Received 8 January 2025, accepted 10 February 2025, date of publication 18 February 2025, date of current version 25 February 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3543441

RESEARCH ARTICLE

Client Selection for Generalization in Accelerated Federated Learning: A Multi-Armed Bandit Approach

DAN BEN AMI[1], KOBI COHEN[1], (Senior Member, IEEE), AND QING ZHAO[2], (Fellow, IEEE)
[1]School of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Be'er Sheva 8410501, Israel
[2]Department of Electrical and Computer Engineering, Cornell University, Ithaca, NY 14853, USA

Corresponding author: Kobi Cohen (yakovsec@bgu.ac.il)

The work of Dan Ben Ami and Kobi Cohen was supported in part by Israel Science Foundation under Grant 2640/20, and in part by the Binational Science Foundation under Grant 2024611. The work of Qing Zhao was supported by the National Science Foundation under Grant CCF-2419622.

ABSTRACT Federated learning (FL) is an emerging machine learning (ML) paradigm used to train models across multiple nodes (i.e., clients) holding local data sets, without explicitly exchanging the data. It has attracted a growing interest in recent years due to its advantages in terms of privacy considerations, and communication resources. In FL, selected clients train their local models and send a function of the models to the server, which consumes a random processing and transmission time. The server updates the global model and broadcasts it back to the clients. The client selection problem in FL is to schedule a subset of the clients for training and transmission at each given time so as to optimize the learning performance. In this paper, we present a novel multi-armed bandit (MAB)-based approach for client selection to minimize the training latency without harming the ability of the model to generalize, that is, to provide reliable predictions for new observations. We develop a novel algorithm to achieve this goal, dubbed Bandit Scheduling for FL (BSFL). We analyze BSFL theoretically, and show that it achieves a logarithmic regret, defined as the loss of BSFL as compared to a genie that has complete knowledge about the latency means of all clients. We conducted evaluations under both i.i.d. and non-i.i.d. scenarios using a synthetic dataset with a linear regression model and two well-known datasets, Fashion-MNIST and CIFAR-10 with CNN-based classification models. The results demonstrate that BSFL outperforms existing methods.

INDEX TERMS Federated learning (FL), client selection, client scheduling, multi-armed bandit (MAB), generalization in machine learning.

I. INTRODUCTION

The increasing demand for ML tasks in wireless networks consisting of a large number of clients (i.e., users or edge devices) has led to the rise of a new ML framework, called federated learning (FL) [2], [3], [4]. FL enables training ML models without extracting the data directly from the edge devices. In the beginning of the FL procedure, the ML model is being initialized. Next, the FL training scheme is done by repeated iterations between the clients that implement local training and the parameter server that aggregates functions of the local trained models. This allows to train the global model without explicitly exchanging the data, which is advantageous in terms of privacy considerations and communication resources. The bandwidth constraint dictates the total number of clients that can be scheduled for transmissions, which can be implemented by traditional digital communications over orthogonal channels [3], [4], [5] or coherent analog transmissions over multiple access channels [6], [7], [8], [9], [10]. To simplify the presentation, we focus here on traditional digital communications.

We consider an FL system with K clients sharing m orthogonal channels at each iteration (e.g., OFDMA), where $m \leq K$. Since the bandwidth is limited by m channels

The associate editor coordinating the review of this manuscript and approving it for publication was Nafees Mansoor.

and the number of clients K is typically high, only a small fraction of the clients can be scheduled for transmission at each iteration [11]. The selection of a subset of m clients among K clients has been studied in the context of scheduling and spectrum access problems in traditional communication schemes (see e.g., [5], [12], [13], [14], [15], [16]). However, solving the problem in FL systems adds new challenges resulting in fundamentally different algorithms and analysis [4]. Each iteration in the FL system includes selecting a subset of m clients, distributing the model from the server to the selected clients, training locally the model by the selected clients, uploading the trained models from the clients to the server (i.e., sharing m orthogonal uplink channels using OFDMA), and finally aggregating the received models at the server to update the global model. An illustration is presented in Fig. 1. Emerging challenges in the FL framework are described below [4], [17], [18]. The first challenge is the heterogeneous data between the clients [9], [18]. The local data is usually subjective to the client, and therefore is likely to be biased and imbalanced. Another challenge is balancing the total latency affected by different clients with different computational and communication resources, as well as different channel states. This is a key challenge in FL systems since each scheduled client experiences a different random latency, and the iteration ends when the server receives the trained models from all clients. In this paper, we tackle these challenges and develop a novel client selection algorithm which is superior to existing methods, and achieves strong performance in terms of reducing the training latency, while not harming the generalization of the model.

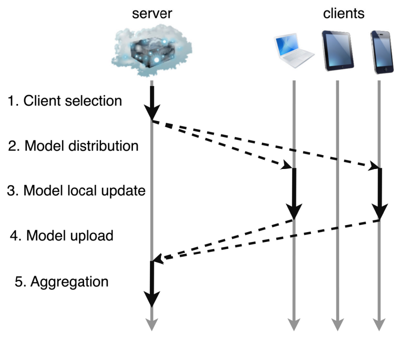


FIGURE 1. An illustration of the FL communications scheme.

A. CLIENT SELECTION METHODS IN FEDERATED LEARNING

Previous studies have shown that client selection has a great impact on the model's performance and the training latency. The standard client selection method is to choose the clients randomly with probability proportional to the amount of data each client contains [2], [19]. The clients can differ from each other in several factors such as the size of their database, the amount of available resources, their channel state, the energy they can invest, and the value of their local loss function. Each iteration in FL systems ends when the last scheduled client uploads its updated model. Thus, variability among clients might lead to poor performance with the standard random selection method. As a result, more recent methods used observations of one or more of the aforementioned factors to infer the iteration latency of each client and subsequently select the appropriate group that leads to efficient training latency [20], [21], [22], [23], [24], [25], [26], [27]. In [20], [21], [22], [23], [25], and [27] the focus was on the effect of client sampling on the learning performance. In [24] and [26] the problem is investigated under the assumption that parameters related to the communication link and latency are known (e.g., channel state, transmission latency). As a result, the optimization problem in those studies is deterministic with respect to the client selection strategy. However, in many scenarios, client latencies are affected by other random factors as discussed earlier (e.g., client energy state, available computational resources, random transmission rate), and are unknown at the time of the client selection. Therefore, recent studies (including this paper) tackle the problem using a fundamentally different approach by treating clients' latencies as unknown random variables drawn from unknown distributions, which leads to a stochastic optimization problem that raises the well-known exploration versus exploitation dilemma. On one hand, it is necessary to explore different actions (i.e., client selection sets) to explore the system state. On the other hand, it is imperative to exploit the information gathered so far to converge to the optimal selection strategy. This approach involves modeling and analyzing the problem as a MAB problem by treating the iteration latency or the local loss function of each client as a reward taken from an unknown distribution and given to the server (modeled as the gambler in MAB) [28], [29], [30], [31], [32]. In [28], the authors developed a MAB method as in the classic i.i.d. MAB that converges to a strategy that selects a small subset of the quickest clients (i.e., with the smallest expected latency). This method, however, suffers from overfitting to the quickest clients' data when used for scheduling in FL systems. In [29], [30], and [31], the focus was on reducing the loss, but the generalization issue remained open.

Unlike traditional client selection methods in federated learning, which often rely on fixed or heuristic-based strategies, our proposed BSFL algorithm introduces a novel multi-armed bandit (MAB) formulation. This formulation specifically addresses the exploration-exploitation dilemma by allowing for dynamic adjustment based on client performance and contributions to generalization (which can depend on data quality, data variability, sensor quality, etc.). Existing methods, such as those employing MAB formulations, do not adequately balance the trade-offs between training latency and model generalization. Our BSFL algorithm optimizes this balance by leveraging a time-varying reward function that considers both client latency and their contribution to the model's generalization. Furthermore, the BSFL algorithm incorporates a rigorous theoretical framework that guarantees logarithmic regret. This ensures that our method not only improves training efficiency but also enhances the robustness and accuracy of the resulting global model. Non-stochastic client selection methods face a critical limitation due to the substantial overhead they impose. These methods rely heavily on gathering detailed context information from each client—such as computing power, channel quality, and energy levels—prior to each iteration. While this can enhance selection accuracy, it also introduces significant communication and computational costs. As the number of clients increases, the process of collecting and processing this metadata becomes a bottleneck, severely limiting scalability and reducing overall training efficiency. In contrast, BSFL operates without relying on context information, making it inherently more scalable. This enables BSFL to maintain efficient and robust performance, even in large-scale federated learning scenarios.

B. COLLECTIVE OBJECTIVE AS OPPOSED TO CONCEPT OF FAIRNESS

To enable generalization by sampling a large number of clients, the concept of fairness was utilized in [28] and [32]. This heuristic approach spreads the sampling among the clients by incorporating a fairness constraint to ensure that each client is selected in a certain proportion of the communication rounds. It is important to notice that the concept of a fairness constraint holds relevance in contexts where each client pursues individual goals, as in routine data transmissions in communication systems. However, in FL, the objective is not personalized but rather focuses on training the best generalized model collectively. Consequently, the identity of the client responsible for model training becomes inconsequential. For instance, if a client's contribution to the training process is minimal, it is actually more advantageous for that client to be excluded. Moreover, even in the scenario characterized by i.i.d. data, employing the fairness-based method leads to convergence towards the group of fastest clients, with occasional selection of slower clients mandated by the fairness constraint. Consequently, incorporating a fairness constraint fails to accurately capture the optimization problem of client selection in FL. Instead, the primary goal is to train a model utilizing clients that offer the greatest contributions to its generalization.

Therefore, in this paper, we propose a fundamentally different approach. Here, we develop a novel MAB formulation for the client selection problem, incorporating a reward function that captures the generalization of the ML task. The MAB process rewards the actions (i.e., client selection decisions) based on the system parameters and the history of the process, aiming to directly select clients to optimize the collective objective. Specifically, the MAB strategy no longer converges to a specific subset of clients but rather to time-varying subsets aimed at optimizing the collective objective. This enables us to rigorously solve the client selection optimization problem in FL.

C. MAIN CONTRIBUTIONS

To solve the client selection problem in FL, one must address an online learning problem with the well-known exploration versus exploitation dilemma. On the one hand, the player (i.e., the server) should explore all arms (i.e., clients) in order to learn their state (i.e., latency distribution, generalization ability) which affects the overall training time of the FL task. On the other hand, it should exploit the information gathered so far to select the most rewarding subset of arms at each given time. In this paper, we developed a rigorous MAB formulation to model this problem, along with a novel learning algorithm to solve it. We provide rigorous theoretical analysis of the algorithm, as well as extensive numerical analysis. Below, we summarize the main contributions of our work in detail.

1) A NOVEL MAB FORMULATION FOR THE CLIENT SELECTION PROBLEM

We present a new MAB formulation that trades-off between the training latency and the generalization of the model in FL by client selection. This trade-off raises new challenges in the learning design. On the one hand, it is desired to reduce the training process time of the model, and on the other hand, it is desired to increase the ability of the model to generalize by avoiding over-fitting. The novel MAB formulation tackles this trade-off by selecting clients based on a time-varying reward influenced by the history of previous selections.

2) ALGORITHM DEVELOPMENT

We develop a novel upper confidence bound (UCB)-based algorithm to solve the problem, dubbed Bandit Scheduling for FL (BSFL). BSFL presents a fundamentally different approach to selecting clients, as the reward function updates the contributions of each client to the FL task based on the history of previous selections. As a result, BSFL does not aim to converge to selecting a fixed subset of clients, but rather aims to increase the ability of the model to generalize. Furthermore, we provide concrete examples of the use of BSFL in both cases where the data is i.i.d. and balanced across clients, and where the data is non-i.i.d. and imbalanced. Since the UCB optimization in BSFL requires to solve a combinatorial problem to select the most rewarding subset of clients at each given time, the time complexity increases exponentially with the number of clients, which might be infeasible for a large number of clients. Hence, we utilize the optimization problem's structure to devise a low-complexity algorithm, named accelerated lightweight simulated annealing (ALSA), which relies on a newly proposed accelerated lightweight simulated annealing technique that we develop for BSFL. Specifically, we show analytically that ALSA reduces the degree-order in the simulated annealing graph from $O(K^2)$ to $O(K)$ (where K is the number of clients), and still keeps the convergence property. This results in a significant accelerated convergence time requires to solve the UCB optimization in BSFL.

