Reputation-Based Model Aggregation and Resource Optimization in Wireless Federated Learning Systems

Jie Feng, *Member, IEEE*, Yanyan Liao, *Graduate Student Member, IEEE*, Lei Liu, *Member, IEEE*, Qingqi Pei, *Senior Member, IEEE*, Ning Zhang, *Senior Member, IEEE*, and Keqin Li, *Fellow, IEEE*

***Abstract*— Federated learning (FL) has received widespread attention from academia and industry because it overcomes traditional security limitations associated with model training data. However, the FL process is vulnerable to manipulation by locally malicious users, who can alter their local data, thus impacting the accuracy of the model's training outcomes. Meanwhile, optimizing delay in FL needs to take individual client fairness into consideration. In this paper, we present a reputation-based model aggregation and resource optimization framework to enhance the efficiency and reliability of training in wireless FL systems. Particularly, we investigate a total delay minimization problem while ensuring fairness among clients, which jointly optimizes client scheduling, transmit rate, bandwidth proportion, and CPU frequency. Considering the non-convexity and high complexity of the objective function, we decoupled the optimal variables and designed an efficient algorithm. By doing this, the client scheduling policy is obtained by deep reinforcement learning. Then, the transmit rate allocation and bandwidth proportion are derived through the Lagrangian dual method. Finally, we attain the CPU frequency allocation via the adaptive harmony algorithm. Simulation results reveal that our algorithm can establish delay fairness among clients and balance convergence performance and delay.**

***Index Terms*— Federated learning, reputation, fairness, deep reinforcement learning, client scheduling, resource allocation.**

Received 15 April 2024; revised 3 November 2024; accepted 26 December 2024. Date of publication 20 January 2025; date of current version 11 April 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62132013, in part by China Postdoctoral Science Foundation under Grant 2024M752525, in part by the Key Research and Development Programs of Shaanxi under Grant 2024GX-YBXM-071, in part by the Basic Strengthening Plan Program under Grant 2023-JCJQ-JJ-0772, and in part by the Fundamental Research Funds for the Central Universities under Grant YJSJ24006. The associate editor coordinating the review of this article and approving it for publication was K. Xue. *(Corresponding author: Qingqi Pei.)*

Jie Feng and Qingqi Pei are with the School of Telecommunication Engineering and Shaanxi Key Laboratory of Blockchain and Secure Computing, Xidian University, Xi'an, Shaanxi 710071, China (e-mail: jiefengcl@163.com; qqpei@mail.xidian.edu.cn).

Yanyan Liao and Lei Liu are with the Guangzhou Institute of Technology, Xidian University, Guangzhou 510555, China (e-mail: yyanliao0909@163.com; tianjiaoliulei@163.com).

Ning Zhang is with the Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON N9B 3P4, Canada (e-mail: ning.zhang@uwindsor.ca).

Keqin Li is with the Department of Computer Science, State University of New York at New Paltz, New Paltz, NY 12561 USA (e-mail: lik@newpaltz.edu).

Digital Object Identifier 10.1109/TWC.2025.3528408

I. Introduction

FEDERATED learning (FL) is a novel distributed machine learning that allows multiple participants to train machine-learning models locally. FL has garnered considerable attention from academia and industry in recent years due to its ability to prevent the leakage of user privacy data by keeping the original data decentralized and not uploaded to cloud servers [1], [2]. However, while FL offers substantial benefits, it also faces significant challenges. Firstly, malicious participants can manipulate data or models to skew FL outcomes, compromising the system's credibility and accuracy [3]. Secondly, FL grapples with resource imbalances and inefficiencies [4]. Moreover, client device channel conditions and computational resource variations can lead to straggler issues during synchronized model updates [5].

To solve the aforementioned challenges, extensive research has focused on strategies for malicious aggregation and resource optimization. Efforts to combat malicious attacks in FL have explored various approaches, such as filtering [6], [7], pruning [8], [9], and aggregation [10], [11], [12]. Among these, Richards et al. [13] highlighted that minimum aggregation is the most effective method for resisting distributed backdoor attacks compared to filtering and pruning. However, it's crucial to acknowledge that while these methods enhance the system's resilience to attacks, they do not eliminate the threat entirely. Moreover, another series of studies has concentrated on defending against attacks from the perspective of client reliability, prioritizing the selection of clients based on their reliability. In [14], a strategy network based on encoder-decoder architecture was designed, which can automatically make decisions based on the data quality learned from the client. In [15], The authors introduced a reputation model utilizing the beta distribution function to assess the credibility of local data, serving as the exclusive criterion for client selection. However, these methods inherently favor clients with higher reliability scores, aiming to enhance network performance while potentially neglecting the principle of fairness among participants.

Therefore, the authors [16] proposed clustering clients into several groups based on their physical performance and selecting only the same type of clients for training in each round to reduce waiting time. In [17], the authors proposed a

more comprehensive clustering method, combining it with a reinforcement learning-based resource allocation algorithm to balance training and communication time, thereby accelerating the FL process. The authors [18] presented an energy-efficient adaptive FL framework at the network edge, which jointly optimized the radio and computation resources to achieve the trade-off between the energy, delay, and performance of FL. Considering the joint optimization of client scheduling and resource allocation, the authors [19] formulated a min-max optimization problem to ensure fairness for FL in vehicular edge computing. However, the works mentioned above did not consider the impact of malicious users on FL. Furthermore, current defense mechanisms against malicious attacks do not adapt well to FL environments affected by stragglers with limited resources.

Driven by the identified challenges, our research delves into a reputation-based scheduling framework coupled with resource allocation within wireless FL systems. We specifically address an optimization challenge aimed at minimizing FL training delays. This involves a holistic approach to optimize client scheduling, client CPU frequency, bandwidth allocation, and transmit rates, all while upholding principles of user fairness and model integrity. The primary contributions of this paper can be outlined as follows:

- We propose an optimization framework of reputation-based model aggregation for wireless FL systems, thereby achieving an integrated optimization of both communication and computational resources. In this framework, the reputation model is introduced to evaluate clients' quality.
- Under the premise of ensuring that the client model is trustworthy, we introduce a min-max optimization framework designed to ensure equitable treatment and fairness for all participating clients. The proposed optimization problem is a mixed-integer nonlinear programming (MINLP) problem, which is difficult to solve directly.
- To address the optimization problem, we strategically decompose the overarching problem into three manageable sub-problems, for which we propose innovative, efficient algorithmic solutions. By this, we develop a deep reinforcement learning algorithm for client scheduling and an iterative algorithm for radio resource allocation, including transmit rate and bandwidth. In addition, we exploit an adaptive harmony search algorithm for CPU frequency allocation.
- The simulation results exhibit the performance of FL under the different datasets. Meanwhile, our proposed algorithms not only excel in reducing delays but also adeptly balance convergence efficiency with time delay, showcasing their effectiveness across various datasets.

The subsequent sections of this article are structured as follows. Section II presents the construction of an optimization problem designed to simultaneously enhance client scheduling and resource allocation. Section III breaks down the main objective function into three sub-problems, with each sub-problem's solution being addressed in turn. Simulation outcomes are demonstrated in Section IV, illustrating the effectiveness of our proposed schemes. Section V provides a summary of the main findings and contributions.

II. System Model and Problem Formulation

A. Federated Learning

We consider a FL framework under wireless networks consisting of a server and K clients. We define the client set, represented $\mathcal{K} = \{1, 2, \ldots, K\}$. The clients train models using their local dataset, denoted by $\mathcal{D}_k = \{e_{kj}, y_{kj}\}$, where e_{kj} and y_{kj} indicates the feature and label of sample j, respectively. Assume that the FL training process contains T epochs, and the global model is updated once in each epoch. In epoch τ, the loss function of client k can be expressed as $F_k(\boldsymbol{W}^\tau) = \frac{1}{D_k}\sum_{j=1}^{D_k} f_k(\boldsymbol{W}^\tau, e_{kj}, y_{kj})$, where D_k is the size of the set $\mathcal{D}_k$, $\boldsymbol{W}^\tau$ is the global model and $f_k(\boldsymbol{W}^\tau, e_{kj}, y_{kj})$ is the loss of a single sample. Then, the global loss function in epoch τ is defined by

$$F(\boldsymbol{W}^\tau) = \sum_{k=1}^{K} \frac{D_k}{D} F_k(\boldsymbol{W}^\tau), \tag{1}$$

where D means the total sample size.

In FL, finding the global optimal solution $\boldsymbol{W}^* = \arg\min F(\boldsymbol{W}^\tau)$ is its ultimate goal [20]. The τ-th epoch of FL is composed of three processes. Firstly, the server selects a subset of N clients from $\mathcal{K}$ for training to reduce the communication cost [21]. We define a binary variable $x_k(\tau) \in \{0, 1\}$, where $x_k(\tau) = 1$ represents that client k participates in the training in epoch τ and $x_k(\tau) = 0$ indicating non-participation. Then, we assume that the number of local training rounds for the selected client is set to I [20]. At the i-th round of local training, the model update process of client k in epoch τ is given by:

$$\boldsymbol{w}_{k,i+1}^\tau = \boldsymbol{w}_{k,i}^\tau - \varkappa \nabla F_k(\boldsymbol{w}_{k,i}^\tau), \tag{2}$$

where $\varkappa$ is the local learning rate, $\boldsymbol{w}_{k,i}^\tau$ is the local model of client k in epoch t, and $F_k(\boldsymbol{w}_{k,i}^\tau) = \frac{1}{D_k}\sum_{j=1}^{D_k} f(\boldsymbol{w}_{k,i}^\tau, e_{kj}, y_{kj})$ is the loss function of the local model on the dataset $\mathcal{D}_k$. After the I round of training, the model transmitted by the client k in epoch τ to the global server is $\boldsymbol{w}_{k,I}^\tau$. In the aggregation process, most of the work follows FedAvg's method. The FedAvg algorithm weights all training models from the perspective of data size ($\boldsymbol{W}^\tau = \sum_{k=1}^{K} x_k(\tau)\frac{D_k}{D}\boldsymbol{w}_{k,I}^\tau$), which cannot effectively reflect the actual utility of the client model on the global model. Research has shown that as data heterogeneity increases, the global model aggregated by the FedAvg algorithm deviates significantly from the local model, resulting in a significant decrease in accuracy. Therefore, it is crucial to restore the actual training value of each client.

We comprehensively evaluate data volume and accuracy to demonstrate the contributions of local client models to the global model. As specified in [22], in epoch τ, we normalize the training accuracy to measure the amount of information on the client k from the accuracy perspective, which is given by

$$in_k(\tau) = -log_2\left(\frac{acc_k(\tau)}{\sum_{k=1}^{K} x_k(\tau) acc_k(\tau)}\right), \tag{3}$$

Algorithm 1 Federated Learning

Input: Set the iteration round T, client sets $\mathcal{K}$, the local learning rate $\varkappa$, the weight factor (Ψ_1, Ψ_2), and the global model parameters W^τ.

