Semi-Synchronous Personalized Federated Learning over Mobile Edge Networks

Chaoqun You, *Member, IEEE,* Daquan Feng, *Member, IEEE,* Kun Guo, *Member, IEEE,* Howard H. Yang *Member, IEEE,* Chenyuan Feng, and Tony Q. S. Quek, *Fellow, IEEE*

***Abstract*—Personalized Federated Learning (PFL) is a new Federated Learning (FL) approach to address the heterogeneity issue of the datasets generated by distributed user equipments (UEs). However, most existing PFL implementations rely on synchronous training to ensure good convergence performances, which may lead to a serious straggler problem, where the training time is heavily prolonged by the slowest UE. To address this issue, we propose a semi-synchronous PFL algorithm, termed as Semi-Synchronous Personalized FederatedAveraging (PerFedS^2), over mobile edge networks. By jointly optimizing the wireless bandwidth allocation and UE scheduling policy, it not only mitigates the straggler problem but also provides convergent training loss guarantees. We derive an upper bound of the convergence rate of PerFedS^2 in terms of the number of participants per global round and the number of rounds. On this basis, the bandwidth allocation problem can be solved using analytical solutions and the UE scheduling policy can be obtained by a greedy algorithm. Experimental results verify the effectiveness of PerFedS^2 in saving the training time as well as guaranteeing the convergence of training loss, in contrast to synchronous and asynchronous PFL algorithms.**

***Index Terms*—Semi-synchronous implementation, personalized federated learning, mobile edge networks**

I. Introduction

FEDERATED Learning (FL) is a new distributed machine learning paradigm that enables model training across multiple user equipments (UEs) without uploading their raw data to a central parameter server [1]. Since its advent, FL has been widely adopted as a powerful tool to exploit the wealth of data available at the end-user devices [2, 3] and foster new applications such as Artificial Intelligence (AI) medical diagnosis [4] and autonomous vehicles [5]. Training a FL model contains three typical steps: (i) a set of UEs conduct local computing based on their own dataset, and upload the resultant parameters to the server, (ii) the server aggregates the UEs' parameters and improve the global model, and (iii) the server feeds back the new model to UEs for another round of local computing. This procedure repeats until the loss function starts to converge and a certain model accuracy is achieved.

With the substantial improvement in sensing capabilities and computational power of edge devices, UEs are producing abundant but diverse data [6]. The increasingly diverse datasets breed a demand for customized services on individual UEs. Typical examples of potential applications include Vehicle-to-everything (V2X) communications, where vehicles in the network may experience various road conditions and driving habits, making the local model disparate to the global model [7, 8]; and recommendation systems, where local servers have potentially heterogeneous customers and share non-independent and identically distributed (non-i.i.d.) item popularities, and thus requiring fine-grained recommendations [9, 10]. However, conventional FL algorithms are proposed to learn a *common* model which may have mediocre performance on certain UEs. And the situation is exacerbating as the ever-developing mobile UEs are generating increasingly diverse data. To address this issue, Personalized Federated Learning (PFL) [11, 12] has been proposed. Specifically, PFL provides an *initial* model that is good enough for the UEs to start with. Using this initial model, each UE can fastly adapt to its local dataset with one or more gradient descent steps using only a few data points. As a result, the UEs (especially with heterogeneous datasets) are able to enjoy fast personalized models by adapting the global model to local datasets.

Nonetheless, most PFL implementations adopt synchronous training to ensure good convergence performance [11, 13–16]. In the synchronous setting, the central server has to wait until the arrival of the parameters of the slowest UE before it can update the global model. As a consequence, synchronous training may cause severe *straggler* problem in PFL, where the deceleration of any UE can delay all other UEs. On the other hand, parameters of the UEs may arrive at the server at different speeds due to reasons such as various CPU processing capabilities and different wireless channel conditions. This difference begets another operation mechanism: asynchronous training. The key idea of asynchronous implementation is to allow all UEs work independently and the server updates the global model every time it receives an update from any UE [17–19]. Although this model updating strategy avoids the waiting time of UEs, the gradient staleness caused by asynchronous updating will further degrade the performance of

This paper was supported in part by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research & Development Programme, in part by MOE ARF Tier 2 under Grant T2EP20120−0006, in part by the National Science and Technology Major Project under Grant 2020YFB1807601, in part by the Shenzhen Science and Technology Program under Grants JCYJ20210324095209025, in part by Shanghai Pujiang Program under Grant No. 21PJ1402600, in part by the National Natural Science Foundation of China under Grant 62201504, in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LGJ22F010001. *(Corresponding author: Daquan Feng)*

C. You and T. Quek are with the Wireless Networks and Design Systems Group, Singapore University of Design and Technology, 487372, Singapore (e-mail: chaoqun_you, tonyquek@sutd.edu.sg).

D. Feng and C. Feng are with the Shenzhen University, Shenzhen 518052, China (e-mail:fdquan, fengchenyuan@szu.edu.cn)

K. Guo is with the East China Normal University, Shanghai 200241, China (e-mail: kguo@cee.ecnu.edu.cn).

H. H. Yang is with the Zhejiang University/University of Illinois at Urbana-Champaign Institute, Zhejiang University, Haining 314400, China (email: haoyang@intl.zju.edu.cn).

the model training. At this point, a semi-synchronous PFL has been a natural choice to balance the disadvantages caused by the synchronous as well as the asynchronous PFL algorithms.

Although there have been several works on semi-synchronous FL algorithms [20–23], the semi-synchronous PFL problem is not well understood. [20] studied the semi-asynchronous protocol for fast FL. [21] proposed a semi-asynchronous FL algorithm in heterogeneous edge computing. [22] introduced a novel energy-efficient semi-asynchronous FL protocol that mixes local models periodically with minimal idle time and fast convergence. At last, [23] proposed a clustered semi-asynchronous FL algorithm that groups UEs by the delay and direction of clients' model update to make the most of the advantage of both synchronous and asynchronous FL. Designing a semi-synchronous PFL in mobile edge networks, however, is particularly challenging due to the following reasons: (1) The convergence rate of a semi-synchronous PFL is *unclear*. Moreover, the loss function of a deep learning model is usually *non-convex*, and whether a semi-synchronous PFL can converge and under what conditions can the algorithm converge is of much interest. (2) The practical wireless communication environments need to be considered. It is *non-trivial* to decide the UE scheduling policy of a semi-synchronous PFL algorithm while considering the wireless bandwidth allocation.

In this paper, we propose a semi-synchronous PFL algorithm over mobile edge networks, named Semi-Synchronous Personalized FederatedAveraging (PerFedS^2) that mitigates the straggler problem in PFL. This is done by optimizing a joint bandwidth allocation and UE scheduling problem. To solve this problem, we first analyse the convergence rate of PerFedS^2 with non-convex loss functions. Our analysis characterizes the upper bound of the convergence rate in terms of two decision variables: the number of scheduled UEs in each communication round, and the number of communication rounds. Based on this upper bound, the joint bandwidth allocation and UE scheduling optimization problem can be solved separately. For the bandwidth allocation problem, we find that for a given UE scheduling policy, there exists infinitely many bandwidth solutions to minimize the overall training time. For the UE scheduling problem, facilitated by the results obtained from the convergence analysis, the optimal number of UEs that are scheduled to update the global model in each communication round and the optimal number of communication rounds can be estimated. These results lead us to designing a greedy algorithm that gives the UE scheduling policy. Finally, with the optimal bandwidth allocation and the UE scheduling policy, we are able to implement PerFedS^2 over mobile edge networks.

To summarize, in this paper we make the following contributions:

- We propose a new semi-synchronous PFL algorithm, i.e., the PerFedS^2, over mobile edge networks. The PerFedS^2 strikes a good balance between synchronous and asynchronous PFL algorithms. Particularly, by solving a joint bandwidth allocation and UE scheduling problem, it not only mitigates the straggler problem caused by the synchronous training but also abbreviates potential divergence issue in asynchronous training.
- We derive the convergence rate of the PerFedS^2. Our analysis characterizes the upper bound of convergence rate as a function with respect to the number of UEs that are scheduled to update the global model in each communication round and the number of communication rounds.
- We solve the optimization problem by decoupling it into two sub-problems: bandwidth allocation problem and UE scheduling problem. While the optimal bandwidth is proved to minimize the overall training time within a range of values, the UE scheduling policy can also be determined using a greedy online algorithm.
- We conduct extensive experiments by using MNIST, CIFAR-100 and Shakespeare datasets to demonstrate the effectiveness of PerFedS^2 in saving the overall training time as well as providing a convergent training loss, compared with four baselines, namely, the synchronous and asynchronous, FL and PFL algorithms, respectively.

The rest of the paper has been organized as follows. In Section II we introduce the basic learning process of PerFedS^2. Then in Section III we formulate a joint bandwidth allocation and UE scheduling problem to quantify and maximize the benefits PerFedS^2 could bring compared with synchronous and asynchronous training. In order to solve the optimization problem, we first analyse the convergence rate of PerFedS^2 in Section IV. Then, we solve the joint optimization problem in Section V. At last, we evaluate the performance of PerFedS^2 in Section VI.

II. Semi-Synchronous Personalized Federated Learning Mechanism

In this section, we propose PerFedS^2 to mitigate the drawbacks of synchronous and asynchronous PFL algorithms. For a better understanding of the proposed algorithm, we commence with reviewing FL and PFL in Section II-A and Section II-B, respectively. Then, we formally introduce PerFedS^2 in Section II-C.

A. Review: Federated Learning

Consider a set of n UEs connected to the server via a BS, where each UE has a local data $(x, y) \in \mathcal{X}_i \times \mathcal{Y}_i$. If we define $f_i : \mathbb{R}^m \to \mathbb{R}$ as the loss function corresponding to UE i, and w as the model parameter that the server needs to learn, then the goal of the server is to solve

$$\min_{w \in \mathbb{R}^m} f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w), \tag{1}$$

where f_i represents the expected loss over the data distribution of UE i, which is formalized as follows,

$$f_i(w) := \mathbb{E}_{(x,y)\sim\mathcal{H}_i}[l_i(w; x, y)], \tag{2}$$

where $l_i(w; x, y)$ measure the error of model w in predicting the true label y, and $\mathcal{H}_i$ is the distribution over $\mathcal{X}_i \times \mathcal{Y}_i$.

Because the dataset resided on different UEs are usually non-i.i.d. and unbalanced, while the global model trained by

FedAvg concentrates on the average performance of all the UEs. The resultant model may perform very poor on certain individual UEs. In response, PFL is proposed to capture the statistical heterogeneity among UEs by adapting the global model to local datasets. We review this scheme in the next subsection.

B. Review: Personalized Federated Learning

In contrast to the standard FL, PFL approaches the solution of (1) via the Model-Agnostic Meta-Learning (MAML). Specifically, the target of PFL is to learn an *initial* model that adapts quickly to each UE through one or more gradient steps with only a few data points on the UEs. Such an initial model is commonly known as the meta model, and the local model after adaptation is referred to as the fine-tuned model.

Formally, if each UE intakes the initial model and updates it via one step of gradient using its own loss function, problem (1) can be written as

$$\min_{w\in\mathbb{R}^m} F(w) := \frac{1}{n}\sum_{i=1}^{n} f_i(w - \alpha\nabla f_i(w)), \tag{3}$$

where $\alpha \geq 0$ is the learning rate at individual UEs. Note that we use the same learning rate for all UEs in this paper for simplification. This assumption can be easily extended to the general case when UEs have diverse learning rate α_i as long as $\alpha_i \geq 0$. For each UE i, its optimization objective F_i can be computed as

$$F_i(w) := f_i(w - \alpha\nabla f_i(w)). \tag{4}$$

Unlike conventional FL, after receiving the current global model, a UE in PFL first adapts the global model to its local data with one step of gradient descent, and then computes local gradients with respect to the model after the adaptation. This step of local adaptation captures the difference between UEs, and the model learned with this new formulation (3) is proved to be a good initial point for any UE to start with for fast adaptation [24, 25].

Many existing works on PFL is limited to the context of synchronous learning, where the faster UEs have to wait until all the others arrive the server to move to the next communication round [11, 13–16]. As a result, the synchronous PFL often suffers from the *straggler* problem due to the prolonged waiting time for the slowest UE. On the other hand, the PFL can also be trained in an asynchronous manner, where the server performs global updating as soon as it receives a local model from any UE. In this scenario, some slower UEs will bring stale gradient updates to the server, thereby degrading the convergence performance of the model training. Therefore, in this paper, we propose a semi-synchronous PFL mechanism that seeks a trade-off between synchronous and asynchronous PFL algorithms, which is detailed in the following subsection.

C. Semi-Synchronous Personalized Federated Learning

We propose a semi-synchronous PFL mechanism, which is a trade-off between synchronous and asynchronous PFL. We term this semi-synchronous PFL algorithm as Semi-Synchronous Personalized FederatedAveraging (PerFedS^2).

