_n^Q(t)$ is given by

$$\phi_n^Q(t) = \frac{1}{M_n}\sum_{d_m^n\in\mathcal{D}_n} \omega_m^n \left(Q_m^n(t) - \frac{1}{M_n}\sum_{d_m^n\in\mathcal{D}_n} Q_m^n(t)\right), \tag{6}$$

where $\omega_m^n$ is service priority of $d_m^n$. $\frac{1}{M_n}\sum_{d_m^n\in\mathcal{D}_n} Q_m^n(t)$ is the average data queue backlog of $g_n$. It is intuitive that the load is more balanced if the deviation between $Q_m^n(t)$ and the average queue backlog is smaller.

Similarly, the load balancing degree of $g_n$ in cloud server $s_k$ is defined as

$$\phi_k^W(t) = \frac{1}{M}\sum_{g_n\in\mathcal{G}}\sum_{d_m^n\in\mathcal{D}_n} \omega_m^n \left(W_{m,k}^n(t) - \frac{1}{M}\sum_{g_n\in\mathcal{G}}\sum_{d_m^n\in\mathcal{D}_n} W_{m,k}^n(t)\right). \tag{7}$$

### 2.4 Queuing Delay Model of Edge Gateway and Cloud Server

Based on *Little's Law* [43] the queuing delay is positively proportional to the queue backlog and inversely proportional to the average data arrival rate. Therefore, the uplink queuing delay for the data of $d_m^n$ stored in $g_n$ is given in (8).

In (8), $\sum_{e=1}^{i-1} x_m^n(e) \sum_{z=(e-1)T_0+1}^{eT_0} A_m^n(z)$ is the total amount of data of $d_m^n$ uploaded to edge gateway $g_n$ in the previous $i-1$ periods, $\sum_{z=(i-1)T_0+1}^{t-1} x_m^n(i)A_m^n(z)$ is the total amount of data of $d_m^n$ uploaded to edge gateway $g_n$ from $((i-1)T_0+1)$-th slot to $(t-1)$-th slot.

The uplink queuing delay for the data of $d_m^n$ which are uploaded by $g_n$ in $s_k$ is

$$L_{m,k}^{n,W}(t) = \frac{W_{m,k}^n(t)}{\frac{1}{t-1}\sum_{e=1}^{t-1} y_{m,k}^n(e) u_{m,k}^n(e)}, \tag{9}$$

where $\frac{1}{t-1}\sum_{e=1}^{t-1} y_{m,k}^n(e) u_{m,k}^n(e)$ is the average amount of data of $d_m^n$ which are uploaded to the cloud server $s_k$ by $g_n$ in the previous $t-1$ slots.

### 2.5 Cloud-Edge-Device Collaborative High-Concurrency Access Management Model

In order to realize the cloud-edge-device collaborative high-concurrency access management, the channel pre-allocation between edge gateways and devices is optimized in large timescale, and the edge-cloud load balancing is optimized in small timescale. The optimization objective is to jointly minimize edge-cloud load balancing degree and queuing delay.

We define $\zeta(t)$ as the weighted sum of edge-cloud load balancing degree and edge-cloud queuing delay, i.e.,

$$\begin{aligned} \zeta(t) = &\sum_{g_n\in\mathcal{G}}\sum_{d_m^n\in\mathcal{D}_n}\left[\frac{1}{N}\phi_n^Q(t) + \lambda\frac{1}{K}\sum_{s_k\in\mathcal{S}}\phi_k^W(t)\right] + \\ &\lambda_L \sum_{g_n\in\mathcal{G}}\sum_{d_m^n\in\mathcal{D}_n}\left[\frac{1}{N}L_m^{n,Q}(t) + \lambda\frac{1}{K}\sum_{s_k\in\mathcal{S}} L_{m,k}^{n,W}(t)\right], \end{aligned} \tag{10}$$

where $\lambda$ is a weight to adjust the balance between edge side and cloud side. $\lambda_L$ is the weight of queuing delay in order to unify the order of magnitude between load balancing degree and queuing delay. Cloud-edge-device collaborative high-concurrency access management model is formulated as

$$
\begin{aligned}
&\min_{\{x_m^n(i), y_{m,k}^n(t)\}} \zeta(t) \\
\text{s.t.}\quad & C_1: x_m^n(i), y_{m,k}^n(t) \in \{0,1\}, \forall d_m^n \in \mathcal{D}_n, \forall g_n \in \mathcal{G}, \\
& \forall s_k \in \mathcal{S}, \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \\
& C_2: \sum_{d_m^n \in \mathcal{D}_n} x_m^n(i) \le q_n, \forall i \in \mathcal{I}, \forall g_n \in \mathcal{G}, \\
& C_3: \mathbb{E}\left[\frac{1}{I}\sum_{i=1}^{I} x_m^n(i)\right] \ge \epsilon_m^n, \forall d_m^n \in \mathcal{D}_n, \forall g_n \in \mathcal{G}, \\
& C_4: \sum_{s_k \in \mathcal{S}} y_{m,k}^n(t) = 1, \forall g_n \in \mathcal{G}, \forall t \in \mathcal{T}, \forall d_m^n \in \mathcal{D}_n, \\
& C_5: \sum_{g_n \in \mathcal{G}} \sum_{d_m^n \in \mathcal{D}_n} y_{m,k}^n(t) \le num_k, \forall s_k \in \mathcal{S}, \forall t \in \mathcal{T}, \\
& C_6: \sum_{g_n \in \mathcal{G}} \sum_{d_m^n \in \mathcal{D}_n} \left(W_{m,k}^n(t) + y_{m,k}^n(t) u_{m,k}^n(t)\right) \le W_k^{\max}, \\
& \forall s_k \in \mathcal{S}, \forall t \in \mathcal{T},
\end{aligned}
\tag{11}
$$

where $q_n$ is the maximum number of channels that are pre-allocatable to devices within the communication range of $g_n$; $\epsilon_m^n$ is the required access success rate for the device $d_m$. $W_k^{\max}$ is the maximum data queue backlog allowed by $s_k$. $num_k$ is the maximum number of data queues which are allowed to access to $s_k$.

$C_1$ is the constraint of large-timescale device-edge channel pre-allocation variable and small-timescale edge-cloud data access control variable. $C_2$ is the constraint of pre-allocation channel numbers. $C_3$ is the constraint of device access success rate. $C_4$ indicates that each edge gateway can access to only one cloud server. $C_5$ is the constraint on the data queue number of high-concurrency access of of cloud server $s_k$, i.e., at most $num_k$ data queues are allowed to access to $s_k$. $C_6$ is the constraint on the data queue backlogs of high-concurrency access of cloud server $s_k$. A data queue is not allowed to access to $s_k$ if the total data queue backlog exceeds $W_k^{\max}$.

