Vu, T. T., Ngo, D. T., Ngo, H. Q., Dao, M. N., Tran, N. H., & Middleton, R. H. (2022). Joint Resource Allocation to Minimize Execution Time of Federated Learning in Cell-Free Massive MIMO. *IEEE Internet of Things Journal, 9*(21), 21736–21750.

Available online:  https://doi.org/10.1109/JIOT.2022.3183295
Joint Resource Allocation to Minimize Execution Time of Federated Learning in Cell-Free Massive MIMO

Tung T. Vu, Member, IEEE, Duy T. Ngo, Senior Member, IEEE, Hien Quoc Ngo, Senior Member, IEEE, Minh N. Dao, Nguyen H. Tran, Senior Member, IEEE, and Richard H. Middleton, Fellow, IEEE

Abstract—Due to its communication efficiency and privacy-preserving capability, federated learning (FL) has emerged as a promising framework for machine learning in 5G-and-beyond wireless networks. Of great interest is the design and optimization of new wireless network structures that support stable and fast operation of FL. Cell-free massive multiple-input-multiple-output (CFmMIMO) turns out to be a suitable candidate, which allows each communication round in the iterative FL process to be stably executed within a large-scale coherence time. Aiming to reduce the total execution time of the FL process in CFmMIMO, this paper proposes choosing only a subset of available users to participate in FL. An optimal selection of users with favorable link conditions would minimize the execution time of each communication round, while limiting the total number of communication rounds required. Toward this end, we formulate a joint optimization problem of user selection, transmit power, and processing frequency, subject to a predefined minimum number of participating users to guarantee the quality of learning. We then develop a new algorithm that is proven to converge to the neighbourhood of the stationary points of the formulated problem. Numerical results confirm that our proposed approach significantly reduces the FL total execution time over baseline schemes. The time reduction is more pronounced when the density of access point deployments is moderately low.

Index Terms— Cell-free massive MIMO, federated learning, execution time minimization.

I. INTRODUCTION

The numbers of mobile devices and connections have been growing significantly in recent years. According to Cisco [1], while the number of global mobile devices is expected to reach 13.1 billion by 2023, more than 10% of this figure will have 5G connections. These devices generate a vast amount of data, which in turn enable a wide range of on-device artificial intelligence (AI) services, such as traffic navigation, indoor localization, image recognition, natural language processing, and augmented reality [2]–[5]. However, it is impractical to use conventional centralized approaches to train AI models (especially those by deep neural networks) at mobile devices. As such approaches store and process data at distant cloud centers, they find it extremely challenging to support delay-critical applications. Moreover, uploading user raw data to distant cloud servers also raises serious concerns about user’s data privacy [6].

Federated learning (FL) has recently emerged as a promising solution for AI model training at wireless devices with a certain guarantee of data privacy [7]–[12]. An FL process is iterative and involves several communication rounds. In each communication round, users (UEs) compute their local model updates by using their local training data, followed by sending these updates to a central server. The central server aggregates the received local updates to generate a global model update, which is then sent back to the UEs for their subsequent local computation. The FL process terminates when a prescribed level of learning accuracy is attained; at which point, a learning model is established. Here, since only model updates (instead of raw training data) is shared between UEs and a central server, data privacy of each UE is protected. Furthermore, as a model update is much smaller in size than raw training data, sending a model update requires a much shorter amount of time.

In the literature, there are two main research directions that study FL in wireless network environments. The learning-oriented direction aims to develop FL frameworks that improve learning performance such as test accuracy. They do so by mitigating the detrimental effects of wireless transmissions, such as channel fading and estimation errors, on FL [13]–[16]. On the other hand, the communication-oriented direction aims to develop communication schemes that optimize certain performance metrics for communications. Examples of these metrics include execution time (in seconds) and energy consumption (in Joule) of an FL process executed “over-the-air” [17]–[21].

The focus of this paper is on the communication-oriented direction. Here, [17]–[20] develop new wireless network designs that use frequency-division multiple access (FDMA) and time-division multiple access (TDMA) to support FL. In these works, the transmission time of each FL communication round could be significantly prolonged when the number of UEs is large. To enable stable and fast FL over wireless media, [21]–[23] propose to use cell-free massive multiple-input multiple-output (CFmMIMO) to assist the process of FL. In a CFmMIMO network, UEs are simultaneously served over the same frequency band, and hence, CFmMIMO can offer very high macro diversity and multiplexing gains [24]. As a result, it can uniformly provide very high data rates for all users in the network which enables the stable and fast FL. Paper [21] shows that the CFmMIMO network can achieve a much shorter FL execution time than that of conventional TDMA/FDMA networks. We note that in a wireless network, the information transmitted is not necessarily private. However, in a wireless network with FL, user privacy is better...
protected since only the model updates (instead of raw data) are sent from devices to a central processing unit. While the risk of privacy leakage exists in a wireless FL network, it does not overtake the benefits of FL. We also note that more advanced FL algorithms to secure FL in wireless networks have been recently studied (see [25] and references therein).

**Advanced FL Algorithms**

We also note that more advanced FL algorithms to secure FL in wireless networks have been recently studied (see [25] and references therein). These algorithms are designed to protect the model updates (instead of raw data) sent from devices to a central processing unit. While the risk of privacy leakage exists in a wireless FL network, it does not overtake the benefits of FL.

**Scope of Research**

In this paper, we employ CFmMIMO as the underlying wireless network structure and propose novel resource allocation schemes to support a wireless FL process. Here, we use a conjugate beamforming/matched filtering scheme at the APs. Implemented locally at each AP, this scheme has been shown to perform well when the number of APs is large. The overarching objective is to minimize the total FL execution time, which is a product of the number of FL communication rounds and the execution time of each round.

**UE Selection**

To achieve the above design objective, we propose allowing only a subset of UEs with favorable links to participate in the learning process. Doing so would reduce the execution time of one FL communication round, albeit at the expense of requiring more FL communication rounds for the FL process to converge. Our aim is to devise a UE selection policy that strikes an optimal balance between these two conflicting outcomes, thereby offering a minimum total FL execution time.

This important observation leads to two research questions:

Q1) What is the minimum number of participating UEs $N_{QoL}$ above which the test accuracy remains acceptable?

Q2) For a given $N_{QoL}$, how to select UEs with favorable links to minimize the total FL execution time?

In this paper, we focus on Q2) and leave Q1) for future research.

**New Optimization-Based Design and Results**

UE selection has tight relationships with the allocation of other resources. Specifically, the decision which UE is selected affects how to allocate transmit power and computing frequency, and vice versa. Therefore, it is not clear and difficult to see how to find manually or heuristically the favorable links and determine the exact number of the selected UEs for minimizing the FL execution time. Motivated by this observation, we propose a joint optimization approach for UE selection and resource allocation. In particular, to minimize the execution time of one FL process, the FL execution time is first formulated as a function of UE selection, transmit power, and computation frequency. A mixed-integer two-stage stochastic nonconvex problem of minimizing the FL execution time is then formulated and subjected to the constraints that reflect the practical relationships among all the variables, and the practical constraints on maximum powers at the APs and UEs, imperfect channel estimation, and a minimum number of selected UEs. The considered problem is different from that in [22] and [23] which aim to reduce the execution time in an FL communication round, rather than minimizing the total FL execution time.

To solve the formulated problem, we propose a new algorithm that is proven to converge to the neighborhood of its stationary points. The mathematical challenges lie in the binary nature of UE selection variables and the coupling among the optimizing variables. Structure-wise, this optimization problem is much more complex than those considered in [21] and [23]. The algorithm developed in this paper is able to handle the binary constraints efficiently, whilst meeting all the
strict conditions for solving two-stage stochastic nonconvex problems [32]. Numerical results with practical parameter settings verify the convergence of the proposed algorithm. They also show that our UE selection approach can cut the total FL execution time by more than half compared to the baseline schemes.

Notation and Paper Organization: In this paper, bold-faced symbols are used for vectors and capitalized boldface symbols for matrices. $\mathbb{R}^d$ is a space where its elements are real vectors with length $d$. $X^*$ and $X^H$ are the conjugate and conjugate transposition of a matrix $X$, respectively. $CN(0, Q)$ denotes the circularly symmetric complex Gaussian distribution with zero mean and covariance $Q$. $N(0, V)$ denotes the normal distribution with zero mean and variance $V$. $\nabla g$ is the gradient of a function $g$. $E\{x\}$ denotes the expected value of a random variable $x$. For ease of reference, the symbols frequently used throughout the paper are listed in Table I.