3) PERFORMANCE ANALYSIS

We analyze the performance of BSFL theoretically and numerically. In terms of theoretical analysis, as commonly done in the MAB literature, we evaluate the performance by regret, defined as the loss of BSFL as compared to a genie that has complete knowledge about the latency means of all clients. We analyze the performance of BSFL rigorously, and show that it achieves a logarithmic regret with time. In terms of numerical analysis, we present extensive simulations using synthetic and real datasets. All simulations demonstrate that BSFL is superior to existing methods in the regret, prediction accuracy and efficiency.

The BSFL algorithm is well-suited for applications where balancing latency and model generalization is desired to achieve real-time performance and reliability. Examples include IoT networks, autonomous driving systems, smart cities, and next-generation wireless networks, where efficient client selection optimizes data-driven decision-making and supports seamless operation. These applications rely on robust algorithms to manage latency, bandwidth, and model generalization in real-time scenarios. Our proposed MAB-based approach significantly improves the efficiency of training processes by minimizing latency while maintaining model generalization. This advancement is especially significant in real-world applications that require timely and accurate model updates. By optimizing client selection, our method ensures that FL systems can be more effectively deployed in practical scenarios, leading to better performance and faster training process.

II. SYSTEM MODEL

We consider a common FL system with star topology. The FL system consists of a set $\mathbb{K}$ of clients with cardinality $|\mathbb{K}| = K$, where the clients communicate directly with the server via m orthogonal channels (e.g., OFDMA), where $m \leq K$. Since the bandwidth is limited by m channels and the number of clients K is typically high, only a small fraction of the clients can be scheduled for transmission at each iteration [11]. Due to operation or communication constraints, the server interacts with a set $\mathbb{A}_t \subseteq \mathbb{K}$ of clients which are available to participate in the FL task at iteration t. The number of available clients $|\mathbb{A}_t|$ is assumed to be much higher than the number of channels m. At each iteration, the server selects m clients from the set $\mathbb{A}_t$ of clients to participate in the FL task, each client is assigned to a dedicated orthogonal channel. We denote the set of all possible client selections by $H(\mathbb{K}) = \{\mathcal{S} \subset \mathbb{K} : |\mathcal{S}| = m\}$, and the set of all possible client selections at iteration t by $H(\mathbb{A}_t)$, i.e., $H(\mathbb{A}_t) = \{\mathcal{S} \subset \mathbb{A}_t : |\mathcal{S}| = m\} \subseteq H(\mathbb{K})$.

Each client $k \in \mathbb{K}$ holds a local database X_k. The local data is not being shared in FL systems due to privacy or communication constraints, and only the output model of a local training procedure is transmitted to the server. The FL task by the server is to minimize the global loss function denoted by:

$$\mathcal{L}(W, X) = \sum_{k \in \mathbb{K}} \frac{|X_k|}{|X|} \cdot \mathcal{L}(W, X_k) \tag{1}$$

with respect to the model's parameters denoted as W, where $X = \cup_{k \in \mathbb{K}} X_k$ and $\mathcal{L}(W, X_k)$ is the local loss of client k. The server's action at iteration t is defined by the selection of the participating set of clients. We denote this action in the algorithm that we aim to develop by $\mathcal{A}_t \subseteq H(\mathbb{A}_t)$. We also denote the history of all previous client selections by the server before iteration t by $\mathcal{H}_t = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_{t-1}\}$.

Each iteration consumes a random processing time depending on the participating clients due to distributing the model to the clients, local training and updating the model at each client, and transmitting each local model back to the server for global aggregation. We denote $\tau_{k,t}$ as the total random latency consumed by client k at iteration t. At each iteration t, the FL process proceeds to the next iteration after receiving all the trained models from all the selected clients. Therefore, the total iteration time τ_t is dictated by the slowest client, i.e.

$$\tau_t = max_{k \in \mathcal{A}_t} \tau_{k,t}. \tag{2}$$

III. MAB-BASED FORMULATION FOR THE CLIENT SELECTION PROBLEM

In this section, we present a novel MAB formulation for the client selection problem in order to minimize the training latency without harming the ability of the model to generalize, i.e., to provide reliable predictions for new observations.

A. TRADING-OFF BETWEEN THE GENERALIZATION AND ITERATION TIME VIA MAB FORMULATION

MAB problems are often illustrated with the example of a player or gambler facing a row of slot machines, selecting arms to pull at each time and obtaining rewards accordingly. In our context, the server acts as the player, and the clients act as the arms. At each iteration, the server selects a subset of clients, which contribute to the learning process rate accordingly. Solving the MAB problem, and specifically the client selection problem in this paper, requires addressing the well-known exploration versus exploitation dilemma in online learning. To better illustrate the trade-off between exploration and exploitation in the MAB framework, consider a scenario where a server selects clients with unknown latencies for model training. In the early stages, the server may explore by selecting different clients to gather information about their latencies. The server might initially choose each client multiple times to estimate their upper confidence bound. As more observations gathered, the server shifts towards exploitation, selecting the client group that provides the best balance between low latency and contribution to model generalization.

The exploration versus exploitation dilemma has been rigorously addressed in MAB optimization [33]. As a result, recent studies have modeled and analyzed the client selection problem in FL as a MAB problem, treating the iteration latency or local loss function of each client as a reward obtained from an unknown distribution and given to the server [28], [29], [30], [31], [32], as discussed in Section I.a. However, all these methods converge to selecting the fastest clients (except for the times of selecting slower clients due to the fairness constraint in [32]). Consequently, they suffer from overfitting to the data of the quickest clients when applied to scheduling in FL systems, leading to reduced generalization ability and prediction accuracy.

In this paper, we overcome this limitation by developing a novel MAB formulation for the client selection problem consisting of a time-varying reward function that captures both the client latency as well as the client ability to contribute to the generalization of the ML task. It should be noted that previous work on rotting bandits [34] considered a reward function that decreases with the activation rate of selected arms. However, the model here is fundamentally different, since the time-varying reward function depends on the overall subset selections of correlated clients in the systems. This leads to a novel combinatorial MAB (CMAB) formulation [35], in which the design and analysis is fundamentally different from rotting bandits. Specifically, our formulation trades-off rigorously between the generalization and latency, as each client in FL systems might have different characteristics such as computing power, task operations, channel states, etc. This results in different latency distributions in training and transmitting the local models across clients in the FL system [4]. On the one hand, to reduce the total iteration time at each iteration, it is desired to select clients with small latencies at each iteration. On the other hand, it is desired to select as many clients as possible during the training process (even slower clients) for the model to be robust and generalized for both i.i.d. balanced data and non-i.i.d. imbalanced data [32], [36], [37], [38]. In the next subsection we describe the generalization function used in the CMAB formulation.

B. THE GENERALIZATION FUNCTION

For simplicity and maintaining generality, we denote by Θ all system constant parameters that contribute to the reward (which will be described later with examples). The level of contribution of client k to the generalization ability and robustness of the model is evaluated by a generalization function $g_{k,\Theta} : \mathcal{H}_t \to [-1, 1]$. It gives at each iteration t a value that represents the contribution of client k to the generalization of the global model, depending on the model constant parameters Θ and the selection history before time t, $\mathcal{H}_t$. We next present concrete examples of the function $g_{k,\Theta}$ in both cases of i.i.d. and balanced data, and non-i.i.d. and imbalanced data across clients.

We start by considering the case of i.i.d. balanced data. We use the selection history to extract a counter $c_{k,t}$ for each client that counts the number of iterations that client k was selected until the beginning of iteration t, i.e., $c_{k,t} = \sum_{i=1}^{t-1} \mathbb{1}_{\{k \in \mathcal{A}_i\}}$. In order to achieve high generalization in this case, it is important that each client is selected an equal number of times [19], [32], [36], [37], [38]. In this case, we let Θ store the number of clients and the number of channels, i.e., $\Theta = \{K, m\}$. Then, an effective design of the function $g_{k,\Theta}$ is given by:

$$g_{k,\Theta}(\mathcal{H}_t) = g_{k,\Theta}\left(\frac{c_{k,t}}{t}\right) = \left|\frac{m}{K} - \frac{c_{k,t}}{t}\right|^{\beta} \cdot sgn\left(\frac{m}{K} - \frac{c_{k,t}}{t}\right), \tag{3}$$

where β is a tuning parameter (natural number), and $sgn(\cdot)$ is the sign function that returns ± 1. An illustration is presented in Fig. 2.

The key idea behind the design is that the more the ML model is trained with new data points or alternatively with data that was used too little, the more its ability to generalize increases [36], [38]. It is thus desired to select clients uniformly with rate m/K over time. Therefore, the generalization function $g_{k,\Theta}$ is designed to be monotonically decreasing so that it provides incentive (i.e., positive value in terms of contributing to generalization) to select clients that were selected too little (i.e., $\frac{c_{k,t}}{t} < m/K$), and returns zero when the client's counter reaches the desired selection rate (i.e., $\frac{c_{k,t}}{t} = m/K$). Clients that were selected too frequently are given negative values in terms of contributing to generalization.

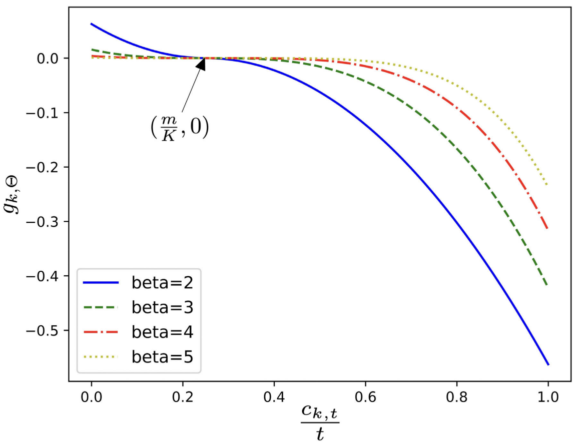


FIGURE 2. The value of $g_{k,\Theta}$ as a function of $\frac{c_{k,t}}{t}$ with different values of β.

In the second scenario, consider non-i.i.d. and imbalanced data, where the data quality varies between clients. In this case, it is desired to take into account the participation rate of each client, as well as the quantity and quality of the data, when selecting clients for transmission. The quantity of each client's data (i.e., its database size $|X_k|$) is useful for the generalization, as it more likely to represent the true distribution of data [39]. The second aspect stems from the desire to train the model on reliable, high-quality, and meaningful data [40], [41], [42], [43]. Generally, in ML, and particularly FL, the data for each client might have different quality due to several factors such as device heterogeneity, environment heterogeneity, etc. We define the client's data quality by $q_k \in [0, 1]$, where 1 stands for the best quality, and thus the generalization function $g_{k,\Theta}$ is designed to prioritize clients with higher data quality. Since the quality for each client (say client k) is aggregated over all its data points X_k, we denote the overall significance of client k by $d_k = q_k \cdot |X_k|$. Therefore. the model parameters in this case is given by $\Theta = \left\{K, m, \{(|X_k|, q_k) : k \in \mathbb{K}\}\right\}$, and the generalization function is given by:

$$g_{k,\Theta}(c_{k,t}) = \left|\frac{md_k}{\sum_{k\in\mathbb{K}} d_k} - \frac{c_{k,t}}{t}\right|^{\beta} \cdot sgn\left(\frac{md_k}{\sum_{k\in\mathbb{K}} d_k} - \frac{c_{k,t}}{t}\right). \tag{4}$$

The operation of the generalization function can be illustrated through different data scenarios. In the case of i.i.d. data, where each client has a balanced dataset, the generalization function ensures that clients are selected uniformly over time. For example, if the system has four clients with similar data distributions, the generalization function will aim to select each client roughly 25% of the time, thus ensuring a fair representation of all data in the global model. In contrast, with non-i.i.d. data, where clients have skewed data distributions, the generalization function prioritizes clients with more diverse or underrepresented data. In realistic scenarios involving unbalanced and non-i.i.d. data, the generalization function can utilize various metadata attributes and data distribution of the clients. For example, in the case of the training digits-classifier task (such as MNIST), the generalization function would assign a higher value to clients with a larger dataset, those that have participated less frequently, those with uniformly distributed samples across all 10 digits, and those with higher quality cameras. The generalization function quantifies the attributes mentioned above, allowing BSFL to balance them against the client's training time within the MAB framework.