Output: $\boldsymbol{W^T}, F(\boldsymbol{W}^T)$

for $\tau = 1, \ldots, T$ **do**
  The server chooses a subset based on the client scheduling policy.
  for client k in the subset **do**
    Client k downloads the global model $\boldsymbol{W}^\tau$.
    Initialize $\boldsymbol{w}_{k,0}^\tau = \boldsymbol{W}^\tau$.
    for $i = 0, 1, \ldots, I-1$ **do**
      $\boldsymbol{w}_{k,i+1}^\tau = \boldsymbol{w}_{k,i}^\tau - \varkappa \nabla F_k(\boldsymbol{w}_{k,i}^\tau)$
    end for
    Transfers the local model $\boldsymbol{w}_{k,I}^\tau$ to the server for global aggregation. $\boldsymbol{W}^\tau = \sum_{k=1}^K a_k(\tau) weight_k(\tau) \boldsymbol{w}_{k,I}^\tau$.
  end for
end for

where $acc_k(\tau)$ denote the train accuracy of the client k in epoch τ. Generally, under the same global initial model, the lower the training accuracy, the more complex information the client contains. Therefore, we used a minus sign before the formula.

Then, we propose an aggregation weight that combines data size and accuracy, which can meet the contribution requirements. Algorithm 1 shows the FL process in detail.

$$weight_k(\tau) = \frac{\Psi_1 D_k}{\sum_{k=1}^K x_k(\tau) D_k} + \frac{\Psi_2 in_k(\tau)}{\sum_{k=1}^K x_k(\tau) in_k(\tau)}, \quad (4)$$

where Ψ_1 and Ψ_2 are positive weighting factors.

B. Client Reputation Model

There are many participants in the FL system, which is likely to include one or more malicious clients. Malicious clients can upload incorrect local model parameters, with the potential risk of revealing information from other honest users. This can compromise the system's availability, confidentiality, and integrity, causing severe deviations from the local model. Common poisoning attacks attack training models by contaminating data, such as adding incorrect labels or biased data (label flipping attacks). Because the client's local data and training process are not visible to the server, verifying the reliability of updates uploaded by specific clients is difficult. To tackle this problem, we use a subjective logical model to evaluate the quality of each client, which helps the server detect malicious clients and select clients with high reputations.

1) Subjective Logical Model: The subjective logical framework is represented by a tuple $\{b_k, z_k, u_k\}$, where b_k, z_k, u_k indicates trust, distrust and uncertainty of client k [23], which are given by

$$\begin{cases} b_k = \dfrac{pos_k}{pos_k + neg_k + 1} \\ z_k = \dfrac{neg_k}{pos_k + neg_k + 1} \\ y_k = \dfrac{1}{pos_k + neg_k + 1} \end{cases} \quad (5)$$

where we assume that the server and client k have successfully communicated $pos_k + neg_k$ times. pos_k and neg_k represent the positive and negative behavior of client k, respectively. Then, the reputation of client k can be given by

$$RE_k(\tau) = b_k + \frac{y_k}{2} = \frac{pos_k + \frac{1}{2}}{pos_k + neg_k + 1}. \quad (6)$$

2) Reputation's Update: Once the client uploads the local update successfully, the server assesses the updated local model using the test set, denoted by $\mathcal{D}^{test} = \{x^{test}, y^{test}\}$. We characterize the client's effectiveness $\rho_k(\tau)$ by using the gap between the loss of the previous round of aggregation model on the test set $F^{test}(\boldsymbol{W}^{\tau-1}, x^{test}, y^{test})$ and the loss of the local model on the test set $F^{test}(\boldsymbol{w}_{k,I}^\tau, x^{test}, y^{test})$.

$$\rho_k(\tau) = F^{test}(\boldsymbol{W}^{\tau-1}) - F^{test}(\boldsymbol{w}_{k,I}^\tau). \quad (7)$$

If $\rho_k(\tau) \leqslant 0$, the client's training in epoch τ is a positive behavior; otherwise, it is considered a negative behavior. The $\rho_k(\tau)$ of malicious clients is usually tiny and differs significantly from that of honest clients. Based on the evaluated update validity, the server can determine the reputation of local data. We combine the aging weight proposed by [24] to eliminate accidental factors to update pos_k and neg_k as follows:

$$\begin{cases} pos_k^{new} & = \big(w_e\, pos_k + p_1\, U(\rho_k^\tau)\big) x_k(\tau) \\ & \quad + pos_k\big(1 - x_k(\tau)\big), \quad \text{if } \rho_k(\tau) \geqslant 0, \\ neg_k^{new} & = \big(w_e\, neg_k - p_2\, U(\rho_k^\tau)\big) x_k(\tau) \\ & \quad + neg_k\big(1 - a_k(\tau)\big), \quad \text{else}, \end{cases} \quad (8)$$

where w_e is the aging weight, ranging from [0,1]. p_1 and p_2 are positive and negative weight factors, respectively, where meet $p_1 + p_2 = 1$. $U(\cdot)$ is defined as a contribution utility function, then $U(\rho_k^\tau) = tanh(\zeta \rho_k^\tau)$, where ζ is a positive factor. Substituting the updated pos_k^{new} and neg_k^{new} into (6) yields the new reputation of the client, which is used to guide the client's scheduling for the next epoch.

C. Delay Model

After I round of training locally, the training delay of client k in epoch τ is defined as

$$t_k^{train}(\tau) = \frac{I d_k C_k}{f_k(\tau)} x_k(\tau), \quad (9)$$

where C_k is the number of CPU cycles needed to calculate one sample data of client k, d_k is the mini-batch size, and $f_k(\tau)$ is the CPU frequency of client k in epoch τ.

We assume there is no interference in the interaction between the clients and the server, such as OFDMA communication. In the system, all clients share the total bandwidth B. Let $h_k(\tau)$ and $P_k(\tau)$ denote the channel gain and the transmit

power between client k and the server in epoch τ. Then, the reachable transmit rate of client k in epoch τ can be given by

$$o_k(\tau) = \alpha_k(\tau) B \log_2 \left(1 + \frac{h_k(\tau) P_k(\tau)}{\alpha_k(\tau) B N_0}\right), \tag{10}$$

where N_0 is the Gaussian noise power and $\alpha_k(\tau)$ indicate the proportion of bandwidth allocated to client k in epoch τ. Accordingly, the transmit delay of the client k in epoch τ is denoted as

$$t_k^{up}(\tau) = \frac{A_k}{o_k(\tau)} x_k(\tau), \tag{11}$$

where A_k is the model data size transferred by client k.

In epoch τ, the total delay in updating a global model can be expressed as

$$t^{total}(\tau) = \max_{k \in \mathcal{K}} t_k = \max_{k \in \mathcal{K}} \{(t_k^{train}(\tau) + t_k^{up}(\tau))\}. \tag{12}$$

D. Energy Model

By flexibly adjusting processor Frequency, client devices can reduce energy consumption and shorten computation time by leveraging Dynamic Voltage and Frequency Scaling (DVFS) technology. The computing energy consumption of client k completing I rounds of training locally is denoted as

$$E_k^{train}(\tau) = u I d_k C_k f_k(\tau)^2 x_k(\tau), \tag{13}$$

where u is the power coefficient, depending on the chip architecture. According to (10), the transmit power of client k can be derived as

$$P_k(\tau) = \frac{\alpha_k(\tau) B N_0}{h_k(\tau)} \left(2^{\frac{o_k(\tau)}{\alpha_k(\tau) B}} - 1\right). \tag{14}$$

Consequently, the transmit energy consumption of client k in τ epoch can be defined by

$$E_k^{up}(\tau) = \frac{\alpha_k(\tau) B N_0 A_k}{h_k(\tau) o_k(\tau)} \left(2^{\frac{o_k(\tau)}{\alpha_k(\tau) B}} - 1\right) x_k(\tau). \tag{15}$$

E. Problem Formulation

In this paper, we investigate the total delay minimization problem of the worst-outcome client during the entire FL training process. Such delays predominantly arise from slow network speeds or suboptimal device performance, extending the time required for these clients to update parameters and complete their model training. In that case, we propose a joint optimization of the client scheduling $\boldsymbol{x(\tau)} = (x_k(\tau))$, transmit rate $\boldsymbol{o(\tau)} = (o_k(\tau))$, bandwidth proportion $\boldsymbol{\alpha(\tau)} = (\alpha_k(\tau))$, and CPU frequency $\boldsymbol{f(\tau)} = (f_k(\tau))$. The formulated problem is as follows:

$$\begin{aligned}
\text{P1}: \min_{\boldsymbol{x(\tau)}, \boldsymbol{f(\tau)}, \boldsymbol{o(\tau)}, \boldsymbol{\alpha(\tau)}} & \sum_{\tau=1}^{T} \max_{k \in \mathcal{K}} \left\{ \frac{A_k}{o_k(\tau)} x_k(\tau) \right. \\
& \left. + \frac{I d_k C_k}{f_k(\tau)} x_k(\tau) \right\} \\
\text{s.t.} \quad & (\text{C1}): x_k(\tau) \in \{0,1\}, \quad \forall k, \tau, \\
& (\text{C2}): \sum_{k=1}^{K} x_k(\tau) \leqslant N, \quad \forall k, \tau,
\end{aligned} \tag{16}$$

$$\begin{aligned}
& (\text{C3}): RE_k(\tau) \geqslant RE_k^{require}, \quad \forall k, \tau, \\
& (\text{C4}): f_k^{min} \leqslant f_k(\tau) \leqslant f_k^{max}, \quad \forall k, \tau, \\
& (\text{C5}): o_k(\tau) \geqslant 0, 0 \leqslant \alpha_k(\tau) \leqslant 1, \quad \forall k, \tau, \\
& (\text{C6}): \sum_{k=1}^{K} \alpha_k(\tau) \leqslant 1, \quad \forall \tau, \\
& (\text{C7}): E_k^{train}(\tau) + E_k^{up}(\tau) \leqslant E^{max}, \quad \forall k, \tau.
\end{aligned}$$

In problem (P1), $RE_k^{require}$ is the reputation necessary of client k. f_k^{min} and f_k^{max} are the minimum and maximum CPU frequency needed for the client k. E^{max} is the maximum energy consumption. (C1) and (C2) denote the client scheduling constraints. (C3) indicate that the reputation of client k must greater than $RE_k^{require}$. (C4) is the constraint of client k CPU frequency. (C5) and (C6) guarantees the transmit rate requirements of client k and bandwidth allocation constraints. (C7) is the peak energy consumption constraint.

III. Client Scheduling and Resource Allocation Algorithm

Problem (P1) is a typical mixed integer nonlinear programming problem (MINLP) because it has binary and continuous variables. It is difficult and complex for traditional optimization algorithms to deal with this problem. In this paper, we decompose the original problem into three sub-problems. We first use deep reinforcement learning (DRL) to tackle the client scheduling problem. It is well known that reinforcement learning is a promising method for solving combinatorial optimization problems [25]. Then, we exploit an algorithm to obtain the optimal solution for transmit rate, bandwidth ratio, and CPU frequency.

A. Reinforcement Learning for Client Scheduling

For given $\boldsymbol{o(\tau)}$, $\boldsymbol{\alpha(\tau)}$, and $\boldsymbol{f(\tau)}$, the client scheduling problem from (P1) can be recast as

$$\begin{aligned}
\text{SUB1}: \quad & \min_{\boldsymbol{x(\tau)}} \sum_{\tau=1}^{T} t^{total}(\tau) \\
\text{s.t.} \quad & (\text{C1}): x_k(\tau) \in \{0,1\}, \\
& (\text{C2}): \sum_{k=1}^{K} x_k(\tau) \leqslant N, \\
& (\text{C3}): RE_k(\tau) \geqslant RE_k^{require}, \forall k, \\
& (\text{C7}): E_k^{train}(\tau) + E_k^{up}(\tau) \leqslant E^{max}, \quad \forall k, \tau.
\end{aligned} \tag{17}$$

In this paper, we resort to the DRL algorithm to solve SUB1. The DRL algorithm can guide the agent to take better actions by perceiving the state of the environment, thereby achieving more excellent benefits. To implement the algorithm, we act the server as the agent and model the client scheduling as a Markov decision process (MDP). The main elements of MDP can be designed as follows:

- State: state provides feedback to the agent and produces different state transitions based on the agent's decisions.