## 3. High-Concurrency Access Management Algorithm Based on Multi-Timescale Joint Optimization of Channel Pre-Allocation and Load Balancing Degree

The problem formulated in (11) is a mixed integer non-linear programming problem involving multiple timescales, which is NP-hard. The optimization objective contains the load balancing degree and queuing delay of edge servers and cloud servers. Therefore, we ensure the load balancing degree and queuing delay performance of edge servers by optimizing device-edge channel pre-allocation in large timescale, and on this basis, we optimize data access control in small timescale to ensure the load balancing degree and queuing delay performance of edge servers and cloud servers simultaneously. Based on the above analysis, we decompose the formulated problem into a large-timescale device-edge channel pre-allocation subproblem and a small-timescale edge-cloud data access control subproblem. Then, we propose the high-concurrency access management algorithm based on multi-timescale joint optimization of channel pre-allocation and load balancing degree to solve the above subproblems, which is shown in Algorithm 1. The

**Algorithm 1** High-concurrency access management algorithm based on multi-timescale optimization of channel pre-allocation and load balancing degree

**For** $t$=1,2,...,$T$ **do**
  **Large-timescale device-edge channel pre-allocation based on device-edge collaborative access priority**
  **If** $t$=$T_0$,..,$iT_0$,...,$IT_0$, **then**
    Calculate service similarity degree $\eta_{m,h}^n(i)$ based on (12) and determine the device access type.
    Calculate the edge-side empirical data queue backlog performance deviation $\Delta Q_m^n(i)$, $\Delta A_m^n(i)$, and $\Delta U_m^n(i)$ based on (13), (14), and (15).
    Calculate the device-side access success rate deviation based on (16).
    Obtain the device-edge collaborative access priority score $Score_m^n(i)$ based on (17).
    Pre-allocate channels based on $\mathcal{D}_n^{Sco}(i)$ for $q_n$ devices, and set $x_m^n(i) = 1$ when device is pre-allocated to a channel by $g_n$.
  **Else**
  **Small-timescale edge-cloud collaborative data access control based on discounted empirical matching with perception of high-concurrency number and queue backlog**
  **Initialization:** Initialize $\Gamma_k(t) = \varnothing$, $\Omega = \Theta$, and $\Upsilon = \mathcal{S}$.
  **Discounted empirical weighted performance-based preference list construction:** Calculate $\alpha_{m,k}^n(t)$ and $\beta_{m,k}^n$ based on (20) and (21) and construct the preference list $\mathcal{F}_m^n$ and $\mathcal{F}_k$.
  **Perception of high-concurrency number and queue backlog-based matching iteration:**
  **While** $\mathcal{F}_m^n \neq \varnothing$, **do**
    **For** $\theta_m^n \in \Theta$, **do**
      $\theta_m^n$ sends matching proposal to its most preferred cloud server based on $\mathcal{F}_m^n$.
    **End for**
    **For** $s_k \in \Upsilon$, **do**
      **If** $|\Gamma_k(t)| \le num_k$ and $\sum_{\theta_m^n \in \Gamma_k(t)} [W_{m,k}^n(t) + u_{m,k}^n(t)] \le W_k^{\max}$, **then**
        $s_k$ establishes temporary matching relationship with the matching options which have sent match proposals to it.
      **Else**
        $s_k$ establishes temporary matching relationship with the first $num_k$ matching options in the $\mathcal{F}_k$. Remove unmatched options from $\Gamma_k(t)$.
        The matching option with the smallest preference value in $\Gamma_k(t)$ is removed successively until $\sum_{\theta_m^n \in \Gamma_k(t)} [W_{m,k}^n(t) + u_{m,k}^n(t)] \le W_k^{\max}$.
      **End if**
      The matched matching options in $\Gamma_k(t)$ are temporarily removed from $\Omega$. $s_k$ rejects other matching options that send matching proposals to $s_k$.
    **End for**
    The rejected matching option $\theta_m^n$ updates the preference list $\mathcal{F}_m^n$ by removing $s_k$.
  **End while**
**End for**

proposed algorithm solves the large-timescale device-edge channel pre-allocation subproblem by the device-edge collaborative access priority scoring mechanism. According to the large-timescale channel pre-allocation strategy, the proposed algorithm addresses the small-timescale edge-cloud data access control subproblem by the DEM with the perception of high-concurrency number and queue backlog.

### 3.1 Large-Timescale Device-Edge Channel Pre-Allocation Based on Device-Edge Collaborative Access Priority

We construct device-edge collaborative access priority score to solve the large-timescale channel pre-allocation subproblem. Firstly, we construct the service data feature vector based on the unique feature fields which appear frequently. The classification of the service carried by the device is performed by calculating the similarity degree between the device data traffic feature vector and the service data traffic feature vector. Then, based on the service classification results, we calculate the edge-side empirical data queue backlog performance deviation. Afterwards, we calculate the device-side access success rate performance deviation. Finally, the device-edge collaborative access priority score is derived by combining edge-side empirical data queue backlog performance deviation and device-side access success rate deviation. The access channel pre-allocation decision is determined based on the access priority score and the constraint of device-edge pre-allocation channel numbers.

#### 3.1.1 Power Service Classification

The edge gateway determines the device access type in each period. If $d_m^n$ is determined as a new access device, the service similarity degree is calculated by comparing the data traffic feature vector of $d_m^n$ with the service data traffic feature vector to achieve service classification. Otherwise, the edge gateway uses the historical service classification result of $d_m^n$. The service similarity degree between $d_m^n$ and the $h$-th type service is given by

$$\eta_{m,h}^n(i) = \frac{1}{\| \mathbf{F}_m^n(i) - \mathbf{X}_h \|_2}, \tag{12}$$

where $\mathbf{F}_m^n(i)$ is the data traffic feature vector of $d_m^n$, and $\mathbf{X}_h$ is the data feature vector of the $h$-th type service. When $\eta_{m,h}^n(i)$ is the largest one among all types of services, the data of device $d_m^n$ are identified as the $h$-th type service. $d_m^n$ is added to the device set of the $h$-th type service $O_h(i)$, i.e., $O_h(i) = O_h(i) \cup \{d_m^n\}$.