In the next subsection we present the CMAB formulation for the client selection problem. In the empirical study in this paper, we used the forms of $g_{k,\Theta}$ as detailed above, which demonstrated very good performance. It should be noted, however, that the CMAB formulation is general. It does not depend on a specific form of $g_{k,\Theta}$, and other forms can be used. For example, it is possible to define $g_{k,\Theta}$ to be dependent on the local loss value of each client, which is advantageous in improving the convergence in the long term [44].

C. CMAB-BASED FORMULATION WITH HISTORY-DEPENDENT REWARD

We now formulate the client selection problem as a CMAB problem with history-dependent reward. At each iteration t, the server (i.e., the player) selects a subset of clients $\mathcal{A}_t \in H(\mathbb{A}_t)$ (i.e., arms), and at the end of the iteration observes the clients' iteration latencies. The reward given at the end of iteration t is defined by:

$$r(t) = \min_{k\in\mathcal{A}_t}\{\frac{\tau_{min}}{\tau_{k,t}}\} + \frac{\alpha}{m}\sum_{k\in\mathcal{A}_t} g_{k,\Theta}(\mathcal{H}_t), \tag{5}$$

where α is a hyper-parameter of the generalization, balancing between the amount of the generalization and the iteration time. $\tau_{\min}$ denotes the shortest conceivable latency for an iteration, encompassing the duration to transmit and retrieve a model, presuming no training time for the client. This is primarily utilized to normalize the iteration speed expression in the reward definition for eliminating the effects of any potential high rewards that may harm the learning process of the algorithm and for analytical considerations, which can be found in the appendix. For purposes of analysis, we further assume that the reward given at the end of each iteration is quantized and the smallest difference between two different rewards is denoted by Δ_{min}, as commonly assumed in analyzing MAB-based problems. Since the iteration time is determined by the maximal latency under the selected subset of clients $\mathcal{A}_t$, the first term on the RHS of (5) represents the reward due to the iteration time, and the second term represents the reward due to the generalization. The smaller the value of α, the higher priority to select faster clients. On the one hand, this reduces the iteration time. On the other hand, it tends to select the same faster clients at each iteration, which reduces the generalization, and increases the total number of iterations needed to achieve reliable predictions.

An important insight from the CMAB formulation is that if a slow client is given high priority based on the reward function and is selected for transmission at the current iteration, then since the iteration time is determined by the slowest client, there is no motivation to choose fast clients at the same iteration. Instead, clients with high rewards in terms of generalization, even if they are slow, should be prioritized.

Note that the formulation of our CMAB problem is fundamentally different from the classical CMAB problem. Specifically, here the reward does not depend on the new observations solely (as in classic CMAB), but also on the selection history up to the current iteration. Thus, optimizing the reward in classic MAB leads to a fixed selection of arms, while in our problem the optimal selection is time-varying depending on the selection history, as will be detailed in the next subsection.

D. THE OBJECTIVE

We aim to find the best selection policy $\Pi = \{\mathcal{A}_1, \mathcal{A}_2, \ldots\}$ (where $\mathcal{A}_t$ stands for the selection at iteration t) which minimizes the training process time while preserving the ability of the model to perform the generalization. The performance of online learning algorithms is commonly evaluated by the regret, defined as the loss of an algorithm as compared to a genie with side information on the system. Here, to measure the performance of our proposed policy we define the regret as follows. We denote the maximal mean reward at iteration t that can be obtained by genie that has complete knowledge about the latency means of all clients by $r^*(t)$, where genie's selection at time t is given by:

$$\mathcal{G}_t = \arg\max_{\mathcal{S}\in H(\mathbb{A})} \left\{ \min_{k\in\mathcal{S}} \mu_k + \frac{\alpha}{m}\sum_{k\in\mathcal{S}} g_{k,\Theta}(\mathcal{H}_t) \right\}, \tag{6}$$

where μ_k stands for the speed mean of client k (i.e., $\mu_k = \mathbb{E}[\frac{\tau_{min}}{\tau_k}]$).

The regret of the policy (say Π) at time n is defined by the accumulated loss by time n between the reward obtained by genie and the expected reward obtained by Π:

$$R(n) = \mathbb{E}_\Pi\left[\sum_{t=1}^{n} r^*(t) - r(t)\right]. \tag{7}$$

The objective is thus to find a policy Π that minimizes the regret order with time.

IV. THE PROPOSED BSFL ALGORITHM

In this section we start by presenting the BSFL algorithm to solve the objective. Then, we analyze the regret (7) analytically.

A. DESCRIPTION OF THE ALGORITHM

The BSFL algorithm is based on a novel UCB-type design for client selection in FL. The pseudocode for the BSFL algorithm is provided in Algorithm 1. We now discuss the steps of the BSFL algorithm in detail.

BSFL observes realizations of the random speed of the selected clients (i.e., $\frac{\tau_{min}}{\tau_{k,t}}$) and estimates its mean accordingly. Let

$$\overline{\mu_{k,t}} = \frac{1}{c_{k,t}}\sum_{i=1}^{t-1}\frac{\tau_{min}}{\tau_{k,i}}\cdot \mathbb{1}_{\{k\in\mathcal{A}_i\}} \tag{8}$$

be the sample-mean speed of client k after t iterations. We design the UCB function of client k's speed after t iterations by:

$$\text{ucb}(k,t) = \overline{\mu_{k,t}} + \sqrt{\frac{(m+1)\ln t}{c_{k,t}}}. \tag{9}$$

To maintain an updated $\overline{\mu_{k,t}}$ and $\text{ucb}(k,t)$ for each client, each time a client is selected to participate in the FL iteration, the algorithm observes its speed $\frac{\tau_{min}}{\tau_{k,t}}$ and updates its counter and the speed's sample-mean as follows:

$$c_{k,t} \leftarrow c_{k,t-1} + 1, \tag{10}$$

$$\overline{\mu_{k,t}} \leftarrow \frac{\overline{\mu_{k,t-1}}\cdot c_{k,t-1} + \frac{\tau_{min}}{\tau_{k,t}}}{c_{k,t}}. \tag{11}$$

Then, for each client $k\in\mathbb{K}$, the UCB function is updated as follows:

$$\text{ucb}(k,t) \leftarrow \overline{\mu_{k,t}} + \sqrt{\frac{(m+1)\ln t}{c_{k,t}}}. \tag{12}$$

At the initialization step, for each client $k \in \mathbb{K}$, $c_{k,0}$ and $\overline{\mu_{k,0}}$ are set to 0, and $\text{ucb}(k,0)$ is set to infinity. Later, in the main loop, the algorithm selects clients at each iteration according to:

$$\mathcal{A}_t = \arg\max_{\mathcal{S}\in H(\mathbb{A})} \left\{ \min_{k\in\mathcal{S}} \text{ucb}(k,t-1) + \frac{\alpha}{m}\sum_{k\in\mathcal{S}} g_{k,\Theta}(\mathcal{H}_t) \right\}. \tag{13}$$

Afterwards, for each client $k \in \mathcal{A}_t$, the algorithm observes $\tau_{k,t}$ and updates $c_{k,t}$, $\overline{\mu_{k,t}}$ using (10), (11). At the end of every iteration, $g_{k,\Theta}(\mathcal{H}_{t+1})$ and $\text{ucb}(k,t)$ for all client $k\in\mathbb{K}$ are updated using (12). Note that two different trade-off mechanisms are manifested during the algorithms. The first is between exploration and exploitation of the client latencies, while the second is between the iteration latency and the generalization ability.

Algorithm 1 BSFL Algorithm

```
   Input: Set of client indices $\mathbb{K}$
    Initialization:
1:  $\forall k \in \mathbb{K} : c_{k,0} \leftarrow 0, \text{ucb}(k,0) \leftarrow \infty, \overline{\mu_{k,0}} \leftarrow 0$
    Main loop:
2:  for iterations $t = 1, 2, \ldots$ do:
3:      Select a set of $m$ clients using (13) (ties are broken arbitrarily)
4:      Execute FL iteration
5:      For each client $k \in \mathcal{A}_t$ observe $\tau_{k,t}$ and update $c_{k,t}$, $\overline{\mu_{k,t}}$ using
          (10), (11).
6:      For each client $k \in \mathbb{K}$ update $g_{k,\Theta}(\mathcal{H}_{t+1})$ accordingly, and
        ucb$(k,t)$ using (12)
7:  until convergence
```

B. REGRET ANALYSIS

In this subsection, we analyze the regret (7) achieved by BSFL analytically, and show that it has a logarithmic order with time. To evaluate the regret of BSFL, we define $r(\mathcal{S},\mathcal{H}_t)$ as the reward that could have been obtained at the end of iteration t, given the selection history $\mathcal{H}_t$, if selection $\mathcal{S}$ had been made. Let Δ_{max} be the maximum difference between the expected reward obtained by $\mathcal{G}_t$ and by any selection $\mathcal{S}$, given any selection history, i.e.,

$$\Delta_{max} = \max_{\mathcal{S}\in H(\mathbb{K}),\mathcal{H}_t\subset H(\mathbb{K}),t\in\mathbb{N}} \mathbb{E}\left[r(\mathcal{G}_t,\mathcal{H}_t) - r(\mathcal{S},\mathcal{H}_t)\right]. \tag{14}$$

Note that Δ_{max} is bounded by:

$$\Delta_{max} \le 2\alpha + \mu_{max} - \mu_{min},$$

where μ_{max} and μ_{min} are the expected speeds of the fastest and slowest clients, respectively.

Theorem 1: At each iteration n, the regret of BSFL is upper bounded by:

$$\Delta_{max}\cdot K\cdot\left(\frac{4(m+1)\ln n}{\Delta_{min}^2} + 1 + \frac{\pi^2}{3}\right). \tag{15}$$

Here, Δ_{min}, as stated earlier, represents the smallest possible difference between two rewards based on quantization.

The proof can be found in Appendix A.

To offer some intuition behind the regret bound in the theorem, note that it captures the impact of the client selection process (characterized by the number of clients K and channels m) and the difference in client speeds ($\mu_{max} - \mu_{min}$) on the overall regret. The logarithmic growth ensures that, even as the system scales or client diversity increases, the BSFL algorithm maintains robust efficiency over a large number of iterations. Figures 5 and 6, along with an explanation in Section VI, clearly illustrate the exploration-exploitation tradeoff in relation to the algorithm parameters.

Theorem 1 implies that BSFL achieves a logarithmic regret order with time $O(\ln n)$.

It is worth noting that the selection policy of Algorithm 1, i.e., finding the maximum value in the update rule stated in (13), can be computationally infeasible due to its exponential complexity. Therefore, in the following section, we will develop a practical solution to solve the maximization problem.

V. COMPLEXITY REDUCTION USING A NOVEL ACCELERATED LIGHTWEIGHT SIMULATED ANNEALING

The most computationally challenging aspect of the BSFL algorithm is selecting clients at the beginning of each iteration, as indicated by line 3 of the pseudocode. Specifically, the server needs to find the client selection that maximizes the expression in the update rule (13). Classic deterministic methods for finding this selection have a time complexity of $O\binom{|\mathbb{A}_t|}{m}$, which is computationally infeasible when the number of channels (m) is large. Therefore, heuristic methods are often used to find good approximate solutions to optimization problems in a finite amount of time. In particular, simulated annealing (SA) is a heuristic optimization algorithm that has strong theoretical properties of convergence guarantee as the number of time steps increases. In the subsequent sections, we present a new SA-type algorithm that leverages the unique structure of the MAB-based FL scheduling problem to accelerate the stochastic optimization process through the design of a lightweight search space. The proposed SA-type method demonstrates significantly faster convergence compared to the classical method while still preserving the strong theoretical convergence guarantee property as the number of time steps used to execute line 3 of the pseudocode increases.