Considering the time delay is closely related to the current channel state, the client's reputation greatly influences the scheduling decision. So, we define environment state in epoch τ as $\boldsymbol{s(\tau)} = [\boldsymbol{s_1(\tau)}, \ldots, \boldsymbol{s_k(\tau)}, \ldots, \boldsymbol{s_K(\tau)}]$, where $\boldsymbol{s_k(\tau)} = [h_k(\tau), RE_k(\tau), f_k^{ava}(\tau)]$ represents the state information of the client k in epoch τ, where $f_k^{ava}(\tau)$ is the CPU resources available to the client k.

- Action: let $\mathcal{N}(\tau)$ be the set of schedulable clients. We need to select N clients from K clients to participate in each epoch. This operation will result in a huge action space $\binom{K}{N}$ and seriously increase the algorithm's complexity. Therefore, we propose to represent the action space as K continuous probability variable, denoted by $\boldsymbol{a(\tau)} = (a_k(\tau))$, where $a_k(\tau)$ indicates the probability of client k being selected. Combined with reputation constraints, we derive a new probability vector $\upsilon_k(\tau)$ for the client k.

$$\upsilon_k(\tau) = \begin{cases} 0, & \text{if k} \notin \mathcal{A}(\tau), \\ \dfrac{a_k(\tau) RE_k(\tau)}{\sum_{k \in \mathcal{A}(\tau)} a_k(\tau) RE_k(\tau)}, & \text{else}, \end{cases} \tag{18}$$

where $\mathcal{A}(\tau) = \{k | RE_k(\tau) > RE_k^{require}\}$ represents the available clients set.

- Reward: reward is feedback signals received by agents when executing actions in the environment, which are used to evaluate the decision quality of intelligent agents. We aim to select clients with less time delay for FL. Thus, the reward function is

$$r(\tau) = -t^{total}(\tau). \tag{19}$$

At each epoch τ, the agent will get a state signal $\boldsymbol{s(\tau)}$ from the environment. Then, the agent conducts an optimal action $\boldsymbol{a(\tau)}$ in accord with the policy π. Next, The environment will provide immediate feedback $r(\tau)$ to the agent and enter the next moment. In the end, MDP consists of the sequence $\{\boldsymbol{s(1)}, \boldsymbol{a(1)}, r(1)), \boldsymbol{s(2)}, \ldots, \boldsymbol{s(\tau)}, \boldsymbol{a(\tau)}, r(\tau))\}$. The goal of DRL is to obtain the optimal policy π, which maximize the discounted cumulative reward $O_\tau = \sum_{l=0}^{T} \gamma^l r(\tau + l)$, where γ is the discount factor. To obtain unbiased rewards for epoch τ, we set the discount factor to 1. We apply the Proximal Policy Optimization (PPO) algorithm to achieve this goal. PPO is a viral DRL algorithm based on actor-critic (AC) architecture [26]. By restricting the size of policy updates by adjusting the target function, PPO ensures that the new policy does not deviate too much from the old one. This bounded strategy update helps to keep the training process stable.

Fig. 1 shows our AC architecture, including an Actor network and a Critic network. The Critic network evaluates the performance of the current state and outputs a state value to guide the Actor network. The input to the Actor network is the current state at the moment, and the output is the action value. After the action is executed, the environment will provide feedback on reward information and the information from the next moment. Status, action, and reward information will be saved in a replay buffer. Until the size of the replay buffer reaches ψ, two networks will update their network parameter based on the replay buffer data. In the initialization phase, the network randomly outputs the actions and values. However, with the network update, the Critic network's state value is more accurate, and the output action of the Actor network is more optimal.

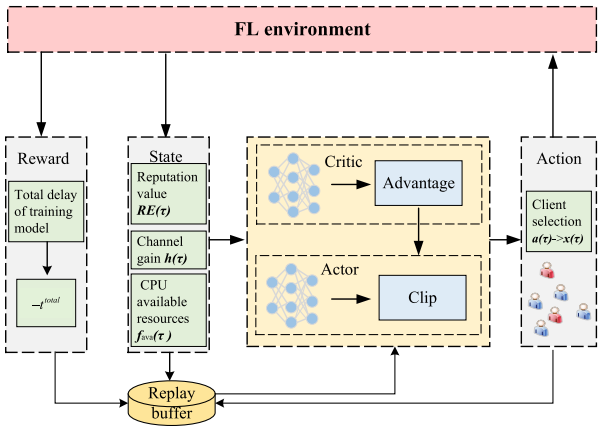


Fig. 1. DRL framework.

The Critic network learns a state value estimation function based on the data collected from the interaction between the actor and the environment. The state value estimation function provides an estimate for each state, indicating how much return is expected from the current strategy. It evaluates the effectiveness of actions in the current state, thereby assisting the actor in updating strategies. The state value function at state $\boldsymbol{s}_\tau$ is defined as follows:

$$V_\pi(\boldsymbol{s(\tau)}; \Omega) = E_\pi[r(\tau) + \gamma V_\pi(\boldsymbol{s(\tau + 1)}; \Omega)], \tag{20}$$

where Ω is the critic network parameter. $V_\pi(\boldsymbol{s(\tau + 1)})$ can determine the current state value and guide the actor to update through temporal differential error.

The Actor network learns an approximate strategy that maximizes the long-term cumulative reward for the selected action under the current strategy. It receives the current state as input, outputs the probability distribution of the agent choosing different actions in each state, and controls the magnitude of policy updates by comparing the similarity between old and new policies. Define $\Gamma^{clip}(\theta)$ as the loss function of the Actor network:

$$\Gamma^{clip}(\theta) = E[min(\Pi_\tau(\theta) \hat{A}_\tau^{GAE(\gamma,\lambda)}, clip(\Pi_\tau(\theta), 1 - \iota, 1 + \iota) \cdot \hat{A}_\tau^{GAE(\gamma,\lambda)})], \tag{21}$$

where $\Pi_\tau(\theta) = \frac{\pi_\theta(\boldsymbol{a(\tau)}|\boldsymbol{s(\tau)})}{\pi_{\theta_{old}}(\boldsymbol{a(\tau)}|\boldsymbol{s(\tau)})}$ represents the ratio of the probability of executing $\boldsymbol{a(\tau)}$ under the new strategy θ and the old strategy θ_{old}, $clip(\Pi_\tau(\theta), 1-\iota, 1+\iota)$ means that $\Pi_\tau(\theta)$ has a lower bound $1-\iota$ and a higher bound $1+\iota$. If the ratio is closer to 1, the difference between the new and old strategies is smaller. And $\hat{A}_\tau^{GAE(\gamma,\lambda)}$ represents the advantage value of an action relative to the average expected value [27], which can help the agent decide which action is a better choice, denoted by:

$$\hat{A}_\tau^{GAE(\gamma,\lambda)} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta(\tau + 1), \tag{22}$$

Algorithm 2 Client Scheduling Solution Based on DRL

Input: T, $\mathcal{K}$, N, B, N_0, $\boldsymbol{h(\tau)}$, $\boldsymbol{f(\tau)}$
Output: Ω, θ

1: **Training:**
2: **for** $iter = 1, 2, 3 \ldots$ **do**
3: **for** $\tau = 1, 2, 3 \ldots, T$ **do**
4: Perceive the state of the environment $\boldsymbol{s(\tau)}$.
5: Obtain $\boldsymbol{a(\tau)}$ based on actor network. And sample randomly N clients based on $\boldsymbol{v(\tau)}$.
6: Call Algorithm 3 and Algorithm 4 to obtain the allocation of transmit rate $\boldsymbol{o(\tau)}$, bandwidth proportion $\boldsymbol{\alpha(\tau)}$, and CPU frequency $\boldsymbol{f(\tau)}$.
7: Conduct FL training process and update $\boldsymbol{RE}(\tau)$.
8: Calculate $r(\tau)$ based on (19).
9: Put $(\boldsymbol{s(\tau)}, \boldsymbol{a(\tau)}, r(\tau), V(\boldsymbol{s(\tau)}; \Omega), \boldsymbol{s(\tau+1)})$ into the replay buffer.
10: **if** the number of the replay buffer reaches ψ **then**
11: **for** $\Upsilon = 1, .., \Lambda$ **do**
12: Take the derivative of (21) and (23) to update θ and Ω.
13: **end for**
14: **end if**
15: **end for**
16: **end for**

where λ is the bias-variance trade-off factor. And $\delta(\tau)$ represents temporal differential error, denoted by $\delta(\tau) = r(\tau) + \gamma V_{\pi_\theta}(\boldsymbol{s(\tau)}; \Omega_{old}) - V_{\pi_\theta}(\boldsymbol{s(\tau)}; \Omega_{old})$, where Ω_{old} is the parameter before the update. The main target of the Critic network is to predict the expected reward value under a given state. The loss function of the Critic network is composed of the gap between the predicted value function and the actual return, which can be expressed as the Mean Squared Error (MSE) based on the state value function. The loss function of the critic network is defined as follows:

$$\Gamma(\Omega) = E[(V_\tau^{target} - V(\boldsymbol{s(\tau)}; \Omega))^2], \tag{23}$$

where $V_\tau^{target} = \hat{A}_\tau^{GAE(\gamma,\lambda)} + V(\boldsymbol{s(\tau)}; \Omega_{old})$. By adding the advantage function to the estimated value of the state value function, we can get V_τ^{target} closer to the actual value, which helps the Critic network to better estimate the state value function.

The particular algorithm is presented in Algorithm 2. In Lines 4-5, the actor network outputs $\boldsymbol{a}(\tau)$ based on $\boldsymbol{s}(\tau)$ and the server executes $\boldsymbol{x}(\tau)$ based on $\boldsymbol{a}(\tau)$. In Line 6, call algorithms to obtain the optimal $\boldsymbol{o(\tau)}, \boldsymbol{\alpha(\tau)}$ and $\boldsymbol{f(\tau)}$. In Line 7, the selected clients train locally and transmit the updated model to the server. Then, the server updates the reputation of the clients. In Lines 8-9, the agent obtains reward $r(\tau)$ and puts the tuple $(\boldsymbol{s(\tau)}, \boldsymbol{a(\tau)}, r(\tau), V(\boldsymbol{s(\tau)}; \Omega), \boldsymbol{s(\tau+1)})$ into the replay buffer. In Lines 10-14, the AC network updates θ and Ω based on the data in the replay buffer when the size reaches ψ.