Algorithm 1: Semi-Synchronous Personalized Federated Averaging (PerFedS^2)

1 **for** $k = 0, 1, \ldots, K-1$ **do**
2 **Processing at Each UE** i
3 **if** *Receive* w_k *from the server* **then**
4 Compute local gradient $\tilde{\nabla} F_i(w_k)$ by Eq. (7)
  Upload $\tilde{\nabla} F_i(w_k)$ to the server
5 **end**
6 **Processing at the Parameter Server**
7 $\mathcal{A}_k = \varnothing$
8 **while** $|\mathcal{A}_k| < A$ **do**
9 Receive local gradient $\tilde{\nabla} F_i(w_k)$ from UE i
10 $\mathcal{A}_k = \mathcal{A}_k \cup \{i\}$
11 **end**
12 Update global model to w_{k+1} by Eq. (8)
13 **for** $i \in \mathcal{U}$ **do**
14 **if** $i \in \mathcal{A}_k$ *or* $\tau_k^i > S$ **then**
15 Distribute w_{k+1} to UE i
16 **end**
17 **end**
18 **end**

PerFedS^2 is formally described in Alg. 1. At the UE side (Line 2-5), upon receiving a global model, or equivalently, the meta model w_k, the UE adapts w_k to its local dataset to obtain the gradient of local functions, which in this case, the gradient ∇F_i, that is given by

$$\nabla F_i(w_k) = (I - \alpha\nabla^2 f_i(w_k))\nabla f_i(w_k - \alpha\nabla f_i(w_k)). \tag{5}$$

At the server side (Line 6-12), let $\mathcal{A}_k$ be the set of UEs participating in the global updating in round k, with the carnality being $|\mathcal{A}_k| = A$. Let τ_k^i be the interval between the current round k and the last received global model version by UE i. Such an interval reflects the *staleness* of local updates. With this notion, we can write the gradient received by the BS at round k from UE i as $\nabla F(w_{k-\tau_k^i})$. Upon receiving A local gradients, the server updates the global model parameter as follows:

$$w_{k+1} = w_k - \frac{\beta}{A}\sum_{i\in\mathcal{A}_k} \nabla F_i(w_{k-\tau_k^i}), \tag{6}$$

where $\beta > 0$ is the global step size. Then, the server distributes the new global model w_{k+1} to either (a) the UEs in $\mathcal{A}_k$ or (b) those with a staleness larger than the staleness threshold S.

Due to the vast volume of dataset, computing the exact gradient for each UE is costly. Therefore, we use the stochastic gradient descent (SGD) [26] as a proxy. Specifically, a generic UE i samples a subset of data points to calculate an unbiased estimate $\tilde{\nabla} f_i(w_k; \mathcal{D}_i)$ of $\nabla f_i(w_k)$, where $\mathcal{D}_i$ represents a portion of UE i's local dataset with size $|\mathcal{D}_i| = D_i$. Similarly, the Hessian ∇^2 in (5) can be replaced by its unbiased estimate $\tilde{\nabla}^2 f_i(w_k; \mathcal{D}_i)$. At this point, the actual gradient computed by UE i is the stochastic gradient of local loss function $\tilde{\nabla} F_i(w_k)$, which is given by

$$\tilde{\nabla} F_i(w_k) =$$

$$(I-\alpha\tilde{\nabla}^2 f_i(w_k;\mathcal{D}_i^{\mathrm{h}}))\tilde{\nabla} f_i(w_k-\alpha\tilde{\nabla} f_i(w_k;\mathcal{D}_i^{\mathrm{in}});\mathcal{D}_i^{\mathrm{o}}), \quad (7)$$

where $\mathcal{D}_i^{\mathrm{in}}$, $\mathcal{D}_i^{\mathrm{o}}$ and $;\mathcal{D}_i^{\mathrm{h}}$ are independently sampled datasets with total size denoted by $d_i = D_i^{\mathrm{in}} + D_i^{\mathrm{o}} + D_i^{\mathrm{h}}$. This stochastic gradient is then uploaded to the central server for global model update as follows:

$$w_{k+1} = w_k - \frac{\beta}{A}\sum_{i\in\mathcal{A}_k}\tilde{\nabla}F_i(w_{k-\tau_k^i}) \quad (8)$$

III. System Model and Problem Formulation

In the last section, we introduce the basic learning process of PerFedS^2. This alone is not enough to quantify the benefits a semi-synchronous training manner brings to implementation, because the communication related parameters and the training hyperparameters remain to be unclear. Therefore, our next step is to formulate an optimization problem for PerFedS^2, with the wireless bandwidth allocation and the UE scheduling policy to be determined. In this section, We introduce some notations and concepts in Section III-A and III-B that are used to formulate the optimization problem in Section III-C.

A. Communication Model

To implement PerFedS^2 in mobile edge networks, the wireless communication environments should also be considered to maximize the benefit a semi-asynchronous learning manner brings to the learning algorithm. Note that in PerFedS^2, one local iteration of UE i may last for a few global communication rounds, we focus on describing the wireless communication processes of UE i within such a local iteration. The learning time of UE i during one local iteration consists of two parts: communication time and computation time. As for the communication time over mobile edge networks, we consider that UEs access the BS through a channel partitioning scheme, such as orthogonal frequency division multiple access (OFDMA) [27], with total bandwidth B. Meanwhile, the bandwidth allocation to UE i in round k is denoted as b_k^i. The uplink rate of UE i transmitting its local gradients to the BS can be computed as follows [28, 29],

$$r_k^i = b_k^i \ln(1+\frac{p_i h_k^i \|c_i\|^{-\kappa}}{b_k^i N_0}), \quad (9)$$

where p_i is the transmit power of UE i, κ is the path loss exponent, and N_0 is the noise power spectral density. $h_k^i\|c_i\|^{-\kappa}$ is the channel gain between UE i and the BS at round k with c_i being the distance between UE i and the BS and h_k^i being the small-scale channel coefficient. In this paper, we assume that the small-scale channel coefficients across communication rounds h_k^i follow Rayleigh distribution [30]. With r_k^i, the uplink transmission delay of UE i can be specified as follows,

$$Tcom_k^i = \frac{Z_k^i}{r_k^i}, \quad (10)$$

where Z_k^i denotes the number of bits UE i transmits in round k. Meanwhile, Z denotes total size of the gradient UE i transmits each time. Since the transmit power of the BS is much higher than the UEs', the downlink transmission latency is much smaller than that in the uplink. Meanwhile, we care more about the transmit power allocation on individual UEs rather than that on the server, hence we ignore the downlink delay for simplicity.

As for the computation time, let c_i denote the number of CPU cycles for UE i to execute one sample of data, ϑ_i denote the CPU-cycle frequency of UE i, and d_i denote the number of sampled data points on UE i, then the computation time of UE i per local iteration can be expressed as follows [28],

$$Tcmp_k^i = \frac{c_i d_i}{\vartheta_i}. \quad (11)$$

As such, given that for semi-synchronous training, each local iteration of UE i may last several global rounds, the total time UE i spent in round k is given by

$$T_k^i = \begin{cases} Tcom_k^i + Tcmp_k^i, \\ \quad \text{when UE } i \text{ starts a new local iteration in round } k, \\ Tcom_k^i, \quad \text{otherwise.} \end{cases} \quad (12)$$

B. Illustrative Example

We give an example to facilitate the understanding of PerFedS^2. Consider the scenario depicted in Fig. 1, where $A = 2$. This network has four UEs. In the first communication round, UE 3 and 4 are stragglers. Therefore, once the stochastic gradients uploaded by UE 1 and 2 arrive at the server in round 1, the server updates the global model from w_0 to w_1, leaving the gradients computed by UE 3 and 4 to be integrated into the global model in round 2 and round 3, respectively.

Scheduling policy: Let $\pi_k^i \in \{0,1\}$ be an indicator to denote whether the gradient uploaded from UE i arrives at the server in round k. That is, $\pi_k^i = 1$ if the update from UE i is included in the global model in round k, and $\pi_k^i = 0$ otherwise. Then, $\mathbf{\Pi} \triangleq [\mathbf{\Pi}_1, \mathbf{\Pi}_2, \ldots, \mathbf{\Pi}_K]$ denotes the scheduling decision matrix up to round K, where $\mathbf{\Pi}_k \triangleq [\pi_k^1, \pi_k^2, \ldots, \pi_k^n]$. For the example given in Fig. 1, the computation has been carried out five rounds and the scheduling decision matrix $\mathbf{\Pi}$ can be written as

$$\mathbf{\Pi} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}. \quad (13)$$

From the above, we can see that the entries in each row of $\mathbf{\Pi}$ satisfy the following relationship

$$\sum_{i=1}^{n}\pi_k^i = A. \quad (14)$$

We further introduce a concept, coined as the relative participation frequency, to reflect the statistical property of the scheduling policy. Specifically, for UE i, we denote its relative participation frequency as η_i, which represents the fraction of time this UE participates in the global iteration. Such a notion

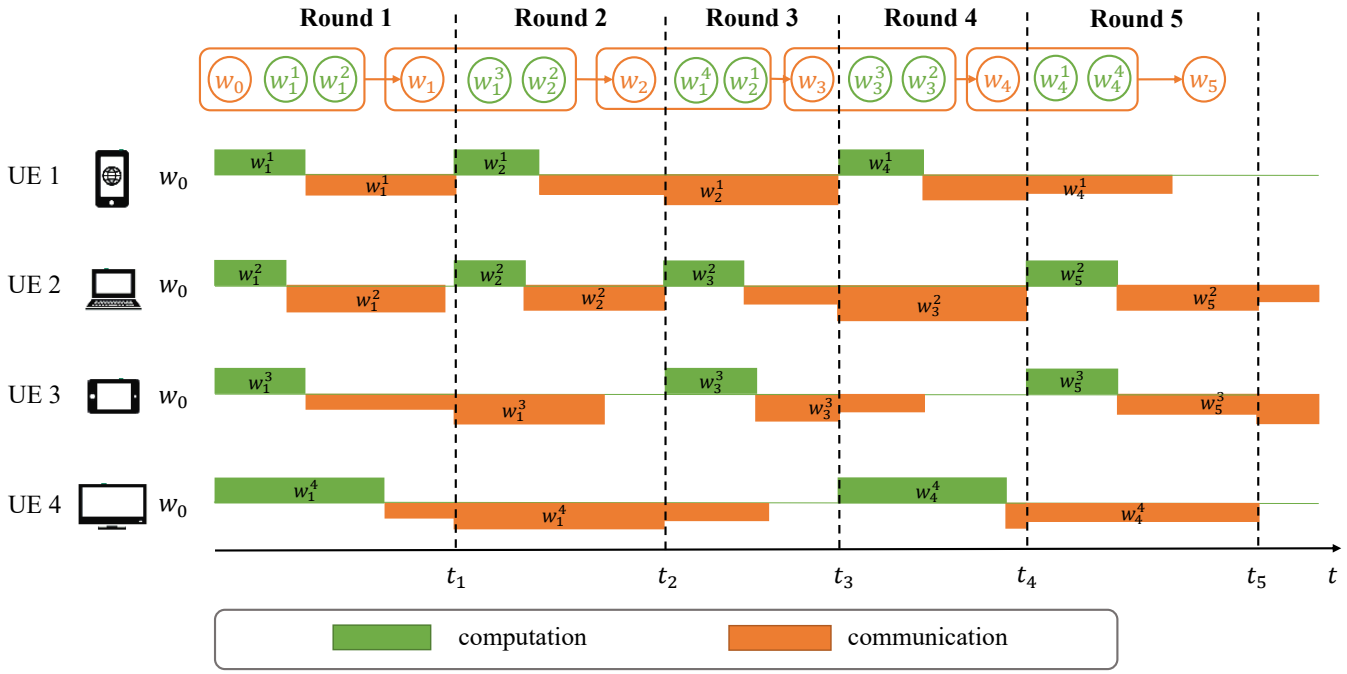


Fig. 1: Example of the PerFedS^2 mechanism when $A = 2$.

is formally defined as

$$\eta_i = \frac{\sum_{k=1}^{K} \pi_k^i}{\sum_{k=1}^{K}\sum_{i=1}^{n} \pi_k^i} = \frac{\sum_{k=1}^{K} \pi_k^i}{AK}. \tag{15}$$

Notably, the staleness bound S provides a lower bound of η_i, that is, $\eta_i \geq S/K$ ($\forall i \in \mathcal{U}$).