A. THE STOCHASTIC OPTIMIZATION FLOW

To implement SA-type algorithm to solve (13), we need to define a multi-state environment, in which each state s has neighboring states denoted as N_s. Each state also has an associated energy, represented by $E(s)$. The objective of the algorithm is to find the state with the highest energy (or, alternatively, the lowest energy, depending on the problem formulation). To facilitate movement between states, a temperature parameter T_i for time step i is introduced. This temperature affects the probability of transitioning from the current state to a state with lower energy. The algorithm begins by randomly selecting an initial state s_0. It then proceeds through a series of time steps, during which the temperature is updated and a neighboring state $u \in N_{s_i}$ is chosen at random. The next state s_{i+1} is then updated according to:

$$s_{i+1} \leftarrow \begin{cases} u, & \text{if } E(u) \geq E(s_i), \\ u, & \text{w.p., } e^{\frac{E(u)-E(s_i)}{T_i}} \text{ if } E(u) < E(s_i), \\ s_i, & \text{otherwise.} \end{cases} \tag{16}$$

In our MAB-based FL scheduling setting, each client selection determines a state, which means that in each FL round t, the number of states of the stochastic optimization problem would be $\binom{|\mathbb{A}_t|}{m}$. The energy of each state would be:

$$E(\mathcal{S}) = \min_{k\in\mathcal{S}}\left\{\text{ucb}(k,t) + \frac{\alpha}{m}\sum_{k\in\mathcal{S}} g_{k,\Theta}(\mathcal{H}_t)\right\}. \tag{17}$$

As previously mentioned, the SA dynamics requires that each state has neighboring states. A direct implementation of the classic SA method results in a structure where any two client selections (which represent two states), $\mathcal{S}$ and $\mathcal{U}$, will be considered neighboring selections only if they differ in only one client. Formally,

$$\mathcal{S} \in N_{\mathcal{U}} \ , \ \mathcal{U} \in N_{\mathcal{S}} \Leftrightarrow \left|\mathcal{S}\cap\mathcal{U}\right| = m-1, \tag{18}$$

where m is the number of channels, same as before. It can be seen that through this construction, the neighborhood is symmetrical between the selections (states) and that for each selection there are $m(K-m)$ neighbors. From [45], setting the temperature at each time step to $T_i = \frac{\Delta_{max}}{\log(i+1)}$ guarantees convergence to the selection with the maximum energy as the number of time steps in the SA algorithm approaches infinity, where Δ_{max} is defined in (14). Despite the strong theoretical convergence guarantee, a recognized disadvantage of the SA method is its rate of convergence which can be quite slow. Thus, in the subsequent section, we present a novel SA-type algorithm that addresses this issue while still maintaining the theoretical convergence guarantee.

B. THE PROPOSED ACCELERATED LIGHTWEIGHT SIMULATED ANNEALING (ALSA) ALGORITHM

Building on the concept of SA-type dynamics, we present a novel accelerated SA-type method that capitalizes on the specific structure of the MAB-based FL scheduling problem. Our proposed method accelerates the optimization process by designing a lightweight search space, resulting in faster convergence compared to the classic SA technique. This is achieved by taking into account the unique features of the MAB-based FL scheduling problem.

When applying the classic SA method to our problem, the number of neighbors for each state is $m(K-m)$, which becomes $O(K^2)$ when $m = O(K)$. In our proposed ALSA method, we exploit the characteristics of the multi-armed

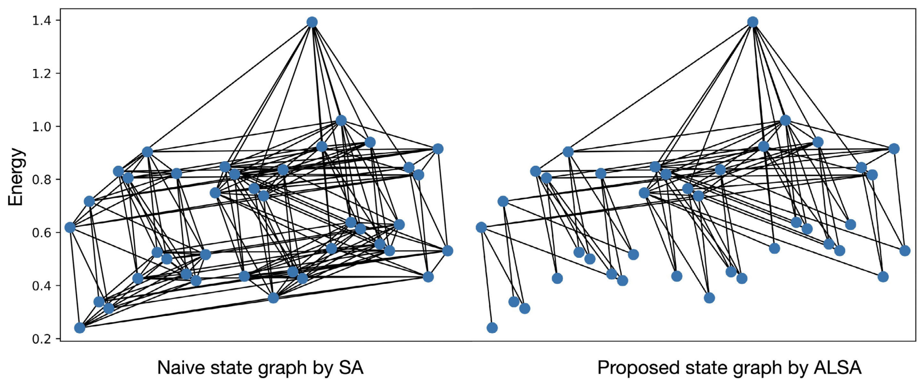


FIGURE 3. An illustration of the sub-graphs of states resulting by ALSA and SA. For clarity, only sub-graphs of the complete graphs are shown.

bandit optimization to reduce the number of connections between states and create a lightweight state graph for the search space. This modification allows for faster convergence while still maintaining the theoretical convergence guarantee of the SA method (which will be shown later). Specifically, we define a new neighborhood rule that only allows for client selections that differ by a single client from the current selection, and requires that this client has the lowest ucb value or the lowest $g_{k,\Theta}$ value in one of the selections. Formally,

$$\begin{aligned}&\mathcal{S} \in N_{\mathcal{U}} \ , \ \mathcal{U} \in N_{\mathcal{S}} \\ &\quad \Leftrightarrow \left(\left|\mathcal{S} \cap \mathcal{U}\right| = m-1 \right) \text{ and} \\ &\left(\mathcal{S} \backslash \mathcal{U} \subset \left\{ \arg\min_{k\in\mathcal{S}} \mathrm{ucb}(k,t), \arg\min_{k\in\mathcal{S}} g_{k,\Theta}(\mathcal{H}_t) \right\} \text{ or} \right. \\ &\left. \mathcal{U} \backslash \mathcal{S} \subset \left\{ \arg\min_{k\in\mathcal{U}} \mathrm{ucb}(k,t), \arg\min_{k\in\mathcal{U}} g_{k,\Theta}(\mathcal{H}_t) \right\} \right). \qquad (19)\end{aligned}$$

It is important to note that each selection $\mathcal{S}$ has at most $2(K-m)$ other selections that differ by only one client k which is in the set

$$\left\{ \arg\min_{k\in\mathcal{S}} \mathrm{ucb}(k,t), \arg\min_{k\in\mathcal{S}} g_{k,\Theta}(\mathcal{H}_t) \right\}.$$

The neighborhoods in this formulation are symmetrical, and the average number of neighbors for each selection is no more than $4(K-m)$, resulting in a total of $O(K)$ neighbors regardless of the value of m.

Denote the state with the global maximum energy as s^*. The theoretical convergence of ALSA is shown next.

Theorem 2: Implementing ALSA with temperature $T_i = \frac{\Delta_{max}}{\log(i+1)}$ at time step i yields:

$$\lim_{i\to\infty} E(s_i) = E(s^*). \qquad (20)$$

The proof can be found in Appendix B.

Theorem 2 implies that as the number of time steps for running ALSA increases, the result of executing ALSA will converge to the optimal solution of (13). By using ALSA, the execution of line 3 in the BSFL algorithm (Algorithm 1) becomes efficient.

C. A COMPARISON OF SA AND ALSA FOR SELECTING CLIENTS

The classic SA method, when applied to our problem, results in a graph structure in which each state has $O(K^2)$ neighbors. However, by exploiting the characteristics of the MAB-based optimization, our proposed ALSA method is able to reduce the number of connections between states to create a lightweight graph for the search space, while still maintaining the theoretical convergence guarantee as previously demonstrated. This construction results in a graph structure in which each state has $O(K)$ neighbors.

To demonstrate the effectiveness of our proposed method, we conducted thousands of runs with varying numbers of clients and selection sizes. In the vast majority of runs (98.3%), ALSA reached a state with a higher energy within the fixed time period compared to the classic SA method. Furthermore, we observed that the majority of the additional edges in the classic SA method are between low-energy states or states with similar energy, which do not contribute to the convergence to the global maximum state and instead cause the algorithm to wander for a longer time between these low-energy states. This is illustrated in Figure 3, which shows a comparison between the graph structures created by ALSA and SA for a simulation of selecting 4 clients out of 8 with drawn $\mathrm{ucb}(k,t)$ and g_k values. As can be observed in Fig. 4, the judicious removal of excess edges in the graph created by ALSA leads to a significantly faster rate of convergence to high-energy states compared to SA.

VI. SIMULATION RESULTS

We begin by illustrating the influence of the alpha and beta parameters on the selection algorithm. To this end, we conducted an experiment using a scenario where the data is i.i.d. across clients. The clients were divided into 10 groups, each containing 50 clients with a uniform average iteration time within the group. The average iteration times for the groups ranged from 0.1 to 1.0 in increments of 0.1. The selection algorithm was then executed using four different parameter settings. Every 500 iterations, we calculated the

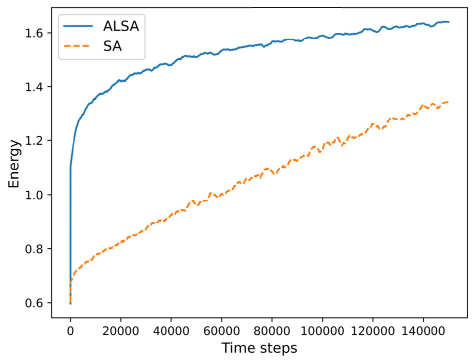


FIGURE 4. The energy of the current state as a function of the time step of ALSA and SA runs is depicted in this figure. The simulation in this figure is based on the selection of 25 clients out of 500, for which it is computationally infeasible to directly solve(3) ($\approx 2^{139}$ different selections).

TABLE 1. Results for varying Alpha in the i.i.d. scenario on CIFAR10.

Alpha	0.1	1	2	3	10
Accuracy	0.58	0.63	0.67	0.66	0.62
Minimum Loss	1.66	1.18	0.83	1.05	1.15

average frequency of selection for clients within each group, referred to as the average mean observations. The results of this experiment are depicted in Figures 5 and 6, showing the effects of varying the alpha and beta parameters, respectively. In the upper left bar chart in figure 5, where alpha is set to 0, the exploration-exploitation trade off is clearly illustrated. During the first 500 iterations, the algorithm selects clients from all groups almost equally, emphasizing exploration. However, as the iterations progress, the algorithm increasingly favors the faster group (with the 0.1 iteration time) (exploitation), while continuing to explore clients from slower groups to some extent. Additionally, as the alpha value increases (in the bottom graphs), the algorithm selects more clients from slower groups to enhance model generalization and prevent overfitting on the data from the fastest group. Similarly, in figure 6 as the beta parameter decreases, the generalization function differentiates more sharply between the contributions of different clients, leading the algorithm to favor clients that have been selected less frequently, thus, the algorithm promotes greater equality in the frequency with which each client is selected. We also performed a sensitivity test of the alpha and beta hyperparameters on performance. The tables 1, 2 demonstrate how varying hyperparameters, such as alpha and beta, affects the algorithm's performance in the i.i.d. data scenario on the CIFAR10 dataset using the ResNet18 CNN model. The columns in the tables represent the test set results obtained by retraining the model with the specified hyperparameter values from Table 3, except for the hyperparameter that is varied in the corresponding table.

Next, we present the results of our simulations, which include two types of datasets and models. The first is synthetic data for linear regression and the second is image

TABLE 2. Results for varying Beta in the i.i.d. scenario on CIFAR10.

Beta	0.1	0.8	1	1.2	5
Accuracy	0.60	0.65	0.67	0.66	0.61
Loss	1.11	1.10	0.83	1.14	1.61

data from two well-known datasets, Fashion-MNIST and CIFAR-10, for a convolutional neural network (CNN) model.

We consider two scenarios for each simulation. In the first scenario, the data (images or synthetic data) is divided i.i.d. between the clients, and for the image databases each client has an equal number of images from each class. In the second and more challenging scenario, each client has a different amount of data and, in the image datasets, a different distribution of images among the ten classes.

To evaluate the effectiveness of the proposed BSFL algorithm in selecting clients that enable rapid model convergence without compromising generalization, we conducted simulations comparing our approach to several state-of-the-art (SOTA) client selection algorithms: (i) CS-UCB: Proposed in [28], this algorithm applies the MAB framework to the i.i.d. scenario for improved training efficiency but does not address generalization as BSFL does; (ii) CS-UCB-Q: Also introduced in [28], this algorithm extends CS-UCB to the non-i.i.d. scenario using the MAB framework; (iii) RBCS-F: Proposed in [32], this algorithm utilizes a contextual MAB approach, where clients report their available computing power as context to optimize iteration time. It also incorporates a fairness mechanism to avoid consistently excluding slower clients; (iv) Power-of-Choice [46]: This method selects clients with the highest loss in their previous local training iteration; (v) Random Selection: In the i.i.d. scenario, clients are selected uniformly at random. In the non-i.i.d. scenario, the selection is weighted based on the data volume of each client [19]. Table 3 presents the details of all the training experiments. In all our simulations, we utilized the StepLR scheduler to adjust the learning rate during training, promoting better convergence.