B. Radio Resources Allocation

Once the client scheduling $\boldsymbol{x(\tau)}$ is obtained, the set of schedulable clients is denoted by

$$\mathcal{N}(\tau) = \{n | x_n(\tau) = 1, n \in \mathcal{K}\}. \tag{24}$$

For given $\boldsymbol{x(\tau)}$ and $\boldsymbol{f(\tau)}$, the proposed algorithm aims to minimize the total delay of the worst-outcome client for one epoch. Then, the optimization problem of bandwidth proportional division and transmit rate allocation can be written as

$$\begin{aligned}
\text{SUB2}: \min_{\boldsymbol{o(\tau)}, \boldsymbol{\alpha(\tau)}} \max_{n \in \mathcal{N}(\tau)} & \left\{ \frac{A_n}{o_n(\tau)} + \frac{Id_n C_n}{f_n(\tau)} \right\} \\
\text{s.t.} \quad & (\text{C5}): o_n(\tau) \geqslant 0, 0 \leqslant \alpha_n(\tau) \leqslant 1, \\
& \qquad \forall n \in \mathcal{N}(\tau), \tau, \\
& (\text{C6}): \sum_{n=1}^{N} \alpha_n(\tau) \leqslant 1, \quad \forall \tau, \\
& (\text{C7}): uId_n C_n f_n(\tau)^2 \\
& \quad \frac{\alpha_n(\tau) B N_0 A_n}{h_n(\tau) o_n(\tau)} (2^{\frac{o_n(\tau)}{\alpha_n(\tau) B}} - 1) \\
& \leqslant E^{max}, \quad \forall n \in \mathcal{N}(\tau), \tau.
\end{aligned} \tag{25}$$

By introducing a new variable G, we convert the primordial problem (26) into a smooth optimization problem. Specifically, the min-max problem can be recast as

$$\begin{aligned}
\min_{\boldsymbol{o(\tau)}, \boldsymbol{\alpha}(\tau)} \quad & G \\
\text{s.t.} \quad & (\text{C5}), (\text{C6}), (\text{C7}), \\
& (\text{C8}): \frac{A_n}{o_n(\tau)} + \frac{Id_n C_n}{f_n(\tau)} \leqslant G, \quad \forall n \in \mathcal{N}(\tau), \tau.
\end{aligned} \tag{26}$$

Due to the nonconvexity of (26), it is difficult to solve the optimal value through traditional methods. Thus, we introduce a new variables $\vartheta_n(\tau) = \frac{o_n(\tau)}{\alpha_n(\tau)}$, which is represented as $\boldsymbol{\vartheta(\tau)} = (\vartheta_n(\tau))$. When $\alpha_n(\tau) = 0$, we let client n not participate in the training and have $\vartheta_n(\tau) = 0$. So, (26) can be rewritten as follows:

$$\begin{aligned}
\min_{\boldsymbol{\vartheta(\tau)}, \boldsymbol{\alpha(\tau)}} \quad & G \\
s.t. \quad & (\text{C5}'): \vartheta_n(\tau) \geqslant 0, 0 \leqslant \alpha_k(\tau) \leqslant 1, \\
& \qquad \forall n \in \mathcal{N}(\tau), \tau, \\
& (\text{C6}): \sum_{n=1}^{N} \alpha_n(\tau) \leqslant 1, \quad \forall \tau, \\
& (\text{C7}): uId_n C_n f_n(\tau)^2 + \frac{B N_0 A_n}{h_n(\tau) \vartheta_n(\tau)} (2^{\frac{\vartheta_n(\tau)}{B}} - 1) \\
& \qquad \leqslant E^{max}, \quad \forall n \in \mathcal{N}(\tau), \tau, \\
& (\text{C8}): \frac{A_n}{\alpha_n(\tau) \vartheta_n(\tau)} + \frac{Id_n C_n}{f_n(\tau)} \leqslant G, \\
& \qquad \forall n \in \mathcal{N}(\tau), \tau.
\end{aligned} \tag{27}$$

Theorem 1: Optimization problem (27) is jointly convex about variables $\boldsymbol{\alpha(\tau)}$ and $\boldsymbol{\vartheta(\tau)}$.

Proof: See Appendix I.

Note that we can solve (27) in a convex optimization manner based on Theorem 1. In addition, (27) satisfies Slater's condition with zero dual gaps. Thus, we can obtain the optimal solution through Lagrangian dual decomposition. The Lagrangian form of problem (27) can be given by (28), shown at the bottom of the page, where $\xi \geqslant 0, \boldsymbol{\nu} = \{\nu_1, \nu_2, \ldots, \nu_N\} \succeq 0, \boldsymbol{\varphi} = \{\varphi_1, \varphi_2, \ldots, \varphi_N\} \succeq 0$ are dual variables, corresponding to constraints (C6), (C7), and (C8), respectively.

In order to solve the optimal solution of $\boldsymbol{\vartheta(\tau)}, \boldsymbol{\alpha(\tau)}$ and G, firstly we need to minimize $L(G, \boldsymbol{\vartheta(\tau)}, \boldsymbol{\alpha(\tau)}, \xi, \boldsymbol{\nu}, \boldsymbol{\varphi})$ with fixed $\xi, \boldsymbol{\nu}, \boldsymbol{\varphi}$:

$$\begin{aligned}\min_{\boldsymbol{\vartheta(\tau)},\boldsymbol{\alpha(\tau)},G} \; & G + \xi \sum_{n=1}^{N} \alpha_n(\tau) + \sum_{n=1}^{N} \varphi_n \Big(\frac{A_n}{\vartheta_n(\tau)\alpha_n(\tau)} - G\Big) \\ & + \sum_{n=1}^{N} \nu_n \Big(\frac{BN_0A_n}{h_n(\tau)\vartheta_n(\tau)}\big(2^{\frac{\vartheta_n(\tau)}{B}} - 1\big)\Big) \\ s.t. \quad & (\mathrm{C5}'): \vartheta_n(\tau) \geqslant 0, 0 \leqslant \alpha_n(\tau) \leqslant 1, \\ & \forall n \in \mathcal{N}(\tau), \tau. \end{aligned} \tag{29}$$

By applying the Karush-Kuhn-Tucker (KKT) condition [28], the optimal solution for the transmit rate and the bandwidth proportion can be obtained. The formula for taking the derivative of $\alpha_n(\tau)$ and $\vartheta_n(\tau)$ are as follows:

$$\frac{\partial L(G, \boldsymbol{\vartheta(\tau)}, \boldsymbol{\alpha(\tau)}, \xi, \boldsymbol{\nu}, \boldsymbol{\varphi})}{\partial \alpha_n(\tau)} = \xi - \frac{\varphi_n \frac{A_n}{\vartheta_n(\tau)}}{\alpha_n(\tau)^2} = 0. \tag{30}$$

$$\begin{aligned}\frac{\partial L(G, \boldsymbol{\vartheta(\tau)}, \boldsymbol{\alpha(\tau)}, \xi, \boldsymbol{\nu}, \boldsymbol{\varphi})}{\partial \vartheta_n(\tau)} &= -\frac{\varphi_n \frac{A_n}{\alpha_n(\tau)}}{(\vartheta_n(\tau))^2} \\ &+ \frac{\frac{B\nu_n N_0 A_n}{h_n(\tau)}\big(\frac{\vartheta_n(\tau)}{B} 2^{\frac{\vartheta_n(\tau)}{B}} \ln 2 - 2^{\frac{\vartheta_n(\tau)}{B}} + 1\big)}{(\vartheta_n(\tau))^2} = 0. \end{aligned} \tag{31}$$

1) The Optimal Solution Structure of $\alpha_n(\tau)$: By dealing the equation (30), we can obtain the optimal solution of $\alpha_n(\tau)$ as follows:

$$\alpha_n^*(\tau) = clip\Big(\sqrt{\frac{\varphi_k A_n x_n(\tau)}{\vartheta_n(\tau)\xi}}, 0, 1\Big). \tag{32}$$

2) The Optimal Solution Structure of $o_n(\tau)$: We let $c_5 = \frac{B\nu_n N_0 A_n}{h_n(\tau)}, c_6 = \varphi_n \frac{A_n}{\alpha_n(\tau)}$. Solving the equation (31), we can get the optimal solution of $o_n(\tau)$.

$$o_n^*(\tau) = \vartheta_n^*(\tau)\alpha_n^*(\tau) = max\left\{\alpha_n^*(\tau) B \log_2 \frac{\beta}{W(\frac{\beta}{e})}, 0\right\}, \tag{33}$$

where $\beta = \frac{c_6}{c_5} - 1$ and $W(\cdot)$ is Lambert function. The specific process of solving the optimal solution of $o_n(\tau)$ is shown in Appendix II.

3) The Optimal Solution Structure of G: For (26), We can only obtain the solution for $\boldsymbol{o(\tau)}$ and $\boldsymbol{\alpha(\tau)}$ when G is given. So, we need to discuss the solution of G further.

$$\begin{aligned} \min_{G} \quad & (1 - \sum_{n=1}^{N} \varphi_n) G \\ s.t. \quad & (\mathrm{C8}). \end{aligned} \tag{34}$$

Then, the optimal solution of G can be designed by

$$G^* = \begin{cases} max_n \left\{\frac{A_n}{o_n^*(\tau)} + \frac{Id_nC_n}{f_n(\tau)}\right\}, \text{if } \sum_{n=1}^{N} \varphi_n \leqslant 1, \\ +\infty, \qquad\qquad \text{else.} \end{cases} \tag{35}$$

4) Lagrange Multipliers Update: From (32) and (33), we can obtain the optimal bandwidth proportion and transmit rate with fixed $\xi, \boldsymbol{\nu}$, and $\boldsymbol{\varphi}$ only. Thus, we need to get the value of dual variables next. According to Lagrangian duality theory, we can obtain the value of $\xi, \boldsymbol{\nu}$, and $\boldsymbol{\varphi}$ by solving the dual problem of the problem (27), which is represented by

$$\begin{aligned} \max_{\xi,\boldsymbol{\nu},\boldsymbol{\varphi}} \quad & F(\xi, \boldsymbol{\nu}, \boldsymbol{\varphi}) \\ s.t. \quad & \xi \geqslant 0, \boldsymbol{\nu} \succeq 0, \boldsymbol{\varphi} \succeq 0, \end{aligned} \tag{36}$$

where $F(\xi, \boldsymbol{\nu}, \boldsymbol{\varphi}) = min_{G, \boldsymbol{\vartheta(t)}, \boldsymbol{\alpha(t)}} L(G, \boldsymbol{\vartheta(t)}, \boldsymbol{\alpha(t)}, \xi, \boldsymbol{\nu}, \boldsymbol{\varphi})$. From (28), Since $F(\xi, \boldsymbol{\nu}, \boldsymbol{\varphi})$ are linear function relative to $\xi, \boldsymbol{\nu}$, and $\boldsymbol{\varphi}$, it can be proven that (36) is constant convex. So, (36) can be solved using the subgradient projection theory. Then, we can obtain the updates of the dual variables as follows:

$$\begin{aligned} \xi(\varpi + 1) &= [\xi(\varpi) + z_1(\varpi)\nabla\xi(\varpi)]^+, \\ &+ \nu_n(\varpi + 1) = [\nu_n(\varpi) + z_2(\varpi)\nabla\nu_n(\varpi)]^+, \\ &+ \varphi_n(\varpi + 1) = [\varphi_n(\varpi) + z_3(\varpi)\nabla\varphi_n(\varpi)]^+, \end{aligned} \tag{37}$$

where $[num]^+ \triangleq max\{0, num\}$, ϖ is the subscript of the round of iterations, $z_1(\varpi), z_2(\varpi)$ and $z_3(\varpi)$ are very small positive step size. We set $z_1(\varpi) = z_2(\varpi) = z_3(\varpi) = \frac{0.1}{\varpi}$ [29]. And $\nabla\xi(\varpi), \nabla\nu_n(\varpi), \nabla\varphi_n(\varpi)$ are represented by

$$\begin{aligned} \nabla\xi(\varpi) &= \sum_{n=1}^{N} \alpha_n^*(\tau) - 1. \\ \nabla\nu_k(\varpi) &= uId_nC_nf_n(\tau)^2 + \frac{BN_0A_n}{h_n(\tau)\vartheta_n^*(\tau)}\big(2^{\frac{\vartheta_n^*(\tau)}{B}} - 1\big) - E^{max}. \\ \nabla\varphi_n(\varpi) &= \frac{Id_nC_n}{f_n(\tau)} + \frac{A_n}{\vartheta_n^*(\tau)\alpha_n^*(\tau)} - G^*. \end{aligned} \tag{38}$$