The rest of this paper is organized as follows. Section II introduces a CFmMIMO system model to support a standard FL framework. Section III formulates the optimization problem of minimizing the FL execution time, whereas Section IV proposes a new algorithm to solve this problem. Section V verifies the performance of the developed algorithm through numerical examples. Finally, Section VI concludes the paper.

II. CELL-FREE MASSIVE MIMO SYSTEM MODEL TO SUPPORT WIRELESS FEDERATED LEARNING

We consider the CFmMIMO network model illustrated in Fig. 1. Let $\mathcal{N} = \{1, \ldots, N\}$ be the UE set in the network. The UEs are served by a set of APs $\mathcal{M} = \{1, \ldots, M\}$ via wireless access links with the same time-frequency resource [33], [34]. The transmission between the APs and UEs are under the time-division duplexing operation with channel reciprocity. The APs and UEs are each equipped with a single antenna. These APs are assumed to connect to a central processing unit (CPU) via backhaul links with high-capacity. As such, the transmission times between the CPU and all the APs are considered negligible [24]. The CPU and UEs act as the central server and the clients in an FL process, respectively. Here, the APs relay the model updates between the CPU and the UEs.

### A. UE Selection Model and Standard Federated Learning Framework

First, we propose to select a subset of UEs out of $\mathcal{N}$ original UEs to participate in an FL process. Let variable $a_k$ indicate whether a UE $k \in \mathcal{N} = \{1, \ldots, N\}$ is selected to partake in an FL process, i.e.,

$$ a_k \triangleq \begin{cases} 1, & \text{if UE } k \text{ is selected}, \\ 0, & \text{otherwise}. \end{cases} $$

Let $\tilde{\mathcal{N}}$ be the set of selected UEs. Denote by $\tilde{\mathcal{N}} \triangleq \sum_{k \in \mathcal{N}} a_k \leq N$ the cardinality of $\tilde{\mathcal{N}}$ or the number of selected UEs, and by $a \triangleq [a_1, \ldots, a_N]^T$ the vector of UE selection. We insist that

$$ \sum_{k \in \mathcal{N}} a_k \geq N_{\text{QoL}}, $$

where $N_{\text{QoL}}$ is a predefined minimum number of participating UEs, chosen such that an acceptable quality of learning (QoL) is achieved. As noted in Sec. I, how to determine the exact value of $N_{\text{QoL}}$ is out of the scope of this paper.

Next, the above $\tilde{\mathcal{N}}$ selected UEs will participate in an FL process that adopts the standard FL framework with a synchronous aggregation mode² [29], [30], [39]–[42]. In general, it is an iterative process consisting of multiple communication rounds. Each communication round involves the following basic steps (S1)–(S4):

(S1) A central server sends the global update to all the selected UEs.

(S2) Upon receiving the global update from the central server, the UEs solve their local learning problems (based on their local data set), and compute their local model updates.

²FL algorithms with the synchronous aggregation mode wait to receive all local model updates sent from UEs before aggregation, while the FL algorithms with an asynchronous aggregation mode do not. The FL algorithms with synchronous aggregation normally offer better convergence rate and accuracy than the FL algorithms operating with asynchronous aggregation. FL algorithms with synchronous aggregation are well studied [38], while research on improvement of learning performance of the FL algorithms with asynchronous aggregation is still in its infancy. As such, our paper focuses on resource allocation for supporting FL with synchronous aggregation [29], [30], [39]–[42].
Upon receiving all the local updates from UEs, the central server computes the global update by aggregation.

The communication rounds repeat until a certain level of test accuracy is attained at the server. At which point, the server terminates the whole FL process.

**B. CFmMIMO System Model for Wireless Federated Learning**

Now, we will describe in detail how a communication round outlined in Sec. II-A is realized by CFmMIMO.

**Assumption (A1):** All selected UEs execute steps (S2) and (S3) in a synchronized manner. Within a step (S2) or (S3), if a UE completes its operation before other UEs, it must wait until all other UEs complete their respective operation before proceeding to the next step.

**Remark 1.** It should be noted that synchronising the operations of UEs at the Steps (S2) and (S3) would cause some waste of radio resources because of the wait time among the UEs. Instead, an asynchronous design may potentially reduce the execution time by removing the wait time. However, in the asynchronous design, some UEs may finish Step (S2) while other UEs have not even completed Step (S1) yet. Therefore, it is difficult for the APs to know exactly when they need to switch to the uplink mode to receive local model updates from the UEs. The solutions for overcoming this difficulty may make the designs of signal processing and signalling much more complex. Based on this observation, this work focuses on the proposed synchronous communication design for ease of implementation, and leaves the asynchronous design for future work.

**Assumption (A2):** Each FL communication round is completely executed within a large-scale coherence time of the wireless channel [21].

**Remark 2.** Assumption 2 is realistic in many practical scenarios, such as outdoor systems with moderately low user mobility (e.g., dense urban areas) and indoor systems (e.g., office buildings, stadiums, cinema theaters, factory). An example is a system that supports users’ speed of \( v \leq 7.5 \text{ m/s} = 27 \text{ km/h} \). With a conventional carrier frequency \( f_c = 1 \text{ GHz} \) [34], the coherence time of the channels is \( T_c \geq \frac{\alpha}{v_c} = 10 \text{ ms} \), where \( \alpha = 3 \times 10^8 \text{ m/s} \) is the speed of light. According to the measurements in [43], the large-scale coefficients can stay relatively unchanged for at least 100 times longer than the coherence time. Therefore, the large-scale coherence time is \( T_c \geq 100T_c \geq 1 \text{ s} \). Let \( S = 4 \text{ KB} = 32 \text{ Kb} \) be the size of global and local updates [42]. For a 5G system with a user-experienced downlink rate of \( R_d = 100 \text{ Mbps} \) and uplink rate of \( R_u = 50 \text{ Mbps} \) [44], the time durations for downlink and uplink transmissions are \( T_d = \frac{S}{R_d} = 0.32 \text{ ms} \), \( T_u = \frac{S}{R_u} = 0.64 \text{ ms} \), respectively. Now, let \( D_s = 10^6 \) be the number of data samples at each UE, \( c = 20 \) cycles/sample be the number of CPU cycles required to process one data sample, \( f = 3 \times 10^9 \) cycles/s be the frequency of processing CPU cycles, and \( N_I = 5 \) be the number of iterations of processing the local data at UEs [42]. Then, the time duration needed to compute the local update is \( T_c = N_L \frac{cF_p}{T} = 33.3 \text{ ms} \). Therefore, the execution time of one FL communication round is \( T_c = T_d + T_u + T_u = 34.26 \text{ ms} \ll 1 \text{ s} \leq T_c \). Note that the number of collected data samples at UEs in FL applications is normally small (e.g., \( 10^3 \leq 10^5 \)) [45]. Therefore, the time of computing local updates should be much shorter than that in our example. This means in practice, the executive time of one FL communication round can even be much smaller than the large-scale coherence time.

On the other hand, Assumption 2 does not impose any practical challenge for deploying cell-free massive MIMO in indoor scenarios. The issues of deployment complexity and high front-haul capacity of cell-free massive MIMO in indoor scenarios can be effectively resolved by using an appropriate architecture, namely radio stripe system [46]–[48]. In this system, each antenna element is effectively an AP. Then, a large number of small-size antennas are put in a cable or a radio stripe. Inside the radio stripe, antennas are connected to their associated antenna processing units (APUs). The APUs and antennas are power-supplied via a shared bus that is serially located in the radio stripe. Finally, each radio stripe is connected to one or multiple central processing units. The radio stripe system turns the long front-haul cables into plug-and-play radio stripes, which effectively and flexibly implement the star topology and improve the coverage of cell-free massive MIMO. The required front-haul capacity of each radio stripe is proportional to the sum rate of data streams for transmitting to or receiving from the users at the maximum network load. Since there is a large number of antennas in the radio stripe system, the required front-haul capacity can be reduced by reducing the number of users served per radio stripe with a proper design of UE-radio stripe association [46].

With the above assumptions, we are ready to implement Steps (S1)-(S4) by CFmMIMO, as illustrated in Fig. 2.

1) **Step (S1):** This step has the following two phases, both executed within a coherence block.

**Uplink channel estimation:** Uplink pilot sequences are sent by all the UEs to all the APs simultaneously in each small-scale coherence time to help estimate channels. These estimates will later be used to construct downlink signal beams.

Here, we assume time-division duplexing where both uplink and downlink channel information is acquired at the APs via uplink channel estimation [24]. Denote by \( \tau_c \) the number of samples of each coherence block, and by \( \tau_t \) (samples) the length of one pilot sequence. Let \( \sum_{k=1}^{N_c} \varphi_k \in \mathbb{C}^{N_c 	imes 1} \) be the pilot sequence transmitted from a UE \( k \in \mathcal{N} \), where \( \| \varphi_k \|^2_2 = 1, \forall k \in \mathcal{N} \). Denote by \( g_{mk} = (\beta_{mk})^{1/2} \hat{g}_{mk} \) the channel from a UE \( k \) to an AP \( m \), where \( \beta_{mk} \) and \( \hat{g}_{mk} \sim \mathcal{C}\mathcal{N}(0, 1) \) are the large-scale fading and small-scale fading channel coefficients, respectively. At the AP \( m \), \( g_{mk} \) is estimated by using the received pilots and the minimum mean-square error (MMSE) estimation. The MMSE estimate \( \hat{g}_{mk} \) of \( g_{mk} \) is a random variable distributed according to \( \mathcal{C}\mathcal{N}(0, \hat{\sigma}_{mk}^2) \), where \( \hat{\sigma}_{mk}^2 = \sum_{k \in \mathcal{N}} \frac{\tau_t \beta_{mk}}{\tau_c} \left( |\varphi_k|^2_2 \right)^{1/2} \) [24].

**Downlink transmission:** Denote by \( S_k \) (bits) the data size of the (same) global downlink model update for each selected
UE. Since the global downlink model update can include thousands of model weights, $S_d$ is typically large. At the CPU, the global model update intended for UE $k$ is encoded into many symbols $s_{d,k,i} \sim \mathcal{CN}(0, 1), \forall i \in \{1, \ldots, L_k\}$. The number of symbols $L_k$ depends on the data size of the model update as well as the data rate. For ease of presentation, hereafter, we drop the index $i$ from $s_{d,k,i}$ and let $R_{d,k}$ be the data rate of the global update intended for a UE $k \in \mathcal{N}$.

To transmit the symbols received from the CPU, the APs first use conjugate beamforming to precode these symbols before broadcasting them to all the selected UEs. Specifically, the transmitted signal at an AP $m$ is given as $x_{d,m} = \sqrt{\rho_d} \sum_{k \in \mathcal{M}} \sqrt{\eta_{mk}} (g_{mk})^* s_{d,k}$, where $\rho_d$ is the maximum normalized transmit power at each AP and $\eta_{mk}, \forall m \in \mathcal{M}, k \in \mathcal{N}$, is a power control coefficient. The transmitted power at the AP $m$ is required to meet the average normalized power constraint, i.e., $E\{|x_{d,m}|^2\} \leq \rho_d$, which can also be expressed as the following per-AP power constraint:

$$\sum_{k \in \mathcal{N}} \sigma^2_{mk} \eta_{mk} \leq 1, \forall m. \quad (3)$$

Since no power should be allocated to the non-selected UEs, we have

$$\forall k \in \mathcal{N} : a_k = 0, \text{ then } \forall m \in \mathcal{M}, \eta_{mk} = 0. \quad (4)$$

The achievable downlink data rate at a UE $k$ is given in (5) [24, (24)] (shown at the top of next page), where $\eta \triangleq \{\eta_{mk}\}_{m \in \mathcal{M}, k \in \mathcal{N}}$, and $B$ is the bandwidth. The numerator in the log function represents the power of the desired signals normalized by noise. The three terms in the denominator in the log function represent the normalized-by-noise power of the pilot contamination, inter-UE interference and noise, respectively.

The transmission time from the APs to a UE $k$ is given by

$$t_{d,k}(a_k, \eta) = \frac{a_k S_d}{R_{d,k}(\eta)}, \quad (6)$$

where $a_k \neq 0$. Here, (6) captures the fact that the non-selected UEs do not receive any intended downlink transmission, i.e., $t_{d,k} = 0$, for any $k \in \mathcal{N}$ with $a_k = 0$.

2) Step (S2): After receiving the global update, each UE $k$ computes its local model update by using its local dataset.

**Computation delay:** Let $c_k$ (cycles/sample) be the number of processing cycles required to process one data sample at a UE $k$. Assume that $c_k$ is known a priori by an offline measurement [49]. Denote by $D_k$ (samples) and $f_k$ (cycles/s) the size of the local data set and the processing frequency of the UE $k$, respectively. The total computing time at the UE $k$ is expressed as [42]

$$t_{c,k} = \frac{a_k LD_{k} c_k}{f_k}, \quad (7)$$

where $a_k \neq 0$, $L$ is the number of local computing iterations, and $D_{k} c_k$ is the computing time of each iteration over the local training data set at the UE $k$. Similarly, because the non-selected UEs do not compute their local models, we have

$$\forall k \in \mathcal{N} : a_k = 0, t_{c,k} = 0. \quad (8)$$

3) Step (S3): This step has the following two phases, both executed within a coherence block.

**Uplink channel estimation:** This phase uses the same pilot assignment and channel estimation techniques in the uplink channel estimation of Step (S1). The MMSE estimate $\hat{g}_{mk}$ of $g_{mk}$ is thus a random variable distributed according to $\mathcal{CN}(0, \sigma^2_{mk})$, where $\sigma^2_{mk} = \sum_{c \in \mathcal{C}, k \in \mathcal{M}} |\beta_{mc}\tau_{pc}\beta_{md}|^2|\psi_{kd}|^2 + 1$.

**Uplink transmission:** Similar to Step (S1), for ease of presentation, a symbol $s_{u,k} \sim \mathcal{CN}(0, 1)$ is encoded for the local model update of a UE $k$. The symbol $s_{u,k}$ is then allocated a transmit amplitude value $\sqrt{\rho_a \zeta_k}$ to generate a baseband signal $x_{u,k}$ for wireless transmission, i.e., $x_{u,k} = \sqrt{\rho_a \zeta_k} s_{u,k}$. The UE $k$ is subjected to the average transmit power constraint, i.e., $E\{|x_{u,k}|^2\} \leq \rho_a$, which can also be expressed as a per-UE constraint:

$$0 \leq \zeta_k \leq 1, \forall k \in \mathcal{N}. \quad (9)$$

Because no power should be allocated to the non-selected UEs, we have

$$\forall k \in \mathcal{N} : a_k = 0, \text{ then } \zeta_k = 0. \quad (10)$$

The achievable uplink data rate $R_{u,k}$ at the CPU for the UE $k$ is then given in (11) shown at the top of next page, where $\zeta \triangleq \{\zeta_k\}_{k \in \mathcal{N}}$ [24, Eq. (27)].

Let $S_u$ (bits) and $R_{u,k}$ (bps) be the same data size of the local model updates and the data rate of transmitting the local model update from a UE $k$ to the CPU, respectively. The transmission time from the UE $k$ to the APs is given by

$$t_{u,k}(a_k, \zeta) = \frac{a_k S_u}{R_{u,k}(\zeta)}, \quad (12)$$

where $a_k \neq 0$. Since the non-selected UEs do not have any uplink transmission, we have

$$\forall k \in \mathcal{N} : a_k = 0, t_{u,k} = 0. \quad (13)$$

4) Step (S4): The CPU computes the global update by using all the received local updates. Since the CPU’s computational capability is much higher than that of the UEs, the time required to compute the global update is assumed negligible.
III. JOINT UE SELECTION AND RESOURCE ALLOCATION FOR MINIMIZING TOTAL FL EXECUTION TIME IN CFmMIMO: OPTIMIZATION PROBLEM FORMULATION

A. Total Execution Time of an FL Process

The execution time of one FL communication round comprises the execution times of Steps (S1)–(S3) [as Step (S4) is assumed not to incur any time delay]. The execution time of each of these steps is the longest execution time within that step. As such, the execution time of one FL communication round is expressed as

\[ T_{o}(a, \eta, f, \zeta) = \max_k t_{d,k}(a_k, \eta) + \max_k t_{e,k}(a_k, f_k) + \max_k t_{u,k}(a_k, \zeta) \]

(13)

where \( k \in \{k \in \mathcal{N} | a_k \neq 0\} \) and \( a_k \) (UE selection) is determined before an FL process is executed, while \( \eta, f, \) and \( \zeta \) are optimized in each FL communication round. The mathematical expression of the execution time of one FL communication round must have a two-timescale structure. Therefore, we introduce a new metric named “effective execution time of one FL communication round” \( E[T_{o}(a, \eta, f, \zeta)] \), which is the average of \( T_{o}(a, \eta, f, \zeta) \) over large-scale fading realizations. The values of \( a \) in \( E[T_{o}(a, \eta, f, \zeta)] \) remains unchanged, while those of \( \eta, f, \) and \( \zeta \) in \( T_{o}(a, \eta, f, \zeta) \) are optimized in each large-scale coherence time.

Let \( G(a) \) be the number of communication rounds in an FL process. By using the tightest convergence rates of FL [30, Theorem 1], we have

\[ G(a) = \frac{q}{\sum_{k \in \mathcal{N}} a_k}. \]

(14)

Here, \( q \) is a known constant which depends on the specific characteristics of the FL learning problems. The total execution time of an FL process is then given by

\[ T_{e}(a, \eta, f, \zeta) \triangleq G(a) E[T_{o}(a, \eta, f, \zeta)]. \]

(15)

Eqs. (13)–(15) clearly shows the effects of UE selection on the overall execution time. According to (13), if we only allow users with favorable link conditions i.e., strong channel gains, weak pilot contamination and interference) to partake in the FL process, the execution time of one FL communication round is shorter. However, selecting only a subset of all UEs also means increasing the required number of communication rounds as shown in (14). Therefore, minimizing the total execution time in (15) involves finding an optimal set of UEs to be selected, in order to balance between the two contradicting effects.

B. Problem Formulation

We are now ready to formulate the main design problem as the following optimization problem.

\[
\begin{align*}
\min_{a, \eta, f, \zeta} & \quad T_e(a, \eta, f, \zeta) \\
\text{s.t.} & \quad (1) - (4), (9), (10) \\
& \quad 0 \leq \eta_{mk}, \forall m, k \\
& \quad 0 \leq \zeta_k, \forall k \\
& \quad 0 \leq f_k \leq f_{k,\text{max}}, \forall k.
\end{align*}
\]

Problem (16) has a nonconvex stochastic, mixed-integer mixed-timescale structure, along with binary constraints and tight coupling among the optimizing variables. Finding its globally optimal solution is challenging.

IV. JOINT UE SELECTION AND RESOURCE ALLOCATION FOR MINIMIZING TOTAL FL EXECUTION TIME IN CFmMIMO: PROPOSED ALGORITHM

First, to deal with the binary constraint (1), we observe that \( x \in \{0, 1\} \Leftrightarrow x \in [0, 1] \) and \( x - x^2 \leq 0 \) [50, 51]. Therefore, (1) is equivalent to the following two constraints

\[ \sum_{k \in \mathcal{N}} (a_k - a_k^2) \leq 0 \]

(17)

\[ 0 \leq a_k \leq 1, \forall k. \]

(18)

Since the values of \( \{a_k\} \) in (17) and (18) are now real, it is easier to handle (17) and (18) than (1). We then use (17) and (18) to rewrite problem (16) as

\[
\begin{align*}
\min_{a, \eta, f, \zeta} & \quad T_e(a, \eta, f, \zeta) \\
\text{s.t.} & \quad (2) - (4), (9), (10) - (16b) - (16d), (17), (18).
\end{align*}
\]

Next, to deal with the max functions in \( T_o(a, \eta, f, \zeta) \), we rewrite problem (19) in a more tractable epigraph form as

\[
\begin{align*}
\min_{a, \eta, f, \zeta} & \quad G(a) \mathbb{E}[T_o(t_{d,k}, t_{e,k}, t_{u,k})] \\
\text{s.t.} & \quad (2) - (4), (9), (10) - (16b) - (16d), (17), (18).
\end{align*}
\]
where $\bar{T}_o(t_d, t_c, t_u) = t_d + t_c + t_u$, $\mathbf{x} \triangleq \{a, \eta, \zeta, r_d, r_u, t_d, t_c, t_u\}$; $\mathbf{r}_d \triangleq \{r_{d,k}\}$, $\mathbf{r}_u \triangleq \{r_{u,k}\}$, $\forall k \in \mathcal{N}$, $t_d$, $t_c$, and $t_u$ are additional variables.

Our main problem (16) is now transformed to problem (20) which is a stochastic nonconvex optimization problem. In (20), the UE selection variables $\mathbf{a}$ are optimized in a long-term timescale (before any FL process happens), while $\bar{\mathbf{x}} \triangleq \bar{\mathbf{x}} \setminus \mathbf{a}$ including transmit power and processing frequency are optimized in a short-term timescale (in each FL communication round). To solve this type of optimization problem, we adopt the general framework of [32]. Specifically, we decompose problem (20) into a short-term subproblem and a long-term master problem, and solve these resulting problems in an alternating manner. The mathematical derivations are detailed in the following.

For a given $\bar{\mathbf{a}}$, in each large-scale coherence time, the short-term subproblem is expressed as:

$$\min_{\bar{\mathbf{x}}} \bar{T}_o(t_d, t_c, t_u)$$

s.t. (3), (4), (9), (10), (16b) – (16d), (20b) – (20g).

For given optimal solutions $\bar{\mathbf{x}}$ to problems (21), we have $t_d = \frac{a_k S_d}{R_{d,k}}$, where $k^* \triangleq \arg\max_{k \in \mathcal{N}} \frac{a_k S_d}{R_{d,k}}$. Therefore, we can obtain $t_d = \mathbf{a}^T \mathbf{t}_d$, where $\mathbf{t}_d$ is the vector whose elements are 0 except for the $k^*$-th element, and the value of this element is $\frac{S_d}{R_{d,k^*}}$.

Similarly, we have $t_c^* = \frac{a_r LD + c_c}{f_k}$ and $t_u = \frac{a_r S_u}{R_{u,k^*}}$, where $i^* \triangleq \arg\max_{i \in \mathcal{V}} \frac{a_r LD + c_c}{f_k}$ and $j^* \triangleq \arg\max_{j \in \mathcal{M}} \frac{a_r S_u}{R_{u,k^*}}$. Then, we also have $t_e = \mathbf{a}^T \mathbf{t}_e$ and $t_u = \mathbf{a}^T \mathbf{t}_u$. Here, $\mathbf{t}_e$ is the vector whose elements are 0 except for the $i^*$-th element, and the value of this element is $\frac{LD + c_c}{f_k}$. Similarly, the concave lower bound $\bar{R}_{d,k}(\mathbf{v})$ of $R_{d,k}(\mathbf{v})$ is given by (27) [52] (see the top of the next page), where $\Phi_k(\mathbf{v}) = \rho_k \sum_{m \in \mathcal{M}} \sum_{\ell \in \mathcal{N}_k \setminus \mathcal{N}_m} v_{m\ell} \beta_{mk}^2 \beta_{ml}^2 |\mathbf{p}_k^H \mathbf{p}_\ell|^2$.

Similarly, the concave lower bound $\bar{R}_{u,k}(\mathbf{u})$ of $R_{u,k}(\mathbf{u})$ is given by (28) at the top of the next page [52], where $\Phi_k(\mathbf{u}) = \rho_k \sum_{m \in \mathcal{M}} \sum_{\ell \in \mathcal{N}_k \setminus \mathcal{N}_m} u_{m\ell} \beta_{mk}^2 \beta_{ml}^2 |\mathbf{p}_k^H \mathbf{p}_\ell|^2$.

As such, (26f) and (26g) can be approximated by the following convex constraints

$$r_{d,k} \leq \bar{R}_{d,k}(\mathbf{v}), \forall \mathbf{v} \in \mathcal{N}$$

$$r_{u,k} \leq \bar{R}_{u,k}(\mathbf{u}), \forall \mathbf{u} \in \mathcal{N}.$$
\[ \tilde{R}_{d,k}(v) \triangleq \frac{\tau_v - \tau_k}{\tau_v \log 2} B \left[ \log \left( 1 + \frac{(Y_k^{(v)})^2}{\Pi_k^{(v)}} \right) - \frac{(Y_k^{(v)})^2}{\Pi_k^{(v)}} + 2 \frac{Y_k^{(v)} \Upsilon_k}{\Pi_k^{(v)}} \left( \frac{(Y_k^{(v)})^2}{\Pi_k^{(v)}} + \Xi_k \right) \right] \leq \tilde{R}_{d,k}(v), \] (27)

\[ \tilde{R}_{u,k}(u) \triangleq \frac{\tau_u - \tau_k}{\tau_u \log 2} B \left[ \log \left( 1 + \frac{(\Psi_k^{(u)})^2}{\Xi_k^{(u)}} \right) - \frac{(\Psi_k^{(u)})^2}{\Xi_k^{(u)}} + 2 \frac{\Psi_k^{(u)} \Psi_k}{\Xi_k^{(u)}} \left( \frac{(\Psi_k^{(u)})^2}{\Xi_k^{(u)}} + \Xi_k \right) \right] \leq \tilde{R}_{u,k}(u), \] (28)

**Algorithm 1** Successive convex approximation approach for solving the short-term subproblem (21)

1. **Initialization:** Set \( \kappa = 0 \) and choose a random point \( \tilde{x}^{(0)} \in \mathcal{F} \).
2. **repeat**
3. Update \( \kappa = \kappa + 1 \) Solve the approximated problem (31) of (21) using CVX to obtain its optimal solution \( \tilde{x}^* \) and update \( \tilde{x}^{(\kappa)} = \tilde{x}^* \)
4. **until** convergence

**Proposition 1.** Algorithm 1 converges to a Karush-Kuhn-Tucker (KKT) solution of (21).

**Proof.** It is true that \( \tilde{R}_{d,k}(v) \) and \( \tilde{R}_{u,k}(u) \) satisfy the key properties of general inner approximation functions [54, Properties (i), (ii), and (iii)]. The feasible set \( \mathcal{F} \) also satisfies the Slater’s constraint qualification for convex programs. Therefore, Algorithm 1 converges to a KKT solution of (26) when starting from a point \( \tilde{x}^{(0)} \in \mathcal{F} \) [54, Theorem 1]. By using the variable transformations (24) and (25), it can be seen that the KKT solutions of (26) satisfy the KKT conditions of (23) as well as (21).

**B. Solving the Long-Term Master Problem (22)**

Let \( V(a) \triangleq \sum_{k \in \mathcal{N}} (a_k - a_k^0) \), then (17) becomes \( V(a) \leq 0 \). To deal with this non-convex constraint, we consider the problem

\[ \min_a L(a) \triangleq g(a) + \lambda^T V(a) \] (32)

s.t. (2), (18),

where \( L(a) \) is the Lagrangian of (22), \( \lambda \geq 0 \) is the Lagrangian multiplier corresponding to (17). Let \( \mathcal{H} \triangleq \{(2), (18)\} \) be the feasible set of problem (32).

**Proposition 2.** The following statement holds:

(i) The value \( V_{\lambda} \) of \( V \) at the solution of (32) corresponding to \( \lambda \) is decreasing to 0 as \( \lambda \to +\infty \).

(ii) Problem (32) has the following property, i.e.,

\[ \min_{a \in \mathcal{H}} g(a) = \sup_{a \in \mathcal{H}} L(a, \lambda) \] (33)

and is therefore equivalent to (22) at the optimal solution \( \lambda^* \geq 0 \) of the sup-min problem in (33).

**Proof.** The proof follows from [51, Proposition 1] and [50, Proposition 1]; hence, omitted for brevity.

Theoretically, it is required to have \( V_{\lambda} = 0 \) in order to obtain an optimal \( \lambda^* \). According to Proposition 2, \( V_{\lambda} \) decreases to 0 as \( \lambda \to +\infty \). Since there is always a numerical tolerance in computation, it is sufficient to accept \( V_{\lambda} < \varepsilon \) for some small \( \varepsilon \) with a sufficiently large value of \( \lambda \) chosen. In our numerical experiment, for \( \varepsilon = 10^{-3} \), we see that \( \lambda = 1 \) is enough to ensure \( V_{\lambda} \leq \varepsilon \). Note that this way of choosing \( \lambda \) has been widely used in the literature, e.g., [50], [51], [55], [56].

Problem (32) is still challenging due to the expectation operator in \( g(a) \) in its cost function \( L(a) \). Following the procedure proposed in [32], we approximate \( L(a) \) by its surrogate function \( \tilde{L}(a) \), which is given as

\[ \tilde{L}^{(n+1)}(a) \triangleq \tilde{g}^{(n+1)}(a) + \lambda V^{(n+1)}(a) \] (34)

\[ \tilde{g}^{(n+1)}(a) \triangleq (1 - \phi^{(n+1)}(a))g^{(n)}(a) + \phi^{(n+1)}T(a) \] (35)

\[ V^{(n+1)}(a) \triangleq V^{(n+1)} + \left( (\nabla V^{(n+1)})^T (a - a^{(n+1)}) \right) + \tau \| a - a^{(n+1)} \|^2 \] (36)

where \( \phi^{(n+1)}(a) \) is a weighting parameter and \( \tau \) can be any positive constant. Here, the surrogate function \( \tilde{g}^{(n+1)}(a) \) at iteration \( n + 1 \) depends on the surrogate function \( \tilde{g}^{(n)}(a) \) at iteration \( n \) and the approximate functions \( \tilde{T}(a) \) of \( T(a) \).

The function \( \tilde{g}^{(n+1)}(a) \) is approximately updated as

\[ \tilde{g}^{(n+1)}(a) \triangleq g^{(n)} + (\nabla g)^{(n)}(a - a^{(n+1)}) \] (37)

and \( \tilde{T}(a) \) is updated as

\[ \tilde{T}(a) \triangleq T^{(n+1)} + \frac{(\nabla T)^{(n+1)}(a - a^{(n+1)})}{\| a - a^{(n+1)} \|^2} \] (38)

Now, with (37) and (38), (35) becomes:

\[ \tilde{L}^{(n+1)}(a) \triangleq L^{(n+1)} + \left( (\nabla L)^{(n+1)}(a - a^{(n+1)}) \right) + \tau \| a - a^{(n+1)} \|^2 \] (39)

where

\[ g^{(n+1)} = (1 - \phi^{(n+1)}(a))g^{(n)} + \phi^{(n+1)}(a)T^{(n+1)} \] (40)

\[ (\nabla g)^{(n+1)} = (1 - \phi^{(n+1)}(a)) (\nabla g)^{(n)} + \phi^{(n+1)}(a) (\nabla T)^{(n+1)} \]

with \( g^{(0)} = 0 \), \( (\nabla g)^{(0)} = 0 \). Here,

\[ g^{(n+1)} = (\tilde{g}^{(n+1)}(a^{(n+1)})) - 1 \left( (\tilde{g}^{(n+1)}(a^{(n+1)})) - (\tilde{g}^{(n+1)}(a^{(n+1)})) \right) \] (41)

From (36) and (39), (34) can be written as

\[ \tilde{L}^{(n+1)}(a) \triangleq L^{(n+1)} + \left( (\nabla L)^{(n+1)}(a - a^{(n+1)}) \right) + \tau \| a - a^{(n+1)} \|^2 \] (42)

where

\[ L^{(n+1)} = g^{(n)} + \lambda V^{(n+1)} \] (43)

\[ (\nabla L)^{(n+1)} = (\nabla g)^{(n+1)} + \lambda (\nabla V)^{(n+1)} \] (44)

At the large-scale coherence time or iteration \( n + 1 \), problem (32) is approximated by the following convex problem:

\[ \min \tilde{L}^{(n+1)}(a) \] (45)
C. Solving the Overall Problem (20)

Algorithm 2 outlines the main steps to solve the overall problem (20) (hence (16)). In the large-scale coherence time $n$ (i.e., iteration $n$), for a given random value of $a^{(n+1)} \in H$, the short-term subproblem (21) is solved by Algorithm 1 after $I_S(n)$ iterations to obtain an approximate solution to problem (22) using CVX to obtain its optimal solution $(a^*, f^*, \zeta^*)$. Therefore, (31) requires a complexity of $O(n^2)$ for $n \in (0, 1)$, and

\[
\sum_{n} \phi(n)^2 < \infty, \quad \sum_{n} \pi(n)^2 < \infty, \quad \text{and} \quad \lim_{n \to \infty} \pi(n)^2 = 0.
\]

(B3): $\phi(n) \to 0$, $\frac{1}{\phi(n)} \leq O(1)$ for $n \in (0, 1)$, and

\[
\sum_{n} \phi(n)^2 < \infty.
\]

(B2): $\pi(n) \to 0$, $\sum_{n} \pi(n)^2 = \infty$, $\sum_{n} \pi(n)^2 < \infty$, and

\[
\lim_{n \to \infty} \pi(n)^2 = 0.
\]

Eventually, the converged UE selection solution $a^*$ provided by Algorithm 2 is used within the subsequent FL process as described in Sec. II-B. In the FL training process, one FL communication round and one STO time block are executed in one large-scale coherence time. Here, the values of transmit power and processing frequency obtained in the STO time block are not those obtained in the STO time block when executing Algorithm 2, but rather are computed by the same Algorithm 1. The detailed execution of one FL communication round is discussed in Section II-B and illustrated in Figure 2. Once the network information changes, we will re-engage Algorithm 2 to get a new solution, ready for FL to execute again.

Algorithm 2 Online successive convex approximation approach for solving the overall problem (20)

1: Initialization: Set $n = 0$, select a random $a^{(n+1)} \in H$
2: repeat
3: Solve the short-term subproblem (21) to obtain its optimal solution $(\eta^*, f^*, \zeta^*)$ by using Algorithm 1, and update $(\eta(n+1), f(n+1), \zeta(n+1)) = (\eta^*, f^*, \zeta^*)$
4: Solve the approximate long-term master problem (45) of (22) using CVX to obtain its optimal solution $(a^*)^{(n+1)}$
5: Update $a^{(n+2)}$ by (46) and $n = n + 1$
6: until convergence

Output: $a^* = a^{(n+1)}$

D. Convergence and Complexity Analyses of Algorithm 2

Definition 1. A solution $(a^*, x^*)$ is called a stationary solution of problem (20) (or (16)) if $x^*$ is a KKT solution of the short-term subproblem (21) for $a = a^*$, and $a^*$ is a KKT solution of the long-term master problem (22) for $x = x^*$.

Proposition 3. Algorithm 2 converges to a neighbourhood of the stationary solutions to problem (16).

Proof. See Appendix A.

Theoretically, $I_S(n) \to \infty$ and $\lambda \to \infty$ are required for Algorithm 2 to converge to the stationary solutions of problem (16). When $I_S(n)$ and $\lambda$ are finite, Algorithm 2 converges to approximate stationary solutions of problem (16).

The computational complexities of solving (31) at each iteration of Algorithm 1 and solving (45) at each iteration of Algorithm 2 are polynomial in the number of variables and constraints. In particular, (31) can be transformed into an equivalent optimization problem that involves $N_{v,1} \triangleq (2MN + 4N + 3)$ real-valued scalar decision variables, $N_{u,1} \triangleq (2MN + M + 4N + 3)$ linear constraints and $N_{q,1} \triangleq (MN + 5N)$ quadratic constraints. Therefore, (31) requires a complexity of $O(\sqrt{N_{v,1} + N_{q,1}}N_{u,1} + N_{v,1} + N_{q,1}N_{v,1}^2)$ [57, 58]. Problem (45) involves $N_{v,2} \triangleq N$ real-valued scalar decision variables, $N_{u,2} \triangleq (N + 1)$ linear constraints. As such, (45) requires a complexity of $O(\sqrt{N_{v,2} + N_{q,2}}N_{v,2}^2)$. 

Fig. 3. Overview of Algorithm 2’s operation to assist CFmMIMO-based wireless FL.
V. NUMERICAL EXAMPLES

This section provides numerical results to analyze the effectiveness of the proposed Algorithm 2 in minimizing the execution time of the considered standard FL process [29], [30]. As previously discussed in Sec. I, the ultimate question is how to find an optimal set of UEs to reduce the FL execution time without satisfying the test accuracy too much. To answer this question, we need to answer the fundamental questions (Q1) and (Q2). Assuming that $N_{QoL}$ is known in advance, our work focuses on answering question (Q2). It is shown in this section that the number of selected UEs in our example is always larger than $N_{QoL}$. Therefore, the test accuracy obtained with this set of selected UEs is always acceptable. This means with the UE selection of our approach, the test accuracy of an FL process using real datasets is expected to be the same as that in [29], [30], and hence, not shown in this paper. On the other hand, if we try to make an ultimate analysis on the test accuracy of our UE selection scheme, we first have to set $N_{QoL}$ correctly, which leads us back to answering question (Q1). However, answering question (Q1) requires extensive efforts that are out of the scope of this work. Based on this observation, we leave the analysis of the test accuracy of an FL process with our UE selection scheme for future work.

A. Network Setup with Non-Uniform UE/AP Distribution

We consider a CFmMIMO network in a square of $D \times D$ km$^2$ whose edges are wrapped around to avoid the boundary effects. We examine the following two cases.

- Case (C1): The UEs are more likely to stay near some fixed locations (e.g., coffee shops, restaurants) in the considered area; and the APs are uniformly distributed across the considered area.
- Case (C2): Both the APs and UEs are more likely to stay close to some fixed locations in the considered area.

1) Modeling of Case (C1): First, a set of $x_p$ fixed locations are uniformly distributed over the network square. To generate UE locations, we create a grid that has $x_l$ vertical and $x_l$ horizontal lines. Then, $N$ UE locations are chosen as the points on this grid that are closest to $x_p$ fixed locations. To generate AP locations, we create a new grid that is the same as the UE location grid, but the lines of these two grids are interleaved. Finally, $M$ AP locations are uniformly chosen in the points on the latter grid. Here, we consider $x_l = 15, x_p = 20$.

2) Modeling of Case (C2): Case (C2) is modeled in the same way for Case (C1), except that the $M$ AP locations are chosen from the points on the AP location grid that are closest to $x_{AP}$ fixed locations. Here, we choose $x_{AP} = 3$, and the $x_{AP}$ fixed locations are uniformly selected out of $x_p$ fixed locations without replacement. An example of the network
setup is shown in Fig. 4.

3) Setup for each network realization: Since each FL communication round happens in one large-scale coherence time (in the order of seconds), the total execution time of an FL process is expected to be around several minutes. Therefore, we assume that the UEs only move around their current locations during the FL process. Here, in each communication round of an FL process, we let each UE move within a circle of radius 5 m around its current location, while the AP locations remain unchanged.

B. Parameter Settings

We model large-scale fading coefficients $\beta_{mk}$ as [59]:

$$\beta_{mk} = 10^{- \frac{p_{rmk}}{10}} \times 10^{- \frac{p_{rk}}{10}}, \quad (47)$$

where $10^{- \frac{p_{rmk}}{10}}$ represents the path loss, and $10^{- \frac{p_{rk}}{10}}$ represents the shadowing effect with $F_{mk} \in \mathcal{N}(0, 4^2)$ (in dB). Here, $PL_{mk}$ (in dB) is given by [59]

$$PL_{mk} = -30.5 - 36.7 \log_{10} \left( \frac{d_{mk}}{1 \text{ m}} \right), \quad (48)$$

and the correlation among the shadowing terms from the AP $m$, $\forall m \in \mathcal{M}$ to different UEs $k, \ell \in \mathcal{N}$ is expressed as:

$$\mathbb{E}\{ F_{mk}F_{j\ell} \} \triangleq \alpha (4^{2q - \delta_{k\ell}/9m}, \quad \text{if } j = m,$$

$$0, \quad \text{otherwise}, \quad \forall j \in \mathcal{M}, \quad (49)$$

where $\delta_{k\ell}$ is the physical distance between UEs $k$ and $\ell$.

For channel estimation, we use a random pilot assignment scheme. Specifically, the pilot of each user is randomly chosen from a predefined set of $\tau_i = N$ orthogonal pilot sequences, each having a length of $\tau_i$ samples. We set $\tau_c = 200$ samples, $S_d = S_u = 5$ MB, noise power $\sigma_0^2 = -92$ dBm, $L = 5$, $f_{max} = 3 \times 10^9$ cycles/s, $D_k = D = 5 \times 10^9$ samples, $c_k = 20$ cycles/samples [42], for all $k$, $\alpha = 2 \times 10^{-29}$. We choose $q = 90$. Let $\hat{\rho}_d = 1$ W, $\hat{\rho}_u = 0.2$ W, $\hat{\rho}_l = 0.2$ W be the maximum transmit powers of the APs, UEs, and uplink pilot sequences, respectively. Here, $\rho_d$, $\rho_u$, and $\rho_l$ are the normalized values of $\hat{\rho}_d$, $\hat{\rho}_u$, and $\hat{\rho}_l$ with respect to the noise power. We set $\pi^{(n)} = \frac{1}{10000}$ and $\phi^{(n)} = \frac{1}{n^{0.75}}$ to satisfy conditions (B1) and (B2) in Section IV-C. Finally, we choose $\lambda = 1$.

C. Results and Discussions

1) Effectiveness of Algorithm 2: First, we evaluate the convergence behavior of the proposed Algorithm 2. As seen from Fig. 5, Algorithm 2 converges within 30 iterations for an arbitrary network realization. Note that each iteration of Algorithm 2 involves solving simple convex programs (31) and (45). It is therefore expected that Algorithm 2 has a low computational complexity.

Next, we compare Algorithm 2 (denoted by OPT in the figures) with the following baseline schemes:

- **Baseline 1 (BL1):** Let $\hat{N}$ be an integer uniformly drawn from the interval $[N_{QoL}, N]$. Then, before an FL process is executed, we select participating UEs for this FL process by uniformly selecting $\hat{N} \leq N$ UEs without replacement. The UE selection result of this scheme is represented by a vector $a_{BL1}$. The FL execution time by BL1 is thus $T_c(a_{BL1}, \eta, f, \zeta) = G(a_{BL1}) \mathbb{E}\{ T_c(a_{BL1}, \eta, f, \zeta) \}$. Since $a_{BL1}$ is known, solving the problem of minimizing $T_c(a_{BL1}, \eta, f, \zeta)$ only requires solving the short-term subproblem (21) of optimizing $(\eta, f, \zeta)$ to minimize the execution time of one FL communication round. Here, (21) is solved by using Algorithm 1 for a given $a_{BL1}$.

- **Baseline 2 (BL2):** In this baseline, we let all N UEs participate in an FL process but only $K \leq N$ UEs participate in each FL communication round. Here, the integer $K$ is uniformly drawn from the interval $[N_{QoL}, N]$. Then, $K$ UEs are uniformly chosen out of $N$ original UEs without replacement. The UE selection result of this scheme in each FL communication round is represented by a vector $a_{BL2}$. Note that BL2 is identical to the opportunistic UE sampling scheme proposed in [29], [30]. Since BL2 performs UE selection in each FL communication round, the number of FL communication rounds is $\hat{G} = \frac{q}{\tilde{q}} + \tilde{q}(1 - \frac{K}{2})$ [30, Theorem 1], where $q, \tilde{q}$ depend on the specific characteristics of the FL learning problems and are assumed known in advance. Here, since there is no UE sampling in each FL communication round in our proposed approach, all $\hat{N} = \sum_{k \in \mathcal{N}} a_{k}$ selected UEs participate in an FL process, i.e., $K = \hat{N}$. Therefore, the number of FL communication rounds are $\frac{\hat{G}}{\hat{N}}$ as in (14). In this work, we choose $\tilde{q} = q$ for simplicity. Then, the execution time of an FL process using BL2 is measured by $T_c(a_{BL2}, \eta, f, \zeta) = G \mathbb{E}\{ T_c(a_{BL2}, \eta, f, \zeta) \}$. Since $a_{BL2}$ is known in each FL communication round, solving the problem of minimizing $T_c(a_{BL2}, \eta, f, \zeta)$ only
Fig. 7. Impact of the number of APs on: (a) FL execution time; (b) number of selected UEs. In this example, we set $N = 15$ and $N_{QoL} = 5$.

![Graph of FL execution time vs number of APs](image1)

![Graph of number of selected UEs vs number of APs](image2)

Fig. 8. Impact of the threshold $N_{QoL}$ on: (a) FL execution time; (b) number of UEs being selected. In this example, we set $M = 40$, $N = 15$.

![Graph of FL execution time vs $N_{QoL}$](image3)

![Graph of number of selected UEs vs $N_{QoL}$](image4)

requires solving the short-term subproblem (21) of optimizing $(\eta, f, \zeta)$ to minimize the execution time of one FL communication round. Here, (21) is solved by using Algorithm 1 for a given $a_{BL2}$.

Fig. 6 compares the total execution time of an FL process by all the considered schemes. As seen, our OPT scheme is the best performer. In particular, while BL1 and BL2 perform quite similarly, OPT cuts the execution time by a substantial amount, e.g., by up to 50% in Case (C1) and 66% in Case (C2) with $M = 20$ and $D = 1.5$ km. These results show the significant advantage of an optimal UE selection over heuristic UE selections.

Fig. 6 also shows the importance of optimal UE selection for reducing the FL execution time when the AP density defined as the number of APs over a geographical area is moderately low. Specifically, in both Cases (C1) and (C2) with a large value of $D$ or a low value of $M$, the reduction in the FL execution time by OPT is at least 44%. This is reasonable because in these cases, there is a high probability of having UEs with unfavorable links. This leads to a significantly low execution time of one FL communication time, and hence, the whole FL execution time.

2) Impact of the number of APs on the number of selected UEs: Fig. 7 shows that a larger number of APs corresponds to a larger number of UEs being selected. Here, as the AP-UE distances are smaller, there are potentially more UEs with favorable links hence being selected. The only time the number of UEs being selected decreases is in Case (C2) with $M = \{20, 40\}$ and $D = 1.5$ km. In this case, there are a high probability of having UEs with unfavorable links, which leads to the two largest execution times as shown in Fig. 7(a). Therefore, the numbers of selected UEs still need to be reduced to shorten the execution time of one FL communication, and hence, the whole FL execution time.

3) Impact of $N_{QoL}$ on the execution time of an FL process: Fig. 8(a) shows that increasing $N_{QoL}$ leads to a dramatic increase in the FL execution time in a network that has a significantly low density of APs and non-uniformly distributed AP locations. This network is presented in Case (C2) with $D = 1.5$ km, where there is a high probability of having UEs with unfavorable links. In this case, for a larger value of $N_{QoL}$, more UEs are required to participate in an FL communication as shown in Fig. 8(b). While this reduces the number of FL communication rounds, a stronger inter-user interference is also resulted. In our example, such a reduction in the number of communication round is not sufficient to compensate for the increase in the execution time of each FL communication round.

Fig. 8(a) also shows that CFmMIMO networks with a high density of APs potentially provides low-latency FL services for everyone. When there are fewer UEs with unfavorable links as in Case (C1) with $D = \{0.75, 1.5\}$ km and Case (C2) with
\[ D = 0.75 \text{ km}, \] the FL execution times are nearly the same when \( N_{\text{QoL}} \) is increased.

VI. CONCLUSION

In this work, we have proposed a novel approach that jointly designs UE selection, transmit power, and processing frequency to minimize the execution time of an FL process in CFmMIMO networks. We formulate a mixed-integer mixed-timescale stochastic nonconvex problem under practical requirements on the maximum transmit powers and the minimum number of selected UEs to guarantee quality of learning. Utilizing online successive convex approximation, we have successfully developed a novel algorithm to solve the formulated problem. The proposed algorithm has been proved to converge to the neighbourhood of stationary points. Numerical results have shown that our approach significantly reduces the FL execution time over the baseline schemes.

APPENDIX A

PROOF OF PROPOSITION 3

The proof involves two steps. The first step is to prove that the solution \( \mathbf{x}^* \) obtained from Algorithm 2 is a KKT solution of the short-term subproblem (21). This proof has already been provided in Proposition 1. The second step is to prove that the solution \( a^{(n+1)} \) obtained from Algorithm 2 is a KKT solution of the long-term master problem (22). The details of this proof are as follows.

It can be confirmed that problem (20) satisfies the conditions of Assumption 1 on the main problem in the general framework [32]. It is worth noting that we do not verify Assumption 1-5) and Assumption 1-6) of [32] for the following reasons. Assumption 1-5) on Mangasarian-Fromovitz constraint qualification is used to ensure the existence of KKT solutions of the short-term subproblem (21). In this work, since Proposition 1 shows that a KKT solution of (21) can be obtained by Algorithm 1, this assumption is unnecessary. Assumption 1-6) is used to guarantee convergence to an exact stationary point of the short-term subproblem (21). However, [32] confirms that Assumption 1-6) can be removed when we allow an approximate convergence by solving the short-term subproblem (21) with a finite number of iterations.

From the definitions of \( \tilde{R}_{d,k}(\mathbf{v}) \) and \( \tilde{R}_{u,k}(\mathbf{u}) \) in (27) and (28), it can be verified that \( \tilde{R}_{d,k}(\mathbf{v}) \) and \( \tilde{R}_{u,k}(\mathbf{u}) \) have the following properties:

- \( \tilde{R}_{d,k}(\mathbf{v}(n)) = R_{d,k}(\mathbf{v}(n)) \),
- \( \tilde{R}_{u,k}(\mathbf{u}(n)) = h_{u,k}(\mathbf{u}(n)) \),
- \( \nabla \tilde{R}_{d,k}(\mathbf{v}(n)) = \nabla R_{d,k}(\mathbf{v}(n)) \),
- \( \nabla \tilde{R}_{u,k}(\mathbf{u}(n)) = \nabla h_{u,k}(\mathbf{u}(n)) \);
- \( -\tilde{R}_{d,k}(\mathbf{v}) \), and \( -\tilde{R}_{u,k}(\mathbf{u}) \) are strongly convex;
- \( \tilde{R}_{d,k}(\mathbf{v},\mathbf{v}(n)) \) and \( \tilde{R}_{u,k}(\mathbf{u},\mathbf{u}(n)) \) are Lipschitz continuous in both \( \mathbf{v},\mathbf{v}(n) \) and both \( \mathbf{u},\mathbf{u}(n) \), respectively.

Algorithm 1 thus satisfies all the conditions of Assumption 2 on the short-term algorithm, within the general framework [32]. Since \( \phi^{(n)}, \pi^{(n)} \) are chosen to satisfy conditions (B1) and (B2) in Sec. IV-C, they satisfy all the conditions of Assumption 5 in [32]. When Assumptions 1, 2 and 5 in [32] is satisfied, it is confirmed by [32, Corollary 1] that the surrogate function \( \tilde{L}(\mathbf{a}) \) in (45) satisfies the Assumptions 3 and 4 in [32] on the properties and asymptotic consistency of surrogate functions.

Since Assumptions 1-5 in [32] are all satisfied, it follows from [32, Lemma 1] that:

(i) The sequence \( \{ (a^{(n+1)}, (\pi^{(n+1)})_n \}_{n=1}^{\infty} \) generated over iterations of Algorithm 2 has the following property:

\[
\lim_{n \to \infty} \| a^{(n+1)} - (\pi^{(n+1)})_n \| = 0, \quad (50)
\]

(ii) Let \( a_\ast \) be a limit point of a subsequence \( \{ a^{(n_j+1)} \}_{j=1}^{\infty} \) and

\[
\lim_{j \to \infty} |\tilde{L}(a^{(n_j+1)}) - \tilde{L}(a_\ast)| = 0, \quad (51)
\]

\[
\lim_{j \to \infty} |\nabla \tilde{L}(a^{(n_j+1)}) - \nabla \tilde{L}(a_\ast)| = 0. \quad (52)
\]

Without loss of generality, we assume that \( a^{(n+1)} \to a_\ast \) as \( n \to \infty \). Then, (51) and (52) imply that

\[
\lim_{n \to \infty} |\tilde{L}(a^{(n+1)}) - \tilde{L}(a^{(n+1)})_n| = 0, \quad (53)
\]

\[
\lim_{n \to \infty} |\nabla \tilde{L}(a^{(n+1)}) - \nabla \tilde{L}(a^{(n+1)})_n| = 0. \quad (54)
\]

It can be seen that there always exists one interior point in \( \mathcal{H} \). Therefore, the convex problem (45) satisfies the Slater’s constraint qualification condition. Its optimal solution \( (a^{\ast(n+1)})_n \) is thus a KKT solution to (32), and hence, (22) when \( \lambda \to \infty \) (see Proposition 2), i.e.,

\[
\nabla \tilde{L}(a^{\ast(n+1)}) + \sum_{j=1}^q \nu_j \nabla \delta_j((a^{\ast})^{(n+1)}) = 0, \quad (55a)
\]

\[
\nu_j \delta_j((a^{\ast})^{(n+1)}) = 0, \quad \forall j \in \{1, \ldots, q\}, \quad (55b)
\]

where \( \delta_j(a) \), \( \forall j \in \{1, \ldots, q\} \) represent the functions in the constraints (2), (17) and (18). It follows from (50) and (54) that the gap between \( a^{(n+1)} \) and \( a^{\ast(n+1)} \) and that between \( \nabla \tilde{L}(a^{(n+1)}) \) and \( \nabla \tilde{L}(a^{(n+1)})_n \) converge to zero as \( n \to \infty \). Therefore, (55) implies

\[
\nabla \tilde{L}(a^{(n+1)}) + \sum_{j=1}^q \nu_j \nabla \delta_j(a^{(n+1)}) = 0, \quad (56a)
\]

\[
\nu_j \delta_j(a^{(n+1)}) = 0, \quad \forall j \in \{1, \ldots, r\}, \quad (56b)
\]

which means \( a^{(n+1)} \) is a KKT solution of the long-term master problem (22).

As such, the convergence of Algorithm 2 to a stationary point of problem (20) in the sense of Definition 1 is guaranteed if the numbers of iterations of Algorithms 1 and 2 are infinity, i.e., \( I_S^{(n)} \to \infty \), \( I_L \to \infty \), and \( \lambda \to \infty \). In practice, it is acceptable to choose finite \( \{ I_S^{(n)} \}_{n=0}^{\infty} \), \( I_L \), and \( \lambda \) for an approximate convergence. Therefore, Algorithm 2 is guaranteed to converge to the neighborhood of the stationary solutions of problem (20), and hence, (16).

REFERENCES

[1] Cisco, “Cisco annual internet report,” Mar. 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/executive-perspectives/annual-internet-report/index.html

[2] J. Dong, M. Noreikis, Y. Xiao, and A. Ylä-Jääski, “Vinav: A vision-based indoor navigation system for smartphones,” IEEE Trans. Mobile Comput., vol. 18, no. 6, pp. 1461–1475, Jun. 2019.

[3] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y. A. Zhang, “The roadmap to 6G: AI empowered wireless networks,” IEEE Commun. Mag., vol. 57, no. 8, pp. 84–90, Aug. 2019.

[4] F. D. Calabrese, L. Wang, E. Ghadimi, G. Peters, L. Hanzo, and P. Soldati, “Learning radio resource management in RANs: Framework, opportunities, and challenges,” IEEE Commun. Mag., vol. 56, no. 9, pp. 138–145, Sep. 2018.
[52] V. D. Nguyen, T. Q. Duong, H. D. Tuan, O. S. Shin, and H. V. Poor, “Spectral and energy efficiencies in full-duplex wireless information and power transfer,” *IEEE Trans. Commun.*, vol. 65, no. 5, pp. 2220–2233, May 2017.

[53] M. Grant and S. Boyd, “CVX: Matlab software for disciplined convex programming, version 2.1,” http://cvxr.com/cvx, Mar. 2014.

[54] B. R. Marks and G. P. Wright, “A general inner approximation algorithm for nonconvex mathematical programs,” *Operations Research*, vol. 26, no. 4, pp. 681–683, 1978.

[55] E. Che, H. D. Tuan, and H. H. Nguyen, “Joint optimization of cooperative beamforming and relay assignment in multi-user wireless relay networks,” *IEEE Trans. Wireless Commun.*, vol. 13, no. 10, pp. 5481–5495, Oct. 2014.

[56] U. Rashid, H. D. Tuan, H. H. Kha, and H. H. Nguyen, “Joint optimization of source precoding and relay beamforming in wireless MIMO relay networks,” *IEEE Trans. Commun.*, vol. 62, no. 2, pp. 488–499, Feb. 2014.

[57] H. H. M. Tam, H. D. Tuan, D. T. Ngo, T. Q. Duong, and H. V. Poor, “Joint load balancing and interference management for small-cell heterogeneous networks with limited backhaul capacity,” *IEEE Trans. Wireless Commun.*, vol. 16, no. 2, pp. 872–884, 2017.

[58] Y. Nesterov and A. Nemirovski, *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, 1994.

[59] E. Björnson and L. Sanguinetti, “Making cell-free massive MIMO competitive with MMSE processing and centralized implementation,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 77–90, Jan. 2020.