In the methods comparison, we start by comparing the regret between the algorithms. To compare the regret, we first had to search through all $\binom{K}{m}$ client selections in each iteration and determine which selection truly maximized (6), i.e., find the selection $\mathcal{G}_t$. Therefore, in order to compare the regret, we chose a relatively small number of clients (namely, 20) and a selection size of 5. As shown in Figure 7 (left), BSFL achieves a logarithmic regret compared to the other algorithms, which reached a linear regret. Additionally, Figure 7 (right) illustrates that BSFL achieves the smallest loss values among the algorithms.

In addition, while still operating within the i.i.d. case, we conducted simulations on real data using a ResNet-18 model with 500 clients and a selection size of 25. The ResNet-18 model features a residual architecture comprising convolutional layers and skip connections to facilitate efficient learning. In Figures 8, 9, we present the results of

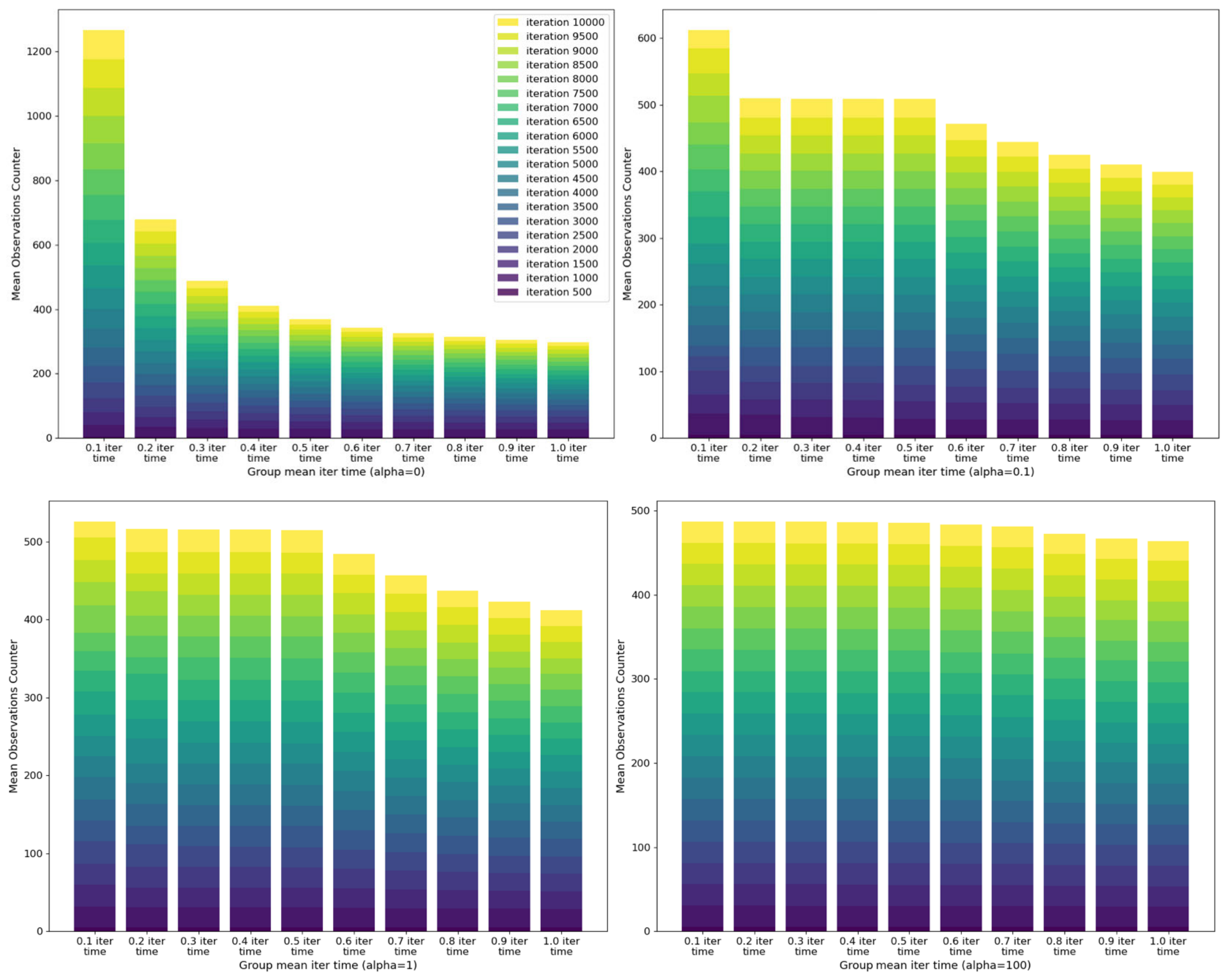


FIGURE 5. Training experiments conducted with four different alpha values. Each experiment involved 10,000 iterations, and the results are displayed as bar charts showing the mean number of observations for each group during the training process, evaluated every 500 iterations. Each bar represents a group of 50 clients with the same iteration time. The colors within each bar indicate the mean number of times clients in the group were selected ("mean observations counter") during the last 500 training iterations leading up to the corresponding iteration.

TABLE 3. Dataset Configurations and Training Details for BSFL Evaluation.

Datasets	Task	Number of Clients	Selection Size	Learning Rate	BSFL's Beta	BSFL's Alpha	Training Time
Synthetic Dataset divided i.i.d.	Linear Regression	20	5	5×10^{-3}	1.2	3	200
Synthetic Dataset divided non-i.i.d.	Linear Regression	20	5	3×10^{-3}	1.2	3	130
Fashion-MNIST divided i.i.d.	Image Classification	500	25	5×10^{-5}	1	2	400
Fashion-MNIST divided non-i.i.d.	Image Classification	500	25	1×10^{-5}	1.2	3	200
CIFAR-10 divided i.i.d.	Image Classification	500	25	9×10^{-5}	1	2	400
CIFAR-10 divided non-i.i.d.	Image Classification	500	25	5×10^{-6}	1.2	3	200

our simulations for each of the algorithms. The simulation results demonstrate the performance of the global model on test data throughout the training process. The figures show that BSFL significantly outperforms the other algorithms in terms of both loss and accuracy percentage on the test data.

In the non-i.i.d. scenario, as before, to evaluate the regret, we divided the synthetic dataset into a small number of clients, and compared the regret of each algorithm. As shown in Figure 10 (left), even in the non-i.i.d. scenario, BSFL achieves a logarithmic regret order, in contrast to the other algorithms which achieve a linear regret order. Furthermore, Figure 10 (right) illustrates how these results in terms of regret lead to faster convergence in terms of the loss calculated on the test data, indicating superior generalization of the model trained using BSFL.

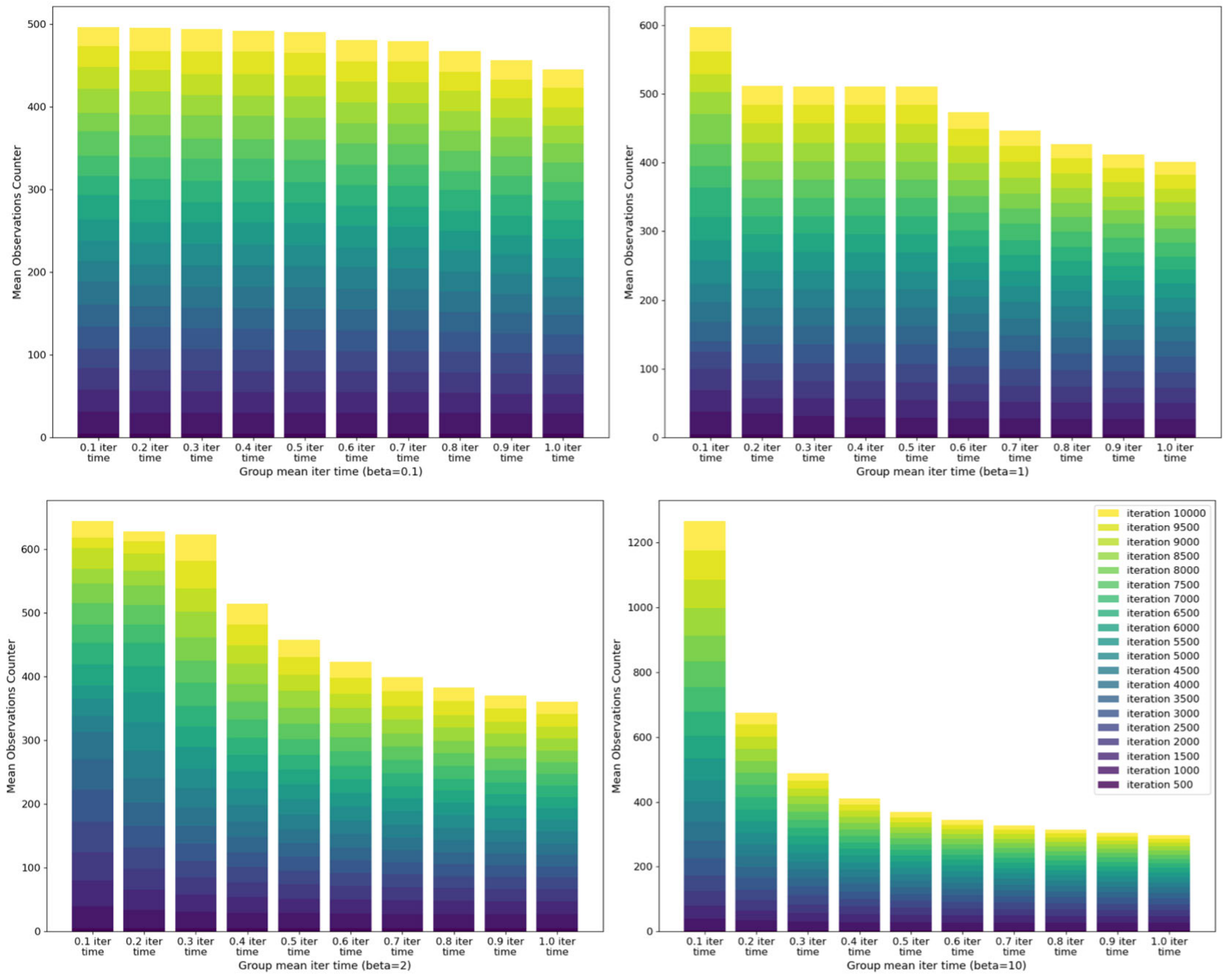


FIGURE 6. Training experiments conducted with four different beta values. Each experiment involved 10,000 iterations, and the results are displayed as bar charts showing the mean number of observations for each group during the training process, evaluated every 500 iterations. Each bar represents a group of 50 clients with the same iteration time. The colors within each bar indicate the mean number of times clients in the group were selected ("mean observations counter") during the last 500 training iterations leading up to the corresponding iteration.

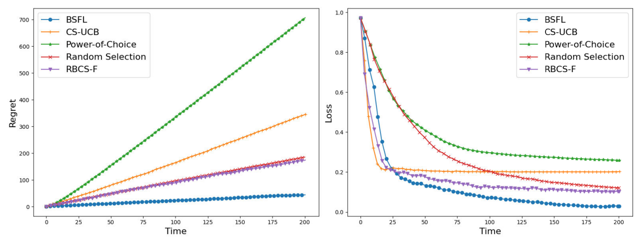


FIGURE 7. Linear regression model with synthetic data in the i.i.d. scenario. Figure (left): Regret as a function of iterations. Figure (right): Test loss as a function of latency.

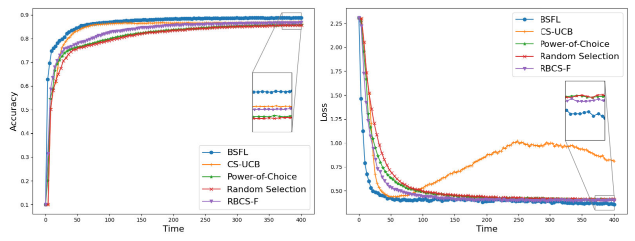


FIGURE 8. CNN model with Fashion-MNIST data in the i.i.d. scenario. Figure (left): Test accuracy as a function of latency. Figure (right): Test loss as a function of latency.

We conducted a similar experiment using real datasets and divided the data into a larger set of 500 clients, with a selection size of 25 clients per iteration. Each client contained a different number of images and varying amounts of images from each class. As shown in Figures 11 and 12, in this scenario as well, BSFL leads to faster convergence in terms of both loss and accuracy on the test data. All evaluations of the global model's performance, depicted in graphs, were conducted using test data that was not utilized for training and was not accessible to any of the clients.

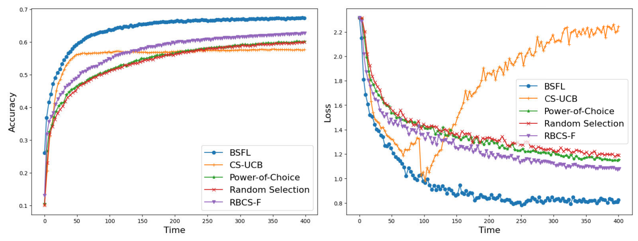


FIGURE 9. Resnet18 model with CIFAR10 data in the i.i.d. scenario. Figure (left): Test accuracy as a function of latency. Figure (right): Test loss as a function of latency.

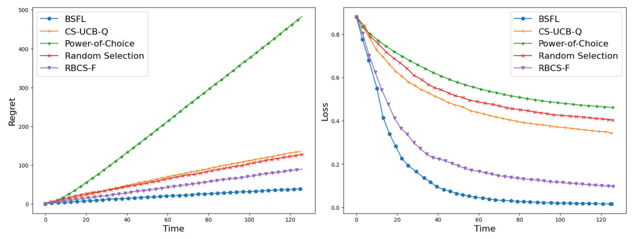


FIGURE 10. Linear regression model with synthetic data in the non-i.i.d. scenario. Figure (left): Regret as a function of iterations. Figure (right): Test loss as a function of latency.

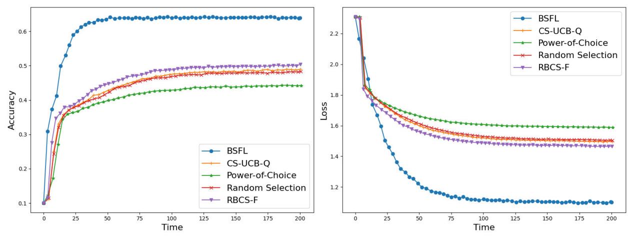


FIGURE 11. CNN model with Fashion-MNIST data in the non-i.i.d. scenario. Figure (left): Test accuracy as a function of latency. Figure (right): Test loss as a function of latency.

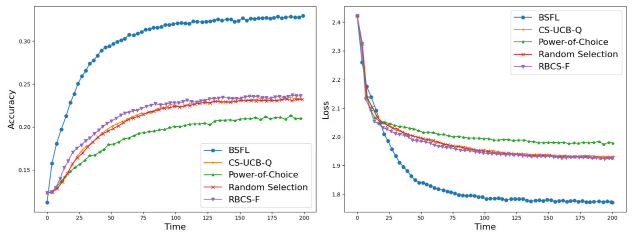


FIGURE 12. CNN model with CIFAR10 data in the non-i.i.d. scenario. Figure (left): Test accuracy as a function of latency. Figure (right): Test loss as a function of latency.

These simulation results demonstrate the strong performance of BSFL for client selection in federated learning compared to existing methods. The superior performance of the BSFL algorithm is due to several key factors. First, in terms of client data distribution, BSFL excels at managing both i.i.d. and non-i.i.d. data distributions. In i.i.d. scenarios, it prioritizes clients based on their speed while ensuring the model trains on data from all clients to support generalization. In non-i.i.d. cases, it prioritizes clients with diverse or underrepresented data, improving the model's generalization capabilities. This adaptability helps BSFL avoid overfitting and maintain robust performance across varying data distributions, a common limitation of traditional methods. Second, in terms of computing efficiency and speed, BSFL incorporates an adaptive exploration-exploitation strategy that optimizes client selection by considering both latency and the client's contribution to overall model generalization. This enables BSFL to balance the computational load and efficiently select clients contributing to the global model. Moreover, leveraging the ALSA algorithm to solve a combinatorial optimization problem improves computational efficiency and strategically selects clients with similar speeds. This approach minimizes iteration time by reducing the waiting period for slower clients, thereby accelerating the training process, a significant advantage over other methods that may suffer delays from including slower clients in the same iteration as faster ones. Third, in terms of resilience to data imbalance, BSFL demonstrates strong robustness in handling imbalances where clients vary in data volume or quality. By integrating both latency and generalization into its reward function, BSFL ensures the global model remains accurate and robust even when faced with skewed or biased data distributions, an area where other methods often struggle to maintain performance.

VII. CONCLUSION

We developed a novel MAB-based approach for client selection in FL systems, aimed at minimizing training latency while preserving the model's ability to generalize. We developed a novel algorithm to achieve this goal, dubbed Bandit Scheduling for FL (BSFL). BSFL was shown to achieve a logarithmic regret, defined as the difference in loss between BSFL and a genie with complete knowledge of all clients' latency means. Simulation results demonstrated that BSFL is superior to existing methods. As federated learning systems scale to accommodate a large number of clients, challenges arise in managing computational overhead and ensuring efficient communication and computation across diverse participants. The BSFL algorithm demonstrates strong scalability, enabled by the ALSA method, which reduces the complexity of the client selection process and facilitates the management of extensive networks. Our simulations included FL tasks with networks of up to 500 clients, consistently showing BSFL's superior performance over alternative methods, underscoring its ability to handle large-scale networks effectively. This makes BSFL a robust choice for real-world scenarios with extensive client participation. While the BSFL algorithm is highly effective in many scenarios, its reliance on the MAB framework, which assumes a stochastic process for the learning process and focuses on asymptotic regret analysis, may face challenges in systems with rapidly changing and highly dynamic environments. In such cases, reinforcement learning approaches that adapt more quickly to changes by approximating value functions could serve as complementary strategies. Future work could explore integrating such approaches with BSFL to enhance adaptability while preserving its strengths in optimizing client selection.

APPENDIX A
PROOF OF THEOREM 1

In this appendix we provide the proof for Theorem 1. The regret of BSFL (7) can be written as:

$$\begin{aligned}
R(n) &= \mathbb{E}_{\Pi}\left[\sum_{t=1}^{n} r^*(t) - r(t)\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{n} r(\mathcal{G}_t, \mathcal{H}_t) - r(\mathcal{A}_t, \mathcal{H}_t)\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{n} \Delta_t \cdot \mathbb{1}\left\{(r(\mathcal{G}_t, \mathcal{H}_t) \neq r(\mathcal{A}_t, \mathcal{H}_t))\right\}\right] \\
&\leq \Delta_{max} \cdot \mathbb{E}\left[\sum_{\mathcal{S}\in H(\mathbb{K})} N_{\mathcal{S}}(n)\right],
\end{aligned} \tag{21}$$

where Δ_t is the difference between the reward achieved by the algorithm to the highest reward that could have been achieved at iteration t (i.e., $\Delta_t = r(\mathcal{G}_t, \mathcal{H}_t) - r(\mathcal{A}_t, \mathcal{H}_t) \geq 0$), and $N_{\mathcal{S}}(n)$ is the number of iterations up until the nth iteration in which selection $\mathcal{S} \in H(\mathbb{K})$ was selected and the reward given from it was strictly less than the reward that would have been received by genie's selection in the same iteration. In addition, we define a K-dimensional counter vector $\widetilde{\mathbf{N}}(n) = (\widetilde{\mathbf{N}}_1(n), \widetilde{\mathbf{N}}_2(n), \ldots, \widetilde{\mathbf{N}}_K(n))$, corresponding to the K clients as follows. For each iteration, in which the selection $\mathcal{A}_t \in H(\mathbb{A}_t)$ achieves a lower reward than the reward of genie's selection, i.e., $r(\mathcal{A}_t, \mathcal{H}_t) < r(\mathcal{G}_t, \mathcal{H}_t)$, then the counter of the client that has been selected the fewest number of times up to this iteration among all the selected clients that were selected in this iteration is incremented by 1. Formally, for each client (say i) let $\mathcal{T}_i(n)$ denote the set of time indices up to time n that satisfy the following conditions: (i) Client i was selected, i.e., $i \in \mathcal{A}_t$ for all $t \in \mathcal{T}_i(n)$; (ii) the counter $c_{i,t}$ of client i is the minimal among all selected clients, i.e., $i = \arg\min_{k\in\mathcal{A}_t} c_{k,t}$, for all $t \in \mathcal{T}_i(n)$; and (iii) the selection $\mathcal{A}_t \in H(\mathbb{A}_t)$ achieves a lower reward than genie's selection, i.e., $r(\mathcal{A}_t, \mathcal{H}_t) < r(\mathcal{G}_t, \mathcal{H}_t)$ for all $t \in \mathcal{T}_i(n)$. Then,

$$\widetilde{\mathbf{N}}_i(n) = |\mathcal{T}_i(n)|. \tag{22}$$

Next, we aim at upper bounding $N_{\mathcal{S}}(n)$ for each $\mathcal{S} \in H(\mathbb{K})$. Note that based on the definition of $\widetilde{\mathbf{N}}(n)$, for every iteration in which the selection $\mathcal{A}_t$ has a lower reward than genie's selection, one of the coordinates in the vector $\widetilde{\mathbf{N}}(n)$ is incremented by 1. Therefore,

$$\sum_{\mathcal{S}\in H(\mathbb{K})} N_{\mathcal{S}}(n) = \sum_{k=1}^{K} \widetilde{\mathbf{N}}_k(n), \tag{23}$$

which implies

$$\mathbb{E}\left[\sum_{\mathcal{S}\in H(\mathbb{K})} N_{\mathcal{S}}(n)\right] = \sum_{k=1}^{K} \mathbb{E}\left[\widetilde{\mathbf{N}}_k(n)\right]. \tag{24}$$

Let $I_k(t)$ be the indicator for the event that $\widetilde{\mathbf{N}}_k(t)$ is incremented by 1 at iteration t. Hence, we obtain

$$\begin{aligned}
\widetilde{\mathbf{N}}_i(n) &= \sum_{t=1}^{n} \mathbb{1}\left\{(I_i(t) = 1)\right\} \\
&\leq 1 + \sum_{t=\lceil\frac{\mathbb{K}}{m}\rceil+1}^{n} \mathbb{1}\left\{(I_i(t) = 1)\right\} \\
&\leq l + \sum_{t=\lceil\frac{\mathbb{K}}{m}\rceil+1}^{n} \mathbb{1}\left\{(I_i(t) = 1, \widetilde{\mathbf{N}}_i(t) \geq l)\right\} \\
&\leq l + \sum_{t=\lceil\frac{\mathbb{K}}{m}\rceil+1}^{n} \mathbb{1}\left\{(\min_{k\in\mathcal{G}_t} \mathrm{ucb}(k,t) + \frac{\alpha}{m}\sum_{k\in\mathcal{G}_t} g_{k,\Theta}(\mathcal{H}_t)\right. \\
&\qquad \left. < \min_{k\in\mathcal{A}_t} \mathrm{ucb}(k,t) + \frac{\alpha}{m}\sum_{k\in\mathcal{A}_t} g_{k,\Theta}(\mathcal{H}_t), \widetilde{\mathbf{N}}_i(t) \geq l)\right\},
\end{aligned} \tag{25}$$

where the last inequality follows since the algorithm chooses action $\mathcal{A}_t \neq \mathcal{G}_t$ that solves (13) although the reward is maximized by $\mathcal{G}_t$. Note that according to the definition of $\widetilde{\mathbf{N}}$, for all $k \in \mathcal{A}_t$ we have: $\widetilde{\mathbf{N}}_i(t) \leq c_{k,t}$. Therefore, since in the indicator function there is an intersection with the event that $\widetilde{\mathbf{N}}_i(t) \geq l$ we have for all $k \in \mathcal{A}_t$ that: $l \leq \widetilde{\mathbf{N}}_i(t) \leq c_{k,t}$ in (25). Denote $h_{c_{k,t}} = \sqrt{\frac{(m+1)\ln t}{c_{k,t}}}$, and $\overline{\mu_{c_k}}$ denotes the sampled mean of the $\frac{\tau_{min}}{\tau_k}$ of client k after c_k observations. Using these notations, we can upper bound $\widetilde{\mathbf{N}}_i(n)$ by:

$$\begin{aligned}
&\widetilde{\mathbf{N}}_i(n) \\
&\quad\leq l + \sum_{t=\lceil\frac{\mathbb{K}}{m}\rceil+1}^{n} \mathbb{1}\left\{\min_{k\in\mathcal{G}_t}\{\overline{\mu_{k,t}} + h_{c_{k,t}}\} + \frac{\alpha}{m}\sum_{k\in\mathcal{G}_t} g_{k,\Theta}(\mathcal{H}_t)\right. \\
&\quad\left. < \min_{k\in\mathcal{A}_t}\{\overline{\mu_{k,t}} + h_{c_{k,t}}\} + \frac{\alpha}{m}\sum_{k\in\mathcal{A}_t} g_{k,\Theta}(\mathcal{H}_t), \widetilde{\mathbf{N}}_i(t) \geq l\right\} \\
&\quad\leq l + \sum_{t=\lceil\frac{\mathbb{K}}{m}\rceil+1}^{n} \mathbb{1}\left\{\min_{0\leq c_{\hat{k}_{1,t}}, c_{\hat{k}_{2,t}}, \ldots, c_{\hat{k}_{m,t}} \leq t}\right. \\
&\quad\left\{\min_{j\in\{1,\ldots,m\}}\{\overline{\mu_{c_{\hat{k}_{j,t}}}} + h_{c_{\hat{k}_{j,t}}}\} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t)\right\} \\
&\quad< \max_{l\leq c_{\tilde{k}_{1,t}}, c_{\tilde{k}_{2,t}}, \ldots, c_{\tilde{k}_{m,t}} \leq t}\left\{\min_{j\in\{1,\ldots,m\}}\{\overline{\mu_{c_{\tilde{k}_{j,t}}}} + h_{c_{\tilde{k}_{j,t}}}\}\right. \\
&\qquad\left.\left. + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\tilde{k}_{j,t},\Theta}(\mathcal{H}_t)\right\}\right\} \\
&\quad\leq l + \sum_{t=\lceil\frac{\mathbb{K}}{m}\rceil+1}^{n} \sum_{c_{\hat{k}_{1,t}}=1}^{t} \cdots \sum_{c_{\hat{k}_{m,t}}=1}^{t} \sum_{c_{\tilde{k}_{1,t}}=l}^{t} \cdots \sum_{c_{\tilde{k}_{m,t}}=l}^{t} \\
&\qquad \cdot \mathbb{1}\left\{\min_{j\in\{1,\ldots,m\}}\{\overline{\mu_{c_{\hat{k}_{j,t}}}} + h_{c_{\hat{k}_{j,t}}}\} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t)\right. \\
&\quad\left. < \min_{j\in\{1,\ldots,m\}}\{\overline{\mu_{c_{\tilde{k}_{j,t}}}} + h_{c_{\tilde{k}_{j,t}}}\} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\tilde{k}_{j,t},\Theta}(\mathcal{H}_t)\right\},
\end{aligned} \tag{26}$$

where $\{\hat{k}_{j,t} : 1 \le j \le m\}$ and $\{\tilde{k}_{j,t} : 1 \le j \le m\}$ are the clients in genie's selection and the server's selection at iteration t, respectively. Let $\hat{k}'_t$, $\tilde{k}'_t$, respectively, be the clients in genie's selection and the server's selection at iteration t that minimizes the expression in the upper bound we derived, i.e.,

$$\hat{k}'_t = \underset{\hat{k}_j \in \{\hat{k}_1,\ldots,\hat{k}_m\}}{\arg\min} \{\overline{\mu_{c_{\hat{k}_{j,t}}}} + h_{c_{\hat{k}_{j,t}}}\}, \tag{27}$$

$$\tilde{k}'_t = \underset{\tilde{k}_j \in \{\tilde{k}_1,\ldots,\tilde{k}_m\}}{\arg\min} \{\overline{\mu_{c_{\tilde{k}_{j,t}}}} + h_{c_{\tilde{k}_{j,t}}}\}. \tag{28}$$

Now, we claim that the event

$$\Bigg\{\overline{\mu_{\hat{k}'_t}} + h_{c_{\hat{k}'_t}} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t) < \overline{\mu_{\tilde{k}'_t}} + h_{c_{\tilde{k}'_t}} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\tilde{k}_{j,t},\Theta}(\mathcal{H}_t)\Bigg\}$$

implies that at least one of the 3 following events must occur:

(i) $\overline{\mu_{\hat{k}'_t}} + h_{c_{\hat{k}'_t}} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t) \le \mu_{\hat{k}'_t} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t)$;

(ii) $\overline{\mu_{\tilde{k}'_t}} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\tilde{k}_{j,t},\Theta}(\mathcal{H}_t) \ge \mu_{\tilde{k}'_t} + h_{c_{\tilde{k}'_t}} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\tilde{k}_{j,t},\Theta}(\mathcal{H}_t)$;

(iii) $\mu_{\hat{k}'_t} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t) < \mu_{\tilde{k}'_t} + 2h_{c_{\tilde{k}'_t}} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\tilde{k}_{j,t},\Theta}(\mathcal{H}_t)$.

We next prove this claim by contradiction. Assume that all three inequalities do not hold. Therefore, it follows that:

$$\begin{aligned}
\overline{\mu_{\hat{k}'_t}} + h_{c_{\hat{k}'_t}} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t) &> \mu_{\hat{k}'_t} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t) \\
&\ge \mu_{\tilde{k}'_t} + 2h_{c_{\tilde{k}'_t}} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\tilde{k}_{j,t},\Theta}(\mathcal{H}_t) \\
&> \overline{\mu_{\tilde{k}'_t}} + h_{c_{\tilde{k}'_t}} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\tilde{k}_{j,t},\Theta}(\mathcal{H}_t),
\end{aligned} \tag{29}$$

where the first transition is derived from (i), the second from (iii), the last from (ii), and all three together contradict the event. Now, we aim at upper bounding the probabilities Pr(i), Pr(ii) that events (i) and (ii) will occur:

$$\begin{aligned}
Pr(\text{i}) &= Pr\Bigg(\overline{\mu_{\hat{k}'_t}} + h_{c_{\hat{k}'_t}} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t) \le \mu_{\hat{k}'_t} + \frac{\alpha}{m}\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t)\Bigg) \\
&= Pr\big(\overline{\mu_{\hat{k}'_t}} + h_{c_{\hat{k}'_t}} \le \mu_{\hat{k}'_t}\big) \\
&\le e^{-2c^2_{\hat{k}'_t}\cdot\frac{(m+1)\ln t}{c_{\hat{k}'_t}}\cdot\frac{1}{c_{\hat{k}'_t}}} = t^{-2(m+1)},
\end{aligned} \tag{30}$$

where the inequality is due to Hoeffding's inequality. Similarly, we can upper bound Pr(ii) by the same upper bound and obtain:

$$Pr(\text{ii}) \le t^{-2(m+1)}. \tag{31}$$

To ensure that (iii) will not occur we need to put a lower bound on l (i.e., the minimum number of times a client should be selected when he has the minimum number of selections so far among the clients in the current selection):

$$\Big\{\mu_{\hat{k}'_t} + \tfrac{\alpha}{m}\textstyle\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t) < \mu_{\tilde{k}'_t} + 2h_{c_{\tilde{k}'_t}} + \tfrac{\alpha}{m}\sum_{j=1}^{m} g_{\tilde{k}_{j,t},\Theta}(\mathcal{H}_t)\Big\} \Leftrightarrow$$
$$\Big\{\mu_{\hat{k}'_t} + \tfrac{\alpha}{m}\textstyle\sum_{j=1}^{m} g_{\hat{k}_{j,t},\Theta}(\mathcal{H}_t) - \mu_{\tilde{k}'_t} - \tfrac{\alpha}{m}\sum_{j=1}^{m} g_{\tilde{k}_{j,t},\Theta}(\mathcal{H}_t) > 2h_{c_{\tilde{k}'_t}}\Big\}.$$

Denote the LHS of the last inequality by $\Delta_{\mathcal{G}_t,\mathcal{A}_t,\alpha}$. Then, for the last inequality to hold, we can demand that for every $\mathcal{A}_t$ and $\mathcal{G}_t$ selections by the algorithm and genie, respectively (which satisfy $r(\mathcal{A}_t,\mathcal{H}_t) < r(\mathcal{G}_t,\mathcal{H}_t)$):

$$\Delta_{\mathcal{G}_t,\mathcal{A}_t,\alpha} \ge 2\sqrt{\frac{(m+1)\ln t}{c_{\tilde{k}'_t}}}, \tag{32}$$

and because we have already shown that $\forall k \in \mathcal{A}_t : l \le c_{k,t}$ and $t \le n$, it is sufficient to demand

$$\Delta_{\mathcal{G}_t,\mathcal{A}_t,\alpha} \ge 2\sqrt{\frac{(m+1)\ln n}{l}}. \tag{33}$$

Therefore, for $l \ge \frac{4(m+1)\ln n}{\Delta^2_{\mathcal{G}_t,\mathcal{A}_t,\alpha}}$ for every t, or alternatively, we can choose $l = \lceil\frac{4(m+1)\ln n}{\Delta^2_{min}}\rceil$ and obtain that inequality (iii) will not be met. Hence, only one of the first two inequalities must occur, and we obtain

$$\begin{aligned}
&\mathbb{E}\Big[\widetilde{\mathbf{N}}_i(n)\Big] \\
&\le l + \sum_{t=\lceil\frac{\mathbb{K}}{m}\rceil+1}^{n}\sum_{c_{\hat{k}_{1,t}}=1}^{t}\cdots\sum_{c_{\hat{k}_{m,t}}=1}^{t}\sum_{c_{\tilde{k}_{1,t}}=l}^{t}\cdots\sum_{c_{\tilde{k}_{m,t}}=l}^{t}(Pr(\text{i})+Pr(\text{ii})) \\
&\le \left\lceil\frac{4(m+1)\ln n}{\Delta^2_{min}}\right\rceil + \sum_{t=\lceil\frac{\mathbb{K}}{m}\rceil+1}^{\infty}\sum_{c_{\hat{k}_{1,t}}=1}^{t}\cdots\sum_{c_{\hat{k}_{m,t}}=1}^{t}\sum_{c_{\tilde{k}_{1,t}}=1}^{t}\cdots\sum_{c_{\tilde{k}_{m,t}}=1}^{t} 2t^{-2(m+1)} \\
&\le \frac{4(m+1)\ln t}{\Delta^2_{min}} + 1 + \frac{\pi^2}{3}.
\end{aligned} \tag{34}$$

Finally, we can upper bound the regret by:

$$R(n) \le \Delta_{max}K\cdot\left(\frac{4(m+1)\ln n}{\Delta^2_{min}} + 1 + \frac{\pi^2}{3}\right).$$

$\square$

APPENDIX B
PROOF OF THEOREM 2

In this appendix we provide the proof for Theorem 2. From [45], the following conditions are sufficient to guarantee the convergence of cooling procedure to the state with the lowest energy (or, alternatively, the highest energy, depending on the problem formulation):

(i) Weak Reversibility: For any energy E and any two states s_1 and s_2, s_1 is reachable at height energy E from state s_2, i.e., there exists a path from s_1 to s_2 that goes only through states with energy E or higher) iff s_2 is reachable from s_1 at height E.

(ii) The temperature is set to $T_i = \frac{d}{\log(i+1)}$, where d is greater then the difference between the energies of the highest local maxima and the minimum energy state.

(iii) The Markov chain is irreducible.

Next, we prove that all conditions are met by ALSA. Condition (i) follows immediately by the definition of the neighborhoods which is symmetrical, i.e., in a graph with symmetric neighborhoods, any path from one node to another can also be in the opposite direction and go through the exact same states. Regarding condition (ii), since Δ_{max} is defined as the largest possible energy difference between any two states, it is in particular greater than the difference in energies between any local maximum and the state with the minimum energy. Therefore, condition (ii) also holds.