The detailed algorithm to get the optimal solution of transmit rate is shown in Algorithm 3. In Lines 2-11, the algorithm iterates ϖ_{max} round at most. At each iteration, the algorithm first obtains $o_n^*(\tau)$, $\alpha_n^*(\tau)$ and G^* based on the fixed dual variable $\xi(\varpi), \boldsymbol{\nu}(\varpi), \boldsymbol{\varphi}(\varpi)$, and then obtains new $\xi(\varpi + 1), \boldsymbol{\nu}(\varpi + 1), \boldsymbol{\varphi}(\varpi + 1)$ based on $\vartheta_n^*(\tau)$, $\alpha_n^*(\tau)$ and G^*.

$$\begin{aligned} L(G, \boldsymbol{\vartheta(\tau)}, \boldsymbol{\alpha(\tau)}, \xi, \boldsymbol{\nu}, \boldsymbol{\varphi}) &= G + \xi\Big(\sum_{n=1}^{N} \alpha_n(\tau) - 1\Big) + \sum_{n=1}^{N} \varphi_n\Big(\frac{A_n}{\vartheta_n(\tau)\alpha_n(\tau)} + \frac{Id_nC_n}{f_n(\tau)} - G\Big) \\ &+ \sum_{n=1}^{N} \nu_n\Big(uId_nC_nf_n(\tau)^2 + \frac{BN_0A_n}{h_n(\tau)\vartheta_n(\tau)}\big(2^{\frac{\vartheta_n(\tau)}{B}} - 1\big) - E^{max}\Big), \forall n \in \mathcal{N}(\tau), \tau. \end{aligned} \tag{28}$$

Algorithm 3 Radio Resource Allocation Algorithm

Input: $\xi(0), \boldsymbol{\nu(0)}, \boldsymbol{\varphi(0)}$,the maximum number of iterations ϖ_{max}, and the specified precision ϵ.
Output: $o_n^*(\tau), \alpha_n^*(\tau)$

```
Initialize ϖ = 0.
while ϖ < ϖ_max do
    Perform bisection search algorithm between [0,1] to obtain.
    Substitute the dual variables ξ(ϖ), ν(ϖ), φ(ϖ) into (32) and (33) to obtain o_n^*(τ) and α_n^*(τ).
    Substitute o_n^*(τ) and α_n^*(τ) into (35) to obtain G^*.
    Update new dual variables ξ(ϖ+1), ν(ϖ+1), φ(ϖ+1) based on (37).
    if ξ(ϖ+1) − ξ(ϖ) < ε, ||ν(ϖ+1) − ν(ϖ)|| < ε and ||φ(ϖ+1) − φ(ϖ)|| < ε then
        o_n^*(τ) = o_n(τ), α_n^*(τ) = α_n(τ).
        Break.
    else
        ϖ = ϖ + 1.
    end if
end while
```

C. Adaptive Harmony Algorithm for CPU Frequency Allocation

To find the optimal solution for CPU frequency, We design a heuristic adaptive search algorithm under given client scheduling $\boldsymbol{x(\tau)}$, bandwidth allocation $\boldsymbol{\alpha(\tau)}$, and transmit rate $\boldsymbol{o(\tau)}$. For given $\boldsymbol{x(\tau)}$, $\boldsymbol{o(\tau)}$, and $\boldsymbol{\alpha(\tau)}$, the CPU frequency optimization problem is written as

$$\begin{aligned}\text{SUB3}: \min_{\boldsymbol{f(\tau)}} \max_{n\in\mathcal{N}(\tau)} & \left\{\frac{A_n}{o_n(\tau)}+\frac{Id_nC_n}{f_n(\tau)}\right\} \\ s.t. \quad & (\text{C4}): f_n^{min}\leqslant f_n(\tau)\leqslant f_n^{max} \quad \forall n\in\mathcal{N}(\tau),\tau, \\ & (\text{C7}): uId_nC_nf_n(\tau)^2+E_n^{up}(\tau)\leqslant E^{max}, \\ & \qquad\qquad \forall n\in\mathcal{N}(\tau),\tau.\end{aligned} \tag{39}$$

For energy constraint (C7), we transfer the problem using a penalty factor.

$$\begin{aligned}\text{SUB3}: \min_{\boldsymbol{f(\tau)}} \quad & M(\boldsymbol{f(\tau)}) \\ s.t. \quad & (\text{C4}): f_n^{min}\leqslant f_n(\tau)\leqslant f_n^{max}, \quad \forall n\in\mathcal{N}(\tau),\tau,\end{aligned} \tag{40}$$

where

$$\begin{aligned}M(\boldsymbol{f(\tau)}) = & \max_{n\in\mathcal{N}}\left\{\frac{A_n}{o_n(\tau)}+\frac{Id_n(\tau)C_n}{f_n(\tau)}\right\} \\ & +\aleph\Big(\sum_{n\in\mathcal{N}(\tau)} max\{0, uId_n(\tau)C_nf_n(\tau)^2 \\ & +E_n^{up}(\tau)-E^{max}\}\Big),\end{aligned} \tag{41}$$

where $\aleph \to +\infty$. According to the theory of the exterior penalty method, (39) and (40) have equivalent solutions. The adaptive harmony search algorithm includes the following parameters [30]:

- NI: it is the maximum number of iterations, indexed by l.
- HMS: it means the number of solutions vectors in harmony memory (HM), where harmony memory (HM) can be expressed as a matrix.

$$\text{HM}=\begin{bmatrix} f_1^1(\tau) & \cdots & f_N^1(\tau) \\ \vdots & \ddots & \vdots \\ f_1^{\text{HMS}}(\tau) & \cdots & f_N^{\text{HMS}}(\tau)\end{bmatrix}, \tag{42}$$

  where $\boldsymbol{f^i(\tau)} = [f_1^i(\tau), f_2^i(\tau), \ldots, f_N^i(\tau)]$ represent the i-th solution vector, that is, harmony variable.
- HMCR: it is the harmony memory considering the rate, following the normal distribution. Generate a random number r_1 between [0,1]. When r_1 is less than HMCR, an old harmony variable is randomly selected; otherwise, a new one is randomly generated.
- PAR: it expresses the pitch adjusting rate. Generally, it obeys the Gaussian distribution.
- BW: it represents the distance bandwidth, which indicates the adjustment amplitude of the new harmonic variable. Generate a random number r_3 between [0, 1]. If $r_3 < 0.5$, add r_4*BW to the new harmony variable. Otherwise, subtract r_4*BW, where r_4 is a random number distributed in [0,1]. To enhance the adaptive efficiency of the algorithm, we dynamically set the value of BW as in [30].

$$\begin{aligned}&\text{BW}(l) \\ &=\begin{cases}\text{BW}^{max}-2l\dfrac{\text{BW}^{max}-\text{BW}^{min}}{\text{NI}}, & \text{if } l<\text{NI}/2, \\ \text{BW}^{min}, & \text{else.}\end{cases}\end{aligned} \tag{43}$$

The particulars of the CPU frequency allocation algorithm are shown in Algorithm 4. In Line 3, generate harmony memory considering rate and pitch adjusting rate. In Lines 4-17, generate the new solution $\boldsymbol{f^{new}(\tau)}$. In Lines 18-20, move $\boldsymbol{f^{new}(\tau)}$ in harmony memory and remove the worst solution vector. In Lines 21-25, regenerate the distribution of PAR and HMCR based on historical results.

IV. Numerical Analysis

In this section, lots of experiments were done to verify the efficiency of the designed algorithm.

A. Simulation Settings

We consider a FL network that clients distribute in a cell with a radius of $R = 500$ m and a server located in the center. In the simulation, we model the channel gain as $h_k(\tau) = hg_k(\tau)(\frac{d_0}{dis_k})^a$, where dis_k is the distance between the server and client k, $g_k(\tau)$ is the small-scale path loss and $\frac{d_0}{dis_k}$ is the large-scale path loss. The other simulation parameters are set as Table I.

To testify to the robustness of the proposed algorithm, we will consider three datasets:

- **MNIST**: MNIST comprises 60000 training and 10000 test samples. To simulate non-iid, we divide 60000 training images into 50 shards, each containing 1200 images. We select 1-5 shards for clients.

Algorithm 4 CPU Frequency Allocation Algorithm

Input: $\mu^{\mathrm{PAR}}, \sigma^{\mathrm{PAR}}, \mu^{\mathrm{HMCR}}, \mathrm{BW}^{min}, \mathrm{BW}^{max}, \sigma^{\mathrm{HMCR}}, \mathrm{NI}$, FLAG, $flag = 1$.

Output: $f_n^*(\tau)$

Initialize HM, $l = 0$.
while $l <$ NI **do**
  Generate HMCR and PAR according to μ^{HMCR}, σ^{HMCR}, μ^{PAR} and σ^{PAR}
  for n in $\mathcal{N}(t)$ **do**
    if $r_1 \leqslant$ HMCR **then**
      $f_n^{new}(\tau) = f_n^i(\tau)$, where $f_n^i(\tau)$ is selected from HM randomly.
      if $r_2 \leqslant$ PAR **then**
        if $r_3 \leqslant 0.5$ **then**
          $f_n^{new}(\tau) = f_n^{new}(\tau) - r_4 * \mathrm{BW}(l)$
        else
          $f_n^{new}(\tau) = f_n^{new}(\tau) + r_4 * \mathrm{BW}(l)$
        end if
      end if
    else
      $f_n^{new}(\tau) = f_k^{min} + r_4 * (f_k^{max} - f_k^{min})$
    end if
  end for
  if $M(\boldsymbol{f^{new}(\tau)}) < M(\boldsymbol{f^{worst}(\tau)})$ **then**
    Move in $\boldsymbol{f^{new}(\tau)}$ and remove $\boldsymbol{f^{worst}(\tau)}$.
  end if
  if $flag ==$ FLAG **then**
    Assess μ^{PAR} and μ^{HMCR} based on PAR and HMCR and let $flag = 1$.
  else
    $flag = flag + 1$
  end if
end while

TABLE I
SUMMARY OF THE SIMULATION PARAMETERS

Parameters	Values
the number clients K	10 or 100 [1]
the number of selected clients N	5 or 20,30,40,50,60,70
transmit data size A_k	2.5×10^4 bit
power coefficient u	10^{-26}
cycle times processing a data sample C_k	10^4 (cycle/ sample)
energy consumption threshold E^{max}	0.25∼0.5 J [31]
white noise power N_0	5×10^{-4} w [32]
global communication round T	80
local training round I	5
bandwidth B	1 MHz [33]
reputation threshold $RE_k^{require}$	0.5
distance dis_k	50∼500m [34]
path loss constant h	1
reference distance d_0	1
path loss exponent a	2

- **CIFAR10**: There are more than ten types of CIFAR data, including tens of thousands of training images and test images in the data set. For non-IID, we divide the 50,000 training images into 50 shards, each shard containing 1,000 images. We choose 1-5 shards for our customers.
- **FASHION-MNIST**: FASHION-MNIST clones all external features of MNIST. The difference is that Fashion MNIST is no longer an abstract symbol but a more concrete human necessity, such as Trousers and Ankle boots. The setup for non-iid is the same as that of MNIST.