#### 3.1.2 Edge-Side Empirical Data Queue Backlog Performance Deviation

Assume $d_m^n \in O_h(i)$. Define $\Delta Q_m^n(i)$ as the deviation between the historical average queue backlog of $d_m^n$ and the historical average queue backlog of the $h$-th type service. Define $\Delta A_m^n(i)$ as the deviation between the historical average queue input of $d_m^n$ and the historical average queue input of the $h$-th type service. Define $\Delta U_m^n(i)$ as the deviation between the historical average queue output of $d_m^n$ and the historical queue output of the $h$-th type service. $\Delta Q_m^n(i)$, $\Delta A_m^n(i)$, and $\Delta U_m^n(i)$ are given by

$$\Delta Q_m^n(i) = \frac{1}{T_0} \sum_{t=(i-2)T_0+1}^{(i-1)T_0} Q_m^n(t) - \frac{1}{| O_h(i) | T_0} \sum_{d_m^n \in O_h(i)} \sum_{t=(i-2)T_0+1}^{(i-1)T_0} Q_m^n(t), \tag{13}$$

$$\Delta A_m^n(i) = \frac{1}{T_0} \sum_{t=(i-2)T_0+1}^{(i-1)T_0} A_m^n(t) - \frac{1}{| O_h(i) | T_0} \sum_{d_m^n \in O_h(i)} \sum_{t=(i-2)T_0+1}^{(i-1)T_0} A_m^n(t), \tag{14}$$

$$\Delta U_m^n(i) = \frac{1}{T_0} \sum_{t=(i-2)T_0+1}^{(i-1)T_0} U_m^n(t) - \frac{1}{| O_h(i) | T_0} \sum_{d_m^n \in O_h(i)} \sum_{t=(i-2)T_0+1}^{(i-1)T_0} U_m^n(t), \tag{15}$$

where $| O_h(i) |$ is the number of devices in the set $O_h(i)$.

#### 3.1.3 Device-Side Access Success Rate Deviation

The deviation between the required access success rate $\epsilon_m^n$ for $d_m^n$ and the historical average access success rate of $d_m^n$ is defined as

$$\Delta\epsilon_m^n(i) = \epsilon_m^n - \frac{1}{i-1} \sum_{e=1}^{i-1} x_m^n(e). \tag{16}$$

#### 3.1.4 Device-Edge Collaborative Access Priority Score

The device-edge collaborative access priority score is calculated based on edge-side queue backlog deviation $\Delta Q_m^n(i)$, queue input deviation $\Delta A_m^n(i)$, queue output deviation $\Delta U_m^n(i)$, and device-side access success rate deviation $\Delta\epsilon_m^n(i)$. Device-edge collaborative access priority score is derived as

$$Score_m^n(i) = \lambda_\epsilon \Delta\epsilon_m^n(i) + \Delta U_m^n(i) - \Delta Q_m^n(i) - \Delta A_m^n(i), \tag{17}$$

where $\lambda_\epsilon$ is the adjustment weight of $\Delta\epsilon_m^n(i)$. The larger $Score_m^n(i)$ is, the higher access priority $d_m^n$ will have in the $i$-th period.

#### 3.1.5 Access Channel Pre-Allocation Decision

Define $Score^{\min}$ as the access channel pre-allocation threshold. The set of devices meeting the threshold is defined as $\mathcal{D}_n^{Sco}(i) = \{d_m^n \mid Score_m^n(i) > Score^{\min}\}$. Arrange the devices in $\mathcal{D}_n^{Sco}(i)$ in descending order according to

$Score^n_m(i)$. If $d^n_m$ ranks top $q_n$ among all the devices in $\mathcal{D}_n$, $g_n$ pre-allocates a channel for $d^n_m$, i.e., $x^n_m(i) = 1$.

### 3.2 Small-Timescale Edge-Cloud Collaborative Data Access Control Algorithm Based on Discounted Empirical Matching with Perception of High-Concurrency Number and Queue Backlog

#### 3.2.1 Matching Model

The small-timescale edge-cloud data access control subproblem involves the optimization of data access control strategy among data queues, edge gateways, and cloud servers. Therefore, we model it as a three-dimensional matching among edge gateways, data queues, and cloud servers. The goal is to construct the matching to minimize the weighted sum of edge-cloud load balancing degree and edge-cloud queuing delay. Considering the high complexity, the three-dimensional matching is reduced into a two-side matching by combining gateways and data queues as new matching options. Define $\Theta = \{\theta^1_1, \ldots, \theta^n_m, \ldots, \theta^N_{M_N}\}$ as the set of matching options, where $\theta^n_m$ represents the data backlog queue of $d^n_m$ in edge gateway $g_n$. The number of matching options in $\Theta$ is $|\Theta| = \sum_{n=1}^{N} M_n$. Some definitions are introduced as the basis to derive the proposed algorithm.

*Definition 1 (Preference relation):* For each side participating matching, the preference relation reflects the degree of mutual preference, which is a complete, reflective and transitive binary relation over the available participants of the other side, i.e., "$\succ$". It is introduced to compare the preferences as

$$\theta^n_m \succ_{s_k} s'_k \Leftrightarrow \alpha^n_{m,k}(t) > \alpha^n_{m,k'}(t), \tag{18a}$$

$$s_k \succ_{\theta^n_m} \theta^{n'}_{m'} \Leftrightarrow \beta^n_{m,k}(t) > \beta^{n'}_{m',k}(t), \tag{18b}$$

where $\theta^n_m \succ_{s_k} s'_k$ means that the matching option $\theta^n_m$ prefers to the cloud server $s_k$ more than the cloud server $s_{k'}$ because the preference value $\alpha^n_{m,k}(t)$ is larger than $\alpha^n_{m,k'}(t)$.

*Definition 2 (Two-sided matching):* The decision of the edge-cloud collaborative data access control is a two-sided matching $\phi$, i.e.

$$\phi(\theta^n_m) \in \mathcal{S} \ \text{ and } \ |\phi(\theta^n_m)| = 1, \forall s_k \in \mathcal{S}, \tag{19a}$$

$$\phi(s_k) \subseteq \Theta \ \text{ and } \ |\phi(s_k)| = num_k, \forall \theta^n_m \in \Theta, \tag{19b}$$

$$\phi(s_k) \subseteq \Theta \ \text{ and } \sum_{\theta^n_m \in \phi(s_k)} \left(W^n_{m,k}(t) + y^n_{m,k}(t) u^n_{m,k}(t)\right) \le W^{\max}_k, \forall \theta^n_m \in \Theta, \tag{19c}$$

$$s_k = \phi(\theta^n_m) \Leftrightarrow \theta^n_m \in \phi(s_k), \tag{19d}$$

where (19a) ensures that each data queue can access to only one cloud server. (19b) indicates that $num_k$ matching options are allowed to access to $s_k$. (19c) indicates that matching options are not allowed to access to $s_k$ if the total data queue backlog exceeds $W^{\max}_k$. (19d) means the cloud server $s_k$ is matched to the matching option $\theta^n_m$ if $\theta^n_m$ is matched to $s_k$.

*Definition 3 (Rational matching):* A matching $\phi$ is individually rational if there does not exist a cloud server which prefers to keep unmatched compared with its current match. This implies that the cloud server in the matching process should not be unacceptable.

*Definition 4 (Blocked):* A matching $\theta$ is said to be blocked by a pair of participants $(\theta^n_m, s_k)$ if $\theta^{n'}_{m'} \notin \phi(s_k)$, $s_k \succ_{\theta^{n'}_{m'}} \phi(s_k)$ and $\theta^n_m \succ_{s_{k'}} s_k$, $s_{k'} \notin \phi(\theta^n_m)$.

*Definition 5 (Stable matching):* A matching $\phi$ is said to be stable if it is individually rational, and it is not blocked by any pair.

#### 3.2.2 Implementation Process of DEM

We propose a DEM mechanism to overcome information uncertainty caused by externalities. Accordingly, the proposed mechanism takes the discounted empirical matching performance into account when constructing the preference list. On the one hand, the empirical matching performance avoids the externality which rely on only historical observation. On the other hand, a discount factor is used to describe the weight of the influence of the historical performance on the calculation of the present preference value at different points in time. The introduction of a discount factor enables temporal adjustment of preference value, ensuring the accuracy of the present matching relationship. DEM contains three steps, i.e., initialization, discounted empirical performance-based preference list construction, and matching iteration base on perception of high-concurrency number and queue backlog.

- **Step 1:** Initialization. Define $\Gamma_k(t)$ as the set of matching options that are currently matched with $s_k$, and denote $|\Gamma_k(t)|$ as the number of matching options in $\Gamma_k(t)$. Denote the set of unmatched matching options as $\Omega$, and the set of unmatched cloud servers as $\Upsilon$. Initialize $\Gamma_k(t) = \varnothing$, $\Omega = \Theta$, and $\Upsilon = \mathcal{S}$.
- **Step 2:** Discounted empirical performance-based preference list construction. The preference value of $\theta^n_m$ for $s_k$ is defined as the discounted empirical weighted sum of load balancing degree and queuing delay of $g_n$, which is given by

$$\alpha^n_{m,k}(t) = -\sum_{z=1}^{t-1} \nu^{t-z} \Big[\omega^n_m \Big(Q^n_m(z) - \frac{1}{M_n} \sum_{d^n_m \in \mathcal{D}_n} Q^n_m(z)\Big) + \lambda_L L^{n,Q}_m(z)\Big]. \tag{20}$$

Similarly, the preference value of $s_k$ for $\theta^n_m$ is defined as the discounted empirical weighted sum of load balancing degree and queuing delay of $s_k$, which is given by

$$\beta^n_{m,k}(t) = -\sum_{z=1}^{t-1} \nu^{t-z} \Big[\omega^n_m \Big(W^n_{m,k}(z) - \frac{1}{M} \sum_{g_n \in \mathcal{G}} \sum_{d^n_m \in \mathcal{D}_n} W^n_{m,k}(z)\Big) + \lambda_L L^{n,W}_{m,k}(z)\Big], \tag{21}$$

where $\nu \in (0,1)$ is the discount factor used to adjust the influence weight of the historical performance at different points in time for the calculation of the present preference value. The discount factor makes the slots in the front have lower weight, ensuring the accuracy of the present matching relationship. Define the $\mathcal{F}_m^n$ as the preference list of $\theta_m^n$ for cloud servers and $\mathcal{F}_k$ as the preference list of $s_k$ for matching options. Then, matching options and cloud servers calculate preference values based on (20) and (21), and construct preference lists by sorting preference values in descending order.

- **Step 3:** Matching iteration base on perception of high-concurrency number and queue backlog. First, $\theta_m^n \in \Omega$ sends matching proposal to its most preferred cloud server based on $\mathcal{F}_m^n$, e.g., $s_k$. Afterward, $s_k$ calculates the number of received matching proposal. If $|\Gamma_k(t)| \leq num_k$ and $\sum_{\theta_m^n \in \Gamma_k(t)}[W_{m,k}^n(t) + u_{m,k}^n(t)] \leq W_k^{\max}$, $s_k$ establishes temporary matching relationship with the matching options which have sent match proposals to it, i.e. $y_{m,k}^n(t) = 1$. Otherwise, $s_k$ establishes temporary matching relationship with the first $num_k$ matching options in the $\mathcal{F}_k$. Remove unmatched options from $\Gamma_k(t)$. The matching option with the smallest preference value among $num_k$ options is removed from $\Gamma_k(t)$ until $\sum_{\theta_m^n \in \Gamma_k(t)}[W_{m,k}^n(t) + u_{m,k}^n(t)] \leq W_k^{\max}$. The matched matching options are temporarily removed from $\Omega$. Then, $s_k$ rejects other matching options that send matching proposals to $s_k$ and the rejected matching option $\theta_m^n$ updates the preference list $\mathcal{F}_m^n$ by removing $s_k$. Finally, the unmatched matching options make new proposals based on the updated preference lists.

Matching iteration ends when each matching option $\theta_m^n$ establishes a matching relationship with a cloud server or its preference list $\mathcal{F}_m^n = \varnothing$.

### 3.2.3 Property Analysis

The proposed algorithm provides an effective solution for high-concurrency access management of massive IoT devices in distribution grids. It focuses on optimizing the device-edge channel pre-allocation strategy in the large timescale to mitigate the capacity shortage of communication and computing resulting from high-concurrency access. Additionally, in the small timescale, the algorithm adopts DEM to determine the data access control strategy. This strategy aims to optimize the load balancing degree and queuing delay of edge-cloud resources in the face of high concurrency data access from IoT devices.

The complexity of the proposed algorithm includes preference list construction and matching iteration. Specifically, the complexity of matching options to construct preference lists is $O(K + K\log(K))$, and the complexity of cloud servers to construct preference lists is $O(M + M\log(M))$. In the matching iteration, based on constraint $C_5$ and constraint $C_6$, the matching option only needs to send matching proposal to the cloud server, and the cloud server establishes a temporary matching relationship with at most $num_k$ matching options, the complexity of both is $O(1)$. At the same time, each iteration has at least one matching options to complete the matching, the algorithm ends for a maximum of $M$ iterations, so the complexity of matching iteration is $O(M)$. Therefore, the complexity of proposed algorithm is $O(K + K\log(K)) + O(M + M\log(M)) + O(M)$, which has a linear logarithmic relationship with the number of matching options. As the number of matching options increases, the algorithm is still applicable.

## 4. Simulation Result

We consider the IEEE 33 node topology model for simulation validation [44]. The topology contains 3 cloud servers, 10 edge gateways, and 3000 IoT devices. Each device implements one type of service. The service priority contains four levels, i.e., the fourth-level service has the highest priority, and the first-level service has the lowest priority. The set of service priorities for four levels $\omega_m^n$ is $[0.8, 0.6, 0.4, 0.2]$, and the set of minimum expected constraints for four levels $\epsilon_m$ is $[0.8, 0.7, 0.6, 0.5]$. The simulation parameters are summarized in Table 1 [45], [46].

The performance of the proposed algorithm is compared with two existing algorithms, which are introduced below.

### 4.1 Load-Aware Channel Allocation (LACA) Algorithm [47]

LACA optimizes the channel pre-allocation optimization based on edge-side load in large timescale, but ignores the small-timescale edge-cloud data access control optimization.

### 4.2 Bipartite Matching-Based Edge-Cloud Collaborative Offloading (BMECO) Algorithm [48]

BMECO pre-allocates access channels based on service priority without considering the constraint of access success rate in large timescale, and optimizes the edge-cloud data access control based on the bipartite matching algorithm in small timescale, which neglects information uncertainty caused by the problem of externalities.

Figure 3 shows the edge-cloud load balancing degree versus time slot. The result shows that the edge-cloud load balancing degree of the proposed algorithm is the best and the fluctuation is the smallest. Compared with LACA and

**Table 1** Simulation parameters.

| Parameter | Value | Parameter | Value |
|---|---|---|---|
| $N$ | 10 | $M$ | $3 \times 10^3$ |
| $K$ | 3 | $I$ | 10 |
| $T$ | 100 | $\tau$ | 50 ms |
| $A_m^{n,thr}(t)$ | $[0.3, 0.7]$ Mbits | $R_m^{n,PLC}(t)$ | [6,8] Mbps |
| $\theta_{m,k}^n(t)$ | $[2, 4] \times 10^3$ cycles/bit | $R_{m,k}^{n,5G}(t)$ | [8,12] Mbps |
| $f_{m,k}^n(t)$ | $2 \times 10^{10}$cycles/s | $\omega_m^n$ | [0.8 0.6 0.4 0.2] |
| $q_n$ | 200 | $W_k^{\max}$ | 200 Mbps |
| $num_k$ | 700 | $\lambda_L$ | 0.1 |

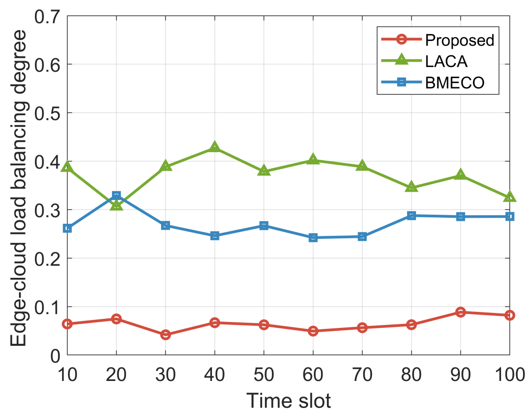


**Fig. 3** The edge-cloud load balancing degree versus time slot.

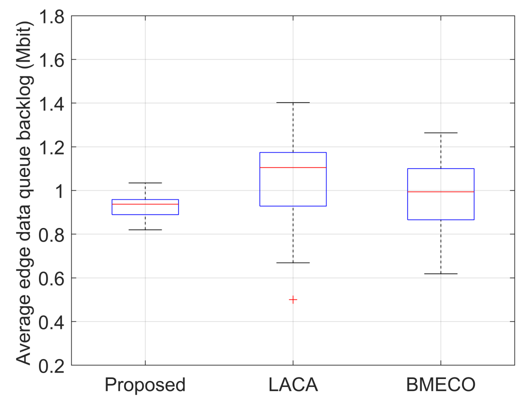


**Fig. 6** The box plots of average edge data queue backlog.

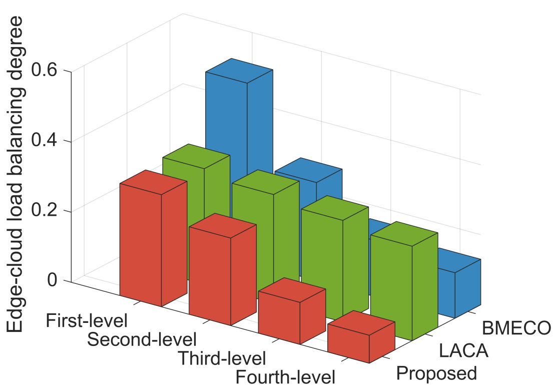


**Fig. 4** The edge-cloud load balancing degree for four-level services with differentiated priorities.

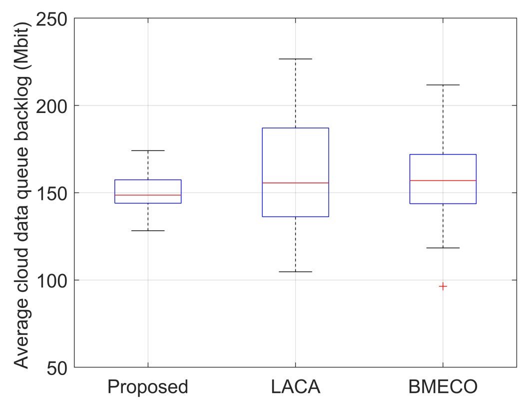


**Fig. 7** The box plots of average cloud data queue backlog.

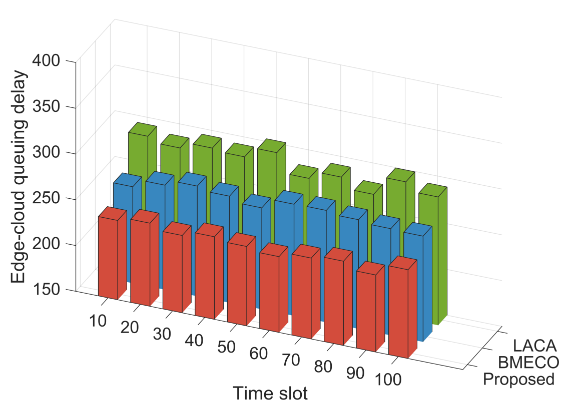


**Fig. 5** The edge-cloud queuing delay versus time slot.

BMECO, the edge-cloud load balancing degree of the proposed algorithm is improved by 74.68% and 71.27%, respectively. The reason is that the proposed algorithm performs channel pre-allocation according to edge-side empirical data queue backlog performance deviation and device-side access success rate deviation in large timescale to balance the queue backlog of different devices. At the same time, the proposed algorithm optimizes the edge-cloud data collaborative data access strategy based on perception of high-concurrency number and queue backlog, achieving greater edge-cloud load balancing degree performance.

Figure 4 shows the edge-cloud load balancing degree for four-level services with differentiated priorities. Compared with LACA and BMECO, the proposed algorithm increases the load balancing degree performance of the fourth-level service with the highest priority by 70.37% and 38.46%. This is because the proposed algorithm considers the service priority in both large-timescale and small-timescale optimizations. In large timescale, more strict constraint of access success rate is imposed for high-priority service. In small timescale, service priority is utilized as a weight to construct the matching preference, which ensures that high-priority data queues are matched to cloud servers with better load balancing degree. LACA ignores the optimization of edge-cloud data access and service priority, which cannot ensure the load balancing performance of high-priority services. BMECO performs better than LACA but worse than the proposed algorithm. It only considers service priorities in large-timescale channel pre-allocation but cannot achieve service priority aware in small-timescale data access control.

Figure 5 shows the edge-cloud queuing delay versus time slot. When $t = 100$, compared with LACA and BMECO, the proposed algorithm improves the edge-cloud queuing delay by 6.97% and 14.83%, respectively. It reduces queuing delay by simultaneously considering the edge-side data queue backlog performance deviation and cloud-side

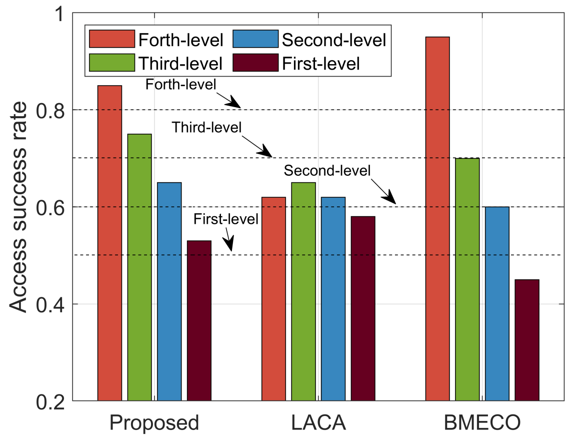


**Fig. 8** Access success rate of different service priorities.

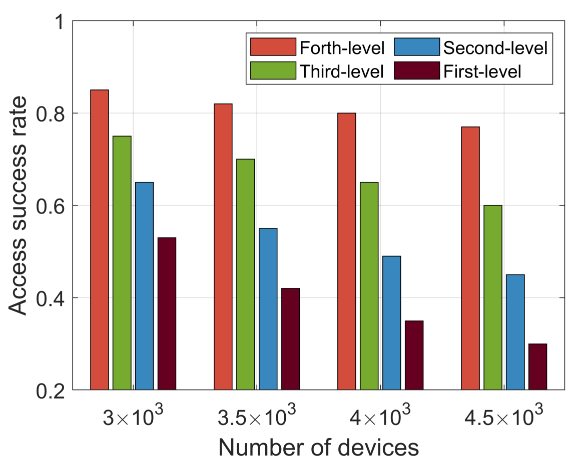


**Fig. 9** Access success rate versus the number of devices.

high-concurrency number and queue backlog. LACA has the worst queuing delay performance due to the large cloud-side queuing delay caused by uncoordinated high-concurrency data access. BMECO cannot construct accurate preference value based on edge-cloud queuing delay because of the information uncertainty caused by externalities. The performance of bipartite matching gets worse since matching is implemented based on the inaccurate preference value.

Figure 6 and Fig. 7 shows the box plots of edge queue backlog and cloud queue backlog. Compared with LACA and BMECO, the proposed algorithm reduces the edge queue backlog fluctuation by 76.67% and 67.19%, and the cloud queue backlog fluctuation by 62.35% and 60.22%. The reasons have been introduced in Fig. 5.

Figure 8 shows the access success rate of different service priorities. The proposed algorithm meets the constraints of access success rate for all services. It considers the device-side access success rate deviation in access priority scores, which is closely related to the service priority. Channels are pre-allocated to devices with larger deviation to increase the access success rate. LACA cannot provide access success rate guarantee for high-priority service because the service priority is ignored in channel pre-allocation. BMECO only considers the service priority but ignores the queue backlog deviation. Channels are aggressively pre-allocated to high-priority services, which reduces the access success rate of low-priority services.

Figure 9 shows the access success rate versus the number of devices. As the number of devices increases from $3 \times 10^3$ to $4.5 \times 10^3$, the access success rate of the forth-level service decreases by 9.41%, while that of the first-level service decreases by 43.39%. With the increase of device number, more channels are pre-allocated to high-priority services, which achieves service priority-aware access success guarantee under high-concurrency access of massive devices.

## 5. Conclusion

In this paper, we proposed a cloud-edge-device collaborative high-concurrency access management algorithm based on multi-timescale joint optimization of channel pre-allocation and load balancing degree for IoT devices in distribution grid. The proposed algorithm achieves joint guarantee of edge-cloud load balancing degree and queuing delay for services with differentiated priorities. Compared with LACA and BMECO, the proposed algorithm respectively improves edge-cloud load balancing degree by 74.68% and 71.27%, and edge-cloud queuing delay by 6.97% and 14.83%. In the future, we will further investigate high-concurrency access management from the perspective of sensing, communication, and computing integration.

## Acknowledgments

This work was supported by Science and Technology Project of State Grid Corporation of China “Key technologies research on security protection for distribution cloud master station and intelligent terminal” (52060021006L).

#### References

[1] H. Gao, W. Ma, S. He, L. Wang, and J. Liu, “Time-segmented multi-level reconfiguration in distribution network: A novel cloud-edge collaboration framework,” IEEE Trans. Smart Grid, vol.13, no.4, pp.3319–3322, 2022.

[2] H. Huang, M. Zhou, S. Zhang, L. Zhang, G. Li, and Y. Sun, “Exploiting the operational flexibility of wind integrated hybrid AC/DC power systems,” IEEE Trans. Power Syst., vol.36, no.1, pp.818–826, 2021.

[3] Y. Mi, C. Liu, J. Yang, H. Zhang, and Q. Wu, “Low-carbon generation expansion planning considering uncertainty of renewable energy at multi-time scales,” Global Energy Interconnection, vol.4, no.3, pp.261–272, 2021.

[4] Y. Xu, Z.Y. Dong, Z. Xu, K. Meng, and K.P. Wong, “An intelligent dynamic security assessment framework for power systems with wind power,” IEEE Trans. Ind. Informat., vol.8, no.4, pp.995–1003, 2012.

[5] F.A. Mourinho and T.M.L. Assis, “A new approach to retrofit plans for distributed energy resources to mitigate adverse impacts on bulk power systems stability,” IEEE Latin Am. Trans., vol.20, no.4, pp.669–676, 2022.

[6] Q. Li, H. Tang, Z. Liu, J. Li, X. Xu, and W. Sun, “Optimal resource allocation of 5G machine-type communications for situation awareness in active distribution networks,” IEEE Syst. J., vol.16, no.3,

pp.4187–4197, 2022.
[7] X. Lu, J. Wang, G. Liu, W. Du, and D. Yang, "Station-and-network–coordinated planning of integrated energy system considering integrated demand response," Global Energy Interconnection, vol.4, no.1, pp.39–47, 2021.
[8] Y. Huo, P. Li, H. Ji, J. Yan, G. Song, J. Wu, and C. Wang, "Data-driven adaptive operation of soft open points in active distribution networks," IEEE Trans. Ind. Informat., vol.17, no.12, pp.8230–8242, 2021.
[9] H. Liao, Z. Zhou, X. Zhao, L. Zhang, S. Mumtaz, 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, 2020.
[10] Y. Sun, Z. Cai, C. Guo, G. Ma, Z. Zhang, H. Wang, J. Liu, Y. Kang, and J. Yang, "Collaborative dynamic task allocation with demand response in cloud-assisted multiedge system for smart grids," IEEE Internet Things J., vol.9, no.4, pp.3112–3124, 2022.
[11] Y. Xu, L. Chen, Z. Lu, X. Du, J. Wu, and P.C.K. Hung, "An adaptive mechanism for dynamically collaborative computing power and task scheduling in edge environment," IEEE Internet Things J., vol.10, no.4, pp.3118–3129, 2023.
[12] M. Babar, M.A. Jan, X. He, M.U. Tariq, S. Mastorakis, and R. Alturki, "An optimized IoT-enabled big data analytics architecture for edge–cloud computing," IEEE Internet Things J., vol.10, no.5, pp.3995–4005, 2023.
[13] Q. Zhang, Q. Zhang, W. Shi, and H. Zhong, "Firework: Data processing and sharing for hybrid cloud-edge analytics," IEEE Trans. Parallel Distrib. Syst., vol.29, no.9, pp.2004–2017, 2018.
[14] Z. Ji, X. Wang, and D. Wu, "Research on task scheduling and concurrent processing technology for energy internet operation platform," Global Energy Interconnection, vol.5, no.6, pp.579–589, 2022.
[15] L. Zhao, E. Zhang, S. Wan, A. Hawbani, A.Y. Al-Dubai, G. Min, and A.Y. Zomaya, "MESON: A mobility-aware dependent task offloading scheme for urban vehicular edge computing," IEEE Trans. Mobile Comput., vol.PP, no.99, pp.1–15, 2023.
[16] S. Mao, L. Liu, N. Zhang, M. Dong, J. Zhao, J. Wu, and V.C.M. Leung, "Reconfigurable intelligent surface-assisted secure mobile edge computing networks," IEEE Trans. Veh. Technol., vol.71, no.6, pp.6647–6660, 2022.
[17] Y. Dong, G. Xu, M. Zhang, and X. Meng, "A high-efficient joint 'cloud-edge' aware strategy for task deployment and load balancing," IEEE Access, vol.9, pp.12791–12802, 2021.
[18] Z. Nezami, K. Zamanifar, K. Djemame, and E. Pournaras, "Decentralized edge-to-cloud load balancing: Service placement for the internet of things," IEEE Access, vol.9, pp.64983–65000, 2021.
[19] K. Govindarajan and T.S. Somasundaram, "A combinatorial optimization algorithm for load balancing in cloud infrastructure," 2017 Ninth International Conference on Advanced Computing (ICoAC), pp.58–63, 2017.
[20] P. Subedi, J. Hao, I.K. Kim, and L. Ramaswamy, "AI multi-tenancy on edge: Concurrent deep learning model executions and dynamic model placements on edge devices," 2021 IEEE 14th International Conference on Cloud Computing (CLOUD), pp.31–42, 2021.
[21] T. Shi, Z. Cai, J. Li, H. Gao, T. Qiu, and W. Qu, "An efficient processing scheme for concurrent applications in the IoT edge," IEEE Trans. Mobile Comput., vol.23, no.1, pp.135–149, 2024.
[22] H. Liao, Z. Zhou, N. Liu, Y. Zhang, G. Xu, Z. Wang, and S. Mumtaz, "Cloud-edge-device collaborative reliable and communication-efficient digital twin for low-carbon electrical equipment management," IEEE Trans. Ind. Informat., vol.19, no.2, pp.1715–1724, 2023.
[23] Z. Zhou, Y. Guo, Y. He, X. Zhao, and W.M. Bazzi, "Access control and resource allocation for M2M communications in industrial automation," IEEE Trans. Ind. Informat., vol.15, no.5, pp.3093–3103, 2019.
[24] S. Zafar, Z. Lv, N.H. Zaydi, M. Ibrar, and X. Hu, "DSMLB: Dynamic switch-migration based load balancing for software-defined IoT network," Computer Networks, vol.214, no.4, pp.109–145, 2022.
[25] F. Wang, H. Yao, Q. Zhang, J. Wang, R. Gao, D. Guo, and M. Guizani, "Dynamic distributed multi-path aided load balancing for optical data center networks," IEEE Trans. Netw. Service Manag., vol.19, no.2, pp.991–1005, 2022.
[26] L. Zhao, Z. Bi, A. Hawbani, K. Yu, Y. Zhang, and M. Guizani, "ELITE: An intelligent digital twin-based hierarchical routing scheme for softwarized vehicular networks," IEEE Trans. Mobile Comput., vol.22, no.9, pp.5231–5247, 2023.
[27] Y. Ju, Y. Chen, Z. Cao, L. Liu, Q. Pei, M. Xiao, K. Ota, M. Dong, and V.C.M. Leung, "Joint secure offloading and resource allocation for vehicular edge computing network: A multi-agent deep reinforcement learning approach," IEEE Trans. Intell. Transp. Syst., vol.24, no.5, pp.5555–5569, 2023.
[28] Y. Wen, L. Liu, J. Li, X. Hou, N. Zhang, M. Dong, M. Atiquzzaman, K. Wang, and Y. Huo, "A covert jamming scheme against an intelligent eavesdropper in cooperative cognitive radio networks," IEEE Trans. Veh. Technol., vol.72, no.10, pp.13243–13254, 2023.
[29] H. Zhang, L. Song, and Y.J. Zhang, "Load balancing for 5G ultra-dense networks using device-to-device communications," IEEE Trans. Wireless Commun., vol.17, no.6, pp.4039–4050, 2018.
[30] L. Zhen, Y. Li, and K. Yu, "A dynamic distributed queueing-based random access protocol for softwarized internet of things," 2022 IEEE Globecom Workshops (GC Wkshps), pp.796–801, 2022.
[31] S. Shahrear Tanzil, O.N. Gharehshiran, and V. Krishnamurthy, "Femto-cloud formation: A coalitional game-theoretic approach," 2015 IEEE Global Communications Conference (GLOBECOM), pp.1–6, 2015.
[32] A.-T.H. Bui, C.T. Nguyen, T.C. Thang, and A.T. Pham, "Design and performance analysis of a novel distributed queue access protocol for cellular-based massive M2M communications," IEEE Access, vol.6, pp.3008–3019, 2018.
[33] J.-D.T. Tom-Ata and D. Kyriazis, "Real-time adaptable resource allocation for distributed data-intensive applications over cloud and edge environments," 2020 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), pp.77–81, 2020.
[34] Y.-H. Xu, C.-C. Yang, M. Hua, and W. Zhou, "Deep deterministic policy gradient (DDPG)-based resource allocation scheme for NOMA vehicular communications," IEEE Access, vol.8, pp.18797–18807, 2020.
[35] X. Zhang, H. Zhang, X. Zhou, and D. Yuan, "Energy minimization task offloading mechanism with edge-cloud collaboration in IoT networks," 2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring), pp.1–7, 2021.
[36] C. Swain, M.N. Sahoo, A. Satpathy, K. Muhammad, S. Bakshi, J.J.P.C. Rodrigues, and V.H.C. de Albuquerque, "METO: Matching-theory-based efficient task offloading in IoT-Fog interconnection networks," IEEE Internet Things J., vol.8, no.16, pp.12705–12715, 2021.
[37] Y. Hou, S. Garg, L. Hui, D.N.K. Jayakody, R. Jin, and M.S. Hossain, "A data security enhanced access control mechanism in mobile edge computing," IEEE Access, vol.8, pp.136119–136130, 2020.
[38] L. Cao, H. Zhao, X. Li, and J. Zhang, "Matching theory for channel allocation in cognitive radio networks," 2016 IEEE 83rd Vehicular Technology Conference (VTC Spring), pp.1–5, 2016.
[39] M. Rahim, A.S. Alfakeeh, R. Hussain, M.A. Javed, A. Shakeel, Q.U. Hasan, A. Israr, A.O. Alsayed, and S.A. Malik, "Efficient channel allocation using matching theory for QoS provisioning in cognitive radio networks," Sensors, vol.20, no.7, p.1872, 2020.
[40] J. Agoun and M.-S. Hacid, "Access control based on entity matching for secure data sharing," Service Oriented Computing and Applications, vol.16, no.1, pp.31–44, 2022.
[41] S. Sekander, H. Tabassum, and E. Hossain, "Matching with externalities for decoupled uplink-downlink user association in full-duplex small cell networks," 2015 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), pp.411–414, 2015.

[42] B. Di, L. Song, and Y. Li, "Radio resource allocation for uplink sparse code multiple access (SCMA) networks using matching game," 2016 IEEE International Conference on Communications (ICC), pp.1–6, 2016.

[43] J.D.C. Little, "A proof for the queuing formula: $L = \lambda W$," Operations Research, vol.9, no.3, pp.383–387, 1961.

[44] F. Chang, Q. Zhu, S. Hu, Z. Li, Z. Zang, and R. Wang, "Fixed change-rate matrix correction algorithm for processing PV nodes in active distribution network power flow calculation," 2019 IEEE Innovative Smart Grid Technologies - Asia (ISGT Asia), pp.1473–1478, 2019.

[45] H. Liao, Z. Jia, Z. Zhou, Y. Wang, H. Zhang, and S. Mumtaz, "Cloud-edge-end collaboration in air–ground integrated power IoT: A semidistributed learning approach," IEEE Trans. Ind. Informat., vol.18, no.11, pp.8047–8057, 2022.

[46] M. Maule, J.S. Vardakas, and C. Verikoukis, "A novel 5G-NR resources partitioning framework through real-time user-provider traffic demand analysis," IEEE Syst. J., vol.16, no.4, pp.5317–5328, 2022.

[47] H. Taramit, L. Orozco-Barbosa, A. Haqiq, J.J.C. Escoto, and J. Gomez, "Load-aware channel allocation for IEEE 802.11ah-based networks," IEEE Access, vol.11, pp.24484–24496, 2023.

[48] X. Zhang, H. Zhang, X. Zhou, and D. Yuan, "Energy minimization task offloading mechanism with edge-cloud collaboration in IoT networks," 2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring), pp.1–7, 2021.

**Shuai Li** received his M.S. degree from Shandong University in 2018. He is now working as a senior engineer in Distribution Technology Center of Shandong Electric Power Research Institute of State Grid. His research direction includes cloud-edge-device collaboration and resource management in distribution grid.

**Xinhong You** is now working as a senior engineer in Distribution Technology Center of Shandong Electric Power Research Institute of State Grid. Her research direction includes cloud-edge-device collaboration and resource management in distribution grid.

**Shidong Zhang** received his Ph.D. degree from Beijing University in 2012. He is now working as a senior engineer in Distribution Technology Center of Shandong Electric Power Research Institute of State Grid. His research direction includes cloud-edge-device collaboration and resource management in distribution grid.

**Mu Fang** received his M.S. degree from Shandong University in 2018. He is now working as a senior engineer in State Grid Shandong Electric Power Company. His research direction includes cloud-edge-device collaboration and resource management in distribution grid.

**Pengping Zhang** is now working as a senior engineer in Distribution Technology Center of Shandong Electric Power Research Institute of State Grid. His research direction includes cloud-edge-device collaboration and resource management in distribution grid.