Cell-Free Massive MIMO for Wireless Federated Learning

Tung Thanh Vu, *Graduate Student Member, IEEE*, Duy Trong Ngo, *Member, IEEE*, Nguyen H. Tran, *Senior Member, IEEE*, Hien Quoc Ngo, *Member, IEEE*, Minh Ngoc Dao, and Richard H. Middleton, *Fellow, IEEE*

***Abstract*— This paper proposes a novel scheme for cell-free massive multiple-input multiple-output (CFmMIMO) networks to support any federated learning (FL) framework. This scheme allows each instead of all the iterations of the FL framework to happen in a large-scale coherence time to guarantee a stable operation of an FL process. To show how to optimize the FL performance using this proposed scheme, we consider an existing FL framework as an example and target FL training time minimization for this framework. An optimization problem is then formulated to jointly optimize the local accuracy, transmit power, data rate, and users' processing frequency. This mixed-timescale stochastic nonconvex problem captures the complex interactions among the training time, and transmission and computation of training updates of one FL process. By employing the online successive convex approximation approach, we develop a new algorithm to solve the formulated problem with proven convergence to the neighbourhood of its stationary points. Our numerical results confirm that the presented joint design reduces the training time by up to 55% over baseline approaches. They also show that CFmMIMO here requires the lowest training time for FL processes compared with cell-free time-division multiple access massive MIMO and collocated massive MIMO.**

***Index Terms*— Cell-free massive MIMO, federated learning.**

Manuscript received December 18, 2019; revised April 30, 2020; accepted June 9, 2020. Date of publication June 24, 2020; date of current version October 9, 2020. The work of Tung T. Vu was supported by the ECR-HDR Scholarship from The University of Newcastle. The work of Duy T. Ngo was supported in part by the Australian Research Council Discovery Project (ARCDP) under Grant DP170100939 and in part by the Vietnam National Foundation for Science and Technology Development (NAFOSTED) under Grant 102.02-2018.320. The work of Nguyen H. Tran was supported in part by the ARCDP under Grant DP200103718 and in part by NAFOSTED under Grant 102.02-2019.321. The work of Hien Quoc Ngo was supported by the U.K. Research and Innovation Future Leaders Fellowships under Grant MR/S017666/1. The work of Minh N. Dao was supported in part by ARCDP under Grant DP160101537 and Grant DP190100555. The associate editor coordinating the review of this article and approving it for publication was M. Xiao. *(Corresponding author: Tung Thanh Vu.)*

Tung Thanh Vu, Duy Trong Ngo, and Richard H. Middleton are with the School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW 2308, Australia (e-mail: thanhtung.vu@uon.edu.au; duy.ngo@newcastle.edu.au; richard.middleton@newcastle.edu.au).

Nguyen H. Tran is with the School of Computer Science, The University of Sydney, Sydney, NSW 2006, Australia (e-mail: nguyen.tran@sydney.edu.au).

Hien Quoc Ngo is with the School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast BT7 1NN, U.K. (e-mail: hien.ngo@qub.ac.uk).

Minh Ngoc Dao is with the Department of Applied Mathematics, The University of New South Wales, Sydney, NSW 2052, Australia (e-mail: daonminh@gmail.com).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TWC.2020.3002988

I. Introduction

THE use of machine learning (ML) techniques in telecommunications industry has been growing dramatically in recent years [1], [2]. One reason for this trend is the fast growing number of mobile devices, wearable devices and autonomous vehicles. They are generating a vast amount of data by using in-built sensors, e.g., microphones, GPS and camera, for critical applications such as traffic navigation, indoor localization, image recognition, natural language processing, and augmented reality [3]. In addition, the computational capabilities of these devices also grow significantly with dedicated hardware architecture and computing engines, e.g., the energy-efficient Qualcomm Hexagon Vector eXtensions on Snapdragon 835 [4]. On-device artificial intelligence (AI) capabilities are predicted to be available on 80% of all smartphones by 2022 [5]. It is therefore critical for telecommunications operators to start investigating into a future communications system that efficiently utilizes the empowered computation resources from mobile devices to solve ML problems.

The typical ML framework used in the current telecommunications systems requires a cloud center to store and process raw data collected by the user equipment (UEs). However, such a centralized structure fails to support real-time applications because of its high latency [6]. The concept of mobile edge computing is introduced to process data at the edge nodes instead of the cloud center [7], [8]. Since the computational capability of mobile devices is growing noticeably, it is possible to even push the network computation further to the mobile device level [9], [10]. On the other hand, serious concerns about data privacy have recently been raised due to data being processed by third-party companies, e.g., Facebook, Apple. This urgently calls for a new class of ML frameworks that not only exploit the computational resources of the UEs for ML applications but also ensure data privacy.

A promising candidate for such ML frameworks is the recently developed Federated Learning (FL) [9], [11]. As shown in Fig. 1, an FL process is an iterative process in which the UEs use their local training data to compute local model training updates, followed by sending the updates to a central server. The central server then aggregates these updates to compute the global training update, which is then sent back

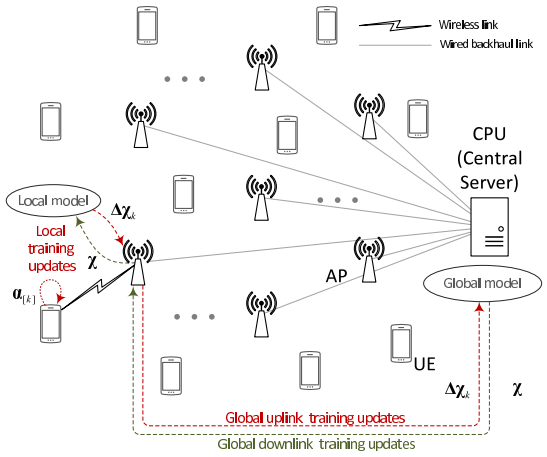

Fig. 1. Illustration of an FL process over communication networks and CFmMIMO network model used to support FL.

to the UEs to further assist their local update computation. This iterative process terminates when a certain learning accuracy level is attained. Data privacy is protected by not sharing the local training *data*, but only the local *training updates* computed at the UEs using local computational resources. Uploading only the local training updates to the central server incurs a significantly lower delay, compared to uploading a large amount of raw data. This distributed approach facilitates a large-scale model training and more flexible data collection, albeit at the expense of UEs' computational resources [10].

With all the promising advantages listed above, FL has attracted much attention from both developer and researcher communities [10], [12]–[17]. In [12], an FL algorithm for keyboard prediction on smartphones is developed by Google. Reference [13] target improving the performance of the general FL algorithms, while [10], [14]–[17] concentrate on optimizing the performance of FL algorithms used in wireless networks. In particular, [13] proposes a new compression framework for a communication-efficient FL system. Reference [10] aims to obtain the best trade-off between FL training time and UE energy consumption. Reference [14] proposes an incentive mechanism that encourages the UEs with high-quality data to participate in FL systems. Reference [15] introduces a control algorithm to achieve the best trade-off between the number of local updates and that of global updates for a given resource budget. A joint device selection and beamforming design is proposed in [16] to enhance the performance of FL. Reference [17] proposes a joint design of user selection, power control and subchannel allocation for minimizing the loss function of FL.

This paper investigates how to implement FL in a wireless network. It is worth noting that the existing works [10], [16] rely on an impractical assumption that the channel state information (CSI) remains unchanged during the whole FL process. In practice however, the channel changes in the order of milliseconds; and as such, certain system parameters for FL performance optimization, e.g., data rates and power control, would have become obsolete even before the FL process terminates. In addition, it might not be most efficient to use orthogonal multiplexing approaches, e.g., orthogonal frequency-division multiple access (OFDMA) [17] and time-division multiple access (TDMA) [10], for UEs to transmit their local updates. With a large number of UEs, the total training time could be significantly prolonged. To both deal with the wireless channel dynamics and to serve UEs at the same time and in the same frequency bands, a new wireless network structure that supports FL is called for.

In this work, we propose using cell-free massive multiple-input multiple-output (CFmMIMO) [18], [19] for FL in a wireless environment. Here, a central processing unit (CPU) (i.e., the central server) is connected to a large number of access point (APs) via backhaul links. These APs then simultaneously serve UEs via wireless links using the same frequency bands with the CSI acquired via uplink (UL) training pilots. An important characteristic of massive MIMO is channel hardening [20], i.e., the effective channel gain at the UEs is close to its expected value—a known deterministic constant [19]. As such, the channels are reasonably stable during one large-scale coherence time $\widetilde{T}_c$.[1] The channel dynamics due to small-scale fading thus have negligible effects on the FL processes. In addition, a CFmMIMO network also provides a high probability of coverage, making the FL processes less prone to the unfavorable UE links.

Our research contributions are summarized as follows.

- We propose, for the first time, a scheme for CFmMIMO networks to support any FL framework. In this scheme, any iterative algorithm can be developed to optimize the FL performance before the FL process is executed. Each instead of all iterations of this FL optimization algorithm or the FL process happens within one $\widetilde{T}_c$ in order to guarantee channel stability during its operation. In each iteration of the FL process, we propose using the APs as relays to transmit the training updates between the CPU and UEs. Doing so allows any beamforming/filtering design to be applied to the APs in order to enhance the performance of training update transmission.
- To show how to optimize the FL performance using the proposed scheme, we consider an existing FL framework [21] as an example and target the key performance metric of "training time minimization". We formulate a mixed-timescale stochastic nonconvex optimization problem that minimizes the time of one FL process. The formulated problem captures the complex interactions among the FL training time, and transmission and computation of FL training updates in a CFmMIMO network. Here, a conjugate beamforming/matched filtering scheme is applied to the APs for ease of implementation. The local accuracy, power control, data rate and UE's processing frequency are jointly designed, subject to the practical constraints on UEs' energy consumption and imperfect channel estimation.

[1]During one large-scale coherence time $\widetilde{T}_c$, the large-scale fading coefficients are reasonably invariant. The value of $\widetilde{T}_c$ can be empirically measured, in the same way for small-scale fading measurement. For indoor communications, the large-scale coherence time can be at least 40 small-scale fading coherence time [19], and it has a time order of seconds.

- Utilizing the general framework in [22], we propose a new algorithm that is proven to converge to at least the neighborhood of the stationary points of the formulated problem. Here, the coupling among the variables makes it challenging to develop a specific algorithm that satisfies all the strict conditions stated in the general framework of [22]. It is also noted that our algorithm only requires channel stability in each but not all iterations. This important feature ensures the problem and its solution remain up-to-date and valid during the running time of the algorithm, despite the channel variations.
- Simulation results verify the convergence of the proposed algorithm, and show that our solution reduces the training time by up to 55% compared with the baseline schemes. They further confirm that CFmMIMO offers the lowest training time compared to cell-free TDMA massive MIMO and collocated massive MIMO.

Paper Organization and Notation: The rest of this paper is organized as follows. Section II proposes a novel scheme for a CFmMIMO network to support a general FL framework. Section III introduces a specific example of the general FL framework considered in this paper. Section IV presents the system model and assumptions. Section V formulates the FL training time minimization problem, whereas Section VI proposes a new algorithm to solve the formulated problem. For comparison, Section VII introduces cell-free TDMA and collocated massive MIMO systems to also support the considered FL framework. Section VIII verifies the performance of the developed algorithm through comprehensive numerical examples. Finally, Section IX concludes the paper.

In this paper, boldfaced symbols are used for vectors and capitalized boldfaced symbols for matrices. $\boldsymbol{X}^*$ and $\boldsymbol{X}^H$ are the conjugate and conjugate transposition of a matrix $\boldsymbol{X}$, respectively. $\mathbb{R}^d$ is a space where its elements are real vectors with length d. $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ means the inner product of vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. $||.||$ denotes the ℓ_2-norm function. $\mathcal{CN}(\boldsymbol{\mu}, \boldsymbol{Q})$ denotes the circularly symmetric complex Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{Q}$. ∇g is the gradient of a function g. $\mathbb{E}\{x\}$ denotes the expected value of a random variable x.

II. Proposed Scheme for CFmMIMO Networks to Support FL

A. The General FL Framework

A global ML problem is solved at a central server with a global training data set partitioned over a number of participating clients. Each client trains their local model by an arbitrary learning algorithm. Let $\mathcal{K} = \{1, \ldots, K\}$ be the set of clients and D_k the size of the local data stored at client k. Then $\widetilde{D} = \sum_{k \in \mathcal{K}} D_k$ is the size of the global training data. Denote by $\mathcal{D} = \{1, \ldots, \widetilde{D}\}$ and $\mathcal{D}_k = \{1, \ldots, D_k\}$ the index sets of the global data samples and the local data samples at a client k, respectively. In a typical supervised learning, a data sample $i \in \mathcal{D}$ is defined as an input-output pair $\{\boldsymbol{x}_i \in \mathbb{R}^d, y_i\}$.

For $\lambda > 0$, the general global ML problem can be posed as the following minimization [11], [21]

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} J(\boldsymbol{w}) \triangleq \frac{1}{\widetilde{D}} \sum_{i \in \mathcal{D}} f_i(\boldsymbol{w}) + \lambda g(\boldsymbol{w}), \tag{1}$$

Algorithm 1 A General FL Framework

1: **Input**: $n = 1$, an initial global DL training update
2: **repeat**
3: (S1) The central server sends the global DL training update to the UEs.
4: **for** $k \in \mathcal{K}$ in parallel **do**
5: (S2) Client k updates and solves its local ML problem on its local data set and then computes the global UL training update
6: (S3) Client k sends its computed global UL training update to the central server
7: **end for**
8: (S4) The central server computes the global DL training update by aggregating the received UL training updates.
9: Update $n = n + 1$
10: **until** convergence with the global accuracy ϵ

where $f_i(\boldsymbol{w})$ is the loss function at data sample i and $g(\boldsymbol{w})$ is a regularization term with a model parameter $\boldsymbol{w}$. Some popular examples are $f_i(\boldsymbol{w}) = \frac{1}{2}(\boldsymbol{x}_i^T \boldsymbol{w} - y_i)^2$ for a linear regression problem and $f_i(\boldsymbol{w}) = \{0, 1 - y_i \boldsymbol{x}_i^T \boldsymbol{w}\}, y_i \in \{-1, 1\}$ for a support vector machine. Here, the learning problem is to find $\boldsymbol{w}$ that characterizes the output y_i with the loss function $f_i(\boldsymbol{w})$ for a given input $\boldsymbol{x}_i$. Note that $f_i(\boldsymbol{w})$ is not necessarily convex.

In a general FL framework to solve the general ML problem (1), this problem is decomposed into K separate local ML problems that are solved at K clients in parallel. For ease of presentation, we make the following definitions.

Definition 1: "Global DL training update" is the information sent from the central server to the clients. Similarly, "global UL training updates" are those from the clients to the central server.

The general FL framework is described in Algorithm 1. Each iteration of Algorithm 1 consists the four key steps (S1)-(S4).

Definition 2: "An FL process" is defined as a full execution of Algorithm 1.

Remark 1: The designs of local ML problems at the clients, the global DL/UL training updates, and the types of aggregation of training updates at the central server are different according to the different designs of FL frameworks for different types of objective functions due to different ML applications [21], [23]–[27].

B. Proposed Cell-Free Massive MIMO Network Structure to Support the General FL Framework

Here, we propose using the CFmMIMO network structure [19] illustrated in Fig. 1 to support the general FL framework discussed above. In this structure, a central processing unit (CPU) is connected to a set of access points (APs) $\mathcal{M} = \{1, \ldots, M\}$ via backhaul links with sufficient capacities. These APs serve a given set $\mathcal{K}$ of participating UEs via wireless access links at the same time and in the same

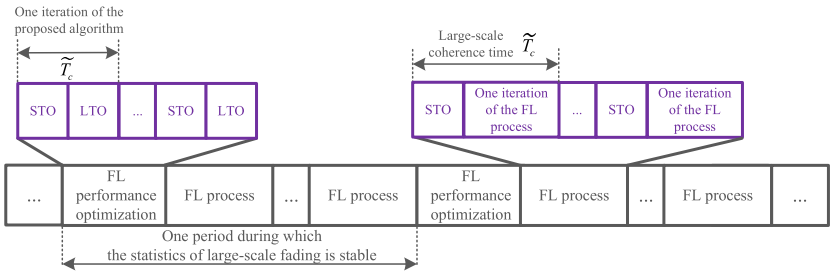

Fig. 2. Proposed scheme of a CFmMIMO system to support FL.

frequency bands.[2] The APs and UEs are each equipped with a single antenna. The CPU and UEs act as the central server and the clients in the general FL framework, respectively. The APs are used to relay the training updates between the CPU and the UEs

Remark 2: Both the AP and UEs can be considered as the clients in the general FL framework. In this paper, we choose the UEs to be the clients instead of the APs [10]. By doing so, data privacy is more protected since raw data does not have to be shared over wireless networks. Moreover, the computational resource of the UEs can be empowered with new computing engines such as Snapdragon 835 [4]. It is possible to push the ML tasks further to the UEs [9], [10].

C. Proposed Scheme for a CFmMIMO to Support the General FL Framework

Now, to realize an FL process in the considered CFmMIMO network structure, we propose a general scheme shown in Fig. 2. As seen from the figure, we divide the time period during which the statistics of large-scale fading is stable into multiple time intervals. The first interval of "FL performance optimization" is used for optimizing the performance of FL. The remaining intervals are reserved for the FL process; hence, termed as "FL process" intervals.

1) Optimizing the Performance of FL: In the "FL performance optimization" interval, any algorithms can be developed to optimize the performance of FL before the FL processes are executed. Here, we denote by "system parameters" the parameters that are designed by the FL optimization algorithm. These parameters are grouped into short-term and long-term parameters. The short-term parameters change in the timescale of large-scale coherence times, whereas the long-term parameters the statistics of large-scale fading. In each iteration of the optimization algorithm, the short-term parameters are designed in the short-term optimization (STO) time blocks, whereas the long-term parameters in the long-term optimization (LTO) time blocks. As a result of the channel hardening effect in massive MIMO, the wireless channel remains unchanged during one large-scale coherence time $\widetilde{T}_c$. In practice, the completion time of the optimization algorithm can be larger than $\widetilde{T}_c$. Therefore, we insist that only one iteration of the algorithm is to happen within $\widetilde{T}_c$.

2) Implementing the FL Process: In the "FL process" intervals, the FL process is executed with the long-term parameters obtained from the "FL performance optimization" interval. The short-term parameters are optimized in the STO time block to enhance the performance of training update transmission before each iteration of the FL process. Since the completion time of the FL process can be larger than $\widetilde{T}_c$, we insist that both STO and "one iteration of the FL process" time blocks are to happen in one $\widetilde{T}_c$. Here, we note that the results of the LTO time blocks remain unchanged for several FL process, while the results of the STO time blocks are invariant only in one iteration of an FL process. Therefore, after the "FL performance optimization" step is executed, only the results of the LTO time block are used in FL processes. In each iteration of an FL process, the results of the STO time block are not the results of the STO time block in the "FL performance optimization" step, but rather are computed by the same method used for the STO time block in the "FL performance optimization" step.

3) Implementing Each Iteration of the FL Process: Fig. 3 shows the time block of "one iteration of the FL process", in which the intervals of the four key steps (S1)-(S4) to implement each iteration of the FL process by the CFmMIMO network model are illustrated. Here, the interval of Step (S1) in Algorithm 1 consists one interval of "DL training update transmission (DLTUT) via backhaul links" from the CPU to the APs, and one interval of "DLTUT via wireless links" from the APs to the UEs. The interval of Step (S3) includes one interval of "UL training udpate transmission (ULTUT) via wireless links" from the UEs to the APs, and one interval of "ULTUT via backhaul links" from the APs to the CPU.

As also seen from Fig. 3, we split each time block of "DLTUT via wireless links" or "ULTUT via wireless links" into multiple intervals. The intervals of "UL training" is used for channel estimation. The remaining intervals are used for DL/UL training update transmission.[3] In these "DLTUT" or "ULTUT" intervals, any beamforming/filtering design can be applied to optimize the performance of training update

[2]In general, since the UEs are geographically distributed, some user may loss the connectivity, and hence, cannot participate in the FL process. However, one of the main strong property of CFmMIMO is that, with very high probability, it can provide uniformly good service for all users in the networks [19]. In the other words, in CFmMIMO, the connectivity probability of each user is very high. Therefore, in the paper, for simplicity, we assume that all UEs participate in the FL process.

[3]The proposed scheme above focuses only on support FL. Therefore, in each small-scale coherence time, all the times not used to estimate the channel are used for transmitting training updates. The scheme that supports FL and data transmission at the same time is left for future works.

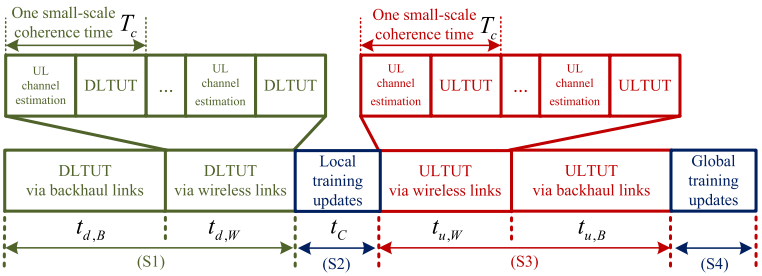

Fig. 3. One iteration of the FL process.

transmission. Here, we insist that the pair of "UL training" and "DLTUT"/"ULTUT" intervals happen in one small-scale coherence time T_c in order to adapt to the variation of small-scale fading.[4] In addition, the data size of training updates may be larger than the amount of data that can be transmitted in T_c. As such, there may be one or several "DLTUT" or "ULTUT" intervals in each "DLTUT via wireless links" or "ULTUT via wireless links" time block in order to complete the transmission of one global DL/UL training update.

Remark 3: The proposed CFmMIMO network model in Section II-B and the proposed scheme in Section II-C can be used to support any version of the general FL framework to solve any global ML problem. This scheme only requires developing specific algorithms to optimize the performance of specific FL frameworks. In the following, we consider a specific example of the general FL framework and investigate an algorithm to optimize the FL performance of this framework.

III. A Specific Example of the General FL Framework

Given the proposed framework in Section II, this section considers a specific example of the general FL framework to show how the performance of an FL process can be optimized in the latter sections. In particular, the FL optimization problem for this example is introduced in Section V, and the algorithms to solve this problem are proposed in Section VI.

Because of all the potential advantages offered by FL, many versions [21], [23]–[27] of the general FL framework have so far been studied despite research on FL is still in its infancy. Here, we consider an existing FL framework of [21] which is briefly introduced as follows. In this FL framework, the considered loss function f_i is convex and the dual problem of (1) is written as

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^{\widetilde{D}}} J_{\widetilde{D}}(\boldsymbol{\alpha}) \triangleq \frac{1}{\widetilde{D}} \sum_{i\in\mathcal{D}} -f_i^*(-\boldsymbol{\alpha}) - \lambda g^*(\boldsymbol{\chi}(\boldsymbol{\alpha})), \tag{2}$$

[4]In this work, we consider time-division duplexing (TDD) where channels are first estimated at the APs via uplink channel estimation. Owing to the channel reciprocity property in massive MIMO, these channel estimates will then be used for: (i) precoding the data symbols in the downlink data transmission, and (ii) used for combining the received signals in the uplink data transmission. Moreover, there is no need for downlink channel estimation because of the channel hardening property in massive MIMO systems. This transmission protocol is widely used in the massive MIMO literature (see, e.g., [28]).

which is a special case of the Fenchel duality [21], where f_i^* and g^* are the convex conjugate functions of f_i and g, respectively; $\boldsymbol{\alpha}$ is a dual variable; $\boldsymbol{\chi}(\boldsymbol{\alpha}) = \frac{1}{\lambda\widetilde{D}}\boldsymbol{X}\boldsymbol{\alpha}$; $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\widetilde{D}}] \in \mathbb{R}^{d\times\widetilde{D}}$. It can be shown that if $\boldsymbol{\alpha}^*$ is an optimal solution of (2), then $\boldsymbol{w}(\boldsymbol{\alpha}^*) = \nabla g^*\left(\frac{1}{\lambda\widetilde{D}}\boldsymbol{X}\boldsymbol{\alpha}^*\right)$ is the optimal solution of (1). This property allows handling the dual variable $\boldsymbol{\alpha} \in \mathbb{R}^{\widetilde{D}}$ instead of $\boldsymbol{w} \in \mathbb{R}^d$. Since each α_i corresponds to each data sample i, $\boldsymbol{\alpha}$ can be distributed in the same way that data are partitioned for the K clients.

Let $(\boldsymbol{\phi})_i$ be the i-th element of vector $\boldsymbol{\phi}$ and $(\boldsymbol{X})_i$ be the i-th column vector of matrix $\boldsymbol{X}$. For any $\boldsymbol{\phi} \in \mathbb{R}^{\widetilde{D}}$, denote by $\boldsymbol{\phi}_{[k]} \in \mathbb{R}^{\widetilde{D}}$ the vector for device k, i.e., $(\boldsymbol{\phi}_{[k]})_i = (\boldsymbol{\phi})_i$ if $i \in \mathcal{D}_k$ and 0 otherwise. Similarly, for any $\boldsymbol{X} \in \mathbb{R}^{d\times\widetilde{D}}$, denote by $\boldsymbol{X}_{[k]} \in \mathbb{R}^{d\times\widetilde{D}}$ the matrix for device k, i.e., $(\boldsymbol{X}_{[k]})_i = (\boldsymbol{X})_i$ if $i \in \mathcal{D}_k$ if $i \in \mathcal{D}_k$ and $\mathbf{0}$ otherwise. The local problem at each client k is then assigned to find the optimal change $\boldsymbol{\phi}_{[k]}$ in the local dual variable $\boldsymbol{\alpha}_{[k]}$, for a given previous $\boldsymbol{\alpha}$ as

$$\max_{\boldsymbol{\phi}_{[k]}\in\mathbb{R}^{\widetilde{D}}} J_k(\boldsymbol{\phi}_{[k]}, \boldsymbol{\chi}(\boldsymbol{\alpha}), \boldsymbol{\alpha}_{[k]}), \tag{3}$$

where

$$\begin{aligned} &J_k(\boldsymbol{\phi}_{[k]}, \boldsymbol{\chi}(\boldsymbol{\alpha}), \boldsymbol{\alpha}_{[k]}) \\ &\quad = -\tfrac{1}{K} g^*(\boldsymbol{\chi}(\boldsymbol{\alpha})) - \left\langle \tfrac{1}{\widetilde{D}} \boldsymbol{X}_{[k]}^T \nabla g^*(\boldsymbol{\chi}(\boldsymbol{\alpha})), \boldsymbol{\phi}_{[k]} \right\rangle \\ &\qquad - \tfrac{\lambda}{2} \left\| \tfrac{1}{\lambda\widetilde{D}} \boldsymbol{X}_{[k]} \boldsymbol{\phi}_{[k]} \right\|^2 - \tfrac{1}{\widetilde{D}} \sum_{i\in\boldsymbol{\mathcal{D}}_k} f_i^*(-(\boldsymbol{\alpha}_{[k]})_i - (\boldsymbol{\phi}_{[k]})_i) \end{aligned} \tag{4}$$

is the quadratic approximation of $J_{\widetilde{D}}$ at the dual variable $(\boldsymbol{\alpha}_{[k]} + \boldsymbol{\phi}_{[k]})$. Here, the only information shared between the clients and the central server is the change in $\boldsymbol{\chi}$.

At iteration n, each client $k \in \mathcal{K}$ solves (3) by an arbitrary iterative algorithm with a local accuracy level of θ in order to obtain its optimal solution $\boldsymbol{\phi}_{[k]}^*$. The local dual variable is updated as

$$\boldsymbol{\alpha}_{[k]}^{(n+1)} = \boldsymbol{\alpha}_{[k]}^{(n)} + \boldsymbol{\phi}_{[k]}^* . \tag{5}$$

Each client $k \in \mathcal{K}$ then shares its local change in $\boldsymbol{\chi}^{(n)}$, i.e.,

$$\Delta\boldsymbol{\chi}_k^{(n)} = \tfrac{1}{\lambda\widetilde{D}} \boldsymbol{X}_{[k]} \boldsymbol{\phi}_{[k]}^*, \tag{6}$$

to the central server. The central server aggregates the local information $\Delta\boldsymbol{\chi}_k^{(n)}$ received from all the clients and updates $\boldsymbol{\chi}$ as

$$\boldsymbol{\chi}^{(n+1)} = \boldsymbol{\chi}^{(n)} + \tfrac{1}{K} \Delta\boldsymbol{\chi}_k^{(n)} . \tag{7}$$

Algorithm 2 FL Framework [21]

Input: $n = 1$, an initial point $\boldsymbol{\alpha}^{(0)}$ and an initial global DL training update $\boldsymbol{\chi}^{(0)} = \frac{1}{\lambda \widetilde{D}} \boldsymbol{X}\, \boldsymbol{\alpha}^{(0)}$
repeat
  (S1) The CPU sends $\boldsymbol{\chi}^{(n)}$ to the UEs
  for $k \in \mathcal{K}$ in parallel **do**
    (S2) UE k solves (3) by an iterative algorithm with a local accuracy θ to obtain an optimal solution $\boldsymbol{\phi}^*_{[k]}$, and then updates the global UL training update $\Delta \boldsymbol{\chi}_k^{(n)}$ by (6)
    (S3) UE k sends $\Delta \boldsymbol{\chi}_k^{(n)}$ to the CPU
  end for
  (S4) The CPU updates $\boldsymbol{\chi}^{(n+1)}$ as (7) and sends it back to the UEs
  Update $n = n + 1$
until convergence with the global accuracy ϵ

$\boldsymbol{\chi}^{(n+1)}$ is finally sent back to the clients to solve (3). This process will terminate when a global accuracy level of ϵ is reached. The FL framework described above is summarized in Algorithm 2. Each iteration of Algorithm 2 also consists of four key steps (S1)-(S4) as that of Algorithm 1.

We assume that each client $k \in \mathcal{K}$ uses optimization algorithms such as stochastic average gradient (SAG) and stochastic variance reduced gradient (SVRG) to solve (3) with a local accuracy level of θ. The number of local iterations is then [26]

$$L(\theta) = \nu \log(\tfrac{1}{\theta}), \tag{8}$$

where $\nu > 0$ depends on the data size and structure of the local problem [26]. On the other hand, for strongly convex objective functions and the global accuracy level of ϵ, the number of global iterations are given by [21]

$$G(\theta) = \frac{\vartheta \log\left(\frac{1}{\epsilon}\right)}{1-\theta}, \tag{9}$$

where $\vartheta > 0$ is a factor that depends on the characteristic and size of the whole data set [21, Theorem 4.2]. Here, we assume that the characteristic and size of the whole data set does not change over the FL processes. Therefore, ϑ is constant and known [10].

Remark 4: In more complex ML models such as deep neural networks, the loss function f_i is usually non-convex. Therefore, instead of the framework in [21], more advanced FL frameworks are needed. In this paper, we choose to consider the FL framework in [21] for simpler ML models because it provides a clear relationship (9) between the number of global training updates and the local accuracy for ease of optimizing FL performance (see the next sections). Such a clear relationship is hard to find in the advanced FL frameworks with non-convex loss functions.

IV. Detailed System Model to Support FL

This section details the CFmMIMO system model used to support the transmission and computation of the training updates in each iteration of the FL process (see Fig. 3).

A. Steps (S1) and (S3) in Each Iteration of the FL Process: Model of Training Update Transmission

1) UL Channel Estimation: Denote by $\tau_c = T_c B_c$ the number of samples of each coherence block, where B_c the coherence bandwidth. UL pilot sequences are sent by all the UEs to all the APs simultaneously. Denote by τ_t (samples) the length of one pilot sequence. Let $\sqrt{\tau_t}\boldsymbol{\varphi}_k \in \mathbb{C}^{\tau_t \times 1}$ be the pilot sequence transmitted from UE $k \in \mathcal{K}$, where $\| \boldsymbol{\varphi}_k \|^2 = 1, \forall k \in \mathcal{K}$. The channel from a UE k to an AP m is modeled as $g_{mk} = (\beta_{mk})^{1/2} \tilde{g}_{mk}$, where β_{mk} and $\tilde{g}_{mk} \in \mathbb{C}$ represent the large-scale fading and small-scale fading channel coefficients, respectively. Assume that $\tilde{g}_{mk}$ is an independent and identically distributed (i.i.d.) $\mathcal{CN}(0,1)$ random variable.

The AP m receives the pilot vector $\boldsymbol{y}_m = \sqrt{\tau_t \rho_t} \sum_{k \in \mathcal{K}} g_{mk} \boldsymbol{\varphi}_k + \boldsymbol{w}_m$, where ρ_t is the normalized signal-to-noise ratio (SNR) of each pilot symbol, and $\boldsymbol{w}_m \in \mathcal{CN}(\boldsymbol{0}, \boldsymbol{I})$ is the additive noise at the AP m. The projection of $\boldsymbol{y}_m$ onto $\boldsymbol{\varphi}_k$ is given as $\hat{y}_{mk} = \boldsymbol{\varphi}_k^H \boldsymbol{y}_m = \sqrt{\tau_t \rho_t} \sum_{\ell \in \mathcal{K}} g_{m\ell} \boldsymbol{\varphi}_k^H \boldsymbol{\varphi}_\ell + \boldsymbol{\varphi}_k^H \boldsymbol{w}_m$. After receiving $\hat{y}_{mk}$, the AP m estimates g_{mk} by using the minimum mean-square error (MMSE) estimation. Given $\hat{y}_{mk}$, the MMSE estimate $\hat{g}_{mk}$ of g_{mk} is obtained as [29]: $\hat{g}_{mk} = \mathbb{E}\{\hat{y}^*_m g_{mk}\} (\mathbb{E}\{|\hat{y}_{mk}|^2\})^{-1} \hat{y}_{mk} = c_{mk} \hat{y}_{mk}$, where $c_{mk} \triangleq \frac{\sqrt{\tau_t \rho_t} \beta_{mk}}{\sum_{\ell \in \mathcal{K}} \tau_t \rho_t \beta_{m\ell} |\boldsymbol{\varphi}_k^H \boldsymbol{\varphi}_\ell|^2 + 1}$. From the property of MMSE channel estimation, $\hat{g}_{mk}$ is distributed according to $\mathcal{CN}(0, \sigma^2_{mk})$, where $\sigma^2_{mk} = \frac{\tau_t \rho_t (\beta_{mk})^2}{\sum_{\ell \in \mathcal{K}} \tau_t \rho_t \beta_{m\ell} |\boldsymbol{\varphi}_k^H \boldsymbol{\varphi}_\ell|^2 + 1}$ [29].

Remark 5: In indoor communications, the time for UL training can be much smaller than the small-scale coherence time T_c. For example, a system supporting users' mobility of $v = 0.75$ m/s $= 2.7$ km/h, delay spread of $T_d = 0.5$ μs and carrier frequency $f_c = 2$ GHz has a small-scale coherence time of $T_c = \frac{c}{4 f_c v} = 50$ ms, and coherence bandwidth $B_c = \frac{1}{2T_d} = 1$ MHz [28]. Suppose $\tau_t = 20$, the time for UL channel estimation is $t_{ce} = \frac{\tau_t}{B_c} = 0.02$ ms $\ll T_c$. Therefore, the time for UL channel estimation can be ignored in one small-scale coherence time and one FL process interval.

2) Step (S1) in Each Iteration of the FL Process: At the CPU, the global DL training update intended for a UE k is encoded into a symbol $s_{d,k} \sim \mathcal{CN}(0,1)$. Here, each DL/UL training update is considered as a data message that is widely used in the literature of wireless communications [20]. The CPU then sends $s_{d,k}, \forall k \in \mathcal{K}$, to all the APs over backhaul links. Let S_d (bits) and $R_{d,k}$ (bps) be the data size and the data rate of the global DL training update for the UE k, respectively. The download latency from the CPU to all the APs is given by

$$t_{d,B}(\boldsymbol{R}_d) = \frac{K S_d}{\sum_{k \in \mathcal{K}} R_{d,k}}, \tag{10}$$

where $\boldsymbol{R}_d \triangleq \{R_{d,k}\}_{k \in \mathcal{K}}$.

For ease of implementation, we apply a conjugate beamforming scheme to the APs to precode the message signals before wirelessly transmitting them to the UEs (using the channel estimates from the UL channel estimation). The transmitted signal at an AP m is expressed as $x_{d,m} = \sqrt{\rho_d} \sum_{k \in \mathcal{K}} \sqrt{\eta_{mk}} (\hat{g}_{mk})^* s_{d,k}$, where ρ_d is the maximum nor-

malized transmit power (normalized by the noise power N_0) at each AP and $\eta_{mk}, \forall m \in \mathcal{M}, k \in \mathcal{K}$, is a power control coefficient. The AP m is required to meet the average normalized power constraint, i.e., $\mathbb{E}\{|x_{d,m}|^2\} \leq \rho_d$, which can also be expressed as the following per-AP power constraint:

$$\sum_{k \in \mathcal{K}} \sigma^2_{mk} \eta_{mk} \leq 1, \quad \forall m. \tag{11}$$

The received signal at the UE k is given by $r_{d,k} = \sum_{m \in \mathcal{M}} g_{mk} x_{d,m} + w_k$, where w_k is the additive noise $\mathcal{CN}(0,1)$ at the UE k. The achievable DL rate at the UE k is

$$R_{d,k} \leq h_{d,k}(\boldsymbol{\eta}), \tag{12}$$

where $\boldsymbol{\eta} \triangleq \{\eta_{mk}\}_{m \in \mathcal{M}, k \in \mathcal{K}}$ and $h_{d,k}(\boldsymbol{\eta})$ is given in (13) shown at the bottom of the next page [19]. Note that in (13), B is the bandwidth. The download latency from the APs to the UE k is given by

$$t_{d,k}(R_{d,k}) = \frac{S_d}{R_{d,k}}. \tag{14}$$

3) Step (S3) in Each Iteration of the FL Process: After updating the local model, the UE k encodes the global UL training update into a symbol $s_{u,k} \sim \mathcal{CN}(0,1)$. The symbol $s_{u,k}$ is then allocated a transmit amplitude value $\sqrt{\rho_u \zeta_k}$ to generate a baseband signal $x_{u,k}$ for wireless transmissions, i.e., $x_{u,k} = \sqrt{\rho_u \zeta_k} s_{u,k}$. The UE k is adhered to the average transmit power constraint, i.e., $\mathbb{E}\{|x_{u,k}|^2\} \leq \rho_u$, which can also be expressed in a per-UE constraint as

$$0 \leq \zeta_k \leq 1, \quad \forall k \in \mathcal{K}. \tag{15}$$

The upload latency from the UE k to the AP m is given by

$$t_{u,k}(R_{u,k}) = \frac{S_u}{R_{u,k}}, \tag{16}$$

where S_u (bits) and $R_{u,k}$ (bps) are the data size and the data rate of the global UL training update, respectively.

The received signal at the AP m is expressed as

$$\begin{aligned} y_{u,m} &= \sum_{k \in \mathcal{K}} g_{mk} x_{u,k} + w_{u,m} \\ &= \sqrt{\rho_u} \sum_{k \in \mathcal{K}} g_{mk} \sqrt{\zeta_k} s_{u,k} + w_{u,m}, \end{aligned} \tag{17}$$

where $w_{u,m} \sim \mathcal{CN}(0,1)$ is the additive noise. To detect the message symbol transmitted from the UE k, the AP m computes and sends $(\hat{g}_{mk})^* y_{u,m}$ to the CPU. The upload latency from the APs to the CPU is expressed as

$$t_{u,B}(\boldsymbol{R}_u) = \frac{K S_u}{\sum_{k \in \mathcal{K}} R_{u,k}}, \tag{18}$$

where $\boldsymbol{R}_u \triangleq \{R_{u,k}\}_{k \in \mathcal{K}}$.

At the CPU, the symbol $s_{u,\ell}$ is detected from the received signal $r_{u,k}$:

$$\begin{aligned} r_{u,k} = \sqrt{\rho_u} &\sum_{m \in \mathcal{M}} \sqrt{\zeta_k} (\hat{g}_{mk})^* g_{mk} s_{u,k} \\ &+ \sqrt{\rho_u} \sum_{m \in \mathcal{M}} \sum_{\ell \in \mathcal{K} \setminus k} \sqrt{\zeta_\ell} (\hat{g}_{mk})^* g_{m\ell} s_{u,\ell} \\ &+ \sum_{m \in \mathcal{M}} (\hat{g}_{mk})^* w_{u,m}. \end{aligned} \tag{19}$$

The achievable UL rate for the UE k is given by

$$R_{u,k} \leq h_{u,k}(\boldsymbol{\zeta}), \tag{20}$$

where $\boldsymbol{\zeta} \triangleq \{\zeta_k\}_{k \in \mathcal{K}}$ and $h_{u,k}(\boldsymbol{\zeta})$ is defined in (21) shown at the bottom of the next page [19].

B. Step (S2) in Each Iteration of the FL Process: Model of Local Training Update Computation at UEs

Denote by c_k (cycles/sample) the number of processing cycles for a UE k to process one data sample. c_k is known *a priori* by an offline measurement [30]. Let D_k (samples) and f_k (cycles/s) be the size of the local data set and the processing frequency of the UE k, respectively. The latency of computing the local training update at the UE k is by

$$t_{C,k}(\theta, f_k) = L(\theta) \frac{D_k c_k}{f_k}, \tag{22}$$

where $L(\theta)$ is the number of local training iterations (see (8)) and $\frac{D_k c_k}{f_k}$ is the time taken to compute the local update over its local training data set in each iteration. Given the limited computational resource at the UEs, we only focus on the delay of computing the local updates at the UEs. Since the computational resource of the CPU is much more abundant than that of the UEs, the latency of aggregating the global UL training updates at the CPU is negligible, and hence ignored.

C. The Model of UE's Energy Consumption

Because the time for UL channel estimation is negligible compared with one FL training interval, the energy consumed in the time block of UL channel estimation is ignored. The energy consumption in the time block of ULTUT at a UE k is given by

$$E_{T,k}(\zeta_k, R_{u,k}) = \rho_u N_0 \zeta_k \frac{S_u}{R_{u,k}}, \tag{23}$$

where ζ_k is the transmitted UL power and $\frac{S_u}{R_{u,k}}$ is the delay incurred by transmitting the global UL training update $\Delta \boldsymbol{\chi}_k$. The energy required for computing local training updates at the UE k is expressed as [10]

$$E_{C,k}(\theta, f_k) = L(\theta) \frac{\alpha}{2} c_k D_k f_k^2, \tag{24}$$

where $\frac{\alpha}{2}$ is the effective capacitance coefficient of the UEs' computing chipset.

V. FL Training Time Minimization: Problem Formulation

To optimize the performance of the FL process using the considered FL framework [21] in the CFmMIMO network model discussed in Sections IV, this paper targets the key performance metric of *training time minimization*.

In each iteration of the FL process, the time of Step (S1) for a UE k involves the transmission delay of sending the global DL training update from the CPU to the APs via backhaul links and that from the APs to UE k via wireless links, i.e.,

$$t^d_{T,k}(\boldsymbol{R}_d) = t_{d,B}(\boldsymbol{R}_d) + t_{d,k}(R_{d,k}) = \frac{K S_d}{\sum_{k \in \mathcal{K}} R_{d,k}} + \frac{S_d}{R_{d,k}}. \tag{25}$$

Similarly, the time of Step (S2) for the UE k consists of the delay of transmitting the global UL training update from it to the APs and from the APs to the CPU, i.e.

$$t^u_{T,k}(\boldsymbol{R}_u) = t_{u,k}(R_{u,k}) + t_{u,B}(\boldsymbol{R}_u) = \frac{S_u}{R_{u,k}} + \frac{KS_u}{\sum_{k\in\mathcal{K}} R_{u,k}}. \tag{26}$$

In the proposed scheme in Section II-C, each of the steps (S1)-(S4) of one iteration of the FL process must be completed for all the UEs before the latter step is executed. Therefore, the time of one iteration of the FL process is

$$\begin{aligned} &T_G(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u) \\ &= \max_{k\in\mathcal{K}} t^d_{T,k}(\boldsymbol{R}_d) + \max_{k\in\mathcal{K}} t_{C,k}(\theta, f_k) + \max_{k\in\mathcal{K}} t^u_{T,k}(\theta, \boldsymbol{f}, \boldsymbol{R}_u) \\ &= t_{d,B}(\boldsymbol{R}_d) + \max_{k\in\mathcal{K}} t_{d,k}(R_{d,k}) + \max_{k\in\mathcal{K}} t_{C,k}(\theta, f_k) \\ &\quad + \max_{k\in\mathcal{K}} t_{u,k}(R_{u,k}) + t_{u,B}(\boldsymbol{R}_u) \\ &\triangleq t_{d,B} + t_{d,W} + t_C + t_{u,W} + t_{u,B}, \end{aligned} \tag{27}$$

where $\boldsymbol{f} \triangleq \{f_k\}_{k\in\mathcal{K}}$; $t_{d,W}$ or $t_{u,W}$ is the maximum delay for a complete DLTUT or ULTUT via wireless links; t_C is the maximum delay for all the UEs to complete their local training update computation. Note again that the time of the global training update at the CPU is ignored as discussed in Section IV-B.

As can be seen from (27), $T_G(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)$ relies on both $(\boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)$ and θ. However, only $(\boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)$ is optimized to reduce the time $T_G(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)$ of one iteration of the FL process in each large-scale coherence time. This is because any change of θ leads to the change in the number of iterations $G(\theta)$ of the FL process as shown in (9). Therefore, $(\boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)$ and θ must be optimized independently in different timescales.

To measure how efficient the time of each iteration of the FL process is optimized over different large-scale coherence times, we introduce a new metric termed "ergodic time of one iteration of the FL process", i.e., $\mathbb{E}\{T_G(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)\}$. Here, $\mathbb{E}\{T_G(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)\}$ is the average of $T_G(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)$ over the large-scale fading realizations. The effective time of one FL process is then defined as

$$\begin{aligned} T_e(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u) &\triangleq G(\theta)\, \mathbb{E}\{T_G(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)\} \\ &= \vartheta \log(\tfrac{1}{\epsilon})\, \mathbb{E}\left\{\frac{T_G(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)}{1-\theta}\right\} \\ &= \vartheta \log(\tfrac{1}{\epsilon})\, \mathbb{E}\{T(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)\}, \end{aligned} \tag{28}$$

where $T(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u) \triangleq \frac{T_G(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)}{1-\theta}$. For ease of presentation, we make the following definition.

Definition 3: An effective training time of FL is the effective time of one FL process and is computed as (28).

The problem of FL training time minimization for the FL framework [21] in the considered CFmMIMO system model is thus formulated as:

$$\min_{\boldsymbol{\eta},\boldsymbol{\zeta},\theta,\boldsymbol{f},\boldsymbol{R}_d,\boldsymbol{R}_u} \; g(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u) \triangleq \mathbb{E}\{T(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)\} \tag{29a}$$

$$\text{s.t. } (11), (12), \; (15), (20)$$

$$E_{T,k}(\zeta_k, R_{u,k}) + E_{C,k}(\theta, f_k) \le E_{k,\max}, \quad \forall k \tag{29b}$$

$$f_{k,\min} \le f_k \le f_{k,\max}, \quad \forall k \tag{29c}$$

$$0 \le \eta_{mk}, \quad \forall m, k \tag{29d}$$

$$0 \le \zeta_k, \quad \forall k \tag{29e}$$

$$0 \le R_{d,k}, \quad \forall k \tag{29f}$$

$$0 \le R_{u,k}, \quad \forall k \tag{29g}$$

$$\theta_{\min} \le \theta \le \theta_{\max}. \tag{29h}$$

Here, problem (29) takes into account the issues related to device performance and user experience, i.e., limiting a maximum energy in (29b) and a maximum frequency processing (29c) in order to ensure that performing FL does not affect much on the UEs' other functions such as data transmission and computation. Problem (29) has a nonconvex stochastic, mixed-timescale structure, along with the tight coupling among the variables. Finding its globally optimal solution is challenging. This paper instead aims to propose a solution approach that is suitable for practical implementation.

Remark 6: At first glance, the optimization problem (29) is only valid for the FL framework [21] because (8) and (9). Nevertheless, on closer observation, the total training time of any FL process (including [21]) normally involves variables that are optimized in different timescales. The variables such as local accuracy θ are optimized in long-term timescales while the variables such as power, rates are optimized in short-term timescales. This leads to the stochastic structure of the training time minimization problems as discussed after (27). In this sense, the optimization problems for other FL frameworks can be different from (35) but their stochastic structures are the same as that of (29). Therefore, we only use the example [21] to show this structure. Moreover, the structure of the algorithms to solve these optimization problems is also the same as that shown in the next section.

$$h_{d,k}(\boldsymbol{\eta}) = \frac{\tau_c - \tau_t}{\tau_c} B \log_2 \left(1 + \frac{\rho_d \left(\sum_{m\in\mathcal{M}} \eta^{1/2}_{mk} \sigma^2_{mk}\right)^2}{\rho_d \sum_{\ell\in\mathcal{K}\setminus k} \left(\sum_{m\in\mathcal{M}} \eta^{1/2}_{m\ell} \sigma^2_{m\ell} \frac{\beta_{mk}}{\beta_{m\ell}}\right)^2 |\boldsymbol{\varphi}^H_\ell \boldsymbol{\varphi}_k|^2 + \rho_d \sum_{\ell\in\mathcal{K}} \sum_{m\in\mathcal{M}} \eta_{m\ell} \sigma^2_{m\ell} \beta_{mk} + 1}\right) \tag{13}$$

$$h_{u,k}(\boldsymbol{\zeta}) = \frac{\tau_c - \tau_t}{\tau_c} B \log_2 \left(1 + \frac{\rho_u \zeta_k \left(\sum_{m\in\mathcal{M}} \sigma^2_{mk}\right)^2}{\rho_u \sum_{\ell\in\mathcal{K}\setminus k} \zeta_\ell \left(\sum_{m\in\mathcal{M}} \sigma^2_{mk} \frac{\beta_{m\ell}}{\beta_{mk}}\right)^2 |\boldsymbol{\varphi}^H_k \boldsymbol{\varphi}_\ell|^2 + \rho_u \sum_{\ell\in\mathcal{K}} \zeta_\ell \sum_{m\in\mathcal{M}} \sigma^2_{mk} \beta_{m\ell} + \sum_{m\in\mathcal{M}} \sigma^2_{mk}}\right) \tag{21}$$

VI. FL Training Time Minimization: Proposed Algorithm

To resolve problem (29), we utilize the online successive convex approximation approach for solving two-stage stochastic nonconvex optimization problems in [22]. Note that while [22] only provides a general description of the solution method, we specifically tailor it to devise a new algorithm for (29).

According to [31], problem (29) can be decomposed into a family of short-term subproblems and a long-term master problem as follows. For a given θ and large-scale fading coefficients $\boldsymbol{\beta} \triangleq \{\beta_{mk}\}_{m\in\mathcal{M},k\in\mathcal{K}}$ in each large-scale coherence time, the short-term subproblem is expressed as:

$$\min_{\boldsymbol{\eta},\boldsymbol{\zeta},\boldsymbol{f},\boldsymbol{R}_d,\boldsymbol{R}_u} T(\boldsymbol{f},\boldsymbol{R}_d,\boldsymbol{R}_u) \quad \text{s.t. (11), (12), (15), (20), (29b)} - \text{(29g)}. \tag{30}$$

For given optimal solutions $\{(\boldsymbol{\eta}^*,\boldsymbol{\zeta}^*,\boldsymbol{f}^*,\boldsymbol{R}_d^*,\boldsymbol{R}_u^*)\}$ to problems (30) at all large-scale coherence times, the long-term master problem is expressed as:

$$\min_{\theta} g(\theta) \triangleq \mathbb{E}\{T(\theta)\} \quad \text{s.t. (29h)}. \tag{31}$$

A. Solving the Short-Term Subproblem (30)

Problem (30) can be rewritten in an epigraph form as follows.

$$\min_{\boldsymbol{x}} \frac{\omega}{1-\theta} \tag{32a}$$

$$\text{s.t. } \omega \geq t_{d,B}(\boldsymbol{R}_d) + \max_{k\in\mathcal{K}} t_{d,k}(R_{d,k}) + \max_{k\in\mathcal{K}} t_{C,k}(\theta, f_k) + \max_{k\in\mathcal{K}} t_{u,k}(R_{u,k}) + t_{u,B}(\boldsymbol{R}_u) \tag{32b}$$

$$\rho_u N_0 \varrho_k S_u + \nu \log\left(\frac{1}{\theta}\right)\frac{\alpha}{2} c_k D_k f_k^2 \leq E_{k,\max}, \quad \forall k \tag{32c}$$

$$\zeta_k \leq \varrho_k R_{u,k}, \quad \forall k \tag{32d}$$

$$\text{(11), (12), (15), (20), (29c)} - \text{(29g)},$$

where $\boldsymbol{x} \triangleq (\boldsymbol{\eta},\boldsymbol{\zeta},\boldsymbol{f},\boldsymbol{R}_d,\boldsymbol{R}_u,\omega,\boldsymbol{\varrho})$, ω and $\boldsymbol{\varrho} \triangleq \{\varrho_k\}_{k\in\mathcal{K}}$ are additional variables. If we let $\boldsymbol{v} \triangleq \{v_{mk}\}_{m\in\mathcal{M},k\in\mathcal{K}}$ and $\boldsymbol{u} \triangleq \{u_k\}_{k\in\mathcal{K}}$ with

$$v_{mk} \triangleq \eta_{mk}^{1/2}, \forall m,k, \tag{33}$$

$$u_k \triangleq \zeta_k^{1/2}, \forall k, \tag{34}$$

then (32) can be rewritten as:

$$\min_{\widetilde{\boldsymbol{x}}} \frac{\omega}{1-\theta} \tag{35a}$$

$$\text{s.t. } \omega \geq \frac{KS_d}{\sum_{k\in\mathcal{K}} R_{d,k}} + t_d + t_C + t_u + \frac{KS_u}{\sum_{k\in\mathcal{K}} R_{u,k}} \tag{35b}$$

$$t_d \geq \frac{S_d}{R_{d,k}}, \quad \forall k \tag{35c}$$

$$t_C \geq \frac{\nu \log\left(\frac{1}{\theta}\right) D_k c_k}{f_k}, \quad \forall k \tag{35d}$$

$$t_u \geq \frac{S_u}{R_{u,k}}, \quad \forall k \tag{35e}$$

$$u_k^2 \leq \varrho_k R_{u,k}, \quad \forall k \tag{35f}$$

$$R_{d,k} \leq h_{d,k}(\boldsymbol{v}), \quad \forall k \tag{35g}$$

$$R_{u,k} \leq h_{u,k}(\boldsymbol{u}), \quad \forall k \tag{35h}$$

$$\sum_{k\in\mathcal{K}} \sigma_{mk}^2 v_{mk}^2 \leq 1, \quad \forall m \tag{35i}$$

$$0 < v_{mk}, \quad \forall m \in \mathcal{M}, k \in \mathcal{K} \tag{35j}$$

$$0 < u_k \leq 1, \quad \forall k \in \mathcal{K} \tag{35k}$$

$$\text{(29c), (29f), (29g), (32c)},$$

where $\widetilde{\boldsymbol{x}} \triangleq \{\boldsymbol{x},\boldsymbol{v},\boldsymbol{u},t_d,t_C,t_u\} \setminus \{\boldsymbol{\eta},\boldsymbol{\zeta}\}$, t_d, t_C and t_u are additional variables. Note that (35) is still challenging due to the nonconvex constraints (35f), (35g), and (35h).

To solve (35), we first rewrite (35f) as

$$z_k(u_k,\varrho_k,R_{u,k}) \leq 0, \quad \forall k. \tag{36}$$

where $z_k(u_k,\varrho_k,R_{u,k}) \triangleq 4u_k^2 - (\varrho_k + R_{u,k})^2 + (\varrho_k - R_{u,k})^2$. Note that, for a given point $(x^{(n)}, y^{(n)})$, a function $f(x,y) = -(x+y)^2$ has an upper bound $\widetilde{f}(x,y) \geq f(x,y)$ as

$$\widetilde{f}(x,y) \triangleq -2(x^{(n)}+y^{(n)})(x+y) + (x^{(n)}+y^{(n)})^2 + \delta((x-x^{(n)})^2 + (y-y^{(n)})^2), \tag{37}$$

where $\delta > 0$ can be any constant. Different from the upper bound used in [32], $\widetilde{f}(x,y)$ is introduced here with the term of $\delta((x-x^{(n)})^2 + (y-y^{(n)})^2)$ to ensure its strong convexity. Now, (35f) can be approximated at iteration $n+1$ by the following convex constraint

$$\widetilde{z}_k(u_k,\varrho_k,R_{u,k}) \leq 0, \quad \forall k. \tag{38}$$

where

$$\widetilde{z}_k(u_k,\varrho_k,R_{u,k}) \triangleq 4u_k^2 - 2(\varrho_k^{(n)} + R_{u,k}^{(n)})(\varrho_k + R_{u,k}) + (\varrho_k^{(n)} + R_{u,k}^{(n)})^2 + (\varrho_k - R_{u,k})^2 + \delta((\varrho_k - \varrho_k^{(n)})^2 + (R_{u,k} - R_{u,k}^{(n)})^2). \tag{39}$$

To deal with (35g) and (35h), we note that a function $f(x,y) = \log\left(1 + \frac{|x|^2}{y}\right)$ has the following lower bound [33]:

$$f(x,y) \geq \log\left(1 + \frac{|x^{(n)}|^2}{y^{(n)}}\right) - \frac{|x^{(n)}|^2}{y^{(n)}} + 2\frac{x^{(n)}x}{y^{(n)}} - \frac{|x^{(n)}|^2(|x|^2+y)}{y^{(n)}(|x^{(n)}|^2 + y^{(n)})}, \tag{40}$$

where $\underline{x} \in \mathbb{R}, y > 0, y^{(n)} > 0$. Therefore, the concave lower bound $\widetilde{h}_{d,k}(\boldsymbol{v})$ of $h_{d,k}(\boldsymbol{v})$ in (35g) is given by

$$\widetilde{h}_{d,k}(\boldsymbol{v}) \triangleq \log_2\left(1 + \frac{(\Upsilon_k^{(n)})^2}{\Pi_k^{(n)}}\right) - \frac{(\Upsilon_k^{(n)})^2}{\Pi_k^{(n)}} + 2\frac{\Upsilon_k^{(n)}\Upsilon_k}{\Pi_k^{(n)}} - \frac{(\Upsilon_k^{(n)})^2(\Upsilon_k^2 + \Pi_k)}{\Pi_k^{(n)}((\Upsilon_k^{(n)})^2 + \Pi_k^{(n)})} \leq h_{d,k}(\boldsymbol{v}), \tag{41}$$

where

$$\Upsilon_k(\{v_{mk}\}_{m\in\mathcal{M}}) = \sqrt{\rho_d}\sum_{m\in\mathcal{M}} v_{mk}\sigma_{mk}^2, \tag{42}$$

$$\Pi_k(\boldsymbol{v}) = \rho_d \sum_{\ell \in \mathcal{K}\setminus k} \Big(\sum_{m\in\mathcal{M}} v_{m\ell}\sigma_{m\ell}^2 \frac{\beta_{mk}}{\beta_{m\ell}} \Big)^2 |\boldsymbol{\varphi}_\ell^H \boldsymbol{\varphi}_k|^2 + \rho_d \sum_{\ell\in\mathcal{K}} \sum_{m\in\mathcal{M}} v_{m\ell}^2 \sigma_{m\ell}^2 \beta_{mk} + 1. \quad (43)$$

Similarly, the concave lower bound $\widetilde{h}_{u,k}(\boldsymbol{u})$ of $h_{u,k}(\boldsymbol{u})$ in (35h) is given by

$$\widetilde{h}_{u,k}(\boldsymbol{u}) \triangleq \log_2\Big(1 + \frac{(\Psi_k^{(n)})^2}{\Xi_k^{(n)}}\Big) - \frac{(\Psi_k^{(n)})^2}{\Xi_k^{(n)}} + 2\frac{\Psi_k^{(n)}\Psi_k}{\Xi_k^{(n)}} - \frac{(\Psi_k^{(n)})^2((\Psi_k^{(n)})^2 + \Xi_k)}{\Xi_k^{(n)}((\Psi_k^{(n)})^2 + \Xi_k^{(n)})} \le h_{u,k}(\boldsymbol{u}), \quad (44)$$

where

$$\Psi_k(u_k) = \rho_u^{1/2} u_k \Big(\sum_{m\in\mathcal{M}} \sigma_{mk}^2\Big), \quad (45)$$

$$\Xi_k(\boldsymbol{u}) = \rho_u \sum_{\ell\in\mathcal{K}\setminus k} u_\ell^2 \Big(\sum_{m\in\mathcal{M}} \sigma_{mk}^2 \frac{\beta_{m\ell}}{\beta_{mk}}\Big)^2 |\boldsymbol{\varphi}_k^H \boldsymbol{\varphi}_\ell|^2 + \rho_u \sum_{\ell\in\mathcal{K}} u_\ell^2 \sum_{m\in\mathcal{M}} \sigma_{mk}^2 \beta_{m\ell} + \sum_{m\in\mathcal{M}} \sigma_{mk}^2. \quad (46)$$

As such, (35g) and (35h) can be approximated by

$$R_{d,k} \le \widetilde{h}_{d,k}(\boldsymbol{v}), \quad \forall k \in \mathcal{K}, \quad (47)$$

$$R_{u,k} \le \widetilde{h}_{u,k}(\boldsymbol{u}), \quad \forall k \in \mathcal{K}. \quad (48)$$

At the iteration $n+1$, for a given point $\widetilde{\boldsymbol{x}}^{(n)}$, problem (35) (hence (30)) can finally be approximated by the following convex problem:

$$\min_{\widetilde{\boldsymbol{x}}\in\widetilde{\mathcal{F}}} \frac{\omega}{1-\theta}, \quad (49)$$

where $\widetilde{\mathcal{F}} \triangleq \{(29c), (29f), (29g), (32c), (35b) - (35e), (35i) - (35k), (38), (47), (48)\}$ is a convex feasible set.

In Algorithm 3, we outline the main steps to solve problem (30). Let $\mathcal{F} \triangleq \{(29c), (29f), (29g), (32c), (35b) - (35k)\}$ be the feasible set of (35). Starting from a random point $\widetilde{\boldsymbol{x}} \in \mathcal{F}$, we solve (49) to obtain its optimal solution $\widetilde{\boldsymbol{x}}^*$. This solution is then used as an initial point in the next iteration. The algorithm terminates when an accuracy level of ε is reached. It can be confirmed that $\widetilde{h}_{d,k}(\boldsymbol{v})$ and $\widetilde{h}_{u,k}(\boldsymbol{u})$ satisfy the key properties of general inner approximation functions [34, Properties (i), (ii), and (iii)]. In the case when the feasible set of problem (49) satisfies Slater's constraint qualification condition, Algorithm 3 converges to a Karush-Kuhn-Tucker (KKT) solution of (35) when starting from a point $\widetilde{\boldsymbol{x}}^{(0)} \in \mathcal{F}$ [34, Theorem 1]. In the worse case when the feasible set of problem (49) does not satisfy Slater's constraint qualification condition, Algorithm 3 converges to a Fritz John (FJ) solution of (35) [32, Proposition 2]. By using the variable transformations (33) and (34), it can be seen that the KKT (respectively, FJ) solutions of (35) satisfy the KKT (respectively, FJ) conditions of (32) as well as of (30).

Algorithm 3 Solving the Short-Term Subproblem (30)

1: **Initialization**: Set $n = 1$ and choose a random point $\widetilde{\boldsymbol{x}}^{(0)} \in \mathcal{F}$.
2: **repeat**
3: Update $n = n + 1$
4: Solving (49) to get its optimal solution $\widetilde{\boldsymbol{x}}^*$
5: Update $\widetilde{\boldsymbol{x}}^{(n)} = \widetilde{\boldsymbol{x}}^*$
6: **until** convergence

Output: $(\boldsymbol{\eta}^*, \boldsymbol{\zeta}^*, \boldsymbol{f}^*, \boldsymbol{R}_d^*, \boldsymbol{R}_u^*)$

B. Solving the Long-term Master Problem (31)

At the large-scale coherence time $n+1$, we replace the cost function of the stochastic nonconvex problem (31) by a sample surrogate function as [22]

$$\widetilde{g}^{(n+1)}(\theta) = (1-\phi^{(n+1)})\widetilde{g}^{(n)}(\theta) + \phi^{(n+1)}\widetilde{T}(\theta), \quad (50)$$

where $\phi^{(n+1)}$ is a weighting parameter. $\widetilde{g}^{(n+1)}(\theta)$ depends on the surrogate function $\widetilde{g}^{(n)}(\theta)$ of the previous large-scale coherence time (n) and the approximate function $\widetilde{T}(\theta)$ of $T(\theta)$. Here, $\widetilde{g}^{(n)}(\theta)$ is approximately updated as

$$\widetilde{g}^{(n)}(\theta) = g^{(n)} + (\nabla g)^{(n)}(\theta - \theta^{(n+1)}), \quad (51)$$

and $\widetilde{T}(\theta)$ is expressed as

$$\widetilde{T}(\theta) = T^{(n+1)} + (\nabla T)^{(n+1)}(\theta - \theta^{(n+1)}) + \tau(\theta - \theta^{(n+1)})^2, \quad (52)$$

where $\tau > 0$ can be any constant.

With (51) and (52), (50) can be rewritten as:

$$\widetilde{g}^{(n+1)}(\theta) = g^{(n+1)} + (\nabla g)^{(n+1)}(\theta - \theta^{(n+1)}) + \tau(\theta - \theta^{(n+1)})^2, \quad (53)$$

where $g^{(n+1)}$ and $(\nabla g)^{(n+1)}$ are updated as

$$g^{(n+1)} = (1-\phi^{(n+1)})(g)^{(n)} + \phi^{(n+1)}T^{(n+1)} \quad (54)$$

$$(\nabla g)^{(n+1)} = (1-\phi^{(n+1)})(\nabla g)^{(n)} + \phi^{(n+1)}(\nabla T)^{(n+1)}, \quad (55)$$

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

$$(\nabla T)^{(n+1)} = \frac{a + b\log(1/\theta^{(n+1)}) - b(1/\theta^{(n+1)} - 1)}{(1-\theta^{(n+1)})^2}, \quad (56)$$

where $a = t_{d,B}^{(n+1)} + t_{d,W}^{(n+1)} + t_{u,W}^{(n+1)} + t_{u,B}^{(n+1)}$ and $b = \nu \max_k \Big(\frac{D_k c_k}{f_k}\Big)$. Since $\widetilde{g}^{(n+1)}(\theta)$ in (53) approximates $g(\theta)$ in (31), problem (31) is finally approximated by the following convex problem:

$$\min_\theta \{g^{(n+1)} + (\nabla g)^{(n+1)}(\theta - \theta^{(n+1)}) + \tau(\theta - \theta^{(n+1)})^2\} \quad \text{s.t. (29h).} \quad (57)$$

C. Solving the Overall Problem (29)

Algorithm 4 outlines the main steps to solve the overall problem (29). In the large-scale coherence time n, a random large-scale fading coefficient $\boldsymbol{\beta}$ is realized. For a given random value of $\theta^{(n+1)} \in (0,1)$, one short-term subproblem (30) is solved by Algorithm 3 after $I^{(n)}$ iterations to obtain a KKT solution. This solution is then used to construct the approximate long-term master problem (57). After solving (57) to obtain an optimal solution θ^*, we update $\theta^{(n+2)}$ as

$$\theta^{(n+2)} = (1-\pi^{(n+1)})\theta^{(n+1)} + \pi^{(n+1)}\theta^*, \tag{58}$$

where $\pi^{(n+1)}$ is a weighting parameter. Here, $\{\phi^{(n)}, \pi^{(n)}\}$ is chosen to satisfy the following conditions [22, Assumption 5].

(C1): $\phi^{(n)} \to 0$, $\frac{1}{\phi^{(n)}} \leq \mathcal{O}(n^{\varsigma})$ for $\varsigma \in (0,1)$, and $\sum_n (\phi^{(n)})^2 < \infty$;

(C2): $\pi^{(n)} \to 0$, $\sum_n \pi^{(n)} = \infty$, $\sum_n (\pi^{(n)})^2 < \infty$, and $\lim_{n\to\infty} \frac{\pi^{(n)}}{\phi^{(n)}} = 0$.

D. The Proposed Algorithm: Implementation and Convergence Analysis

Referring to Fig. 2, Algorithm 4 is executed in the "FL performance optimization" interval. Specifically, Steps 3 and 4 takes place in the time block of STO, while steps $5-8$ in the time block of LTO. Once Algorithm 4 converges, the FL process is then executed using the value of θ given by Algorithm 4. Here, the performance of training update transmission in each iteration of the FL process is enhanced by updating $(\boldsymbol{\eta}, \boldsymbol{\zeta}, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)$ using Algorithm 3 in the STO time block. Whenever the statistics of large-scale fading changes, Algorithm 4 is executed again to make sure the FL performance is optimized with the updated statistics.

The convergence of Algorithm 4 is proved as follows. From the definitions of $\widetilde{z}_k(\varrho_k, R_{u,k})$, $\widetilde{h}_{d,k}(\boldsymbol{v})$, and $\widetilde{h}_{u,k}(\boldsymbol{u})$ in (39), (41) and (44), it can be verified that $\widetilde{z}_k(\varrho_k, R_{u,k})$, $\widetilde{h}_{d,k}(\boldsymbol{v})$ and $\widetilde{h}_{u,k}(\boldsymbol{u})$ have the following properties:

- $\widetilde{z}_k(\varrho_k^{(n)}, R_{u,k}^{(n)}) = z_k(\varrho_k^{(n)}, R_{u,k}^{(n)})$, $\widetilde{h}_{d,k}(\boldsymbol{v}^{(n)}) = h_{d,k}(\boldsymbol{v}^{(n)})$, $\widetilde{h}_{u,k}(\boldsymbol{u}^{(n)}) = h_{u,k}(\boldsymbol{u}^{(n)})$, $\nabla\widetilde{z}_k(\varrho_k^{(n)}, R_{u,k}^{(n)}) = \nabla z_k(\varrho_k^{(n)}, R_{u,k}^{(n)})$, $\nabla\widetilde{h}_{d,k}(\boldsymbol{v}^{(n)}) = \nabla h_{d,k}(\boldsymbol{v}^{(n)})$, $\nabla\widetilde{h}_{u,k}(\boldsymbol{u}^{(n)}) = \nabla h_{u,k}(\boldsymbol{u}^{(n)})$;
- $\widetilde{z}_k(\varrho_k, R_{u,k})$, $-\widetilde{h}_{d,k}(\boldsymbol{v})$, and $-\widetilde{h}_{u,k}(\boldsymbol{u})$ are strongly convex;
- $\widetilde{z}_k(\varrho_k, \varrho_k^{(n)}, R_{u,k}, R_{u,k}^{(n)})$ is Lipschitz continuous in all $\varrho_k, \varrho_k^{(n)}, R_{u,k}, R_{u,k}^{(n)}$; $\widetilde{h}_{d,k}(\boldsymbol{v}, \boldsymbol{v}^{(n)})$ and $\widetilde{h}_{u,k}(\boldsymbol{u}, \boldsymbol{u}^{(n)})$ are Lipschitz continuous in both $\boldsymbol{v}, \boldsymbol{v}^{(n)}$ and both $\boldsymbol{u}, \boldsymbol{u}^{(n)}$, respectively.

Algorithm 4 thus satisfies all the conditions for the short-term algorithm to work, as specified in the general framework [22, Assumption 2]. As such, the convergence of Algorithm 4 to a stationary point of problem (29) is guaranteed if $I^{(n)} \to \infty$ and $N \to \infty$ [22, Theorem 2], where the FJ condition may replace the KKT condition in the definition of the stationary point [22, Definition 1]. In practice, since there are always numerical errors in computation, it is acceptable to choose finite $\{I^{(n)}\}_{n\in\mathcal{N}}$ and N, where $\mathcal{N} \triangleq \{1,\ldots,N\}$. Therefore, Algorithm 4 is then guaranteed to converge to the neighbourhood of the stationary solutions of problem (29) [22, Theorem 3].

Algorithm 4 Training Time Minimization for FL on CFmMIMO Networks

1: **Initialization**: Set $n = 0$ and choose a random point $\theta^{(n+1)} \in (0,1)$.
2: **repeat**
3: A random $\boldsymbol{\beta}$ is realized for one large-scale coherence time
4: Find the optimal solution $(\boldsymbol{\eta}^*, \boldsymbol{\zeta}^*, \boldsymbol{f}^*, \boldsymbol{R}_d^*, \boldsymbol{R}_u^*)$ of the short-term subproblem (30) by using Algorithm 3
5: Update $(\boldsymbol{\eta}^{(n+1)}, \boldsymbol{\zeta}^{(n+1)}, \boldsymbol{f}^{(n+1)}, \boldsymbol{R}_d^{(n+1)}, \boldsymbol{R}_u^{(n+1)}) = (\boldsymbol{\eta}^*, \boldsymbol{\zeta}^*, \boldsymbol{f}^*, \boldsymbol{R}_d^*, \boldsymbol{R}_u^*)$
6: Solve the approximate long-term master problem (57) to obtain its optimal solution θ^*
7: Update $\theta^{(n+2)}$ by (58)
8: Update $n = n+1$
9: **until** convergence

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

The REPEAT-UNTIL loop runs for N iterations before Algorithm 4 converges.

VII. Cell-Free TDMA Massive MIMO and Collocated Massive MIMO for Wireless Federated Learning

For comparison, this section introduces cell-free TDMA massive MIMO and collocated massive MIMO approaches that support wireless FL. Their associated problem formulations and solution algorithms are discussed in the following.

A. Cell-Free TDMA Massive MIMO

The channel estimation of cell-free TDMA massive MIMO networks is equivalent to that of the CFmMIMO networks where all the pilot are pairwisely orthogonal, i.e., $\boldsymbol{\varphi}_\ell^H \boldsymbol{\varphi}_k = 0, \forall \ell \in \mathcal{K}\setminus k$. While cell-free TDMA massive MIMO networks only require the length of the pilot sequence $\widetilde{\tau}_t$ to be 1, CFmMIMO networks require $\tau_t \geq K$ for orthogonal pilots with K being the number of UEs.

In cell-free TDMA massive MIMO networks, the training update transmissions between the APs and K UEs happen in K equal orthogonal time slots. Therefore, a factor of $(1/K)$ is imposed on the achievable DL and UL rates. Specifically, the achievable DL rate for a UE k is

$$R_{d,k} \leq \frac{\tau_c - \widetilde{\tau}_t}{K\tau_c} B \log_2\Big(1 + \frac{\rho_p \big(\sum_{m\in\mathcal{M}} \eta_{mk}^{1/2} \sigma_{mk}^2\big)^2}{\rho_p \sum_{m\in\mathcal{M}} \eta_{mk} \sigma_{mk}^2 \beta_{mk} + 1}\Big), \tag{59}$$

where $\sigma_{mk}^2 = \frac{\widetilde{\tau}_t \widetilde{\rho}_t (\beta_{mk})^2}{\widetilde{\tau}_t \widetilde{\rho}_t \beta_{mk} + 1}$, and $\widetilde{\rho}_t$ is the normalized signal-to-noise ratio (SNR) of each pilot symbol. The achievable UL rate $R_{u,k}$ for a UE k is

$$R_{u,k} \leq \frac{\tau_c - \widetilde{\tau}_t}{K\tau_c} B \times \log_2\Big(1 + \frac{\rho_u \zeta_k \big(\sum_{m\in\mathcal{M}} \sigma_{mk}^2\big)^2}{\rho_u \zeta_k \sum_{m\in\mathcal{M}} \sigma_{mk}^2 \beta_{mk} + \sum_{m\in\mathcal{M}} \sigma_{mk}^2}\Big). \tag{60}$$

Since the training updates are transmitted sequentially via wireless links, the effective training time of FL in cell-free TDMA massive MIMO networks is expressed as

$$\begin{aligned} &T_{\text{e,TDMA}}(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u) \\ &\quad \triangleq G(\theta)\, \mathbb{E}\{T_{\text{G,TDMA}}(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)\} \\ &\quad \triangleq \vartheta \log\left(\tfrac{1}{\epsilon}\right) \mathbb{E}\left\{ \tfrac{1}{1-\theta}\Big(t_{d,B}(\boldsymbol{R}_d) + \sum_{k\in\mathcal{K}} t_{d,k}(R_{d,k}) \right. \\ &\qquad \left. + \max_{k\in\mathcal{K}} t_{C,k}(\theta, f_k) + \sum_{k\in\mathcal{K}} t_{u,k}(R_{u,k}) + t_{u,B}(\boldsymbol{R}_u)\Big)\right\} \\ &\quad \triangleq \vartheta \log\left(\tfrac{1}{\epsilon}\right) \mathbb{E}\{T_{\text{TDMA}}(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)\}. \end{aligned} \tag{61}$$

The problem of FL training time minimization for cell-free TDMA massive MIMO is formulated as:

$$\min_{\boldsymbol{\eta},\boldsymbol{\zeta},\theta,\boldsymbol{f},\boldsymbol{R}_d,\boldsymbol{R}_u} \quad \mathbb{E}\{T_{\text{TDMA}}(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)\} \tag{62a}$$

$$\text{s.t. } (15), (29\text{b}) - (29\text{h}), \ (59), \ (60)$$

$$\sigma_{mk}^2 \eta_{mk} \le 1, \quad \forall m. \tag{62b}$$

Since problem (62) has the same mathematical structure as (29), the former can be solved by a slightly modified version of Algorithm 4 proposed in Section VI.

B. Collocated Massive MIMO

A collocated massive MIMO network is a special case of a CFmMIMO network where all the APs are collocated. Therefore, $\beta_{mk} = \beta_k$ and $\sigma_{mk}^2 = \sigma_k^2, \forall k \in \mathcal{K}$. The DL power control coefficient $\eta_k, \forall k \in \mathcal{K}$, is constrained by

$$\sum_{k\in\mathcal{K}} \sigma_k^2 \frac{\eta_k}{M} \le 1. \tag{63}$$

From (12) and (20), the achievable DL and UL rates for UE k are respectively designed as (64) and (65), shown at the bottom of the next page.

The problem of FL training time minimization for collocated massive MIMO is formulated as:

$$\min_{\boldsymbol{\eta},\boldsymbol{\zeta},\theta,\boldsymbol{f},\boldsymbol{R}_d,\boldsymbol{R}_u} \quad \mathbb{E}\{T(\theta, \boldsymbol{f}, \boldsymbol{R}_d, \boldsymbol{R}_u)\}$$

$$\text{s.t. } (15), (29\text{b}) - (29\text{h}), (63) - (65). \tag{66}$$

Similar to (62), problem (66) can be solved by a slightly modified version of Algorithm 4 in Section VI.

VIII. Numerical Examples

A. Parameters and Setup

We consider a CFmMIMO network with $\tau_c = 200$ samples. The APs and UEs are located in a square of $D \times D$ km^2 whose edges are wrapped around to avoid the boundary effects. The large-scale fading coefficients, e.g., β_{mk}, are modeled in the same manner as [35]: $\beta_{mk} = 10^{\frac{\text{PL}_{mk}^d}{10}} 10^{\frac{\sigma_{shd} z_{mk}^d}{10}}$, where $10^{\frac{\sigma_{shd} z_{mk}^d}{10}}$ represents the log-normal shadowing with the standard deviation σ_{shd} (in dB); and $10^{\frac{\text{PL}_{mk}^d}{10}}$ represents the three-slope path loss. PL_{mk}^d (in dB) is given by

$$\text{PL}_{mk}^d = \begin{cases} -L - 35\log_{10}(d_{mk}), & \text{if } d_{mk} > d_1, \\ -L - 15\log_{10}(d_1) - 20\log_{10}(d_{mk}), & \\ \qquad\qquad \text{if } d_0 < d_{mk} \le d_1, & \\ -L - 15\log_{10}(d_1) - 20\log_{10}(d_0), & \\ \qquad\qquad \text{if } d_{mk} \le d_0, & \end{cases} \tag{67}$$

where L is a constant depending on the carrier frequency, the UE and AP heights. To estimate channels, a scheme of random pilot is used in the time block of UL channel estimation. Specifically, the pilot of each user is randomly chosen from a predefined set of τ_t orthogonal pilot sequences of length τ_t samples.

Here, we choose $\sigma_{shd} = 8$ dB, $d_0 = 10$ m, $d_1 = 50$ m, $L = 140.7$ dB, bandwidth $B = 20$ MHz, noise figure $F = 9$ dB [19], $f_{k,\max} = f_{\max} = 3.0 \times 10^9$ cycles/s, $f_{k,\min} = f_{\min} = 1 \times 10^6$ cycles/s, $D_k = \widehat{D} = 10$ MB, $c_k = c = 20$ cycles/sample, $\forall k$, $\nu = \vartheta = 1$, $S_d = S_u = 5$ MB, $\alpha = 2 \times 10^{-28}$ [10], $E_{k,\max} = E_{\max} = 15$ J, $\theta_{\max} = -10$ dB, $\theta_{\min} = -60$ dB, and $\epsilon = \varepsilon = 10^{-2}$. Noise power $N_0 = k_B T_0 B F = -92$ dBm, where $k_B = 1.381 \times 10^{-23}$ Joules/^{o}K is the Boltzmann constant and $T_0 = 290\,^o$K is the noise temperature. Let $\tilde{\rho}_d = 1$ W, $\tilde{\rho}_u = 0.2$ W and $\tilde{\rho}_t = 0.2$ W be the maximum transmit power of the APs, UEs and UL pilot sequences, respectively. The maximum transmit powers ρ_d, ρ_u and ρ_t are normalized by the noise power. We set $\pi^{(n)} = \frac{1}{n}$ and $\phi^{(n)} = \frac{1}{n^{7/8}}$ which satisfy conditions (C1) and (C2) in Section VI-C.

Remark 7: Our paper does not propose a new FL framework but rather a scheme for a CFmMIMO network to support any FL framework. Here, we consider an existing FL framework [21] as an example to show how to improve the FL performance in terms of training time minimization. Therefore, the simulations on real datasets to see the effectiveness of the considered FL framework has already been performed in [21, Section 6], and hence, they are not considered in this paper. Instead, in what follows, we focus on the numerical results to analyze the effectiveness of the proposed Algorithm 4 to solve the problem (29) of FL training time minimization for the FL framework [21].

B. Results and Discussions

1) Effectiveness of the Proposed Algorithm: First, we evaluate the convergence behavior of the proposed Algorithm 4. Fig. 4 shows the effective training time T_e versus the number of iterations with $D = 0.5$ km, $K = 4, \tau_t = 10$ and $M = \{30, 50\}$ for an arbitrary large-scale fading realization. It can be seen from Fig. 4 that Algorithm 4 converges in fewer than 100 iterations. It is also worth noting that each iteration of Algorithm 4 corresponds to solving simple convex programs (49) and (57). It is therefore expected that Algorithm 4 has a low computational complexity.

To further evaluate the effectiveness of Algorithm 4, we consider the following baseline schemes:

- Baseline 1 (BL1): The DL powers allocated to all UEs are the same, i.e., $\eta_{mk}\sigma_{mk}^2 = 1/K, \forall m, k$. The transmitted

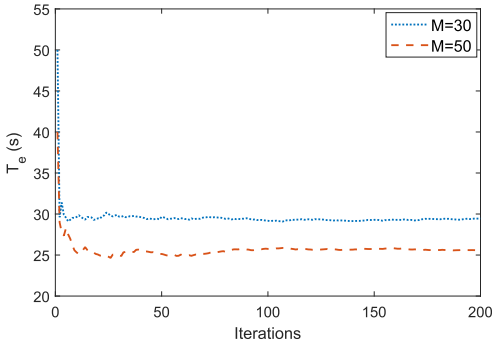

Fig. 4. The convergence of Algorithm 4. Here, $K = 4$.

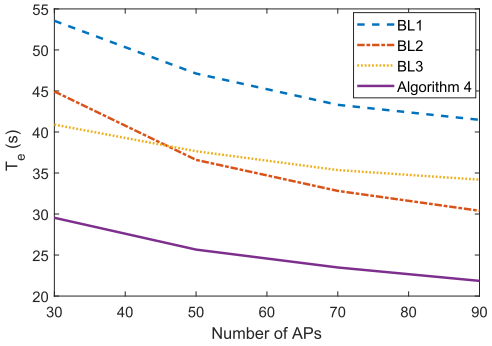

Fig. 5. Comparison among the baselines and Algorithm 4. Here, $K = 4$.

UL power of each UE is maximum, i.e., $\zeta_k = 1, \forall k$. The local accuracy is fixed, i.e., $\theta = \frac{\theta_{\max}+\theta_{\min}}{2}$ dB. The data rates and processing frequencies of UEs are then optimized.

- Baseline 2 (BL2): This baseline is similar to BL1 except that θ is optimized by a slightly modified version of Algorithm 4 (without using Algorithm 3).
- Baseline 3 (BL3): This baseline is similar to BL1 except that the transmitted DL and UL powers are optimized by Algorithm 3. Here, the effective training time of FL is the averaged time of one FL process taken over the large-scale fading realizations.

Figs. 5 and 6 compare the effective training time T_e by the considered schemes. As seen, Algorithm 4 gives the best performance. In particular, compared to BL1, the time reduction by Algorithm 4 is up to 55% with $M = 50$, $K = 8$. Note that BL2 and BL3 also perform much better than BL1, e.g., up to 29% in term of time reduction with $M = 50, K = 2$ and 38% with $M = 50, K = 8$, respectively. Even so, Algorithm 4 still provides substantial time reductions over BL2 and BL3, e.g., up to 49% with $M = 50, K = 8$ and 43% with $M = 90, K = 4$, respectively.

The figures not only show the importance of optimizing transmit power or local accuracy, but also demonstrate the noticeable advantage of joint optimization design. Moreover, thanks to the array gain, the data rates of UEs increase when the number of APs increases. This leads to the decrease in the effective training time as shown in Fig. 5. It can also be observed from Fig. 6 that a dramatic increase in the training time when the number of UEs increases. This is because the mutual interference and pilot contamination become stronger for a larger number of UEs.

2) Impact of key System Parameters on the Effective Training Time: The impact of the local accuracy on the effective training time is shown in Fig. 7. Decreasing the threshold $\theta_{\max}$

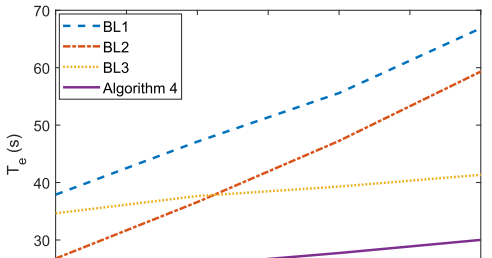

Fig. 6. Comparison among the baselines and Algorithm 4. Here, $M = 50$.

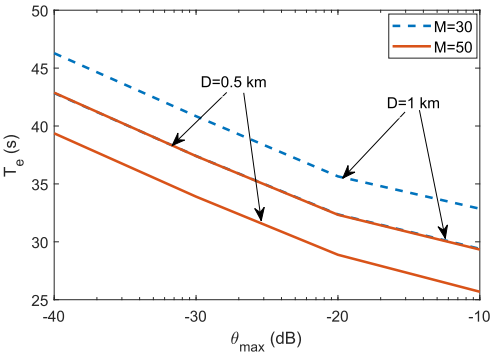

Fig. 7. Impact of the local accuracy on the effective training time. Here, $K = 4$.

leads to a dramatic increase in the effective training time, e.g., by up to 33% with $\theta_{\max} = -40$ dB in comparison to that with $\theta_{\max} = -10$. This is reasonable because at a lower value of θ, more iterations are required for local training. To keep the energy consumption of UEs below $E_{\max}$, the UEs' processing frequencies become smaller. This leads to an increase in the time required to compute the local training updates.

$$R_{d,k} \leq \frac{\tau_c - \tau_t}{\tau_c} B \log_2 \left(1 + \frac{\rho_p M \eta_k \sigma_k^4}{\rho_p \sum_{\ell \in \mathcal{K} \setminus k} M \eta_\ell \left(\sigma_\ell^2 \frac{\beta_k}{\beta_\ell}\right)^2 |\boldsymbol{\varphi}_\ell^H \boldsymbol{\varphi}_k|^2 + \rho_p \sum_{\ell \in \mathcal{K}} \eta_\ell \sigma_\ell^2 \beta_k + 1}\right) \tag{64}$$

$$R_{u,k} \leq \frac{\tau_c - \tau_t}{\tau_c} B \log_2 \left(1 + \frac{\rho_u M \zeta_k \sigma_k^2}{\rho_u \sum_{\ell \in \mathcal{K} \setminus k} \zeta_\ell M \sigma_k^2 \left(\frac{\beta_\ell}{\beta_k}\right)^2 |\boldsymbol{\varphi}_k^H \boldsymbol{\varphi}_\ell|^2 + \rho_u \sum_{\ell \in \mathcal{K}} \zeta_\ell \beta_\ell + 1}\right). \tag{65}$$

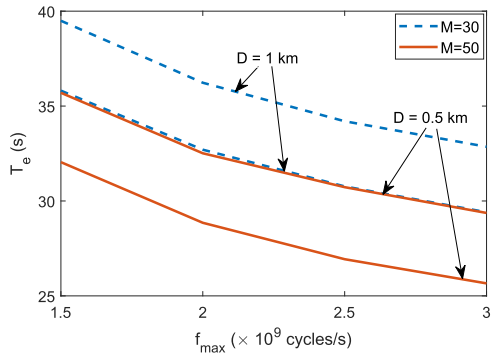

Fig. 8. Impact of UE's processing frequency on the effective training time. Here, $K = 4$.

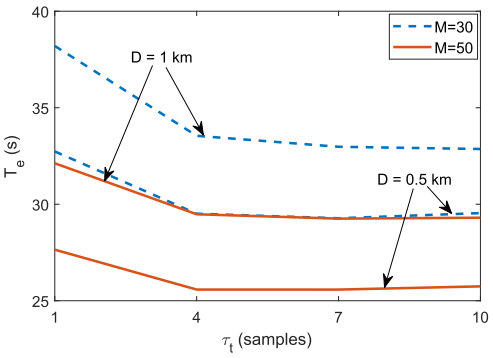

Fig. 10. Impact of the length of UL pilots on the effective training time. Here, $K = 4$.

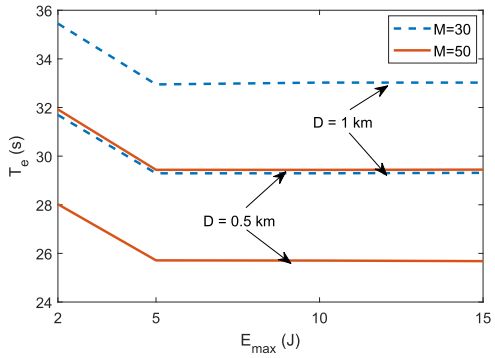

Fig. 9. Impact of UE's energy consumption limit $E_{\max}$ on the effective training time. Here, $K = 4$.

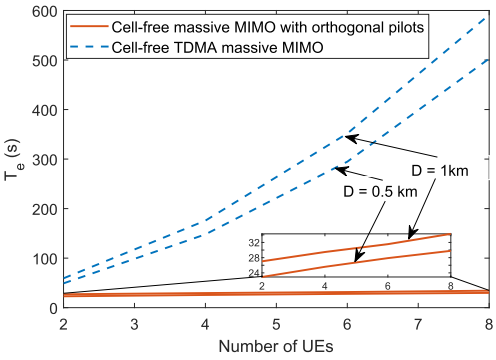

Fig. 11. Comparison between CFmMIMO with orthogonal pilots and cell-free TDMA massive MIMO. Here, $M = 50$.

Fig. 8 shows the impact of UE's processing frequency on the effective training time. As seen, the effective training time increases when the threshold $f_{\max}$ decreases. In particular, the increase is by up to 19% with $f_{\max} = 1.5 \times 10^9$ cycles/s in comparison to that with $f_{\max} = 3 \times 10^9$ cycles/s. This is because at a lower value of UEs' processing frequency, it requires more time to compute the local training updates.

In Fig. 9, the impact of UE's energy consumption limit $E_{\max}$ on the effective training time is revealed. Here, decreasing $E_{\max}$ leads to an increase in the effective time. Specifically, the increase is by up to 9.4% with $E_{\max} = 2$ J, $D = 1$ km in comparison to that with $E_{\max} = 15$ J, $D = 1$ km. This is reasonable because at a low value of $E_{\max}$, the effective time may not approach the optimal value due to a smaller feasible region of the optimization problem (29). We note that in the case of deep fading, the achievable rates of UEs could be small. This may lead to an infeasible problem because the constraint (29b) on the energy consumed at each UE may be violated. However, it should be also emphasized that a cell-free massive MIMO network has many antennas distributed over a potentially large coverage area. As very high small-scale and macro diversity gains can be achieved, the probability of simultaneously experiencing deep fading for all links would be very small. The problem is therefore likely feasible and our proposed solution would apply.

Fig. 10 focuses on the impact of the length of UL pilots on the effective training time. It is clear that too small and too large values of τ_t both increase the effective time. Specially, the effective time increases up to 10% and 1% with $\tau_t = 1$ and $\tau_t = 13$ in comparison to that with $\tau_t = 7$, respectively. This is reasonable because at a large value of τ_t, the factor of $\frac{T_c - \tau_t}{T_c}$ makes the data rate decrease and the transmission time grows. In contrast, at a small value of τ_t, the network suffers more from the pilot contamination, the data rates drop and the training update transmission time increases.

3) Cell-Free Massive MIMO vs. Cell-Free TDMA Massive MIMO: Fig. 11 compares the cell-free massive MIMO system with the cell-free TDMA massive MIMO system. Since the pilot sequences of UEs in the latter are pairwisely orthogonal in the time domain, we choose orthogonal pilot sequences for the former, i.e., $\boldsymbol{\varphi}_\ell^H \boldsymbol{\varphi}_k = 0, \forall \ell \in \mathcal{K} \setminus k$, for a fair comparison. The training durations are then the same for both systems. We also choose $\widetilde{\rho}_t = K\rho_t$ for the amount of energy consumed at the "UL channel estimation" time blocks of the two networks to be the same, and $\tau_t = K$ so that the powers of channel estimate, i.e., $\sigma_{mk}^2, \forall m, k$, are the same in the two networks. From Fig. 11, a significant time reduction (e.g., of up to 94% with $K = 8$) is achieved by the CFmMIMO compared with the cell-free TDMA massive MIMO. This result is expected because in the former, the factor of $(1/K)$ is imposed on the data rates and the training updates are transmitted sequentially. For a large number of UEs, the data rate is significantly small, and as a result, the training update transmission requires a substantially long time.

4) Cell-Free Massive MIMO vs. Collocated Massive MIMO: Finally, we compare the effective training time in CFmMIMO with that in collocated massive MIMO. Fig. 12 shows that

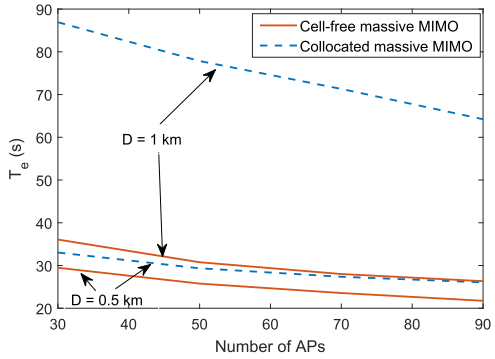

Fig. 12. Comparison between CFmMIMO and collocated massive MIMO. Here, $K = 4$.

the former significantly outperform the latter, e.g., the time reduction is by up to 57% with $M = 30$ and $D = 1$ km. This observation is as expected because CFmMIMO distributes antennas over their coverage area; and as such, their performance suffers less from the UEs with unfavorable links than that of collocated massive MIMO. Higher data rates and a lower training time then follow.

IX. Conclusion

In this paper, we have proposed using CFmMIMO networks to support FL in a wireless environment. We designed a general scheme in which any algorithm and beamforming/filtering approach can be further developed to optimize the performance of any FL framework. Specially here, each iteration of the FL optimization algorithms or the FL process happens in one large-scale coherence time. Targeting training time minimization for the FL framework [21] as example, we jointly design local accuracy, transmit power, data rate, and UE's processing frequency under the practical requirements on the UE's energy consumption limit and maximum transmit powers at the APs and UEs. A mixed timescale stochastic nonconvex optimization problem has been formulated with the objective of minimizing the training time of one FL process. Based on the general online successive convex approximation framework, we have developed a new algorithm to successfully solve the formulated problem. We have proved that the proposed algorithm converges to the neighborhood of stationary points of the optimization problem. For given parameter settings, numerical results show that our joint optimization design significantly reduces the training time of FL over the baselines under comparison. They have also confirmed that CFmMIMO offers the lowest training time when compared with cell-free TDMA massive MIMO and collocated massive MIMO.

References

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

[2] 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.

[3] J. Dong, M. Noreikis, Y. Xiao, and A. Yla-Jaaski, "ViNav: A vision-based indoor navigation system for smartphones," *IEEE Trans. Mobile Comput.*, vol. 18, no. 6, pp. 1461–1475, Jun. 2019.

[4] Qualcomm. (2017). *We are Making on-Device AI Ubiquitous*. [Online]. Available: https://www.qualcomm.com/news/onq/2017/08/16/we-are-making-device-ai-ubiquitous

[5] Gartner. (2018). *Gartner Highlights 10 Uses for AI-Powered Smartphones*. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2018-03-20-gartner-highlights-10-uses-for-ai-powered-smartphones

[6] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Towards an intelligent edge: Wireless communication meets machine learning," 2018, *arXiv:1809.00343*. [Online]. Available: http://arxiv.org/abs/1809.00343

[7] H. A. Alameddine, S. Sharafeddine, S. Sebbah, S. Ayoubi, and C. Assi, "Dynamic task offloading and scheduling for low-latency IoT services in multi-access edge computing," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 668–682, Mar. 2019.

[8] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, Mar. 2018.

[9] T. Li, A. Kumar Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," 2019, *arXiv:1908.07873*. [Online]. Available: http://arxiv.org/abs/1908.07873

[10] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE Conf. Comput. Commun.*, Apr. 2019, pp. 1387–1395.

[11] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, vol. 54, Fort Lauderdale, FL, USA, Apr. 2017, pp. 1273–1282.

[12] A. Hard *et al.*, "Federated learning for mobile keyboard prediction," 2018, *arXiv:1811.03604*. [Online]. Available: http://arxiv.org/abs/1811.03604

[13] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-IID data," 2019, *arXiv:1903.02891*. [Online]. Available: http://arxiv.org/abs/1903.02891

[14] J. Kang, Z. Xiong, D. Niyato, H. Yu, Y.-C. Liang, and D. In Kim, "Incentive design for efficient federated learning in mobile networks: A contract theory approach," 2019, *arXiv:1905.07479*. [Online]. Available: http://arxiv.org/abs/1905.07479

[15] S. Wang *et al.*, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.

[16] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via Over-the-Air computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2022–2035, Mar. 2020.

[17] M. Chen, Z. Yang, W. Saad, C. Yin, H. Vincent Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," 2019, *arXiv:1909.07972*. [Online]. Available: http://arxiv.org/abs/1909.07972

[18] J. Zhang, S. Chen, Y. Lin, J. Zheng, B. Ai, and L. Hanzo, "Cell-free massive MIMO: A new next-generation paradigm," *IEEE Access*, vol. 7, pp. 99878–99888, 2019.

[19] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson, and T. L. Marzetta, "Cell-free massive MIMO versus small cells," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1834–1850, Mar. 2017.

[20] T. L. Marzetta, E. G. Larsson, H. Yang, and H. Q. Ngo, *Fundamentals Massive MIMO*. Cambridge, U.K.: Cambridge Univ. Press, 2016.

[21] C. Ma *et al.*, "Distributed optimization with arbitrary local solvers," *Optim. Methods Softw.*, vol. 32, no. 4, pp. 813–848, Jul. 2017.

[22] A. Liu, V. K. N. Lau, and M.-J. Zhao, "Online successive convex approximation for two-stage stochastic nonconvex optimization," *IEEE Trans. Signal Process.*, vol. 66, no. 22, pp. 5941–5955, Nov. 2018.

[23] J.-H. Ahn, O. Simeone, and J. Kang, "Wireless federated distillation for distributed edge learning with heterogeneous data," 2019, *arXiv:1907.02745*. [Online]. Available: http://arxiv.org/abs/1907.02745

[24] M. Mohri, G. Sivek, and A. Theertha Suresh, "Agnostic federated learning," 2019, *arXiv:1902.00146*. [Online]. Available: http://arxiv.org/abs/1902.00146

[25] H. Zhu and Y. Jin, "Multi-objective evolutionary federated learning," 2018, *arXiv:1812.07478*. [Online]. Available: http://arxiv.org/abs/1812.07478

[26] J. Koneäný, Z. Qu, and P. Richtárik, "Semi-stochastic coordinate descent," *Optim. Methods Softw.*, vol. 32, no. 5, pp. 993–1005, Sep. 2017.
[27] M. Jaggi *et al.*, "Communication-efficient distributed dual coordinate ascent," in *Proc. 27th Int. Conf. Neural Inform. Process. Syst. (NIPS)*, vol. 2, 2014, pp. 3068–3076.
[28] E. Björnson, J. Hoydis, and L. Sanguinetti, "Massive MIMO networks: Spectral, energy, and hardware efficiency," *Found. Trends Signal Process.*, vol. 11, nos. 3–4, pp. 154–655, 2017.
[29] S. M. Kay, *Fundamentals Stat. Signal Processing: Estimation Theory*. Upper Saddle River, NJ, USA: Prentice-Hall, 1993.
[30] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proc. 2nd USENIX Conf. HotCloud*, 2010, p. 19.
[31] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge Univ. Press, 2004.
[32] T. T. Vu, D. T. Ngo, M. N. Dao, S. Durrani, D. H. N. Nguyen, and R. H. Middleton, "Energy efficiency maximization for downlink cloud radio access networks with data sharing and data compression," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 4955–4970, Aug. 2018.
[33] 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.
[34] B. R. Marks and G. P. Wright, "Technical note—A general inner approximation algorithm for nonconvex mathematical programs," *Operations Res.*, vol. 26, no. 4, pp. 681–683, Aug. 1978.
[35] H. Q. Ngo, L.-N. Tran, T. Q. Duong, M. Matthaiou, and E. G. Larsson, "On the total energy efficiency of cell-free massive MIMO," *IEEE Trans. Green Commun. Netw.*, vol. 2, no. 1, pp. 25–39, Mar. 2018.

Tung Thanh Vu (Graduate Student Member, IEEE) received the B.Sc. degree (Hons.) in telecommunications and networking and the M.Sc. degree (Hons.) in telecommunications engineering from the Ho Chi Minh City University of Science in 2012 and 2016, respectively. He is currently pursuing the Ph.D. degree with the School of Electrical Engineering and Computing, The University of Newcastle, Australia. His current research interests include optimization designs and machine learning for cell-free massive MIMO and cloud radio access networks.

Duy Trong Ngo (Member, IEEE) received the B.Eng. degree (Hons.) in telecommunication engineering from The University of New South Wales, Australia, in 2007, the M.Sc. degree in electrical engineering (communication) from the University of Alberta, Canada, in 2009, and the Ph.D. degree in electrical engineering from McGill University, Canada, in 2013.

He is a Senior Lecturer with the School of Electrical Engineering and Computing, The University of Newcastle, Australia, where he is currently involved in the research effort of design and optimization for 5G and beyond wireless communications networks. His current research interests include cloud radio access networks, multiaccess edge computing, and vehicle-to-everything (V2X) communications for intelligent transportation systems.

Nguyen H. Tran (Senior Member, IEEE) received the B.S. degree in electrical and computer engineering from the HCMC University of Technology in 2005 and the Ph.D. degree in electrical and computer engineering from Kyung Hee University in 2011. He was an Assistant Professor with the Department of Computer Science and Engineering, Kyung Hee University, from 2012 to 2017. Since 2018, he has been with the School of Computer Science, The University of Sydney, where he is currently a Senior Lecturer. His research interests include distributed computing, machine learning, and networking. He received the Best KHU Thesis Award in engineering in 2011 and several best paper awards, including at IEEE ICC 2016, APNOMS 2016, IEEE ICCS 2016, and ACM MSWiM 2019. He is currently receiving the Korea NRF Funding for Basic Science and Research 2016–2023 and ARC Discovery Project 2020–2023. He has been an Editor of the IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING since 2016.

Hien Quoc Ngo (Member, IEEE) is currently a Lecturer with Queen's University Belfast, U.K. His main research interests include massive (large-scale) MIMO systems, cellfree massive MIMO, physical layer security, and cooperative communications. He has coauthored many research articles in wireless communications and coauthored the textbook *Fundamentals of Massive MIMO* (Cambridge University Press, 2016). He received the IEEE ComSoc Stephen O. Rice Prize in communications theory in 2015, the IEEE ComSoc Leonard G. Abraham Prize in 2017, and the Best Ph.D. Award from EURASIP in 2018.

Minh Ngoc Dao received the Ph.D. degree in applied mathematics from the University of Toulouse, France, in 2014. He was a Post-Doctoral Fellow with The University of British Columbia, Canada, from 2014 to 2016, and a Research Associate with The University of Newcastle, Australia, from 2016 to 2019. He is currently a Research Associate with The University of New South Wales, Australia. His research interests include nonlinear optimization, nonsmooth analysis, control theory, signal processing, and machine learning. In 2017, he received the Annual Best Paper Award from the *Journal of Global Optimization*.

Richard H. Middleton (Fellow, IEEE) received the Ph.D. degree from the University of Newcastle, Australia, in 1987. He was a Research Professor with the Hamilton Institute, The National University of Ireland, Maynooth, from May 2007 to 2011 and is currently a Professor with the University of Newcastle and the Head of the School of Electrical Engineering and Computing. He has served as the Program Chair (CDC 2006), the Co-General Chair (CDC 2017) CSS Vice President Membership Activities, and Vice President Conference Activities. In 2011, he was the President of the IEEE Control Systems Society. His research interests include a broad range of control systems theory and applications, including communications systems, control of distributed systems, and systems biology. He is a fellow of IFAC.