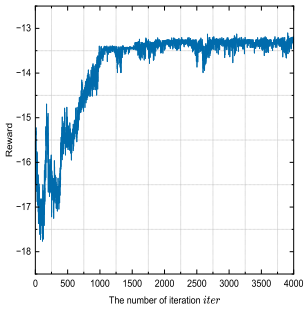


Fig. 2. Convergence of algorithm.

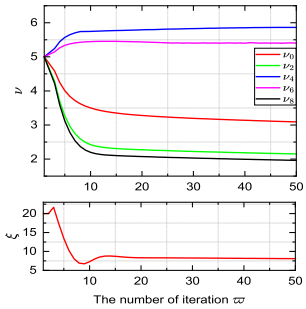


Fig. 3. Convergence of algorithm 3.

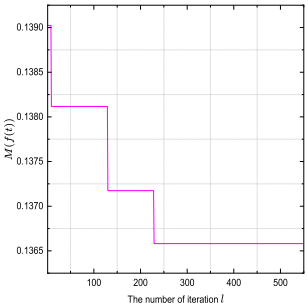


Fig. 4. Convergence of algorithm 4.

B. Convergence Performance of the Proposed Algorithm

In Fig. 2, we show the convergence of Algorithm 2. Algorithm 2 is the main algorithm framework of this paper, whose reward is the target value of the optimization problem. From Fig. 2, it can be observed that the reward eventually converges under slight jitters. Fig. 3 displays the dual variables $\boldsymbol{\nu} = \{\nu_k\}$ and ξ versus iteration ϖ to show the convergence of Algorithm 3. It can be found that the algorithm has a rapid convergence rate. In Fig. 4, we show the convergence of the CPU frequency allocation algorithm. We can find that the

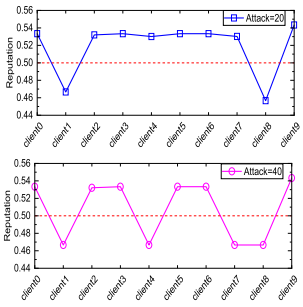


Fig. 5. Clients' reputation when $RE_k^{require} = 0.5$.

algorithm converges about 200 rounds, which also has a rapid convergence rate.

C. Verify the Validity of the Reputation Model

We simulate data attack by flipping all labels of the client to label 0 as in [35]. We conduct training on both containing 20% malicious clients and containing 40% malicious clients and use $Attack = 20$ and $Attack = 40$ below to represent both cases.

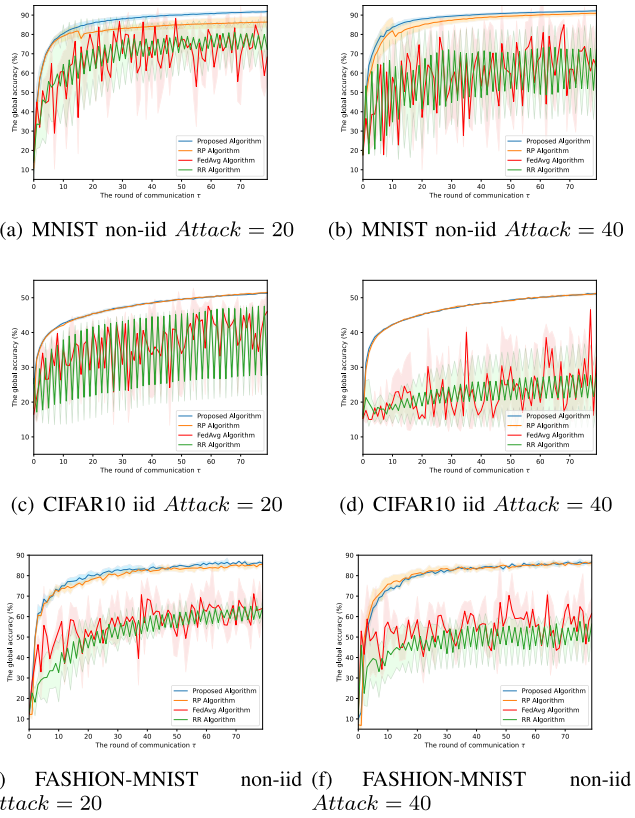
To simulate $Attack = 20$ and $Attack = 40$, we randomly sample 20% clients and 40% clients as malicious clients. In the first FL round, all clients have the same reputation value of 0.5. After training, we can get all clients' real reputations. Fig. 5 shows clients' updated reputation after the first FL epoch. From Fig. 5, it can be observed that our model has indeed identified malicious clients (reputation less than $RE_k^{require}$). In the subsequent training rounds, clients with a reputation of less than $RE_k^{require}$ will forever lose the opportunity to be scheduled. In this way, our reputation model, based on the subjective logic model, can identify malicious clients efficiently.

D. Performance Comparison of the Proposed Algorithm and Other Existing Algorithms

To demonstrate the effectiveness of the client scheduling of the proposed scheme under data attack, we set up the following three benchmark schemes:

- RP [15]: Compared to the proposed scheme, this approach utilizes the beta trust model to establish trust, and the scheduling client is chosen based on this model.
- FedAvg [20]: This scheme has the same optimization variables, except the client selection mechanism adopts a random method.
- RR [36]: The client selection mechanism of this scheme adopts a polling mechanism.

We set independent and identically distributed (IID) and non-IID for these three datasets, but due to space limitations, this paper only presents some of the results. Fig. 6 shows the convergence of global accuracy with the number of global epochs. The solid curve represents the average of 10 experiments, and the shaded envelope represents the upper and lower accuracy bounds. From Fig.6, we can see that the convergence of FedAvg and RR is severely compromised as the percentage of malicious attacks increases. Nevertheless,

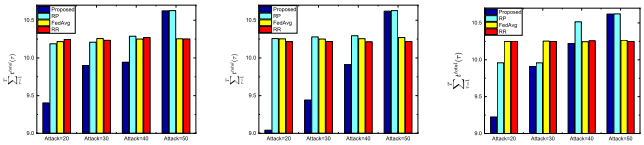


(a) MNIST non-iid $Attack = 20$ (b) MNIST non-iid $Attack = 40$

(c) CIFAR10 iid $Attack = 20$ (d) CIFAR10 iid $Attack = 40$

(e) FASHION-MNIST non-iid $Attack = 20$ (f) FASHION-MNIST non-iid $Attack = 40$

Fig. 6. Comparisons of the test accuracy of FL among the proposed algorithm, RP, FedAvg, and RR under$Attack = 20$ and $Attack = 40$ with $K = 10$, $N = 5$, $T = 80$.



(a) MNIST non-iid (b) CIFAR10 iid (c) FASHION-MNIST non-iid

Fig. 7. Comparisons of the total time delay of T epochs among the proposed algorithm, RP, FedAvg, and RR under different $Attack$ with$K = 10$, $N = 5$, $T = 10$.

both the algorithm and RP still maintain excellent convergence. The reason for this phenomenon is that FedAvg and RR cannot identify the client that contains the malicious data attack, resulting in the model being contaminated. Moreover, the proposed algorithm can reach a faster convergence rate and better convergence precision than RP under a non-iid setting. Hence, our algorithm is more applicable and efficient in ensuring convergence performance.

In Fig. 7, we show the comparisons of the total time delay of T epochs among the proposed Algorithm, RP, FedAvg, and RR. In order to observe the changes in time delay more intuitively, we add two settings: $Attack = 30$ and $Attack = 50$. From Fig. 7, we observe that the proposed algorithm is significantly different in $Attack = 20, 30, 40, 50$ scenarios compared with other algorithms, especially in Fig. 7(b). This phenomenon is because the proposed algorithm will prioritize clients with low latency while ensuring quality. As malicious clients increase, the number of available clients decreases. The

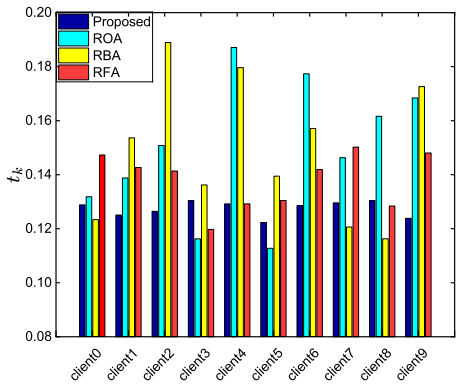


Fig. 8. Comparisons of the time delay of each client among the Proposed algorithm, ROA, RBA, and RFA under $E^{max} = 0.35J$ and $K = 10$, $N = 10$.

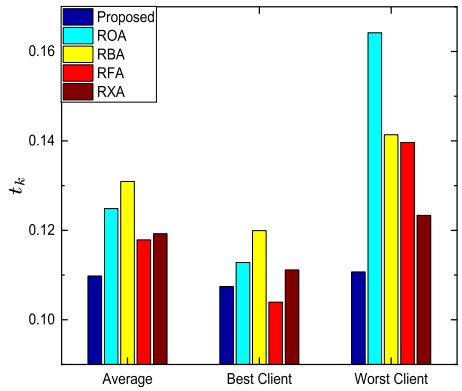


Fig. 9. Comparisons of the time delay of the average value, the best client, and the worst client among the Proposed algorithm, ROA, RBA, RFA and RXA under $E^{max} = 0.35J$ and $K = 10, N = 5$.

proposed algorithm selects clients with lower latency from the remaining clients, and RP always favors clients with good reputations. Therefore, our algorithm is very efficient and balances convergence performance and latency.

To highlight the efficiency of our resource optimization, we set up the following scheme for comparison.

- ROA: The scheme randomly sets the transmit rate and optimizes the client scheduling variable, CPU frequency variable, and bandwidth proportion variable.
- RBA: The scheme randomly sets bandwidth proportion and optimizes the client scheduling variable, CPU frequency variable, and transmit rate variable.
- RFA: The scheme randomly sets CPU frequency and optimizes the client scheduling variable, bandwidth proportion variable, and transmit rate variable.
- RXA: The scheme randomly sets client scheduling and optimizes the CPU frequency variable, bandwidth proportion variable, and transmit rate variable.

Fig. 8 compares the time delay of each client among the proposed algorithm, ROA, RBA, and RFA. From Fig. 8, it can be found that the time delay of the other three algorithms has a significant difference between clients, while the proposed algorithm balances the time delay of each client. In Fig. 9, we further compare the clients' average time delay, best client's time delay, and worst client's time delay among the proposed algorithm, ROA, RBA, RFA, and RXA. We can visually observe that the time delay of ROA and RBA exceeds the RFA and RXA, so the optimation of transmit rate and bandwidth proportion is more beneficial in reducing time delay than CPU frequency and client scheduling. Additionally, compared to the proposed algorithm, ROA sacrifices the efficiency of other clients to improve the average latency, which will cause a serious waste of resources. Moreover, the proposed algorithm has certain advantages over other algorithms. Specifically, the proposed algorithm has a subtle difference in the time delay between the best and worst clients, balancing the time delay of each client and achieving efficiency and fairness. This experiment's results verify the superiority of the proposed algorithm in achieving balanced and fair resource allocation.