Finally, we need to show that condition (iii) holds. We will show this by proving that in the newly structured state graph by ALSA, from every possible state there exists a path that reaches s^*, and due to the symmetric neighborhoods, this will complete the proof. Denote s_0 as some arbitrary state (selection). Define the state s_1 to be $s_1 = \left(s_0 \backslash \{\arg\min_{k\in\mathcal{S}_0} \text{ucb}(k,t)\}\right) \cup \{\arg\min_{k\in\mathcal{S}^*} \text{ucb}(k,t)\}$. Note that s_1 and s_0 are neighboring states. For $j = 1, 2, 3, \ldots$ let us define the rest of the path with two phases (P1,P2) as follows:

(P1) As long as
$\arg\min_{k\in\mathcal{S}^*} \text{ucb}(k,t) \neq \arg\min_{k\in\mathcal{S}_j} \text{ucb}(k,t)$:

$$\text{define } s_{j+1} = \left(s_j \backslash \{\arg\min_{k\in\mathcal{S}_j} \text{ucb}(k,t)\}\right) \cup \left\{\arg\max_{k\in\mathcal{S}^*} \text{ucb}(k,t)\right\}.$$

(P2) After Phase 1 ends, as long as $s^* \neq s_j$:

$$\text{define } s_{j+1} = \left(s_j \backslash \{\arg\min_{k\in\mathcal{S}_0} g_{k,\Theta}\left(\mathcal{H}_t\right)\}\right) \cup \{\arg\max_{k\in\mathcal{S}^*} g_{k,\Theta}\left(\mathcal{H}_t\right)\}.$$

Note that each one of the phases lasts a finite amount of iterations, i.e., $\exists n \in \mathbb{N} : s_n = s^*$. Note that for every $j \in \mathbb{N}$ the state s_j and s_{j+1} are neighbors, which implies that $P = (s_0, s_1, \ldots, s_n = s^*)$ defined by the last phases is a path from s_0 to s^*, which completes the proof. □

ACKNOWLEDGMENT

An earlier version of this paper was presented in part at the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2023 [DOI: 10.1109/ICASSP49357.2023.10095274].

REFERENCES

[1] D. B. Ami, K. Cohen, and Q. Zhao, "Client selection for generalization in accelerated federated learning: A bandit approach," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–5.

[2] 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. Artif. Intell. Statist.*, 2016, pp. 1273–1282.

[3] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Koneč ný, S. Mazzocchi, H. B. McMahan, T. V. Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," in *Proc. Mach. Learn. Syst.*, 2019, pp. 374–388.

[4] T. Gafni, N. Shlezinger, K. Cohen, Y. C. Eldar, and H. V. Poor, "Federated learning: A signal processing perspective," *IEEE Signal Process. Mag.*, vol. 39, no. 3, pp. 14–41, May 2022.

[5] R. Srikant and L. Ying, *Communication Networks: An Optimization, Control, and Stochastic Networks Perspective*. Cambridge, U.K.: Cambridge Univ. Press, 2013.

[6] M. M. Amiri and D. Gündüz, "Federated learning over wireless fading channels," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3546–3557, May 2020.

[7] M. M. Amiri and D. Gündüz, "Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air," *IEEE Trans. Signal Process.*, vol. 68, pp. 2155–2169, 2020.

[8] T. Sery and K. Cohen, "On analog gradient descent learning over multiple access fading channels," *IEEE Trans. Signal Process.*, vol. 68, pp. 2897–2911, 2020.

[9] T. Sery, N. Shlezinger, K. Cohen, and Y. C. Eldar, "Over-the-air federated learning from heterogeneous data," *IEEE Trans. Signal Process.*, vol. 69, pp. 3796–3811, 2021.

[10] R. Paul, Y. Friedman, and K. Cohen, "Accelerated gradient descent learning over multiple access fading channels," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 2, pp. 532–547, Feb. 2022.

[11] J. Koneč ný, H. Brendan McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*.

[12] C. Tekin and M. Liu, "Online learning of rested and restless bandits," *IEEE Trans. Inf. Theory*, vol. 58, no. 8, pp. 5588–5611, Aug. 2012.

[13] I. Bistritz and A. Leshem, "Distributed multi-player bandits—A game of thrones approach," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–11.

[14] T. Gafni and K. Cohen, "Learning in restless multiarmed bandits via adaptive arm sequencing rules," *IEEE Trans. Autom. Control*, vol. 66, no. 10, pp. 5029–5036, Oct. 2021.

[15] T. Gafni and K. Cohen, "Distributed learning over Markovian fading channels for stable spectrum access," *IEEE Access*, vol. 10, pp. 46652–46669, 2022.

[16] T. Gafni, M. Yemini, and K. Cohen, "Learning in restless bandits under exogenous global Markov process," *IEEE Trans. Signal Process.*, vol. 70, pp. 5679–5693, 2022, doi: 10.1109/TSP.2022.3224790.

[17] P. Kairouz et al., "Advances and open problems in federated learning," *Found. Trends Registered Mach. Learn.*, vol. 14, nos. 1–2, pp. 1–210, 2021.

[18] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[19] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," 2019, *arXiv:1905.10497*.

[20] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–7.

[21] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1188–1200, Feb. 2021.

[22] S. Abdulrahman, H. Tout, A. Mourad, and C. Talhi, "FedMCCS: Multicriteria client selection model for optimal IoT federated learning," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4723–4735, Jun. 2020.

[23] E. Rizk, S. Vlaski, and A. H. Sayed, "Federated learning under importance sampling," *IEEE Trans. Signal Process.*, vol. 70, pp. 5381–5396, 2022.

[24] H. H. Yang, A. Arafa, T. Q. S. Quek, and H. V. Poor, "Age-based scheduling policy for federated learning in mobile edge networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 8743–8747.

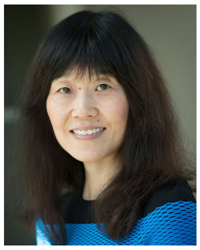
[25] W. Zhang, X. Wang, P. Zhou, W. Wu, and X. Zhang, "Client selection for federated learning with non-IID data in mobile edge computing," *IEEE Access*, vol. 9, pp. 24462–24474, 2021.

[26] M. E. Ozfatura, J. Zhao, and D. Gündüz, "Fast federated edge learning with overlapped communication and computation and channel-aware fair client scheduling," in *Proc. IEEE 22nd Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Sep. 2021, pp. 311–315.

[27] M. Mohamed, A. Houdou, H. Alami, K. Fardousse, and I. Berrada, "NIFL: A statistical measures-based method for client selection in federated learning," *IEEE Access*, vol. 10, pp. 124766–124776, 2022.
[28] W. Xia, T. Q. S. Quek, K. Guo, W. Wen, H. H. Yang, and H. Zhu, "Multi-armed bandit-based client scheduling for federated learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 11, pp. 7108–7123, Nov. 2020.
[29] N. Yoshida, T. Nishio, M. Morikura, and K. Yamamoto, "MAB-based client selection for federated learning with uncertain resources in mobile networks," in *Proc. IEEE Globecom Workshops (GC Wkshps*, Dec. 2020, pp. 1–6.
[30] Y. Jee Cho, S. Gupta, G. Joshi, and O. Yagan, "Bandit-based communication-efficient client selection strategies for federated learning," in *Proc. 54th Asilomar Conf. Signals, Syst., Comput.*, Nov. 2020, pp. 1066–1069.
[31] B. Xu, W. Xia, J. Zhang, T. Q. S. Quek, and H. Zhu, "Online client scheduling for fast federated learning," *IEEE Wireless Commun. Lett.*, vol. 10, no. 7, pp. 1434–1438, Jul. 2021.
[32] T. Huang, W. Lin, W. Wu, L. He, K. Li, and A. Y. Zomaya, "An efficiency-boosting client selection scheme for federated learning with fairness guarantee," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1552–1564, Jul. 2021.
[33] Q. Zhao, "Multi-armed bandits: Theory and applications to online learning in networks," *Synth. Lectures Commun. Netw.*, vol. 12, no. 1, pp. 1–165, Nov. 2019.
[34] N. Levine, K. Crammer, and S. Mannor, "Rotting bandits," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, Jan. 2017, pp. 3074–3083.
[35] W. Chen, Y. Wang, and Y. Yang, "Combinatorial multi-armed bandit: General framework and applications," in *Proc. Int. Conf. Mach. Learn.*, Feb. 2013, pp. 151–159.
[36] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2016, pp. 1–9.
[37] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proc. Int. Conf. Mach. Learn.*, Feb. 2019, pp. 4615–4625.
[38] Y. Shi, H. Yu, and C. Leung, "Towards fairness-aware federated learning," 2021, *arXiv:2111.01872*.
[39] H. Xu and S. Mannor, "Robustness and generalization," *Mach. Learn.*, vol. 86, no. 3, pp. 391–423, Mar. 2012.
[40] X. Zhu and X. Wu, "Class noise vs. Attribute noise: A quantitative study," *Artif. Intell. Rev.*, vol. 22, no. 3, pp. 177–210, Nov. 2004.
[41] A. Atla, R. Tada, V. S. Sheng, and N. K. Singireddy, "Sensitivity of different machine learning algorithms to noise," *J. Comput. Sci. Colleges*, vol. 26, no. 5, pp. 96–103, May 2011.
[42] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, "A study of the effect of different types of noise on the precision of supervised learning techniques," *Artif. Intell. Rev.*, vol. 33, no. 4, pp. 275–306, Apr. 2010.
[43] S. Gupta and A. Gupta, "Dealing with noise problem in machine learning data-sets: A systematic review," *Proc. Comput. Sci.*, vol. 161, pp. 466–474, Jan. 2019.
[44] Y. J. Cho, J. Wang, and G. Joshi, "Towards understanding biased client selection in federated learning," in *Proc. 25th Int. Conf. Artif. Intell. Statist. (AISTATS)*, Mar. 2022, pp. 10351–10375.
[45] B. Hajek, "Cooling schedules for optimal annealing," *Math. Operations Res.*, vol. 13, no. 2, pp. 311–329, May 1988.
[46] Y. Jee Cho, J. Wang, and G. Joshi, "Client selection in federated learning: Convergence analysis and power-of-choice selection strategies," 2020, *arXiv:2010.01243*.

DAN BEN AMI is currently pursuing the Ph.D. degree in electrical and computer engineering with the Ben-Gurion University of the Negev, Israel. His main research interests include statistical inference and machine learning, federated learning, computer vision, and deep learning.

KOBI COHEN (Senior Member, IEEE) received the B.Sc. and Ph.D. degrees in electrical engineering from Bar-Ilan University, Ramat Gan, Israel, in 2007 and 2013, respectively. He was with the Coordinated Science Laboratory, University of Illinois at Urbana–Champaign, from August 2014 to July 2015, and the Department of Electrical and Computer Engineering, University of California at Davis, from November 2012 to July 2014, as a Postdoctoral Research Associate. In October 2015, he joined the School of Electrical and Computer Engineering, Ben-Gurion University of the Negev (BGU), Be'er Sheva, Israel, where he is currently an Associate Professor. He is also a member of the Cyber Security Research Center and the Data Science Research Center, BGU. His main research interests include statistical inference and learning, signal processing, communication networks, decision theory, and stochastic optimization with applications to large-scale systems, cyber systems, and wireless and wireline networks. His awards and honors include, highlighting in top 50 popular paper list, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS (2019 and 2020) for paper: Deep Multi-User Reinforcement Learning for Distributed Dynamic Spectrum Access; and highlighting in popular paper list, *IEEE Signal Processing Magazine* (2022) for paper: Federated Learning: A Signal Processing Perspective. He received the Best Paper Award in the International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt), in 2015; the Feder Family Award (second prize) by the Advanced Communication Center, Tel Aviv University, in 2011; and the President Fellowship (2008–2012) and top Honor List's prizes (2006, 2010, and 2011) from Bar-Ilan University. Since 2021, he has been serving as an Associate Editor for IEEE TRANSACTIONS ON SIGNAL PROCESSING and IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS.



QING ZHAO (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Cornell University, Ithaca, NY, USA, in 2001. In 2015, she joined Cornell University, where she is currently the Joseph C. Ford Professor of engineering. Prior to that, she was a Professor with the ECE Department, University of California at Davis, Davis, CA, USA. Her research interests include sequential decision theory, stochastic optimization, machine learning, and algorithmic theory with applications in infrastructure, communications, and social-economic networks. She is a Marie Skodowska-Curie Fellow of the European Union Research and Innovation Program; and the Jubilee Chair Professor of Chalmers University, from 2018 to 2019 (sabbatical leave). She was a recipient of the 2010 IEEE Signal Processing Magazine Best Paper Award and the 2000 Young Author Best Paper Award from IEEE Signal Processing Society. She is a Distinguished Lecturer of the IEEE Signal Processing Society.