C. Problem Formulation

PerFedS^2 significantly increases the proportion of time UEs spend on computing, as opposed to waiting. Meanwhile, PerFedS^2 also upper bounds the staleness caused by updates from slow UEs. Let T be the overall training time over K communication rounds. Then the objective of PerFedS^2 is to minimize the loss function as well as the overall training time. Formally, the optimization problem of PerFedS^2 is formulated as follows [1],

$$\min_{\mathbf{b},\Pi,A,K} \quad F(w) \tag{P1}$$

$$\text{s.t.} \quad \min_{\mathbf{b}} \sum_{k=1}^{K} \max_{i\in\mathcal{A}_k}\{T_k^i\} = T, \quad \forall i \in \mathcal{U}, \tag{C1.1}$$

$$\sum_{i=1}^{n} b_k^i \leq B, \quad k = 1, 2, \ldots, K, \tag{C1.2}$$

$$\sum_{j=k-\tau_k^i}^{k-\tau_k^i+S} \pi_j^i \geq 1, \quad \forall i \in \mathcal{U} \tag{C1.3}$$

$$\sum_{j=k-\tau_k^i}^{k} Z_j^i \leq Z \tag{C1.4}$$

$$K \geq \frac{S}{\eta_i}, \quad \forall i \in \mathcal{U}, \tag{C1.5}$$

where $\mathbf{b} \triangleq [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_K]$ denotes the bandwidth allocation matrix up to round K, and $\mathbf{b}_k = [b_k^1, b_k^2, \ldots, b_k^n]$. (C1.1) is the overall training time constraint, that for each communication round k, the round time is determined by the maximum of T_k^i over $i \in \mathcal{A}_k$, and the total time up to round K is equal to T. (C1.2) is the bandwidth constraint, that the bandwidth allocation to all UEs in every communication round shall not exceed the available bandwidth B. (C1.3) stipulates the staleness constraint on the updates, that the during any S rounds of communication, UE i must be scheduled to update the global model at least once. (C1.4) limits the number of bit transmitted, note that Z_k^i is determined by b_k^i, and the number of bits that are transmitted during τ_k^i rounds shall not be larger than the size of model parameters. Finally, (C1.5) follows from the lower bound we drawn in the previous subsection.

[1]Besides bandwidth allocation and UE scheduling policy, other decision variables such like transmit power can also be included in the problem formulation. The logic keeps the same, but the parameters that need to be considered might change. Problem (P1) shows the case when we consider the bandwidth allocation and UE scheduling policy as variables, and it is free for the researcher to extend this general formulation to other forms.

IV. Convergence Analysis

In this section, we first introduce some definitions and assumptions on the loss functions of PerFedS^2 in Section IV-A. Then we analyse its convergence rate in Section IV-B.

A. Preliminaries

We consider the *non-convex* loss functions in this paper. Our goal is to find an ϵ-approximate first-order stationary point (FOSP) for PerFedS^2 [13, 25]. The formal definition of FOSP is given as follows.

Definition 1. A random vector $w_\epsilon \in \mathbb{R}^m$ is called an ϵ-FOSP for PerFedS^2 if it satisfies $\mathbb{E}[\|\nabla F(w_\epsilon)\|^2] \leq \epsilon$.

To make the convergence analysis consistent with that of Per-FedAvg, we make the following assumptions [13].

Assumption 1 (Bounded Staleness)**.** *All delay variables τ_k^i's are bounded, i.e., $\max_{k,i} \tau_k^i \leq S$.*

Assumption 2. *For each UE $i \in \mathcal{U}$, its gradient ∇f_i is L-Lipschitz continuous and is bounded by a nonnegative constant C, namely,*

$$\|\nabla f_i(w) - \nabla f_i(u)\| \leq L\|w-u\|, \qquad w,u \in \mathbb{R}^m \tag{17}$$

$$\|\nabla f_i(w)\| \leq C, \qquad w \in \mathbb{R}^m. \tag{18}$$

Assumption 3. *For each UE $i \in \mathcal{U}$, the Hessian of f_i is ρ-Lipschitz continuous:*

$$\|\nabla^2 f_i(w) - \nabla^2 f_i(u)\| \leq \rho\|w-u\|, \qquad w,u \in \mathbb{R}^m. \tag{19}$$

Assumption 4. *For any $w \in \mathbb{R}^m$, $\nabla l_i(w;x,y)$ and $\nabla^2 l_i(w;x,y)$, computed w.r.t. a single data point $(x,y) \in \mathcal{X}_i \times \mathcal{Y}_i$, have bounded variance:*

$$\mathbb{E}_{(x,y)\sim p_i}[\|\nabla l_i(w;x,y) - \nabla f_i(w)\|^2] \leq \sigma_G^2,$$
$$\mathbb{E}_{(x,y)\sim p_i}[\|\nabla^2 l_i(w;x,y) - \nabla^2 f_i(w)\|^2] \leq \sigma_H^2. \tag{20}$$

Assumption 5. *For any $w \in \mathbb{R}^m$, the gradient and Hessian of local loss function $f_i(w)$ and the average loss function $f(w) = 1/n \sum_{i=1}^n f_i(w)$ satisfy the following conditions:*

$$\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(w) - \nabla f(w)\|^2 \leq \gamma_G^2,$$
$$\frac{1}{n}\sum_{i=1}^n \|\nabla^2 f_i(w) - \nabla^2 f(w)\|^2 \leq \gamma_H^2. \tag{21}$$

While Assumption 1 limits the maximum of the staleness, Assumptions 2 to 5 characterize the properties of the gradient and Hessian of $f_i(w)$, which are necessary to deduce the following lemmas and convergence rate analysis.

B. Analysis of Convergence Bound

Before delving into the full details of convergence analysis, we introduce three lemmas inherited from [13] to quantify the smoothness of $F_i(w)$ and $F(w)$, the deviation between $\nabla F_i(w)$ and its estimate $\tilde{\nabla} F_i(w)$, and the deviation between $\nabla F_i(w)$ and $\nabla F(w)$, respectively.

Lemma 1. *If Assumptions 2-4 hold, then F_i is smooth with parameter $L_F := 4L + \alpha\rho C$. As a consequence, the average function $F(w) = 1/n \sum_{i=1}^n F_i(w)$ is also smooth with parameter L_F.*

Lemma 2. *If Assumptions 2-4 hold, then for any $\alpha_i \in (0, 1/L]$ and $w \in \mathbb{R}^m$, we have*

$$\left\|\mathbb{E}\left[\tilde{\nabla} F_i(w) - \nabla F_i(w)\right]\right\| \leq \frac{2\alpha L \sigma_G}{\sqrt{D^{in}}}, \tag{22}$$

$$\mathbb{E}\left[\|\tilde{\nabla} F_i(w) - \nabla F_i(w)\|^2\right] \leq \sigma_F^2. \tag{23}$$

where σ_F^2 is defined as

$$\sigma_F^2 := 12\left[C^2 + \sigma_G^2\left[\frac{1}{D^o} + \frac{(\alpha L)^2}{D^{in}}\right]\right]\left[1 + \sigma_H^2 \frac{\alpha^2}{4D^h}\right] - 12C^2, \tag{24}$$

where $D^{in} = \max_{i\in\mathcal{U}} D_i^{in}$, $D^o = \max_{i\in\mathcal{U}} D_i^o$ and $D^h = \max_{i\in\mathcal{U}} D_i^h$.

Lemma 3. *Given the loss function $F_i(w)$ shown in (4) and $\alpha \in (0, 1/L]$, if the conditions in Assumptions 2, 3, and 5 are all satisfied, then for any $w \in \mathbb{R}^m$, we have*

$$\frac{1}{n}\sum_{i=1}^n \|\nabla F_i(w) - \nabla F(w)\|^2 \leq \gamma_F^2, \tag{25}$$

where γ_F^2 is defined as

$$\gamma_F^2 := 3C^2\alpha^2\gamma_H^2 + 192\gamma_G^2, \tag{26}$$

where $\nabla F(w) = 1/n \sum_{i=1}^n \nabla F_i(w)$.

Based on the three lemmas, we obtain the following theorem to

Theorem 1. *If Assumptions 1 to 5 hold and the steplength L_F in Lemma 1 satisfies*

$$L_F\beta^2 - \beta + 2L_F^2\beta^2 S^2 \leq 1, \tag{27}$$

then the following FOSP condition holds,

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla F(w_k)\|^2] \leq \frac{2(F(w_0) - F(w^*))}{\beta K}$$
$$+ 4(L_F\beta + 2L_F^2\beta^2 S^2)(\sigma_F^2 + \gamma_F^2)\sqrt{A}. \tag{28}$$

Proof: See the Appendix. □

Corollary 1. *Assume the conditions in Theorem 1 are satisfied. Then, if we set the number of total communication rounds as $K = \mathcal{O}(\epsilon^{-3})$, the global learning rate as $\beta = \mathcal{O}(\epsilon^2)$, the staleness threshold as $S = \mathcal{O}(\epsilon^{-1})$, and the number of UEs that updates the global model as $A = \mathcal{O}(\epsilon^{-2})$, Algorithm 1 finds an ϵ-FOSP for PerFedS^2.*

Proof: Note that $2(F(w_0) - F(w^*))$ is constant, then $K = \mathcal{O}(\epsilon^{-3})$ and $\beta = \mathcal{O}(\epsilon^2)$ ensure the first term of right-hand-side of (28) to be equal to $\mathcal{O}(\epsilon)$. Next we examine the second term of (28). Note that $(\sigma_F^2 + \gamma_F^2)$ is constant, then $\beta = \mathcal{O}(\epsilon^2)$ and $S = \mathcal{O}(\epsilon^{-1})$ together make $(2L_F\beta + 4L_F^2\beta^2 S^2) = \mathcal{O}(\epsilon^2)$. At this point, if $A = \mathcal{O}(\epsilon^{-2})$, the second term of (28) is equivalent to $\mathcal{O}(\epsilon)$. □

V. Joint Bandwidth Allocation and UE Scheduling

In this section, we present the steps to solve the optimization problem P1. Particularly, we decouple P1 into P2, a bandwidth

allocation problem, and P3, a UE scheduling problem. Note that individually solving the two sub-problems is equivalent to solving the original P1, which will be elaborated in the sequel.

A. Problem Decoupling

We begin with the bandwidth allocation problem. Given a scheduling pattern $\mathbf{\Pi}$, the bandwidth allocation problem can be written as follows:

$$\min_{\mathbf{b}} \quad T(\mathbf{\Pi}) \tag{P2}$$

$$s.t. \quad \sum_{k=1}^{K} \max_{i\in\mathcal{A}_k}\{T_k^i\} \le T(\mathbf{\Pi}) \tag{C2.1}$$

$$\sum_{i=1}^{n} b_k^i \le B, k = 1, 2, \ldots, K \tag{C2.2}$$

$$\sum_{j=k-\tau_k^i}^{k} Z_j^i \le Z, \quad \forall i \in \mathcal{U}. \tag{C2.3}$$

Then, with the optimal bandwidth allocation and the corresponding minimal overall training time $T^*(\mathbf{\Pi})$, the UE scheduling problem can be written as follows,

$$\min_{K,A,\mathbf{\Pi}} \quad F(w) \tag{P3}$$

$$s.t. \quad \sum_{k=1}^{K} \max_{i\in\mathcal{A}_k}\{T_k^i\} = T^*(\mathbf{\Pi}), \quad \forall i \in \mathcal{U} \tag{C3.1}$$

$$\sum_{j=k-\tau_k^i}^{k-\tau_k^i+S} \pi_j^i \ge 1, \quad \forall i \in \mathcal{U} \tag{C3.2}$$

$$K \ge \frac{S}{\eta_i}, \quad \forall i \in \mathcal{U}. \tag{C3.3}$$

B. Optimal Bandwidth Allocation

In order to solve P2, we introduce the following theorems to explore the relationship between b_k^i and $T(\mathbf{\Pi})$ step by step.

Theorem 2. *If the server updates the global model after receiving A gradients from the UEs in each round, then the optimal bandwidth allocation can be achieved if and only if all the scheduled UEs have the same finishing time.*

Proof: Recall the expression of r_k^i defined in (9), we take a derivative with respect to b_k^i and arrive at the following

$$\begin{aligned}
&\frac{\mathrm{d}}{\mathrm{d}b_k^i}\left(b_k^i \ln\left(1+\frac{p_i h_i \|c_i\|^{-\kappa}}{b_k^i N_0}\right)\right) \\
=&\ln\left(1+\frac{p_i h_i \|c_i\|^{-\kappa}}{b_k^i N_0}\right) - \frac{p_i h_i \|c_i\|^{-\kappa}}{b_k^i N_0 + p_i h_i \|c_i\|^{-\kappa}} \qquad (31)\\
>&\frac{\frac{p_i h_i \|c_i\|^{-\kappa}}{b_k^i N_0}}{1+\frac{p_i h_i \|c_i\|^{-\kappa}}{b_k^i N_0}} - \frac{p_i h_i \|c_i\|^{-\kappa}}{b_k^i N_0 + p_i h_i \|c_i\|^{-\kappa}} \\
=&0, \qquad (32)
\end{aligned}$$

where the inequality follows from the fact that $\ln(1+x) > \frac{x}{1+x}$, for $x > 0$. Therefore, r_k^i monotonically increases with b_k^i. While it is obvious that $r_k^i > 0$, and thus $Tcmp_k^i + Tcom_k^i = Tcmp_k^i + \frac{Z_k^i}{r_k^i}$ monotonically decreases with b_k^i. Therefore, at round k, if any UE $i \in \mathcal{A}_k$ has finished its whole local model update process than the others, we can decrease its bandwidth allocation to make it up for the other slower UEs in $\mathcal{A}_k$. As a result, the round latency which is determined by the slowest UE in $\mathcal{A}_k$ can be reduced. Such a bandwidth compensation is performed until all scheduled UEs in $\mathcal{A}_k$ finish their local iterations at the same time. Consequently, the optimal bandwidth allocation in round k is achieved when all scheduled UEs in $\mathcal{A}_k$ have the same finishing time. □

Theorem 3. *Given the relative participation frequency η_i ($i \in \mathcal{U}$), the UEs would be scheduled in an order with a recurrence pattern. That is, the UEs would periodically participate into the global model update.*

Proof: Recall the formulation of η_i defined in (15), it is obvious that η_i is computed by the number of times UE i has been scheduled during all K rounds. Therefore, if η_i is settled, then $\sum_{k=0}^{K-1} \pi_k^i$ is settled. As a result, if the UEs are scheduled periodically, the times of each UE involved in the global update can be settled, thus matching the relative participation rate it has been assigned with. □

Theorem 4. *The optimal bandwidth allocation that achieves the minimum learning time is given by the following*

$$\begin{cases} \sum_{i\in\mathcal{U}} b_k^i = B, \quad k = 1, \ldots, K \\ b_k^i > \dfrac{Bn\eta_i Z}{(T_i^*(\Pi) - Tcmp_i)(W(-\Gamma_i e^{-\Gamma_i}) + \Gamma_i)}, \\ \sum_{i\in\mathcal{A}_k} b_k^i \le B, \end{cases} \tag{33}$$

where $\Gamma_i \triangleq \frac{N_0 Z}{(T_i^*(\Pi) - Tcmp_i) p_i h_i \|c_i\|^{-\kappa}}$, $W(\cdot)$ is Lambert-W function, and $T_i^*(\Pi)$ is the objective value of (P2).

Proof: From Theorem 3, we know that all UEs update the global model periodically. Let K_p denote the number of communication rounds in each period, then inferring from Theorem 2, all UEs have the same finishing time in each period without any waiting time. That is, we have

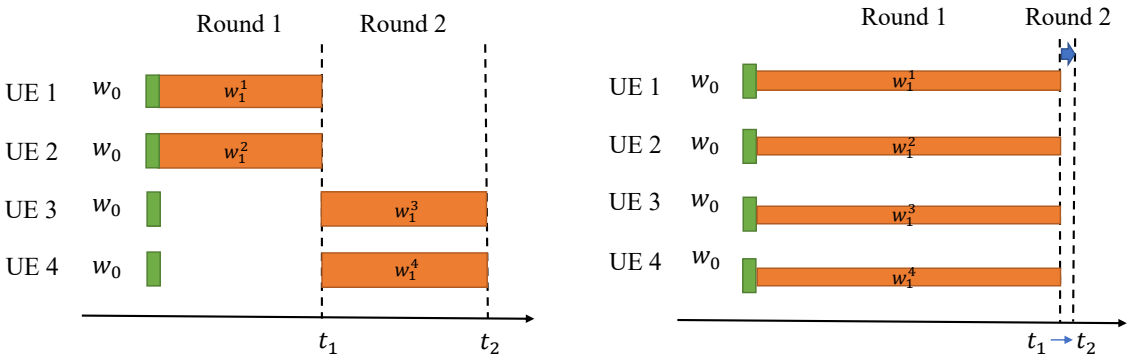
$$\sum_{k=1}^{K_p} T_k^i = \sum_{k=1}^{K_p} T_k^i, \quad \forall i, j \in \mathcal{U}, i \ne j, \tag{34}$$

Meanwhile, we have

$$\sum_{k=1}^{K_p} Z_k^i = \eta_i Z A K_P, \quad \forall i \in \mathcal{U}, \tag{35}$$

where ZAK_P denotes the number of bits that needs to be transmitted during the K_p rounds. This equation indicates that the number of bits transmitted by UE i during K_p rounds is equal to the product of its relative participation frequency η_i and the total number of bits transmitted during that K_p communication rounds. From equation (35), it is easy to indicate that

$$\sum_{k=1}^{K_p} \frac{Z_k^i}{\eta_i} = \sum_{k=1}^{K_p} \frac{Z_k^j}{\eta_j}, \quad \forall i, j \in \mathcal{U}, i \ne j. \tag{36}$$



(a) The largest bandwidth allocation to UEs in $\mathcal{A}_k$

(b) The least bandwidth allocation to UEs in $\mathcal{A}_k$

Fig. 2: Bandwidth allocation example, where all UEs have the same parameters, and $A = 2$.

Now combing (34) and (36), we have

$$\frac{\sum_{k=1}^{K_p} Z_k^i}{\eta_i \sum_{k=1}^{K_p} T_k^i} = \frac{\sum_{k=1}^{K_p} Z_k^j}{\eta_j \sum_{k=1}^{K_p} T_k^j}, \quad \forall i, j \in \mathcal{U}, i \neq j. \tag{37}$$

From equation (37) we observe that $\frac{\sum_{k=1}^{K_p} Z_k^i}{\sum_{k=1}^{K_p} T_k^i}$ denotes the average rate of UE i during K_p rounds. That is, we have

$$\frac{\mathbb{E}(r_k^i)}{\eta_i} = \frac{\mathbb{E}(r_k^j)}{\eta_j}, \quad \forall i, j \in \mathcal{U}, i \neq j. \tag{38}$$

The above equation states a fact that as long as the average rate of each UE is weighted equalized, the optimal solution is achieved. Therefore, there exists *infinitely many solutions* of r_k^i to the above equation. The simplest solution is $\frac{\eta_i}{r_k^i} = \frac{\eta_j}{r_k^j}$ in each round k. Note that r_k^i is determined by b_k^i, and thus there exits infinitely many solutions of b_k^i in each round k.

Our next step is to compute the boundary values of b_k^i. To do this, we first divide UEs into two categories: UEs in $\mathcal{A}_k$ and UEs do not in $\mathcal{A}_k$.

- At one extreme case, only UEs in $\mathcal{A}_k$ are assigned with bandwidth. That is, $\sum_{i \in \mathcal{A}_k} b_k^i = B$. Under this case, the PerFedS^2 algorithm turns out to be a synchronous PerFedAvg algorithm where in each round A UEs are selected to update the global model. Meanwhile, the bandwidth is allocated proportionally to the UEs in $\mathcal{A}_k$ such that $\frac{r_k^i}{\eta_i} = \frac{r_k^j}{\eta_j}$, $\forall i, j \in \mathcal{A}_k, i \neq j$. This extreme case is corresponding to the third inequation of (33).
- At the other extreme case, all UEs in round k share the available bandwith B at a rate $\frac{r_k^i}{\eta_i} = \frac{r_k^j}{\eta_j}$, $\forall i, j \in \mathcal{A}_k, i \neq j$. This case indicates the least bandwidth allocation to UEs in $\mathcal{A}_k$ to ensure their orders to arrive the server in the scheduling pattern. Under this case, $\sum_{i \in \mathcal{U}} b_k^i = B$. Therefore, a closed form of b_k^i is obtained, which is corresponding to the lower bound of b_k^i shown in the second inequation of (33).

□

To better illustrate these approaches, let us take the example in Fig. 2. Assume $A = 2$ and the four UEs have the same η_i, p_i, h_i, and c_i. We can write the scheduling pattern Π of the four UEs as follows:

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ \cdots & \cdots & \cdots & \cdots \end{pmatrix}. \tag{39}$$

The length of the scheduling period is $K_p = 2$. Meanwhile, according to Theorem 4, we have $\mathbb{E}(r_k^1) = \cdots = \mathbb{E}(r_k^4)$. One extreme case of bandwidth allocation is UE 1 and UE 2 share the total bandwidth B in the first round, each of which is assigned $\frac{B}{2}$. At the same time, UE 3 and UE 4 can complete their local computation during round 1. Then, at round 2, all bandwidth B is allocated to UE 3 and UE 4 for their gradients transmission. In this case, according to Theorem 2, in each round, both UEs will finish their gradient transmission at the same time. That is, the duration of round 1 will be minimized when UE 1 and UE 2 share the total bandwidth B equally. At this point, the round duration is $\frac{Z}{r(B/2)}$, where $r(B/2) = \frac{B}{2}\ln(1 + \frac{2p_i h_i \|c_i\|^{-\kappa}}{BN_0})$. Similarly, the duration of round 2 is also $\frac{Z}{r(B/2)}$. Then, the total time of each period is $\frac{2Z}{r(B/2)}$. The other extreme case of bandwidth allocation is for all the four UEs to share the bandwidth equally, then the UEs will finish one time of global update at the same time, which is computed by $\frac{Z}{r(B/4)}$. Note that we set $A = 2$, but in this case if all UEs finish one communication round at the same time then $A = 4$, therefore this extreme situation cannot be achieved but can only be approached infinitely. It is obvious $\frac{Z}{r(B/4)} = \frac{2Z}{r(B/2)}$, this equation indicates that all bandwidth allocation policies between the two extreme cases can lead to the same minimized overall training time.

At this point, according to the features of the optimal bandwidth solutions, we obtain four corollaries. Corollary 2 and 3 are two direct conclusions derived from Theorem 2, which are shown as follows,

Corollary 2. *From Theorem 2, we find that in each round k, UEs in $\mathcal{A}_k$ will finish the communication round at the same*

time. That is, none of the UEs have to wait for the others under the optimal bandwidth allocation policy. Therefore, we have $\sum_{k=1}^{K} \max_{i\in\mathcal{A}}\{T_k^i\} = \sum_{k=1}^{K} T_k^{i*} = T_i^*$ *(*$\forall i \in \mathcal{U}$*).*

Corollary 3. *The optimal overall training time is equivalent to the optimal total training time of any UE* i *from a long-term perspective when* $K \to +\infty$*. That is,* $T^*(\mathbf{\Pi}) = T_i^*$ *(*$\forall i \in \mathcal{U}$ *and a large* K*).*

Next, according to Theorem 4, we extract Corollary 4 to characterize the optimal solutions of Z_k^i, which is determined right after the computation of b_k^i.

Corollary 4. *There exists infinitely many solutions of* Z_k^i *as long as the bandwidth allocation follows the results shown in Theorem 4. Meanwhile,* Z_k^i *is in a range of values from* 0 *to* Z*.*

At last, we introduce Corollary 5 to describe the relationship between the relative participation frequency η_i and the optimal overall training time $T^*(\mathbf{\Pi})$.

Corollary 5. *There is a tradeoff between the relative participation frequency* η_i *(*$i \in \mathcal{U}$*) and the optimal overall training time* $T^*(\mathbf{\Pi})$*. As long as* η *is defined or determined, then according to Theorem 3 the circular scheduling pattern* $\mathbf{\Pi}$ *can be determined. With the scheduling pattern* $\mathbf{\Pi}$*, according to Theorem 4, the optimal bandwidth allocation and the corresponding optimal overall training time* $T^*(\mathbf{\Pi})$ *can be determined.*

Algorithm 2: Greedy PerFedS^2 Scheduling Algorithm

Input: $\eta = \{\eta_1, \eta_2, \ldots, \eta_n\}$, A^*
Initialize $\Pi \leftarrow \varnothing$;
for $k = 1$ **to** K **do**
  for $i = 1$ **to** N **do**
    if *the total number of global updates* $sum(\Pi) = 0$ **then**
      $\hat{\eta}_i = 0$;
    else
      $\hat{\eta}_i = \frac{\text{number of overall updates of UE } i}{\text{number of overall global updates}} = \frac{sum(\Pi[:,i])}{sum(\Pi)}$;
    end
    if *current number of updates in round* k $sum(\Pi[k,:]) < A^*$ *and current relative participation frequency of UE* i $\hat{\eta}_i \le \eta_i$ **then**
      Set $\Pi[k][i] \leftarrow 1$;
      if *current number of updates in round* k $sum(\Pi[k,:]) < A^*$ **then**
        Schedule the first $A^* - sum(\Pi[k,:])$ UEs in current round k;
        i.e., $\Pi[k][0 : A^* - sum(\Pi[k,:])] = 1$;
      end
    else
      $\Pi[k][i] = 0$;
    end
  end
end

C. Scheduling Policy

Based on the optimal bandwidth b_k^i obtained from P2, we turn to P3 to solve the UE scheduling problem. From (C3.2) we have

$$\eta_i A K = \sum_{k=1}^{K} \pi_k^i \ge \frac{K}{S}, \quad \forall i \in \mathcal{U}, \tag{40}$$

which can be further simplified to $A \ge \frac{1}{\eta_i S}$. Meanwhile, note that the minimization of $F(w)$ can be approximated by minimizing the upper bound of $\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla F(w_k)\|^2]$ according to Theorem 1. Therefore, P3 can be approximated by P4 as follows:

$$\min_{K,A,\mathbf{\Pi}} \quad \frac{2(F(w_0)-F(w^*))}{\beta K} + 4(L_F\beta + 2L_F^2\beta^2S^2)(\sigma_F^2+\gamma_F^2)\sqrt{A} \tag{P4}$$

$$s.t. \quad T_i^* = T^*(\mathbf{\Pi}), \quad \forall i \in \mathcal{U} \tag{C4.1}$$

$$A \ge \frac{1}{\eta_i S}, \quad \forall i \in \mathcal{U} \tag{C4.2}$$

$$K \ge \frac{S}{\eta_i}, \quad \forall i \in \mathcal{U}, \tag{C4.3}$$

where (C4.1) is derived from Corollary 2 and 3.

The relationship between A and K has been coarsely analysed in Corollary 1, where $K = \mathcal{O}(\epsilon^{-3})$ and $A = \mathcal{O}(\epsilon^{-2})$. This means that the optimal K^* and A^* can only be estimated in the implementation. Let the first term and the second term of the objective of P4 be equal to ϵ respectively, the optimal solution of K and A can be approximated by

$$K^* \approx \min_{i\in\mathcal{U}}\{\frac{2(F(w_0)-F(w^*))}{\beta\epsilon}, \frac{S}{\eta_i}\} \tag{42}$$

$$A^* \approx \min_{i\in\mathcal{U}}\{\frac{\epsilon^2}{16(L_F\beta+2L_F^2\beta^2S^2)^2(\sigma_F^2+\gamma_F^2)^2}, \frac{1}{\eta_i S}\}. \tag{43}$$

With the optimal value A^*, we use a greedy algorithm to generate the scheduling policy matrix Π, which is shown in Algorithm 2. In each round k, the algorithm is always picking up the UE i with the smallest current relative participation frequency $\hat{\eta}_i$, if $\hat{\eta}_i < \eta_i$ then the algorithm sets $\pi_k^i = 1$. Then the algorithm picks up the second poorest UE j and set $\pi_k^j = 1$. This process repeats until A^* UEs are picked up in round k. For the next round $k+1$, the same process repeats. In this way, the circular scheduling pattern can be achieved and $\mathbf{\Pi}$ is obtained.

VI. Performance Evaluation

In this section, we conduct extensive experiments to (i) verify the effectiveness of PerFedS^2 in saving the overall training time and (ii) examine the effects of different system parameters on the performance of PerFedS^2.

A. Setup

1) Datasets and Models: We consider an FL system that contains multiple UEs located in a cell of radius $R = 200$

TABLE I: System Parameters

Parameter	Value
α (MNIST)	0.03
β (MNIST)	0.07
α (CIFAR-100)	0.02
β (CIFAR-100)	0.06
α (Shakespeare)	0.03
β (Shakespeare)	0.07
B	1 MHz
κ	3.8
N_0	-174 dBm/Hz
p_i	0.01 W

m and a BS located at the center. Meanwhile, the Rayleigh distribution parameter of h_k^i across communication rounds is 40. We conduct the experiments using three datasets: MNIST [31], CIFAR-100 [32] and the Shakespeare [33] datasets. The network model we used for MNIST is a 2-layer deep neural network (DNN) with hidden layer of size 100. The network model we used for CIFAR-100 is LeNet-5 [34] that contains two convolutional layers and three fully connected layers. And the network model we used for the Shakespeare dataset is an LSTM classifier. The number of UEs under the MNIST and the CIFAR-100 datasets is set to be 20, and the number of UEs under the Shakespeare dataset for next-character prediction is 188. The other parameters used in the experiments are summarized in Table I.

2) Baselines: We compare PerFedS^2 with three benchmarks: synchronous, semi-synchronous, and asynchronous FL algorithms. For the synchronous FL benchmark, we consider three algorithms, FedAvg, FedProx [35], and Per-FedAvg (termed as FedAvg-SYN, FedProx-SYN and PerFed-SYN in the figures). FedProx is a FL algorithm that deals with heterogenous datasets. For the semi-synchronous benchmark, we consider only two algorithms besides PerFedS^2, semi-synchronous Federated Learning (FedAvgS^2), which is a semi-asynchronous FL algorithm, and semi-synchronous FedProx (FedProxS^2). For the asynchronous FL benchmark we consider three algorithms, FedAvg-ASY, FedProx-ASY and PerFed-ASY. The above three algorithms are asynchronous FL mechanisms, where the server performs the global updating as soon as it receives a local model from any UE.

3) Dataset Participation: The level of divergence in the distribution of UEs' datasets will affect the overall performance of the system. To reflect this feature, each UE is allocated a different local data size and has $l = 1, 2, \ldots, 10$ of the 10 labels, where l denotes the level of data heterogeneity, the higher l is, the more diverse the datasets are.

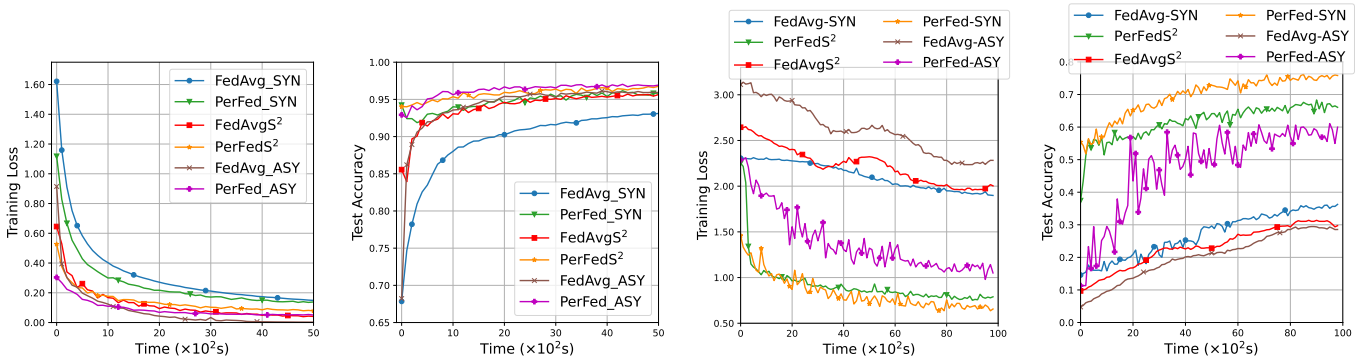
4) Relative Participation Frequency Setting: The relative participation frequency plays a critical role in the system performance as it determines not only the scheduling pattern but also the minimal overall training time. In practice, there are many factors that may affect the value of η. For example, the distances from UEs to the server and the transmit power of each UE. In this paper, we use two sets of η. For the first one, we consider all the UEs have the same η_i, i.e., $\eta_1 = \eta_2 = \cdots = \eta_n$. For the second one, we consider the distances from the UEs to the server is uniformly distributed, while the other parameters of the UEs are the same. Under this setting, the values of η_i among the UEs are unbalanced.

B. Evaluation Results

1) Effect of relative participation frequency η: Fig. 3 shows the convergence performance comparison between PerFedS^2 and other five FL and PFL algorithms, where UEs have the same η_i, and $A = 5$. Then Fig. 4 shows the convergence performance comparison of the six algorithms, where the η_i of each UE is determined by its distance to the server, and the distance is uniformly distributed from 0 to 200 m. At last, Fig. 5 shows the convergence comparison of the six algorithms using Shakespeare dataset, where $A = 50$.

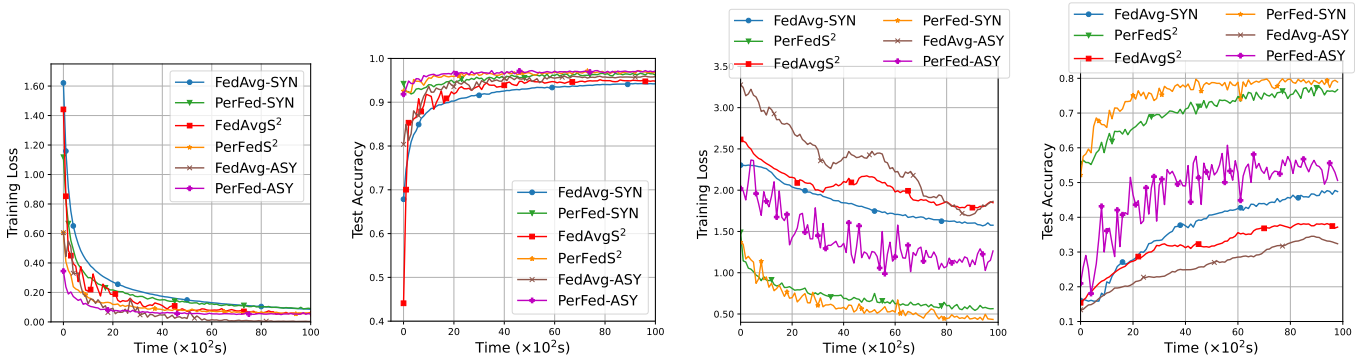
From both figures, we find that for MNIST, generally, it takes synchronous algorithms the most time to achieve the same convergence performance compared with semi-synchronous and asynchronous algorithms, then asynchronous algorithms behaves the best. However, for the CIFAR-100 dataset, generally, semi-synchronous algorithms behaves the best. We attribute this confliction of behavior to the fact that MNIST is a much simpler dataset than CIFAR-100. Commonly, we use asynchronous algorithms to save waiting time for faster UEs and hope that the convergence performance will not be affected by the update staleness. This only works when the dataset is simple and easy to train. Therefore, as we can see in Fig. 3, for the MNIST dataset with a two-layer DNN model, the asynchronous algorithms does behave the best, semi-synchronous algorithms is the second, and synchronous algorithms behave the worst. However, when it comes to the CIFAR-100 dataset with the LeNet-5 model, which is a much larger dataset with a much more complicated model, it is hard for the asynchronous algorithms to convergence. In this case, semi-synchronous algorithms behave the best. This evaluation performance verifies our theoretical result that a proper semi-synchronous algorithm not only mitigates the straggler problem that happened in synchronous algorithms, but also bounds the staleness caused by the stragglers, thereby ensuring the convergence of the learning process. Meanwhile, it is clear that PFL algorithms converge much faster than FL algorithms. This result is derived from the fact the PFL algorithms is designed to adapt and converge fast to new datasets.

Most importantly, we find that compared with Fig. 3, the convergence performance shown in Fig. 4 is poorer. This is because the relative participation frequencies of UEs in Fig. 4 is not equalized. Since the UEs are uniformly distributed in the cell, their distances to the central server are different. The UEs with longer distances to the server have to transmit its gradients for a longer time to reach the server. Therefore, these UEs are naturally slower than the others, leading to smaller η to participate in the global model updates. Given that the datasets among UEs are heterogenous, the less participation of long distance UEs will lead to inadequate training on these UEs, making the global model convergence performance poorer than the ones shown in Fig. 3.



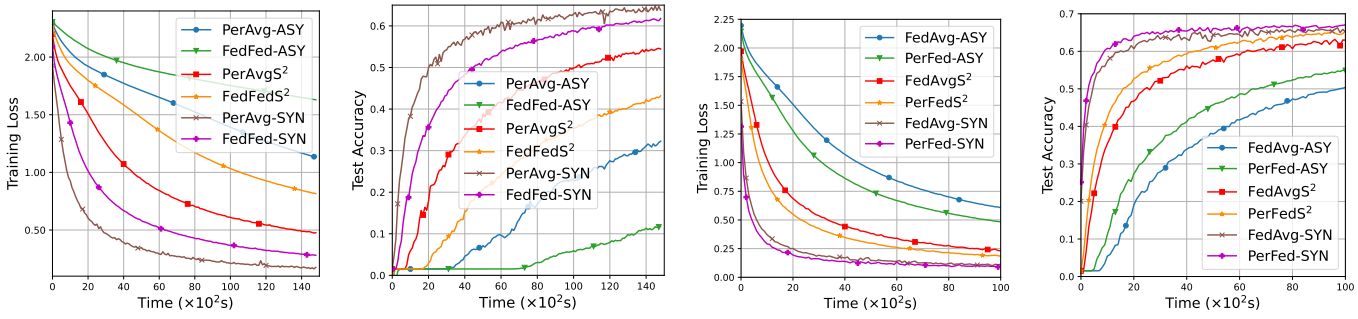
(a) MNIST training loss (b) MNIST test accuracy (c) CIFAR-100 training loss (d) CIFAR-100 test accuracy

Fig. 3: Convergence performance comparison of PerFedS^2, FedAvgS^2, FedAvg-SYN, PerFed-SYN, FedAvg-ASY and PerFed-ASY using MNIST and CIFAR-100 datasets. In this case, $\eta_1 = \eta_2 = \cdots = \eta_n$. Meanwhile, as for the PerFedS^2 and FedAvgS^2 algorithms, we set $A = 5$.



(a) MNIST training loss (b) MNIST test accuracy (c) CIFAR-100 training loss (d) CIFAR-100 test accuracy

Fig. 4: Convergence performance comparison of PerFedS^2, FedAvgS^2, FedAvg-SYN, PerFed-SYN, FedAvg-ASY and PerFed-ASY using MNIST and CIFAR-100 datasets. In this case, the distance from UEs to the server obeys the random distribution from 0 to 200 m. Meanwhile, as for the PerFedS^2 and FedAvgS^2 algorithms, we set $A = 5$.

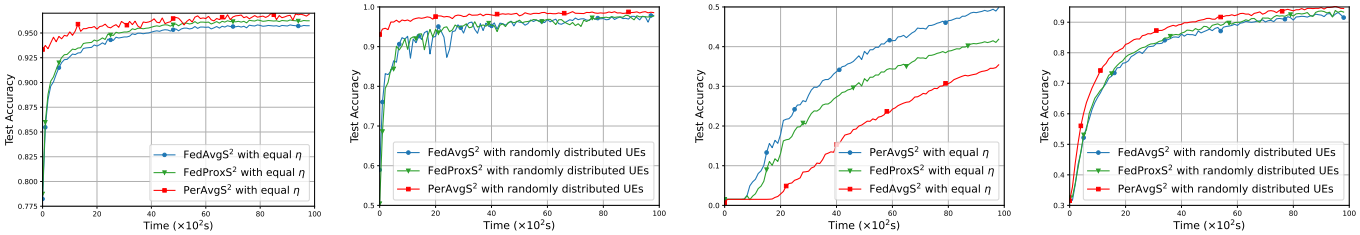


(a) Shakespeare training loss (b) Shakespeare test accuracy (c) Shakespeare training loss (d) Shakespeare test accuracy

Fig. 5: Convergence performance comparison of PerFedS^2, FedAvgS^2, FedAvg-SYN, PerFed-SYN, FedAvg-ASY and PerFed-ASY using the Shakespeare dataset. For (a) and (b), $\eta_1 = \eta_2 = \cdots = \eta_n$, and for (c) and (d), the distance from UEs to the server obeys the random distribution from 0 to 200 m. Meanwhile, as for the PerFedS^2 and FedAvgS^2 algorithms, we set $A = 50$.

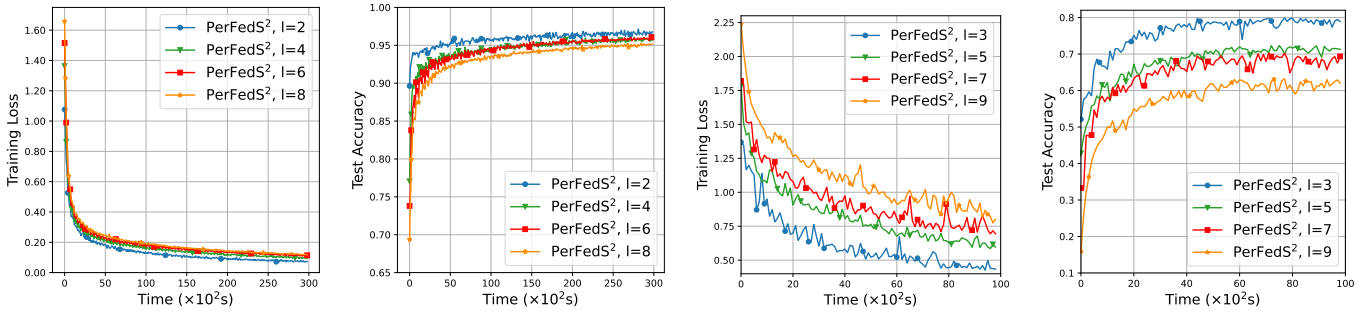
As for the shakespeare dataset, we find that all the conclusions about the comparisons between the 6 algorithms drawn from the above two datasets still stand.

The comparison between FedAvgS^2, FedProxS^2 and PerFedS^2 using the MNIST and Shakespeare datasets is shown in Fig. 6. From the figure it is obvious that PerFedS^2 out-



(a) MNIST (b) MNIST (c) Shakespeare (d) Shakespeare

Fig. 6: Convergence performance comparison of PerFedS^2, FedAvgS^2 and FedProxS^2. For (a), we use the MNIST dataset and $\eta_1 = \eta_2 = \cdots = \eta_n$. For (b), we use the MNIST dataset and the distance from UEs to the server obeys the random distribution from 0 to 200 m. For (c), we use the Shakespeare dataset and $\eta_1 = \eta_2 = \cdots = \eta_n$. And for (d), we use the Shakespeare dataset and the distance from UEs to the server obeys the random distribution from 0 to 200 m. Meanwhile, we set $A = 5$ for the MNIST dataset and $A = 50$ for the Shakespeare dataset.



(a) MNIST training loss (b) MNIST test accuracy (c) CIFAR-100 training loss (d) CIFAR-100 test accuracy

Fig. 7: Convergence performance of PerFedS^2 with respect to the non-i.i.d level l of data sampled from the MNIST and CIFAR-100 datasets. We compare the results when $l = 2, 4, 6, 8$ for data sampled from the MNIST dataset, and $l = 3, 5, 7, 9$ for data sampled from the CIFAR-100 dataset.

performs the other two algorithms. This is reasonable since Per-FedAvg has already been verified in previous works to provide a better convergence performance, and PerFedS^2 is designed based on Per-FedAvg. Therefore, PerFedS^2 inherits this benefit.

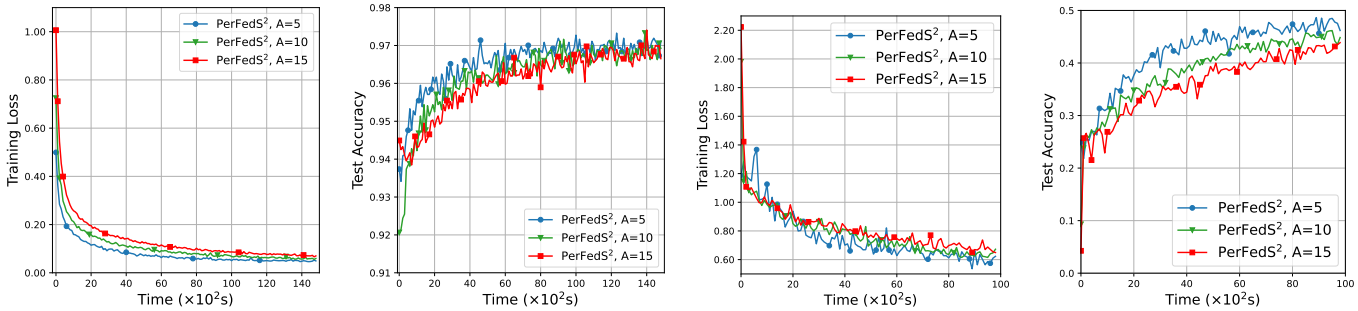
2) Effect of the non-i.i.d. level l: Fig. 7 shows the evaluation results of PerFedS^2 under different non-i.i.d. levels. It is obvious that for both datasets, the higher the heterogenous level is, the worse the convergence performances are. These results are natural and in line with the laws of theory.

3) Effect of the number of participants in each round A: Fig. 8 and Fig. 9 show the convergence performance of PerFedS^2 with respect to different number of participation UEs A in each round, where Fig. 8 is under the case that all UEs have the same η_i, whereas Fig. 9 is under the case that the η_i of each UE is determined by its distance to the central server that follows a random distribution.

As for the MNIST dataset, the result shown in Fig. 8 and Fig. 9 indicates a situation that the larger number of participation UEs in each round, the poorer the convergence performance is. This conclusion is not always true, given that the relative participation frequency vector $\eta = [\eta_i, \eta_2, \ldots, \eta_n]$ in Fig. 9 is generated randomly according to the distances from UEs to the central server, and thus the optimal A to minimize the overall training time is random. We can only conclude that in this very specific case of η, the larger number of participation UEs in each round, the better. Nevertheless, the benefits gained from a smaller value of A is slight in Fig. 9. This is reasonable because, the randomly generated η may result in a scheduling pattern that degrades the influences caused by different number of participation UEs in each round.

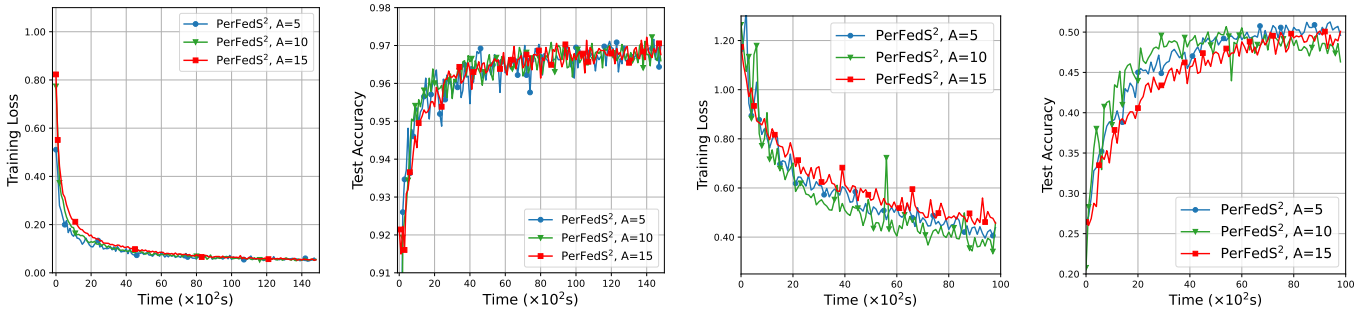
However, as for the CIFAR-100 dataset, although Fig. 8c and 8c still indicate the same conclusion as that in the MNIST dataset, Fig. 9c and 9d indicate another situation where the convergence performance of PerFedS^2 wins when $A = 10$. This result just verified the conclusion we mentioned above, that the conclusion obtained from the MNIST dataset is not always true. The result shown in Fig. 9c and 9d indicate a specific case when $A = 10$ is approaching the optimal A^*.

4) Effect of the staleness threshold S: Finally, we evaluate the effect of the staleness threshold S on the convergence performance of PerFedS^2, where the results are shown in Fig. 10. Here, in order to make the effect of S more clear, we use the simpler setting when all UEs have the same η_i, and $A = 5$. Therefore, when $S \geq 5$, all the scheduled UEs would arrive the server within S rounds. Consequently, we study change of the total training time when $S = 1, 2, 3, 4, 5$.



(a) MNIST training loss (b) MNIST test accuracy (c) CIFAR-100 training loss (d) CIFAR-100 test accuracy

Fig. 8: Convergence performance of PerFedS^2 with respect to the number of UEs A that participate in the global model update in each round using MNIST and CIFAR-100 datasets. In this case, $\eta_1 = \eta_2 = \cdots = \eta_n$. Meanwhile, we compare the results when $A = 5, 10, 15$.



(a) MNIST training loss (b) MNIST test accuracy (c) CIFAR-100 training loss (d) CIFAR-100 test accuracy

Fig. 9: Convergence performance of PerFedS^2 with respect to the number of UEs A that participate in the global model update in each round using MNIST and CIFAR-100 datasets. In this case, the distance from UEs to the server obeys the random distribution from 0 to 200 m. Meanwhile, we compare the results $A = 5, 10, 15$.

Note that in the theoretical analysis, we have the constraint that $\eta_i \geq S/K$. This constraint eliminates the situations when the staleness τ_k^i is larger than the staleness bound S, and thus no updates would be dropped by the central server. However, in practice, η_i is determined by a number of elements, for example, the distances from UEs to the server or the transmit power of individual UEs. Therefore, in practice, the constraint $\eta_i \geq S/K$ cannot be always satisfied. When this happens to UE i, in order to keep η_i constant, other UEs may have to wait until the updates from UE i finally arrives the server, thereby prolonging the overall training time. This conclusion is verified through the results shown in Fig. 10, where the larger S is, the better the convergence performance PerFedS^2 has.

VII. Conclusions

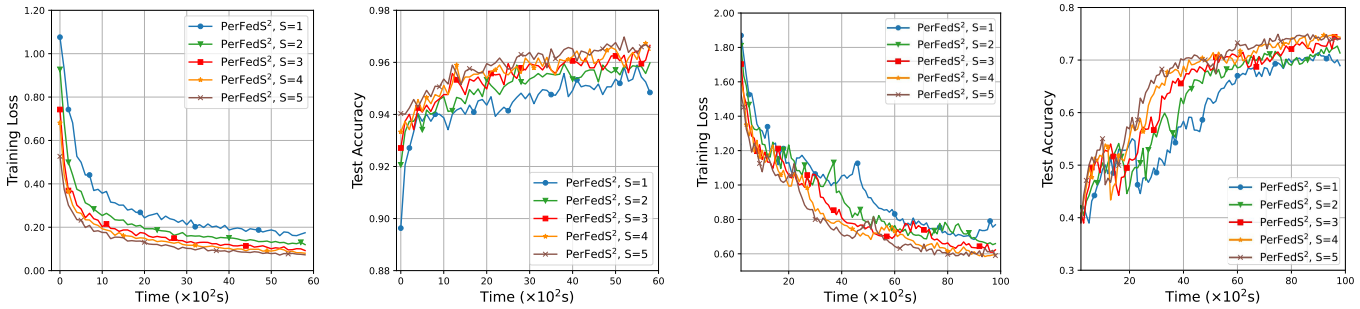
We have proposed a new semi-synchronous PFL algorithm over mobile edge networks, PerFedS^2, that not only mitigates the straggler problem caused by the synchronous training, but also ensures a convergent training loss that may not be guaranteed in the asynchronous training. This is achieved by optimizing the joint bandwidth allocation and UE scheduling problem. In order to solve such an optimization problem, we first have analysed the convergence rate of PerFedS^2, and have proved that there exist a convergent upper bound on the convergence rate. Then, based on the convergence analysis, we have solved the optimization problem by decoupling it into two sub-problems: the bandwidth allocation problem and the UE scheduling problem. For a given scheduling policy, the bandwidth allocations problem has been proved to have infinitely many solutions. Meanwhile, based on the convergence analysis of PerFedS^2, the optimal UE scheduling policy can be determined using a greedy algorithm. We have conducted extensive experiments to verify the effectiveness of PerFedS^2 in saving training time, compared with synchronous and asynchronous FL and PFL algorithms.

Appendix

Proof of Theorem 1

Using Lemma 1, we have

$$\begin{aligned} &F(w_{k+1}) - F(w_k) \\ \leq &\langle \nabla F(w_k), w_{k+1} - w_k \rangle + \frac{L_F}{2} \|w_{k+1} - w_k\|^2 \\ = &- \left\langle \nabla F(w_k), \frac{\beta}{A} \sum_{i \in \mathcal{A}_k} \tilde{\nabla} F_i(w_{k-\tau_k^i}) \right\rangle \end{aligned}$$



(a) MNIST training loss (b) MNIST test accuracy (c) CIFAR-100 training loss (d) CIFAR-100 test accuracy

Fig. 10: Convergence performance comparison of PerFedS^2 with respect to the staleness threshold S using the MNIST and CIFAR-100 datasets. In this case, $\eta_1 = \eta_2 = \cdots = \eta_n$, A=5. Meanwhile, we compare the results $S = 1, 2, 3, 4, 5$.

$$+\frac{L_F}{2}\left\|\frac{\beta}{A}\sum_{i\in\mathcal{A}_k}\tilde{\nabla}F_i(w_{k-\tau_k^i})\right\|^2. \tag{44}$$

From the above inequality, it is obvious that the key is to bound the term $\sum_{i\in\mathcal{A}_k}\tilde{\nabla}F_i(w_{k-\tau_k^i})$. Let

$$\frac{1}{A}\sum_{i\in\mathcal{A}_k}\tilde{\nabla}F_i(w_{k-\tau_k^i}) = X + Y + \frac{1}{A}\sum_{i\in\mathcal{A}_k}\nabla F(w_{k-\tau_k^i}), \tag{45}$$

where

$$\begin{aligned} X &= \frac{1}{A}\sum_{i\in\mathcal{A}_k}(\tilde{\nabla}F_i(w_{k-\tau_k^i}) - \nabla F_i(w_{k-\tau_k^i})), \\ Y &= \frac{1}{A}\sum_{i\in\mathcal{A}_k}(\nabla F_i(w_{k-\tau_k^i}) - \nabla F(w_{k-\tau_k^i})). \end{aligned} \tag{46}$$

Our next step is to upper bound $\mathbb{E}[\|X\|^2]$ and $\mathbb{E}[\|Y\|^2]$ respectively. Recall the Cauchy-Schwarz inequality $\|\sum_{i=1}^n a_i b_i\|^2 \le (\sum_{i=1}^n \|a_i\|^2)(\sum_{i=1}^n \|b_i\|^2)$, as for X, consider the Cauchy-Schwarz inequality with $a_i = \frac{1}{\sqrt{A}}(\tilde{\nabla}F_i(w_{k-\tau_k^i}) - \nabla F_i(w_{k-\tau_k^i}))$ and $b_i = \frac{1}{\sqrt{A}}$, we have

$$\|X\|^2 \le \frac{1}{A}\left(\sum_{i\in\mathcal{A}_k}\|\tilde{\nabla}F_i(w_{k-\tau_k^i}) - \nabla F_i(w_{k-\tau_k^i})\|^2\right). \tag{47}$$

Let $\mathcal{F}_k$ denote the information up to round k. Given that the set of scheduled UEs $\mathcal{A}_k$ is selected according to their relative participation frequency η_i ($i \in \mathcal{A}_k$), hence, by using Lemma 2 along with the tower rule, we have

$$\mathbb{E}[\|X\|^2] = \mathbb{E}[\mathbb{E}[\|X\|^2|\mathcal{F}_k]] \le \sigma_F^2\sum_{i\in\mathcal{A}_k}\eta_i. \tag{48}$$

Meanwhile, as for Y, consider the Cauchy-Schewarz inequality with $a_i = \frac{1}{\sqrt{A}}(\nabla F_i(w_{k-\tau_k^i}) - \nabla F(w_{k-\tau_k^i}))$ and $b_i = \frac{1}{\sqrt{A}}$, we have

$$\|Y\|^2 \le \frac{1}{A}\left(\sum_{i\in\mathcal{A}_k}\|\nabla F_i(w_{k-\tau_k^i}) - \nabla F(w_{k-\tau_k^i})\|^2\right). \tag{49}$$

In a similar way, the mean of $\|Y\|^2$ is the weighted average sum of $\mathbb{E}[\|Y\|^2|\mathcal{F}_k]$, where the weight is the relative participation frequency of UE $i \in \mathcal{A}_k$. By using Lemma 3 along with the tower rule, we have

$$\mathbb{E}[\|Y\|^2] = \mathbb{E}[\mathbb{E}[\|Y\|^2|\mathcal{F}_k] \le \gamma_F^2\sum_{i\in\mathcal{A}_k}\eta_i. \tag{50}$$

Now getting back to the inequality (44), from the fact $\langle a, b\rangle = \frac{1}{2}(\|a\|^2 + \|b\|^2 - \|a-b\|^2)$, we have

$$\begin{aligned} &F(w_{k+1}) - F(w_k) \\ &\le -\frac{\beta}{2}\|\nabla F(w_k)\|^2 - \frac{\beta}{2}\left\|\frac{1}{A}\sum_{i\in\mathcal{A}_k}\tilde{\nabla}F_i(w_{k-\tau_k^i})\right\|^2 \\ &\quad + \frac{\beta}{2}\left\|\nabla F(w_k) - X - Y - \frac{1}{A}\sum_{i\in\mathcal{A}_k}\nabla F(w_{k-\tau_k^i})\right\|^2 \\ &\quad + \frac{L_F\beta^2}{2}\left\|\frac{1}{A}\sum_{i\in\mathcal{A}_k}\tilde{\nabla}F_i(w_{k-\tau_k^i})\right\|^2 \\ &\le -\frac{\beta}{2}\|\nabla F(w_k)\|^2 + L_F\beta^2\underbrace{\|X+Y\|^2}_{T_1} \\ &\quad + \underbrace{\beta\left\|\nabla F(w_k) - \frac{1}{A}\sum_{i\in\mathcal{A}_k}\nabla F(w_{k-\tau_k^i})\right\|^2}_{T_2} \\ &\quad + (L_F\beta^2 - \beta)\left\|\frac{1}{A}\sum_{i\in\mathcal{A}_k}\nabla F(w_{k-\tau_k^i})\right\|^2. \end{aligned} \tag{51}$$

Our next step is to estimate the upper bounds of $\mathbb{E}[T_1]$ and $\mathbb{E}[T_2]$, respectively. As for T_1, we have

$$\mathbb{E}[T_1] \le 2\mathbb{E}[\|X\|^2] + 2\mathbb{E}[\|Y\|^2] = 2(\sigma_F^2 + \gamma_F^2). \tag{52}$$

As for T_2, we have

$$\begin{aligned} T_2 &= \frac{1}{A^2}\left\|\sum_{i\in\mathcal{A}_k}(\nabla F(w_k) - \nabla F(w_{k-\tau_k^i}))\right\|^2 \\ &\le \frac{1}{A}\sum_{i\in\mathcal{A}_k}\left\|\nabla F(w_k) - \nabla F(w_{k-\tau_k^i})\right\|^2 \\ &\le \frac{1}{A}\sum_{i\in\mathcal{A}_k}\left\|L_F(w_k - w_{k-\tau_k^i})\right\|^2 \end{aligned}$$

$$\leq \max_{i\in\mathcal{A}_k} \|L_F(w_k - w_{k-\tau_k^i})\|^2$$
$$= L_F^2 \|(w_k - w_{k-\tau_k^\mu})\|^2, \quad (53)$$

where $\mu = \arg\max_{i\in\mathcal{A}_k} \|L_F(w_k - w_{k-\tau_k^i})\|^2$, the first inequality is obtained from the fact that $\|\sum_{i=1}^n a_i\|^2 \leq n\sum_{i=1}^n \|a_i\|^2$, the second inequality is derived from Lemma 1, and the third inequality comes from the fact that $\frac{1}{n}\sum_{i=1}^n \|a_i\| \leq \max_i \|a_i\|$. It follows that

$$\begin{aligned} T_2 \leq & L_F^2 \|w_k - w_{k-\tau_k^\mu}\|^2 \\ = & L_F^2 \left\| \sum_{j=k-\tau_k^\mu}^{k-1} (w_{j+1} - w_j) \right\|^2 \\ = & L_F^2 \beta^2 \left\| \sum_{j=k-\tau_k^\mu}^{k-1} \frac{1}{A} \sum_{i\in\mathcal{A}_j} \tilde{\nabla} F_i(w_{j-\tau_j^i}) \right\|^2 \\ \leq & L_F^2 \beta^2 S \sum_{j=k-S}^{k-1} \left\| \frac{1}{A} \sum_{i\in\mathcal{A}_j} \tilde{\nabla} F_i(w_{j-\tau_j^i}) \right\|^2 \\ \leq & 2L_F^2 \beta^2 S^2 \|X + Y\|^2 \\ & + 2L_F^2 \beta^2 S^2 \left\| \frac{1}{A} \sum_{i\in\mathcal{A}_j} \nabla F(w_{j-\tau_j^i}) \right\|^2 \end{aligned} \quad (54)$$

Taking expectation on both sides of (54), we have

$$\begin{aligned} \mathbb{E}[T_2] \leq & 4L_F^2 \beta^2 S^2 (\sigma_F^2 + \gamma_F^2) \sum_{i\in\mathcal{A}_k} \eta_i \\ & + 2L_F^2 \beta^2 S^2 \mathbb{E}\left[\left\| \frac{1}{A} \sum_{i\in\mathcal{A}_k} \nabla F(w_{k-\tau_k^i}) \right\|^2 \right]. \end{aligned} \quad (55)$$

Note that $\sum_{i\in\mathcal{A}_k} \eta_i = \sum_{i\in\mathcal{U}} \pi_k^i \eta_i$, we have

$$\begin{aligned} (\sum_{i\in\mathcal{U}} \pi_k^i \eta_i)^2 & \leq \sum_{i\in\mathcal{U}} (\pi_k^i)^2 \sum_{i\in\mathcal{U}} \eta_i^2 \\ & = \sum_{i\in\mathcal{U}} \pi_k^i \sum_{i\in\mathcal{U}} \eta_i^2 = A \sum_{i\in\mathcal{U}} \eta_i^2 \leq A, \end{aligned} \quad (56)$$

where the first equation is derived from the fact that $(\pi_k^i)^2 = \pi_k^i$, the second equation is derived from the fact that $\sum_{i\in\mathcal{U}} \pi_k^i = A$, the last inequation is derived from the fact that $\eta_i < 1$ and $\sum_{i\in\mathcal{U}} \eta_i = 1$. As a result, we have

$$\sum_{i\in\mathcal{A}_k} \eta_i \leq \sqrt{A}. \quad (57)$$

Now getting back to (51), we have

$$\begin{aligned} & \mathbb{E}[F(w_{k+1})] - \mathbb{E}[F(w_k)] \\ \leq & -\frac{\beta}{2} \mathbb{E}[\|\nabla F(w_k)\|^2] \\ & + (2L_F\beta^2 + 4L_F^2\beta^3 S^2)(\sigma_F^2 + \gamma_F^2)\sqrt{A} \\ & + (L_F\beta^2 - \beta + 2L_F^2\beta^2 S^2)\mathbb{E}\left[\left\| \frac{1}{A} \sum_{i\in\mathcal{A}_j} \nabla F(w_{j-\tau_j^i}) \right\|^2 \right] \end{aligned} \quad (58)$$

Summarizing the inequality from $k = 0$ to $k = K - 1$, we have

$$\begin{aligned} & \mathbb{E}[F(w_K)] - f(w_0) \\ \leq & -\frac{\beta}{2} \sum_{k=1}^K \mathbb{E}[\|\nabla F(w_k)\|^2] + \\ & K(2L_F\beta^2 + 4L_F^2\beta^3 S^2)(\sigma_F^2 + \gamma_F^2)\sqrt{A} + \\ & \sum_{k=1}^K (L_F\beta^2 - \beta + 2L_F^2\beta^2 S^2)\mathbb{E}\left[\left\| \frac{1}{A} \sum_{i\in\mathcal{A}_k} \nabla F(w_{k-\tau_k^i}) \right\|^2 \right] \\ \leq & -\frac{\beta}{2} \sum_{k=0}^{K-1} \mathbb{E}[\|\nabla F(w_k)\|^2] \\ & + K(2L_F\beta^2 + 4L_F^2\beta^3 S^2)(\sigma_F^2 + \gamma_F^2)\sqrt{A}, \end{aligned} \quad (59)$$

where the last inequality is due to (27). As a result, the desired result is obtained.

REFERENCES

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017, pp. 1273–1282.

[2] Z. Yang, M. Chen, K.-K. Wong, H. V. Poor, and S. Cui, "Federated learning for 6g: Applications, challenges, and opportunities," *Engineering*, 2021.

[3] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Transactions on Wireless Communications (TWC)*, vol. 20, no. 3, pp. 1935–1949, 2020.

[4] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein *et al.*, "The future of digital health with federated learning," *NPJ Digital Medicine*, vol. 3, no. 1, pp. 1–7, 2020.

[5] H. Xiao, J. Zhao, Q. Pei, J. Feng, L. Liu, and W. Shi, "Vehicle selection and resource optimization for federated learning in vehicular edge computing," *IEEE Transactions on Intelligent Transportation Systems (TITS)*, 2021.

[6] H. Song, J. Bai, Y. Yi, J. Wu, and L. Liu, "Artificial intelligence enabled Internet of Things: Network architecture and spectrum access," *IEEE Computational Intelligence Magazine*, vol. 15, no. 1, pp. 44–51, 2020.

[7] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1146–1159, 2019.

[8] S. Prathiba, G. Raja, S. Anbalagan, S. Gurumoorthy, N. Kumar, and M. Guizani, "Cybertwin-driven federated learning based personalized service provision for 6g-v2x," *IEEE Transactions on Vehicular Technology (TVT)*, 2021.

[9] L. Yang, B. Tan, V. W. Zheng, K. Chen, and Q. Yang, "Federated recommendation systems," in *Federated Learning*. Springer, 2020, pp. 225–239.

[10] Q. Wang, H. Yin, T. Chen, J. Yu, A. Zhou, and X. Zhang, "Fast-adapting and privacy-preserving federated recommender system," *arXiv preprint arXiv:2104.00919*, 2021.

[11] C. T. Dinh, N. H. Tran, and T. D. Nguyen, "Personalized federated learning with moreau envelopes," 2020.

[12] Y. Jiang, J. Konečnỳ, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," *arXiv preprint arXiv:1909.12488*, 2019.

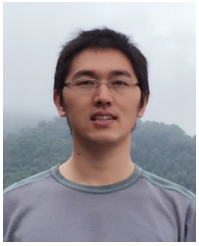
[13] A. Fallah, A. Mokhtari, and A. E. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach." in *International Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[14] Y. Deng, M. M. Kamani, and M. Mahdavi, "Adaptive personalized federated learning," 2020.

[15] A. Shamsian, A. Navon, E. Fetaya, and G. Chechik, "Personalized federated learning using hypernetworks," 2021.

[16] I. Achituve, A. Shamsian, A. Navon, G. Chechik, and E. Fetaya, "Personalized federated learning with gaussian processes," in *International Conference on Neural Information Processing Systems (NeurIPS)*, 2021.
[17] X. Lian, Y. Huang, Y. Li, and J. Liu, "Asynchronous parallel stochastic gradient for nonconvex optimization," vol. 28, 2015, pp. 2737–2745.
[18] C. Xu, Y. Qu, Y. Xiang, and L. Gao, "Asynchronous federated learning on heterogeneous devices: A survey," *arXiv preprint arXiv:2109.04269*, 2021.
[19] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-iid data," in *IEEE International Conference on Big Data (Big Data)*, 2020, pp. 15–24.
[20] W. Wu, L. He, W. Lin, R. Mao, C. Maple, and S. Jarvis, "SAFA: A semi-asynchronous protocol for fast federated learning with low overhead," *IEEE Transactions on Computers (TOC)*, vol. 70, no. 5, pp. 655–668, 2020.
[21] Q. Ma, Y. Xu, H. Xu, Z. Jiang, L. Huang, and H. Huang, "FedSA: A semi-asynchronous federated learning mechanism in heterogeneous edge computing," *IEEE Journal on Selected Areas in Communications (JSAC)*, 2021.
[22] D. Stripelis and J. L. Ambite, "Semi-synchronous federated learning," *arXiv preprint arXiv:2102.02849*, 2021.
[23] Y. Zhang, M. Duan, D. Liu, L. Li, A. Ren, X. Chen, Y. Tan, and C. Wang, "CSAFL: A clustered semi-asynchronous federated learning framework," *arXiv preprint arXiv:2104.08184*, 2021.
[24] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning (ICML)*, 2017, pp. 1126–1135.
[25] A. Fallah, A. Mokhtari, and A. Ozdaglar, "On the convergence theory of gradient-based model-agnostic meta-learning algorithms," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020, pp. 1082–1092.
[26] L. Bottou, "Stochastic gradient descent tricks," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 421–436.
[27] H. Yin and S. Alamouti, "Ofdma: A broadband wireless access technology," in *IEEE Sarnoff Symposium*, 2006, pp. 1–4.
[28] W. Shi, S. Zhou, Z. Niu, M. Jiang, and L. Geng, "Joint device scheduling and resource allocation for latency constrained wireless federated learning," *IEEE Transactions on Wireless Communications (TWC)*, vol. 20, no. 1, pp. 453–467, 2020.
[29] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications (TWC)*, vol. 20, no. 1, pp. 269–283, 2020.
[30] B. Sklar, "Rayleigh fading channels in mobile digital communication systems. i. characterization," *IEEE Communications Magazine*, vol. 35, no. 7, pp. 90–100, 1997.
[31] L. Yann, C. Corinna, and B. Christopher. The mnist dataset. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[32] K. Alex, N. Vinod, and H. Geoffrey. The cifat-10 dataset. [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html
[33] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar, "Leaf: A benchmark for federated settings," *arXiv preprint arXiv:1812.01097*, 2018.
[34] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[35] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine Learning and Systems (MLSys)*, vol. 2, pp. 429–450, 2020.

Chaoqun You (S'13–M'20) is a postdoctoral research fellow in Singapore University of Technology and Design (SUTD). She received the B.S. degree in communication engineering and the Ph.D. degree in communication and information system from University of Electronic Science and Technology of China (UESTC) in 2013 and 2020, respectively. She was a visiting student at the University of Toronto from 2015 to 2017. Her current research interests include mobile edge computing, network virtualization, federated learning, meta-learning, and 6G.



Daquan Feng received the Ph.D. degree in information engineering from the National Key Laboratory of Science and Technology on Communications, University of Electronic Science and Technology of China, Chengdu, China, in 2015. From 2011 to 2014, he was a visiting student with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA. After graduation, he was a Research Staff with State Radio Monitoring Center, Beijing, China, and then a Postdoctoral Research Fellow with the Singapore University of Technology and Design, Singapore. He is now an associate professor with the Shenzhen Key Laboratory of Digital Creative Technology, the Guangdong Province Engineering Laboratory for Digital Creative Technology, the Guangdong-Hong Kong Joint Laboratory for Big Data Imaging and Communication, College of Electronics and Information Engineering, Shenzhen University, Shenzhen, China. His research interests include URLLC communications, MEC, and massive IoT networks. Dr. Feng is an Associate Editor of IEEE COMMUNICATIONS LETTERS, Digital Communications and Networks and ICT Express.

Kun Guo (Member, IEEE) received the B.E. degree in Telecommunications Engineering from Xidian University, Xi'an, China, in 2012, where she received the Ph.D. degree in communication and information systems in 2019. From 2019 to 2021, she was a Post-Doctoral Research Fellow with the Singapore University of Technology and Design (SUTD), Singapore. Currently, she is a Zijiang Young Scholar with the School of Communications and Electronics Engineering at East China Normal University, Shanghai, China. Her research interests include edge computing, caching, and intelligence.

Howard H. Yang (S'13–M'17) received the B.E. degree in Communication Engineering from Harbin Institute of Technology (HIT), China, in 2012, and the M.Sc. degree in Electronic Engineering from Hong Kong University of Science and Technology (HKUST), Hong Kong, in 2013. He earned the Ph.D. degree in Electrical Engineering from Singapore University of Technology and Design (SUTD), Singapore, in 2017. He was a Postdoctoral Research Fellow at SUTD from 2017 to 2020, a Visiting Postdoc Researcher at Princeton University from 2018 to 2019, and a Visiting Student at the University of Texas at Austin from 2015 to 2016. Currently, he is an assistant professor with the Zhejiang University/University of Illinois at Urbana-Champaign Institute (ZJU-UIUC Institute), Zhejiang University, Haining, China. He is also an adjunct assistant professor with the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign, IL, USA

Dr. Yang's research interests cover various aspects of wireless communications, networking, and signal processing, currently focusing on the modeling of modern wireless networks, high dimensional statistics, graph signal processing, and machine learning. He serves as an editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He received the IEEE WCSP 10-Year Anniversary Excellent Paper Award in 2019 and the IEEE WCSP Best Paper Award in 2014.

Chenyuan Feng (S'16-M'21) received the B.E. degree in electrical and electronics engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2016, and the Ph.D. degree in information system technology and design from Singapore University of Technology and Design (SUTD), Singapore, in 2021, respectively. Currently she has been doing postdoctoral work at Shenzhen Key Laboratory of Digital Creative Technology in Shenzhen University. Her research interests include edge computing, federated learning, graph signal processing and recommendation systems. She received the IEEE ComComAp Best Paper Award in 2021.

Tony Q.S. Quek (S'98-M'08-SM'12-F'18) received the B.E. and M.E. degrees in electrical and electronics engineering from the Tokyo Institute of Technology in 1998 and 2000, respectively, and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology in 2008. Currently, he is the Cheng Tsang Man Chair Professor with Singapore University of Technology and Design (SUTD). He also serves as the Director of the Future Communications R&D Programme, the Head of ISTD Pillar, and the Deputy Director of the SUTD-ZJU IDEA. His current research topics include wireless communications and networking, network intelligence, internet-of-things, URLLC, and 6G.

Dr. Quek has been actively involved in organizing and chairing sessions, and has served as a member of the Technical Program Committee as well as symposium chairs in a number of international conferences. He is currently serving as an Area Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and an elected member of the IEEE Signal Processing Society SPCOM Technical Committee. He was an Executive Editorial Committee Member for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, an Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS, and an Editor for the IEEE WIRELESS COMMUNICATIONS LETTERS.

Dr. Quek was honored with the 2008 Philip Yeo Prize for Outstanding Achievement in Research, the 2012 IEEE William R. Bennett Prize, the 2015 SUTD Outstanding Education Awards – Excellence in Research, the 2016 IEEE Signal Processing Society Young Author Best Paper Award, the 2017 CTTC Early Achievement Award, the 2017 IEEE ComSoc AP Outstanding Paper Award, the 2020 IEEE Communications Society Young Author Best Paper Award, the 2020 IEEE Stephen O. Rice Prize, the 2020 Nokia Visiting Professor, and the 2016-2020 Clarivate Analytics Highly Cited Researcher. He is a Fellow of IEEE.