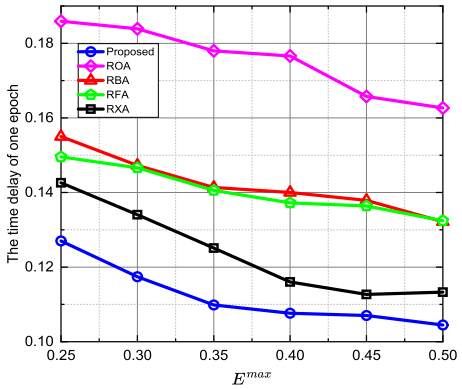


Fig. 10. Impact of maximum transmit energy E^{max} on time delay under $K = 10$ and $N = 5$.

E. Impact of Maximum Transmit Energy E^{max} on Time Delay

Fig. 10 illustrates how the maximum transmit energy E^{max} affects the worst client time delay under the contrastive schemes. When the client's available energy increases, the worst client's time delay for each scheme will correspondingly decrease. Moreover, we also observe that when the energy is below a certain threshold, the worst client time delay decreases significantly with the increase of E^{max}. Until E^{max} exceeds a certain threshold, the worst client time delay decreases slowly with the increase of E^{max}. Furthermore, the proposed algorithm consistently outperforms other algorithms.

F. Impact of the Number of Selecting Clients N on Time Delay and FL

To consider the impact of the number of selecting clients N on time delay, we expand the total number of clients to 100 and conduct $N = 20, 30, 40, 50, 60, 70$. Fig. 11 provides time delay under different N among the proposed algorithm, ROA, RBA, RFA, RXA. From Fig. 11, we find that scheduling more clients will result in higher time delays due to limited bandwidth resources. However, as N increases, the proposed algorithm consistently maintains the lowest delay compared to other existing resource optimization. In addition, the proposed algorithm can reduce the delay increase as much as possible so that the delay increase with N is slower.

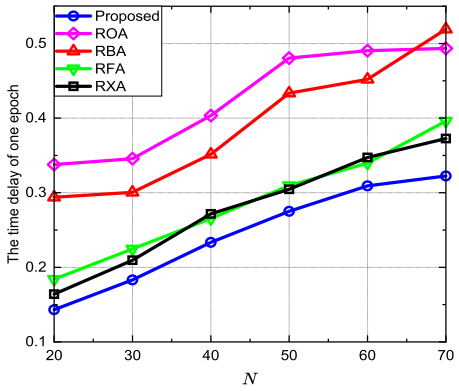


Fig. 11. Impact of the number of selecting clients N on time delay under $K = 100$ and $E^{max} = 0.5J$.

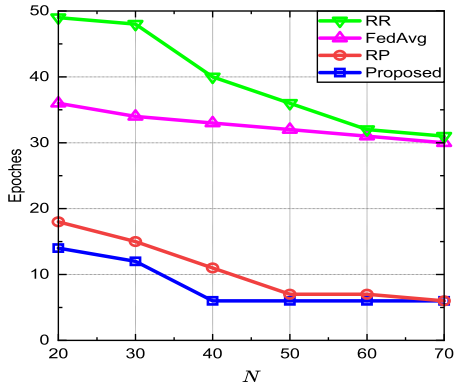


Fig. 12. Impact of the number of selecting clients N on the number of communication rounds to achieve the target accuracy 80%.

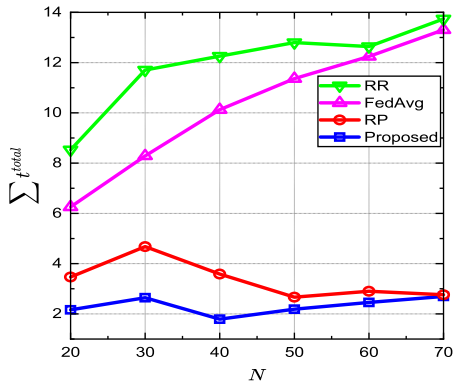


Fig. 13. Impact of the number of selecting clients N on the time delay of total epoch to achieve the target accuracy 80%.

To consider the impact of the number of selecting clients N on FL, we apply the above experimental apparatus to RP, FedAvg, and RR. Due to limited space, we only present the training results on the non-iid MNIST dataset. After massive experiments, we find that FedAvg's best convergence accuracy in the case of attack is 80%. Fig. 12 and Fig. 13 record the number of communication rounds and the time delay required to achieve 80% accuracy, respectively. From Fig. 12, it can be found that scheduling more clients can reduce the number of communication rounds to achieve the target accuracy. Besides, we also find that when N reaches a specific value, the number of communication rounds to achieve the objective accuracy is approaching stability. Fig. 13 shows that the time delay increases with the increase of N, which is consistent with the test result in Fig. 11. Additionally, Fig. 12 and Fig. 13 also indicate that the proposed algorithm can achieve objective accuracy with lower communication rounds and delay.

V. Conclusion

In this paper, we proposed a reputation-based scheduling framework to minimize the total train delay in wireless FL systems. Firstly, we have designed a reputation model in a wireless FL system to evaluate the client's data quality. Then, we jointly optimized client scheduling, transmit rate, bandwidth proportion, and CPU frequency allocation to minimize the total delay. To solve this combinatorial optimization problem, we have decoupled the optimization variables and transformed them into three sub-problems. Then, we designed efficient algorithms for each subproblem. Ultimately, the results manifest that the developed algorithm has excellent convergence speed and can ensure delay fairness among clients. Meanwhile, convergence accuracy and total delay can be achieved in balance. In the future, we will respond to more malicious attacks and strengthen our robustness.

Appendix I
Proof of Theorem 1

(C5), (C6) in (27) are linear constraints. For (C7), we need to make some changes. Let $c_1 = \frac{N_0 A_n}{h_n(\tau)}, c_2 = uId_nC_nf_n(\tau)^2 - E^{max}$, where $c_1(c_1 > 0)$ and c_2 are constants. Then, (C7) can be expressed by $f(\vartheta_n(\tau)) = \frac{c_1 B}{\vartheta_n(\tau)}(2^{\frac{\vartheta_n(\tau)}{B}} - 1) + c_2, \vartheta_n(\tau) \geqslant 0$. According to convex optimization theory [37], the convexity of a unary function depends on whether its second-order derivative is greater than 0. The second-order derivative of $f(\vartheta_n(\tau))$ is expressed by $\frac{d^2 f(\vartheta_n(\tau))}{\vartheta_n(\tau)^2} = c_1\Big(2^{\vartheta_n(\tau)/B}\big(\frac{\vartheta_n(\tau)}{B}\ln 2 - 1\big)^2 + 2^{\vartheta_n(\tau)/B} - 2\Big)/(\vartheta_n(\tau)/B)^3$. Next, we analyze the sign of $\frac{d^2 f(\vartheta_n(\tau))}{\vartheta_n(\tau)^2}$. It is obvious that the sign of $\frac{d^2 f(\vartheta_n(\tau))}{\vartheta_n(\tau)^2}$ depends on the sign of $\pounds(\vartheta_n(\tau)) = 2^{\vartheta_n(\tau)/B}\big(\frac{\vartheta_n(\tau)}{B}\ln 2 - 1\big)^2 + 2^{\vartheta_n(\tau)/B} - 2$. The first-order derivative of $\pounds(\vartheta_n(\tau))$ is expressed by $\frac{d\pounds(\vartheta_n(\tau))}{\vartheta_n(\tau)} = (\vartheta_n(\tau)/B)^3 2^{\vartheta_n(\tau)/B}\ln 2^3 \geqslant 0$. So $\pounds(\vartheta_n(\tau))$ increases with $\vartheta_n(\tau)$, and has the minimum value 0 in $\vartheta_n(\tau) = 0$. Then, we have $\frac{d^2 f(\vartheta_n(\tau))}{\vartheta_n(\tau)^2} \geqslant 0$. Hence, (C7) is a convex function of $\vartheta_n(\tau)$. For (C8), we let $c_3 = A_n, c_4 = \frac{Id_nC_n}{f_n(\tau)} - G$, where $c_3(c_3 > 0)$ and c_4 are constants. Then, (C8) can be expressed by $f(\alpha_n(\tau), \vartheta_n(\tau)) = \frac{c_3}{\alpha_n(\tau)\vartheta_n(\tau)} - c_4, 0 \leqslant \alpha_n(\tau) \leqslant 1, \vartheta_n(\tau) \geqslant 0$. We can get the hessian matrix of $f(\alpha_n(\tau), \vartheta_n(\tau))$, denoted by $H = \begin{pmatrix} 2/\alpha_n(\tau)^3\vartheta_n(\tau) & 1/\alpha_n(\tau)^2\vartheta_n(\tau)^2 \\ 1/\alpha_n(\tau)^2\vartheta_n(\tau)^2 & 2/\alpha_n(\tau)\vartheta_n(\tau)^3 \end{pmatrix}$. It is effortless to prove that H is positive definite by calculating that every principal minor of order is greater than 0. Therefore, we can obtain that (C8) is a convex constraint, and then we demonstrate that (27) is a convex optimization problem.

Appendix II
Proof of the Optimal Solution $o_k^*(\tau)$

Let $\chi = \frac{\vartheta_n(\tau)}{B}$, (31) can be rewritten as $\chi 2^\chi \ln 2 - 2^\chi = \frac{c_6}{c_5} - 1$, where $c_5 = \frac{B\nu_n N_0 A_n}{h_n(\tau)}, c_6 = \varphi_n \frac{A_n}{\alpha_n(\tau)}$.

Let $\frac{c_6}{c_5} - 1 = \beta$. We have $2^\chi \ln 2^\chi - 2^\chi = \beta$. Let $2^\chi = m$. Then, (31) can be rewritten as $m \ln m - m = \beta$. Further transformation leads to $\frac{m}{e} \ln \frac{m}{e} = \frac{\beta}{e}$. $\frac{m}{e} \ln \frac{m}{e} = \frac{\beta}{e}$ can be converted to $\frac{\beta}{e} = \frac{\beta}{m} e^{\frac{\beta}{m}}$. Since $xe^x = y, x = W(y)$, we can obtain $m = \frac{\beta}{W(\frac{\beta}{e})}$. Bring the value of m, χ into $2^\chi = m$, we can get $\vartheta_n(\tau) = B \log_2 \frac{\beta}{W(\frac{\beta}{e})}$. In the end, we can gain the optimal solution $o_n^*(\tau) = \alpha_n^*(\tau) B \log_2 \frac{\beta}{W(\frac{\beta}{e})}$.

REFERENCES

[1] J. Feng, L. Liu, Q. Pei, and K. Li, "Min-max cost optimization for efficient hierarchical federated learning in wireless edge networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 11, pp. 2687–2700, Nov. 2022.

[2] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 72–80, Apr. 2020.

[3] G. Xia, J. Chen, C. Yu, and J. Ma, "Poisoning attacks in federated learning: A survey," *IEEE Access*, vol. 11, pp. 10708–10722, 2023.

[4] J. Feng, W. Zhang, Q. Pei, J. Wu, and X. Lin, "Heterogeneous computation and resource allocation for wireless powered federated edge learning systems," *IEEE Trans. Commun.*, vol. 70, no. 5, pp. 3220–3233, May 2022.

[5] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10700–10714, Dec. 2019.

[6] P. Rieger, T. Duc Nguyen, M. Miettinen, and A.-R. Sadeghi, "DeepSight: Mitigating backdoor attacks in federated learning through deep model inspection," 2022, *arXiv:2201.00763*.

[7] X. Xiao, Z. Tang, L. Yang, Y. Song, J. Tan, and K. Li, "FDSFL: Filtering defense strategies toward targeted poisoning attacks in IIoT-based federated learning networking system," *IEEE Netw.*, vol. 37, no. 4, pp. 153–160, Jul. 2023.

[8] Z. Chen, W. Yi, H. Shin, and A. Nallanathan, "Adaptive model pruning for communication and computation efficient wireless federated learning," *IEEE Trans. Wireless Commun.*, vol. 23, no. 7, pp. 7582–7598, Jul. 2024.

[9] G. Zhou et al., "FedPAGE: Pruning adaptively toward global efficiency of heterogeneous federated learning," *IEEE/ACM Trans. Netw.*, vol. 32, no. 3, pp. 1873–1887, Jun. 2024.

[10] T. D. Nguyen et al., "FLAME: Taming backdoors in federated learning," in *Proc. USENIX Secur. Symp.*, 2022, pp. 1415–1432.

[11] W. Deng, X. Chen, X. Li, and H. Zhao, "Adaptive federated learning with negative inner product aggregation," *IEEE Internet Things J.*, vol. 11, no. 4, pp. 6570–6581, Feb. 2024.

[12] H. Yang et al., "Lead federated neuromorphic learning for wireless edge artificial intelligence," *Nature Commun.*, vol. 13, no. 1, p. 4269, Jul. 2022.

[13] C. Richards, S. Khemani, and F. Li, "Evaluation of various defense techniques against targeted poisoning attacks in federated learning," in *Proc. IEEE 19th Int. Conf. Mobile Ad Hoc Smart Syst. (MASS)*, Oct. 2022, pp. 693–698.

[14] Y. Deng et al., "AUCTION: Automated and quality-aware client selection framework for efficient federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 8, pp. 1996–2009, Aug. 2022.

[15] Z. Song, H. Sun, H. H. Yang, X. Wang, Y. Zhang, and T. Q. S. Quek, "Reputation-based federated learning for secure wireless networks," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1212–1226, Jan. 2022.

[16] T. Zang, C. Zheng, S. Ma, C. Sun, and W. Chen, "A general solution for straggler effect and unreliable communication in federated learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2023, pp. 1194–1199.

[17] S. He, T. Ren, X. Jiang, and M. Xu, "Client selection and resource allocation for federated learning in digital-twin-enabled industrial Internet of Things," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Mar. 2023, pp. 1–6.

[18] C. Battiloro, P. Di Lorenzo, M. Merluzzi, and S. Barbarossa, "Lyapunov-based optimization of edge resources for energy-efficient adaptive federated learning," *IEEE Trans. Green Commun. Netw.*, vol. 7, no. 1, pp. 265–280, Mar. 2023.

[19] H. Xiao, J. Zhao, Q. Pei, J. Feng, L. Liu, and W. Shi, "Vehicle selection and resource optimization for federated learning in vehicular edge computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 11073–11087, Aug. 2021.

[20] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2016, pp. 1–10.

[21] B. Luo, W. Xiao, S. Wang, J. Huang, and L. Tassiulas, "Tackling system and statistical heterogeneity for federated learning with adaptive client sampling," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2022, pp. 1739–1748.

[22] W. Huang, T. Li, D. Wang, S. Du, J. Zhang, and T. Huang, "Fairness and accuracy in horizontal federated learning," *Inf. Sci.*, vol. 589, pp. 170–185, Apr. 2022.

[23] Q. Liu, Y. Liao, B. Tang, and L. Yu, "A trust model based on subjective logic for multi-domains in grids," in *Proc. IEEE Pacific-Asia Workshop Comput. Intell. Ind. Appl.*, vol. 2, Dec. 2008, pp. 882–886.

[24] G. Han, J. Jiang, L. Shu, and M. Guizani, "An attack-resistant trust model based on multidimensional trust metrics in underwater acoustic sensor network," *IEEE Trans. Mobile Comput.*, vol. 14, no. 12, pp. 2447–2459, Dec. 2015.

[25] Z. Wang, S. Yao, G. Li, and Q. Zhang, "Multiobjective combinatorial optimization using a single deep reinforcement learning model," *IEEE Trans. Cybern.*, vol. 54, no. 3, pp. 1984–1996, Mar. 2024.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[27] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2015, *arXiv:1506.02438*.

[28] G. Haeser and A. Ramos, "Constraint qualifications for Karush–Kuhn–Tucker conditions in multiobjective optimization," *J. Optim. Theory Appl.*, vol. 187, pp. 469–487, Sep. 2020.

[29] Y. Li et al., "Energy-efficient subcarrier assignment and power allocation in OFDMA systems with max-min fairness guarantees," *IEEE Trans. Commun.*, vol. 63, no. 9, pp. 3183–3195, Sep. 2015.

[30] Q.-K. Pan, P. N. Suganthan, M. F. Tasgetiren, and J. J. Liang, "A self-adaptive global best harmony search algorithm for continuous optimization problems," *Appl. Math. Comput.*, vol. 216, no. 3, pp. 830–848, Apr. 2010.

[31] H. Xiao, L. Cai, J. Feng, Q. Pei, and W. Shi, "Resource optimization of MAB-based reputation management for data trading in vehicular edge computing," *IEEE Trans. Wireless Commun.*, vol. 22, no. 8, pp. 5278–5290, Aug. 2023.

[32] J. Du et al., "Joint optimization in blockchain- and MEC-enabled space–air–ground integrated networks," *IEEE Internet Things J.*, vol. 11, no. 19, pp. 31862–31877, Oct. 2024.

[33] J. Feng, L. Liu, X. Hou, Q. Pei, and C. Wu, "QoE fairness resource allocation in digital twin-enabled wireless virtual reality systems," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 11, pp. 3355–3368, Nov. 2023.

[34] J. Du et al., "Profit maximization for multi-time-scale hierarchical DRL-based joint optimization in MEC-enabled air-ground integrated networks," *IEEE Trans. Commun.*, early access, Sep. 5, 2024, doi: 10.1109/TCOMM.2024.3454702.

[35] S. Shen, S. Tople, and P. Saxena, "AUROR: Defending against poisoning attacks in collaborative deep learning systems," in *Proc. 32nd Annu. Conf. Comput. Secur. Appl.*, 2016, pp. 508–519.

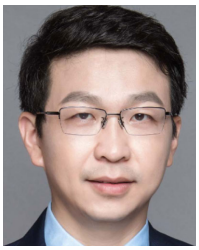
[36] H. H. Yang, Z. Liu, T. Q. S. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 317–333, Jan. 2020.

[37] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

Jie Feng (Member, IEEE) received the Ph.D. degree in communication and information systems from Xidian University, Xi'an, China, in 2020. From 2018 to 2019, she was with Carleton University, Ottawa, ON, Canada, as a Visiting Ph.D. Student. From 2022 to 2023, she was a Post-Doctoral Research Fellow with Nanyang Technological University, Singapore. She is currently an Associate Professor with the School of Telecommunications Engineering, Xidian University. Her current research interests include mobile-edge computing, blockchain, deep reinforcement learning, the device to device communication, resource allocation, and convex optimization, and stochastic network optimization.

Yanyan Liao (Graduate Student Member, IEEE) is currently pursuing the master's degree in electronic information with Xidian University, China. Her current research interests include federated learning, resource allocation, security and privacy, convex optimization, and stochastic network optimization.

Lei Liu (Member, IEEE) received the B.Eng. degree in communication engineering from Zhengzhou University, Zhengzhou, China, in 2010, and the M.Sc. and Ph.D. degrees in communication engineering from Xidian University, Xi'an, China, in 2013 and 2019, respectively. From 2013 to 2015, he worked in a technology company. From 2018 to 2019, he was supported by China Scholarship Council to be a Visiting Ph.D. Student with the University of Oslo, Oslo, Norway. He is currently a Lecturer with the Department of Electrical Engineering and Computer Science, Xidian University. His research interests include vehicular ad hoc networks, intelligent transportation, mobile-edge computing, and the Internet of Things.



Qingqi Pei (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in computer science and cryptography from Xidian University, in 1998, 2005, and 2008, respectively. He is currently a Professor and a member with the State Key Laboratory of Integrated Services Networks. His research interests include privacy preservation, blockchain, and edge computing security. He is also a Professional Member of ACM and a Senior Member of Chinese Institute of Electronics and China Computer Federation.

Ning Zhang (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2015. After that, he was a Post-Doctoral Research Fellow with the University of Waterloo and the University of Toronto, Toronto, ON, Canada. He is an Associate Professor with the Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON, Canada. His research interests include connected vehicles, mobile edge computing, wireless networking, and security. He received several Best Paper Awards from conferences and journals, such as IEEE Globecom, IEEE ICC, IEEE ICCC, IEEE WCSP, and the *Journal of Communications and Information Networks*. He is a Distinguished Lecturer of IEEE ComSoc, a Highly Cited Researcher (Web of Science), as well as the Vice Chair for IEEE Technical Committee on Cognitive Networks and IEEE Technical Committee on Big Data. He serves/served as the Co-Editor-in-Chief for IEEE COMMUNICATION SOCIETY TECHNICAL COMMITTEES NEW LETTERS and an Associate Editor for IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE Communications Surveys and Tutorials, IEEE INTERNET OF THINGS JOURNAL, and IEEE TRANSACTIONS ON COGNITIVE COMMUNICATIONS AND NETWORKING. He also serves/served as a TPC/General Chair for numerous conferences and workshops, such as IEEE ICC, VTC, INFOCOM Workshop, and Mobicom Workshop.

Keqin Li (Fellow, IEEE) is currently a SUNY Distinguished Professor of computer science with the State University of New York at New Paltz. He is also a National Distinguished Professor with Hunan University, China. His current research interests include cloud computing, fog computing and mobile edge computing, energy efficient computing and communication, embedded systems and cyber-physical systems, heterogeneous computing systems, big data computing, high-performance computing, CPU-GPU hybrid and cooperative computing, computer architectures and systems, computer networking, machine learning, and intelligent and soft computing. He is a fellow of Asia–Pacific Artificial Intelligence Association (AAIA). He is also a member of Academician of the Europe (Academia Europaea). He has authored or co-authored more than 850 journal articles, book chapters, and refereed conference papers, and received several best paper awards. He holds more than 70 patents announced or authorized by Chinese National Intellectual Property Administration. He is among the world's top 5 most influential scientists in parallel and distributed computing in terms of both single-year impact and career-long impact based on a composite indicator of Scopus citation database. He has chaired many international conferences. He is also an Associate Editor of the *ACM Computing Surveys* and the *CCF Transactions on High Performance Computing*. He served on the editorial boards for IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON CLOUD COMPUTING, IEEE TRANSACTIONS ON SERVICES COMPUTING, and IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING.