Adaptive Semi-Asynchronous Federated Learning over Wireless Networks

Zhixiong Chen, *Student Member, IEEE*, Wenqiang Yi, *Member, IEEE*, Hyundong Shin, *Fellow, IEEE*, and Arumugam Nallanathan, *Fellow, IEEE*

***Abstract*—Owing to the heterogeneous computation and communication capabilities among clients, the synchronous model aggregation in wireless federated learning (FL) is susceptible to the straggler effect and exhibits low learning efficiency, while asynchronous aggregation encounters delayed gradients that lead to convergence errors and learning performance degradation. To address these obstacles, this work proposes an adaptive semi-asynchronous FL (ASAFL) approach to incorporate the strengths of synchronous and asynchronous FL while mitigating their inherent drawbacks. Specifically, the edge server dynamically adjusts the synchronous degree, i.e., the number of local gradients aggregated in each round, to strike a balance between learning latency and accuracy. Recognizing that data heterogeneity among clients may induce biased global model updating, we propose calibrating the global update by leveraging historical gradients received at the edge server from clients. Following that, we theoretically investigate the impact of synchronous degrees in different rounds on the convergence bound of ASAFL. The results imply that allocating more learning time to the later learning stages to increase the synchronous degree contributes to better learning performance. Based on this, we develop an adaptive synchronous degree control and resource allocation algorithm to enhance the learning performance of FL while adhering to the overall learning latency and wireless resources constraint. Numerical results on the MNIST and CIFAR-10 datasets demonstrate that the proposed approach is capable of attaining faster convergence speed and higher learning accuracy compared to the benchmark FL algorithms.**

***Index Terms*—Data heterogeneity, federated Learning, semi-asynchronous update**

I. Introduction

Federated learning (FL) is a promising distributed learning framework for exploring the massive data generated in wireless networks to support intelligent applications, such as autonomous driving and healthcare [2], [3]. In FL, one edge server orchestrates multiple clients to collaboratively learn a global model through iterative exchanges of model parameters among clients and the server without disclosing client data. This distinctive nature ensures data privacy for clients and reduces the communication overhead compared to centralized learning methods [4], [5].

Despite the advantages of FL, its deployment in wireless networks confronts the following bottlenecks: 1) *Resource Constraints*: The limited wireless resources restrict the participating client number in each round of FL, which substantially restrains the performance of FL [6], [7]. 2) *Device Heterogeneity*: In practical systems, clients typically exhibit substantial diversity in computation and communication capabilities, which induces varied local training time among clients [8], [9]. The conventional synchronous FL approach may suffer from the straggler effect in this presence. 3) *Data Heterogeneity*: The local data distribution among clients are usually non-independent and identically distributed (non-IID). This may induce biased global model updating and learning performance degradation [10]. To enable highly efficient FL, it is important to design innovative FL solutions to conquer the aforementioned challenges.

Existing FL research can be categorized into synchronous FL, asynchronous FL, and semi-asynchronous FL according to the aggregation mechanism. In synchronous FL, the edge server is compelled to wait for all participating clients to complete the local training process before executing the global model update in each round. Consequently, the training speed of FL is dragged by the slowest client [11]. To boost the learning efficiency of synchronous FL, existing studies have primarily concentrated on optimizing resource allocation [12], client selection policies [13]–[15], and their collaborative design [16], [17]. Specifically, in [12], the multi-dimensional optimization in bandwidth allocation, power control, and computation frequency achieved significant energy consumption reduction and ensured the learning performance for FL. By measuring the importance score of clients by the gradient norms, the probabilistic client selection method in [13] effectively improved the convergence speed of FL. In [14], the joint channel and gradient norm-aware client selection approach has been demonstrated to be superior to the selection schemes based on either channel conditions or gradient norms individually. In [15], the client representativity-aware selection approach mitigated the data heterogeneity in FL and captured the trade-off between learning accuracy and latency. The co-design of user selection and resource allocation approach in [16], [17] effectively reduced the FL convergence time and improved the learning accuracy. Although the above approaches effectively tackle resource constraints for FL, the synchronous

Part of this work was presented at the IEEE Vehicular Technology Conference (VTC-Spring), 2024 [1]. (Corresponding author: Arumugam Nallanathan and Hyundong Shin)

Zhixiong Chen is with the School of Electronic Engineering and Computer Science, Queen Mary University of London, E1 4NS London, U.K. (email: zhixiong.chen@qmul.ac.uk).

W. Yi is with the School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, U.K. (email: wy23627@essex.ac.uk).

Hyundong Shin is with the Department of Electronics and Information Convergence Engineering, Kyung Hee University, Yongin-si, Gyeonggido 17104, Republic of Korea (e-mail: hshin@khu.ac.kr).

Arumugam Nallanathan is with the School of Electronic Engineering and Computer Science, Queen Mary University of London, E1 4NS London, U.K., and also with the Department of Electronic Engineering, Kyung Hee University, Yongin-si, Gyeonggido 17104, Korea. (email: a.nallanathan@qmul.ac.uk).

aggregation nature impedes clients from resuming their local training until all participating clients' local updates are aggregated into the global model. Consequently, synchronous FL may pose scalability and learning efficiency challenges, especially in modern wireless networks with massive clients.

To mitigate the straggler effect, asynchronous FL has been developed to enable the server to promptly update the global model upon receiving one local update from an arbitrary client without waiting for the completion of all participating clients [18], [19]. In asynchronous FL, each client seamlessly resumes its local training process after incorporating its gradient into the global model, independent of the training progress of other clients. This characteristic significantly diminishes single-round latency and achieves a fast learning speed [20]. Nevertheless, asynchronous FL suffers from delayed gradients, i.e., clients may return stale gradients that were computed based on an older version of the global model. This may slow down the convergence of asynchronous FL and make the learning process unstable [21]. Moreover, under the data heterogeneity scenarios, the adverse effect of delayed gradients on convergence may be intensified. This is because the global model changes more significantly in adjacent rounds when the data heterogeneity is high [22]. As a result, the delayed gradients become more outdated and inconsistent with the current gradients, exacerbating the overall model convergence.

To avoid the long waiting time caused by the straggler effect in synchronous FL and alleviate the adverse impacts of delayed gradients in asynchronous FL, semi-asynchronous FL has been proposed to strike a balance between learning latency and accuracy. Specifically, in [23], the authors proposed a clustered semi-asynchronous FL framework to mitigate the straggler effect, which classifies clients according to their local training delay and model update directions, and then the clients in each group asynchronously update the group model. The model distribution and client scheduling method in [24] effectively alleviated the impacts of stragglers and model staleness on semi-asynchronous FL. The semi-asynchronous FL approach in [25] employed adaptive learning rate adjustments to balance the contribution of local updates in the global model, thereby effectively reducing the learning time under resource constraints. While demonstrably effective, the above semi-asynchronous FL schemes in [23]–[25] adopt the fixed synchronous degree, i.e., the number of local updates in each round, in the entire learning process, which may not well balance the learning latency and accuracy. To this end, a deep reinforcement learning-based asynchronous FL approach has been proposed in [26], which intelligently varies the number of locally updated models for global model aggregation to minimize the learning time while satisfying the learning performance requirement.

Although the aforementioned semi-asynchronous FL schemes in [23]–[26] effectively address the straggler effect and delayed gradients, the impacts of data heterogeneity have not been considered. In FL, data heterogeneity may result in biased global model updating and significant learning performance degradation. To this end, we propose leveraging clients' historical gradients received at the edge server to calibrate the global update for addressing the adverse impact of non-IID data on FL. The efficacy of this calibration method has been demonstrated in our simulations. In addition, the fixed synchronous degree design in [23]–[25] may not balance the training latency and accuracy well. The reinforcement learning-based synchronous degree control in [26] would increase the learning costs of FL, as it necessitates training an extra reinforcement learning algorithm. Moreover, the relationship between the per-round synchronous degree and the learning performance remains undisclosed. The main differences between our work and the existing works are summarized in Table I. Motivated by this, we theoretically analyze how the synchronous degree in different rounds affects the learning performance. Note that, unlike the existing works, e.g., [18]–[26], which assume the staleness of local gradients is bounded, the convergence analysis in this work mitigates this assumption and investigates the impact of different local gradients' staleness on the learning performance. In addition, our convergence analysis involves the proposed gradient calibration mechanism and reveals that progressively increasing the synchronous degree as the learning progresses contributes to enhanced learning performance. Building on this insight, we develop an adaptive synchronous degree control and resource allocation approach to enhance the learning performance of wireless FL. The primary contributions of this work contain:

- We propose an adaptive semi-asynchronous FL framework, i.e., ASAFL, which dynamically controls the synchronous degree in each learning round to balance the learning accuracy and latency. In addition, to mitigate the negative effect of non-IID data among clients, we propose a gradient calibration mechanism which calibrates the global update by adopting clients' historical gradients to avoid biased global model updating.
- We conduct a theoretical analysis to investigate how the synchronous degree control policy affects the convergence bound of ASAFL, which indicates that augmenting the synchronous degree in each round helps diminish the convergence error. Note that a higher synchronous degree would also result in increased latency. In cases where the time budget for the entire learning process is fixed, our convergence analysis result suggests allocating more learning time to the later rounds to raise their synchronous degree for improving the FL performance.
- According to the convergence findings, we formulate a synchronous degree control and resource allocation problem considering learning latency and wireless bandwidth constraints. To address this long-term stochastic optimization problem and facilitate online synchronous degree control, we convert it into a deterministic form using the Lyapunov optimization framework. Subsequently, we employ a binary search method to find the optimal bandwidth allocation. Following that, we present an efficient synchronous degree control approach that offers an $\mathcal{O}(\sqrt{\mu}, 1/\mu)$ latency-learning balance guarantee where μ is a hyperparameter associated with the algorithm.
- Numerical results corroborate our theoretical findings and demonstrate the effectiveness of the proposed ap-

TABLE I
COMPARISON OF OUR WORK AND RELATED WORKS (✓: CONSIDERED, ✗: NOT CONSIDERED)

Works	Limited wireless resources	Constrained learning latency	Data heterogeneity	Gradient calibration	Convergence analysis	Synchronous degree control	Effects of synchronous degree on learning performance
[18], [20]	✗	✗	✗	✗	✗	✗	✗
[19]	✗	✗	✗	✗	✓	✗	✗
[21], [22]	✗	✗	✓	✗	✓	✗	✗
[23]	✗	✗	✓	✗	✗	✗	✗
[24]	✓	✗	✗	✗	✗	✗	✗
[25]	✓	✗	✓	✗	✓	✗	✗
[26]	✓	✗	✓	✗	✓	✓	✗
Ours	✓	✓	✓	✓	✓	✓	✓

proach. Compared to the benchmark schemes, ASAFL attains 2.9% and 5.78% accuracy gain on the MNIST and CIFAR-10 datasets, respectively. Correspondingly, it provides 2.1x and 1.9x speed up in the learning process on those two datasets to achieve the corresponding target accuracies.

The remainder of this paper is structured as: We presents the proposed ASAFL and system model in Section II. Section III analyzes the convergence bound of ASAFL. Section IV presents the proposed bandwidth allocation and adaptive synchronous degree control scheme to minimize global loss. In section V, we evaluate the proposed approaches by simulation. Finally, we conclude this work in Section VI.

II. SYSTEM MODEL AND LEARNING MECHANISM

In the considered FL system, N clients are coordinated by an edge server to learn a global model $\mathbf{x} \in \mathbb{R}^d$, where d is the dimension of the model. The clients are indexed by $\mathcal{N} = \{1, 2, \cdots, N\}$. Each client n ($n \in \mathcal{N}$) has a local private dataset $\mathcal{B}_n$ with B_n data samples. The entire dataset is $\mathcal{B} = \cup_{n=1}^{N} \mathcal{B}_n$, which has $B = \sum_{n=1}^{N} B_n$ data samples. For each sample $\zeta \in \mathcal{B}$, let $\ell(\zeta; x)$ be the sample-wise loss. Thus, the local loss function of client n is

$$\mathcal{L}_n(\mathbf{x}) = \frac{1}{B_n} \sum\nolimits_{\xi \in \mathcal{B}_n} \ell(\xi; \mathbf{x}). \tag{1}$$

The goal of the FL system is to minimize the global loss function as follows:

$$\mathcal{L}(\mathbf{x}) = \sum\nolimits_{n=1}^{N} q_n \mathcal{L}_n(\mathbf{x}), \tag{2}$$

where q_n is the weight of client n, which satisfies $q_n \geq 0$ and $\sum_{n=1}^{N} q_n = 1$. Similar to [14], [15], we consider the dataset size among clients to be balanced and setting $q_n = \frac{1}{N}$.

A. Adaptive Semi-Asynchronous Federated Learning

To alleviate the straggler effect in synchronous FL and the adverse impact of delayed gradients in asynchronous FL, this work proposes a novel ASAFL approach which dynamically controls the synchronous degree in each round to capture the trade-off between learning latency and accuracy, as depicted in Fig. 1. In addition, due to the data heterogeneity among clients, aggregating partial clients' local updates may induce biased global model updating and learning performance degradation for FL. We propose to utilize clients' historical gradients received by the server to calibrate the global update. The efficacy of this gradient calibration approach is verified in our simulations. Specifically, the server retains a gradient array $\{\mathcal{G}_{n,t} : \forall n \in \mathcal{N}\}$ to cache the latest received gradients from clients. At the start of FL, the server initializes the global model $\mathbf{x}_0$ and broadcasts it to clients. The retained gradient arrays of all clients are initialized as $\mathcal{G}_{n,t} = \mathbf{0}$ ($\forall n \in \mathcal{N}$). Then, the learning process is done by repeating the local training at clients and the global model updating at the edge server, formally described in Algorithm 1. Let T denote the number of global rounds until the training process terminates. In the following, we illustrate the detailed algorithm on the client and server sides, respectively.

- **Client Side** (Line 12-16 in Algorithm 1): Upon receiving the global model $\mathbf{x}_t$, client n ($n \in \mathcal{N}$) updates its model through executing I steps stochastic gradient descent as follows:

$$\mathbf{x}_{n,t}^{i+1} = \mathbf{x}_{n,t}^{i} - \eta \nabla \tilde{\mathcal{L}}_n(\mathbf{x}_{n,t}^{i}), i \in \{0, 1, \cdots, I-1\}, \tag{3}$$

where $\mathbf{x}_{n,t}^{i}$ represents client n's local model in the i-th iteration of round t and $\mathbf{x}_{n,t}^{0} = \mathbf{x}_t$, η is the learning rate. In (3), $\nabla \tilde{\mathcal{L}}_n(\mathbf{x}_{n,t}^{i}) = \frac{1}{A_b} \sum_{\xi \in \mathcal{A}_n} \nabla \ell(\xi; \mathbf{x}_{n,t}^{i})$ is the stochastic gradient, where $\mathcal{A}_n$ is a mini-batch data uniformly sampled from $\mathcal{B}_k$ with $A_b = |\mathcal{A}_n|$ samples. After local training, client n uploads its local gradient, $\widetilde{\mathbf{g}}_{n,t}$, to the edge server, where $\widetilde{\mathbf{g}}_{n,t}$ is

$$\widetilde{\mathbf{g}}_{n,t} = \sum\nolimits_{i=0}^{I-1} \nabla \tilde{\mathcal{L}}_n(\mathbf{x}_{n,t}^{i}) = -\frac{1}{\eta}(\mathbf{x}_{n,t}^{I} - \mathbf{x}_t). \tag{4}$$

- **Edge Server Side** (Line 3-10 in Algorithm 1): In each round t, the edge server determines the *synchronous degree* K_t, i.e., the number of local updates that need to be collected. Then, the edge server waits to receive K_t local gradients from clients. Let $s_{n,t} \in \{0, 1\}$ indicate whether client n uploads its local gradient to the server in round t, where $s_{n,t} = 1$ signifies the upload, and $s_{n,t} = 0$ otherwise. Thus, the set of clients whose local gradients are received at the edge server in round t can be denoted as $S_t = \{n : s_{n,t} = 1, \forall n \in \mathcal{N}\}$. Denote by $\mathbf{g}_{n,t}$ the received local gradient from client n in round t. Since clients are asynchronously involved in the global model updating, $\mathbf{g}_{n,t}$ is not necessarily equal to $\widetilde{\mathbf{g}}_{n,t}$. Denote by $\tau_{n,t}$ the staleness of $\widetilde{\mathbf{g}}_{n,t}$, defined as the interval

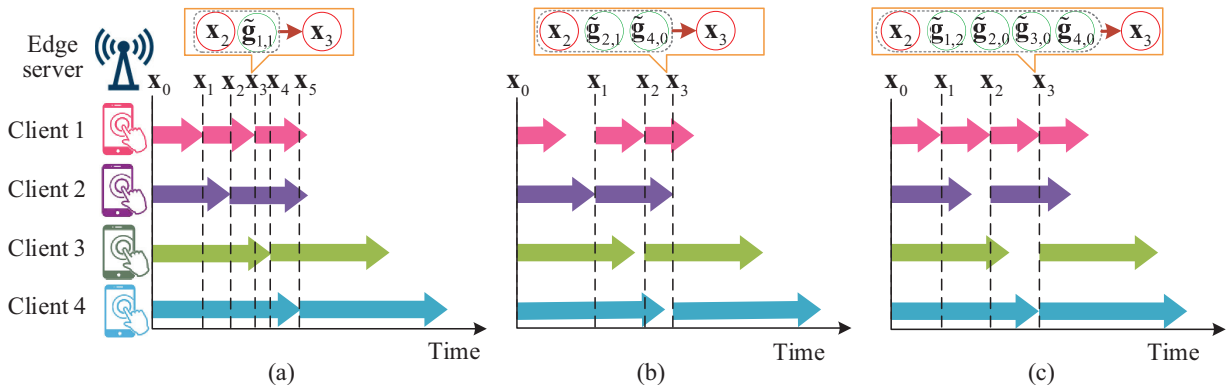


Fig. 1. Comparison of traditional asynchronous FL and ASAFL: (a) Asynchronous FL. (b) Semi-asynchronous FL. (c) The proposed adaptive semi-asynchronous FL.

Algorithm 1 Adaptive Semi-Asynchronous FL

Initialization: The server initials $\{\mathcal{G}_{n,-1} = \mathbf{0} : \forall n \in \mathcal{N}\}$ and the global model $\mathbf{x}_0$, then send the global model to all clients.
Server side:
for $t = 0, 1, \cdots, T-1$ **do**
  Initialize $S_t = \emptyset$, decide the synchronous degree K_t.
  while $|S_t| < K_t$ **do**
    if Receive the gradient from client n **then**
      Updating the gradient array according to (5)
      $S_t = S_t \cup n$
  Update the global model as $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \frac{1}{N} \sum_{n=1}^{N} \mathcal{G}_{n,t}$
  Broadcast $\mathbf{x}_{t+1}$ to the clients in S_t.
Client side:
if Client n received the global model $\mathbf{x}_t$ **then**
  for $i = 0, 1, \cdots, I-1$ **do**
    Execute local training based on (3);
  Calculate local gradient as $\widetilde{\mathbf{g}}_{n,t} = -\frac{1}{\eta}(\mathbf{x}_{n,t}^I - \mathbf{x}_t)$
  Upload $\widetilde{\mathbf{g}}_{n,t}$ to the edge server.

between round t and the round when client n received the last global model. In fact, $\mathbf{g}_{n,t} = \widetilde{\mathbf{g}}_{n,t-\tau_{n,t}}$ and $\widetilde{\mathbf{g}}_{n,t-\tau_{n,t}}$ is computed based on an older version of the global model, i.e., $\mathbf{x}_{t-\tau_{n,t}}$. Once the edge server receiving $|S_t| = K_t$ local gradients, it updates its retained gradients as follows:

$$\mathcal{G}_{n,t} = \begin{cases} \mathbf{g}_{n,t}, & \text{if } n \in S_t, \\ \mathcal{G}_{n,t-1}, & \text{else.} \end{cases} \tag{5}$$

Then, the global model is updated as $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \frac{1}{N} \sum_{n=1}^{N} \mathcal{G}_{n,t}$ and broadcasted to the clients in S_t.

To better explain the proposed ASAFL, we compare it to the conventional asynchronous FL and semi-asynchronous FL in Fig. 1. Consider one server and four clients collaboratively learning a global model. Taking round 2 as an example, in which the global model is updated from $\mathbf{x}_2$ to $\mathbf{x}_3$. The *asynchronous FL* in Fig. 1(a) immediately updates the global model upon receipt of client 1's local gradient $\widetilde{\mathbf{g}}_{1,1}$, i.e., $\mathbf{x}_3 = \mathbf{x}_2 - \eta \widetilde{\mathbf{g}}_{1,1}$. Due to the frequent updating of the global model, asynchronous FL encounters delayed gradients, resulting in unforeseen model accuracy degradation. To relieve the negative effect of delay gradients while mitigating the straggler effect, *semi-asynchronous FL* updates the global model once the edge server collects K $(1 < K < N)$ local gradients, as shown in Fig. 1(b). Specifically, the global model is updated using the local gradients of client 2 and client 4, i.e., $\mathbf{x}_2 = \mathbf{x}_1 - \eta \frac{1}{2}(\widetilde{\mathbf{g}}_{2,1} + \widetilde{\mathbf{g}}_{4,0})$. Although the aforementioned FL approaches effectively enhance learning convergence for FL, the partial client participation in each round can lead to biased global model updating when the data distribution is non-IID. To address this issue, this work calibrates the global update by reusing the received historical local gradients at the edge server from the un-participated clients, as shown in Fig. 1(c). Particularly, upon receiving local gradients from client 1, client 3, and client 4, the edge server updates the global model by incorporating the received gradients from client 1, 3, and client 4 (i.e., $\widetilde{\mathbf{g}}_{1,2}$, $\widetilde{\mathbf{g}}_{3,0}$, and $\widetilde{\mathbf{g}}_{4,0}$), along with the historical gradients of client 2 (i.e., $\widetilde{\mathbf{g}}_{2,0}$), i.e., $\mathbf{x}_2 = \mathbf{x}_1 - \eta \frac{1}{4}(\widetilde{\mathbf{g}}_{1,2} + \widetilde{\mathbf{g}}_{2,0} + \widetilde{\mathbf{g}}_{3,0} + \widetilde{\mathbf{g}}_{4,0})$. In addition, the conventional semi-asynchronous FL approaches fixed the synchronous degree in the learning process, potentially failing to adequately balance learning latency and accuracy. To this end, we propose dynamically adjusting the synchronous degree in each round to achieve better learning performance.

It is worth mentioning that the proposed ASAFL can be further improved by introducing a proper staleness control mechanism to exclude the local gradients with extremely large staleness and prevent model performance degradation. In existing works, e.g., [23], [25], the staleness control is mainly implemented by introducing a staleness threshold τ_0. In the training process, after the per-round global model updating, the edge server distributes the updated global model to clients exceeding the staleness threshold τ_0. Subsequently, these clients halt their local training and restart local training based on the latest received global model. In practical systems, the value of τ_0 should be carefully determined by sufficiently considering the data heterogeneity degree, the system scale, clients' computing capability heterogeneity degree, and other characteristics of the learning system. This is beyond the scope of this work and will be studied in our future work.

B. Learning Latency Model

This subsection characterizes the learning latency for the proposed FL framework, incorporating computation and communication latencies.

- *Computation Latency*: We adopt the central processing unit (CPU) frequency to depict client computation capability. The CPU frequency of client n is denoted as f_n. Let C_n be the required CPU cycles for processing one data sample at client n. The local training latency at client n can be expressed as

$$T_{n,t}^{\text{comp}} = \frac{IA_bC_n}{f_n}, \tag{6}$$

where A_b is the local batch size.
- *Communication Latency*: This work employs the frequency division multiple access technique in the system, where clients share the total bandwidth of B Hz for uploading their local gradients. Let p_n and $h_{n,t}$ represent the transmit power and channel gain of client n in round t, respectively. Denote by $\vartheta_{n,t}$ the ratio of the total bandwidth assigned to client n in the t-th round. We use $\boldsymbol{\vartheta}_t = \{\vartheta_{1,t}, \vartheta_{2,t}, \cdots, \vartheta_{N,t}\}$ to denote the bandwidth allocation policy in round t. Consequently, the transmit rate of client n in round t is expressed as $r_{n,t} = \vartheta_{n,t} B \log_2(1 + \frac{p_n h_{n,t}}{\sigma^2})$, where σ^2 is the variance of Gaussian additive noise. Denote by Q the number of parameters of each local gradient, and each parameter is quantized by q bits. Therefore, the communication latency of client n to transmit its gradient is

$$T_{n,t}^{\text{comm}} = \frac{Qq}{r_{n,t}} = \frac{Qq}{\vartheta_{n,t} B \log_2(1 + \frac{p_n h_{n,t}}{\sigma^2})}. \tag{7}$$

Based on the above models, we characterize the per-round latency in the following. Consider a general round t, where the edge server waits for K_t clients to complete their local training and upload their local gradients. Due to the asynchronous mechanism, clients may have accomplished part of the local training process at the beginning of round t instead of starting training from round t. Thus, the required computation time of client n is usually not equal to $T_{n,t}^{\text{comp}}$. Let $T_{n,t}^{\text{rem}}$ denote the remaining time to complete local training of client n in round t. Thus, the latency of each client n ($n \in S_t$) includes the remaining local training time $T_{n,t}^{\text{rem}}$ and communication time $T_{n,t}^{\text{comm}}$. The latency in round t depends on the slowest client, i.e.,

$$T_t = \max_{n \in S_t} \{T_{n,t}^{\text{rem}} + T_{n,t}^{\text{comm}}\}. \tag{8}$$

Note that the aforementioned analysis does not consider the latency associated with global model broadcasting and updating. This is because the edge server typically possesses larger transmit power and stronger computational capability than the resource-constrained edge clients. Thus, we mainly focus on the performance bottleneck inherent in edge clients.

C. Problem Formulation

This work aims to dynamically adjust the synchronous degree and optimize bandwidth allocation policies in each round to minimize the global loss function while adhering to the total learning latency constraint $T_{\max}$. The optimization problem is formulated as follows:

$$\min_{\{K_t, \boldsymbol{\vartheta}_t\}_{t=0}^{T-1}} \ \mathbb{E}[\mathcal{L}(\mathbf{x}_T)] \tag{9}$$

$$\text{s. t.} \quad \sum_{t=0}^{T-1} T_t \le T_{\max}, \tag{9a}$$

$$\sum_{n=1}^{N} \vartheta_{n,t} \le 1, \forall t, \tag{9b}$$

$$0 \le \vartheta_{n,t} \le 1, \forall n \in \mathcal{N}, \forall t, \tag{9c}$$

$$0 \le K_t \le N, \forall t, \tag{9d}$$

$$K_t \in \mathbb{N}, \forall t, \tag{9e}$$

where (9a) stipulates that the latency of the entire learning process cannot exceed the maximum allowed latency, $T_{\max}$. (9b) and (9c) correspond to the wireless bandwidth restrictions. (9d) and (9e) impose constraints on the synchronous degree, indicating the number of collected gradients in each round cannot surpass the client number.

Problem (9) is intractable to solve as it necessitates deriving the closed-form expression of the global loss function $\mathcal{L}(\mathbf{x}_T)$, which is almost impossible due to the intricacies of the learning process. To tackle this issue, we find an upper bound of $\mathcal{L}(\mathbf{x}_T)$ and transform problem (9) into minimizing this bound for global loss minimization in the following sections. Furthermore, solving problem (9) demands precise time latency information for all clients in all rounds at the beginning of FL. However, such information is unattainable because of unforeseen time-varying channels. To facilitate online synchronous degree control and resource allocation, we transform the long-term decision problem (9) into a deterministic one in each round by leveraging the Lyapunov optimization framework in Section IV.

III. Convergence Analysis

This section investigates the convergence behaviour of ASAFL to explore how the synchronous degree K_t in each round affects its learning performance. For the sake of analysis, we define $\nabla \mathcal{L}_n(\mathbf{x}_{n,t}^i) = \frac{1}{B_n} \sum_{\xi \in \mathcal{B}_n} \ell(\xi; \mathbf{x})$ as the full gradient of client n in the i-th iteration in round t. Let $\mathcal{L}^*$ represent the optimal global loss function. In addition, we impose some assumptions on the loss functions $\mathcal{L}_n(\cdot)$ as follows:

Assumption 1. All the loss functions, $\mathcal{L}_n(\mathbf{x})$ ($\forall n \in \mathcal{N}$), are β-smooth, i.e., for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$, we have $\|\nabla \mathcal{L}_n(\mathbf{x}_1) - \nabla \mathcal{L}_n(\mathbf{x}_2)\| \le \beta \|\mathbf{x}_1 - \mathbf{x}_2\|$, where $\beta > 0$. In addition, $\mathcal{L}_n(\mathbf{x})$ is weak convex, i.e., $\mathcal{L}_n(\mathbf{x}_1) \ge \mathcal{L}_n(\mathbf{x}_2) + \langle \nabla \mathcal{L}_n(\mathbf{x}_2), \mathbf{x}_1 - \mathbf{x}_2 \rangle + \frac{\mu}{2} \|\mathbf{x}_1 - \mathbf{x}_2\|^2$, where $\mu \ge 0$. $\mathcal{L}_n(\mathbf{x})$ is convex if $\mu = 0$, and non-convex if $\mu > 0$ [18].

Assumption 2. The stochastic gradient computed on the uniformly sampled mini-batch data $\mathcal{A}_{n,t}$ of client n is an unbiased estimation of the full gradient $\nabla \mathcal{L}_n(\mathbf{x}_t)$, i.e., $\mathbb{E}\|\nabla \tilde{\mathcal{L}}_n(\mathbf{x}_t)\| = \nabla \mathcal{L}_n(\mathbf{x}_t)$, and the variance is bounded by ς^2, i.e., $\mathbb{E}\|\nabla \tilde{\mathcal{L}}_n(\mathbf{x}_t) - \nabla \mathcal{L}_n(\mathbf{x}_t)\| \le \varsigma^2$.

Assumption 3. The loss function on each client has a global variance bound, i.e., $\frac{1}{N} \sum_{n=1}^{N} \|\nabla \mathcal{L}(\mathbf{x}) - \nabla \mathcal{L}_n(\mathbf{x})\|^2 \le \Gamma^2$.

These assumptions are widely utilized for the convergence analysis of FL, e.g., [15], [27]–[29]. In Assumption 1, the weak convex assumption on loss functions induces the derived analysis result to be effective for both strongly convex and a restricted family of non-convex problems [18]. The convergence analysis in more general non-convex loss function settings is a promising direction, which will be left as a future direction of this work. In addition, the experimentation results in Section V further show that our proposed ASAFL approach based on this assumption works well for practical machine learning models (i.e., VGG-11 and LeNet-5) with non-convex loss functions. In Assumption 3, we utilize a universal bound Γ^2 for quantifying the degree of non-IID. Particularly, $\Gamma^2 = 0$ indicates the data distribution among clients is IID. $\Gamma^2 > 0$ represents the data distributions are non-IID, and large Γ^2 reflects high heterogeneity of the local data distribution.

Lemma 1. *If Assumptions 1 and 2 are satisfied, the deviation of local model from the global model after i iterations is bounded by*

$$\mathbb{E}\left\|\mathbf{x}_{n,t}^{i} - \mathbf{x}_t\right\|^2 \leq 4\eta^2(I-1)(2IG^2+\varsigma^2). \tag{10}$$

Proof. See Appendix A. □

Lemma 2. *If Assumptions 1 and 2 are satisfied, the drift between the global models in two rounds, i.e., t_1 and t_2 ($t_1 > t_2$), is bounded by*

$$\mathbb{E}\left\|\mathbf{x}_{t_1} - \mathbf{x}_{t_2}\right\|^2 \leq 3c_1\eta^2(t_1-t_2)^2, \tag{11}$$

where $c_1 = (I^2+I-1)\varsigma^2 + (I^2+2I(I-1))G^2$

Proof. See Appendix B. □

Based on the above two lemmas, the convergence bound of ASAFL is derived in the following theorem.

Theorem 1. *If Assumptions 1-3 are satisfied, and $\eta \leq \frac{1}{2I\beta}$, the T-round convergence bound of ASAFL is*

$$\mathbb{E}[\mathcal{L}(\mathbf{x}_T) - \mathcal{L}^*] \leq c_2^T\mathbb{E}[\mathcal{L}(\mathbf{x}_0) - \mathcal{L}^*] + c\sum_{t=0}^{T-1} c_2^t + \frac{21\eta^2\beta c_1}{4N}\sum_{t=0}^{T-1} c_2^{T-t-1}\sum_{n=1}^{N}(1-s_{n,t})(\tau_{n,t-1}+1)^2. \tag{12}$$

where $c_2 = 1-\mu\eta I + 2\mu\beta\eta^2 I^2$ *and* $c = 12\eta^2\beta(I-1)IG^2 + (6\eta^2\beta(I-1) + \frac{3}{4}\eta I)\varsigma^2 + 3\eta I\Gamma$.

Proof. See Appendix C. □

According to Theorem 1, the convergence bound of ASAFL is determined by three terms: 1) The initial gap between the global loss and the optimal loss. 2) A system hyperparameters-associated term. 3) The cumulative staleness of clients' local gradients. The first two terms determined by the system hyperparameters and the initial global model are unrelated to bandwidth allocation and synchronous degree control. The third term is closely connected to the synchronous degree control policy. By increasing the synchronous degree K_t in each round to decrease the gradients' staleness, the learning performance of ASAFL can be improved. However, increasing the synchronous degree K_t in each round will increase the corresponding learning latency as the edge server must wait for more clients to transmit their gradients. In the latency-constrained FL system, one should carefully control the synchronous degree K_t in each round to achieve balance between the latency and accuracy.

Remark 1. Note that $0 < c_2 < 1$ due to $\eta \leq \frac{1}{2I\beta}$. As t increases, c_2^{T-t-1} also keeps increasing. This indicates that the impact of gradient staleness intensifies throughout the learning process, with the staleness in later rounds exerting a more substantial influence on learning performance than that in the early rounds. Consequently, under the given overall training time, one should allocate more time to the later rounds to increase the synchronous degree for improving the learning performance.

IV. Adaptive Semi-Asynchronous Federated Learning

In this section, we develop an adaptive asynchronous degree control and bandwidth allocation algorithm to solve problem (9) online. For this purpose, we convert problem (9) to optimizing the convergence bound obtained in Section III. Then, we derive the bandwidth allocation policies and develop a low-time complexity synchronous degree control algorithm to enhance the FL performance.

A. Problem Analysis

Following Theorem 1, minimizing the final term on the right-hand-side (RHS) of (12) leads to global loss minimization. However, the direct minimization of $\sum_{t=0}^{T-1}(1-\mu\eta I + 2\mu\beta\eta^2 I^2)^{T-t-1}\sum_{n=1}^{N}(1-s_{n,t})(\tau_{n,t-1}+1)^2$ presents an intractable challenge due to the presence of unknown parameters, such as the Lipschitz constant β and μ. The value of β and μ are closely related the machine learning models and loss function characteristics. As highlighted in [30], computing the precise value of β and μ for deep learning models is impractical, even in the context of two-layer neural networks. Fortunately, our analysis result in Remark 1 elucidates that allocating more time to the later rounds helps improve the learning performance when the overall learning time is constrained. Building upon this insight, we introduce variables λ_t ($\forall t$) to represent the weight of gradients' staleness in round t to quantify the impact of staleness in each round. Furthermore, we define the objective function as the weighted staleness of local gradients across the learning process, i.e., $\sum_{t=0}^{T-1}\lambda_t\sum_{n=1}^{N}s_{n,t}(\tau_{n,t-1}+1)^2$, and maximize it to achieve global loss minimization. Consequently, we reformulate problem (9) into the following problem:

$$\max_{\{K_t,\boldsymbol{\vartheta}_t\}_{t=0}^{T-1}} \sum_{t=0}^{T-1}\lambda_t\sum_{n=1}^{N}s_{n,t}(\tau_{n,t-1}+1)^2 \tag{13}$$
$$\text{s. t.} \quad (9a), (9b), (9c), (9d), (9e).$$

In problem (13), the weight parameter λ_t is crucial for the learning performance. In the following, we provide some guidelines for selecting λ_t:

1) Based on Remark 1, λ_t should be ascending with respect to the round index t since the gradients' staleness in

later rounds exerts a more significant influence on learning performance than that in the early rounds. This guideline is verified in Section V-B.

2) Based on Theorem 1, the weight of gradients' staleness in each round t is c_2^{T-t-1} $(0 < c_2 < 1)$, which shows a convex shape with respect to t. Thus, in the continuous number domain, λ_t ought to be a convex function with respect to t.

3) The values of λ_t in the early rounds should not be too small. Given a fixed learning latency, overly small values may lead to quite small synchronous degrees and only a few clients participate in the global model updating. Hence, the global model is not well-trained in the earlier rounds. This may induce learning performance degradation.

According to these guidelines, we select $\lambda_t = \frac{1}{T-t}$ as the weighting policy in this work. The effectiveness of these guidelines for selecting λ_t is verified in Section V-D.

Directly solving problem (13) requires optimally dividing the latency budget into each round, which is impractical since it requires precise knowledge of the channel condition and computation schedule of all clients throughout the entire learning process. To enable online dynamic synchronous degree control, we adopt the Lyapunov optimization framework to handle the dependencies between rounds. Thus, we define a virtual queue q_t, which evolves as

$$q_{t+1} = \max\Big(q_t + T_t - \frac{T_{\max}}{T}, 0\Big), \tag{14}$$

with a initial value $q_0 = 0$. Taking inspiration from the drift-plus-penalty algorithm [31], the long-term latency constraint (9a) and the objective function (13) can be transformed into Lyapunov drift-plus-penalty function. Then, we reformulate problem (13) into the following problem to minimize the drift-plus-penalty function and facilitate online synchronous degree control and bandwidth allocation:

$$\begin{aligned} \min_{K_t, \boldsymbol{\vartheta}_t} \quad & -\mu\lambda_t \sum_{n=1}^{N} s_{n,t}(\tau_{n,t-1}+1)^2 + q_t T_t \\ \text{s. t.} \quad & \text{(9b)}, \text{(9c)}, \text{(9d)}, \text{(9e)}. \end{aligned} \tag{15}$$

In problem (15), μ is a weight factor that balances the latency and accuracy. Increasing μ would emphasize accuracy by immolating latency and vice versa.

B. Optimal Bandwidth Allocation

For any given synchronous degree control policy K_t in round t, the participating client set S_t can be determined, which is the first K_t clients that accomplish their local training in round t. With S_t, we decompose the optimal bandwidth allocation problem as

$$\begin{aligned} \min_{\boldsymbol{\vartheta}_t} \quad & \max_{n \in S_t} \{T_{n,t}^{\mathrm{rem}} + T_{n,t}^{\mathrm{comm}}\} \\ \text{s. t.} \quad & \text{(9b)}, \text{(9c)}.. \end{aligned} \tag{16}$$

The optimization problem (16) is a convex [32], its optimal solution can be obtained through the following lemma.

Lemma 3. *The optimal bandwidth allocation policy of problem* (16) *satisfies the following condition:*

$$\vartheta_{n,t} = \frac{Qq}{(T_t^* - T_{n,t}^{\mathrm{rem}}) B \log_2(1 + \frac{p_{n,t} h_{n,t}}{\sigma^2})}, \tag{17}$$

where T_t^* is the optimal latency for the participating client set S_t in round t and satisfy $\sum_{n \in S_t} \vartheta_{n,t} = 1$.

Proof. See Appendix D. □

In Lemma 3, the optimal bandwidth allocation decision includes an unknown variable T_t^*. From (17), it is evident that $\vartheta_{k,t}$ is a monotonically decreasing function concerning T_t^*. Therefore, the binary search method can be utilized to find the optimal bandwidth allocation policy. To facilitate this, we derive the upper and lower bounds of T_t^* in the below. To derive the lower bound for T_t^*, we utilize the fact that the minimum bandwidth ratio allocated to clients is below $\frac{1}{K_t}$, i.e., $\min_{n \in S_t} \vartheta_{n,t}^* \le \frac{1}{K_t}$. Thus, we have

$$\frac{\min_{n \in S_t} \frac{Qq}{B \log_2(1 + \frac{p_{k,t} h_{k,t}}{\sigma^2})}}{\max_{n \in S_t}(T_t^* - T_{n,t}^{\mathrm{rem}})} \le \frac{1}{K_t}. \tag{18}$$

Hence, the lower bound of T_t^* is given by

$$T_t^{\mathrm{low}} = \min_{n \in S_t} \frac{K_t Qq}{B \log_2(1 + \frac{p_{k,t} h_{k,t}}{\sigma^2})} + \min_{n \in S_t} T_{n,t}^{\mathrm{rem}}. \tag{19}$$

To determine the upper bound for T_t^*, we consider $\max_{n \in S_t} \vartheta_{n,t}^* \ge \frac{1}{K_t}$. The upper bound is derived using a comparable approach to the one employed for the lower bound and is therefore skipped for conciseness, it is given by

$$T_t^{\mathrm{up}} = \max_{n \in S_t} \frac{K_t Qq}{B \log_2(1 + \frac{p_{k,t} h_{k,t}}{\sigma^2})} + \max_{n \in S_t} T_{n,t}^{\mathrm{rem}}. \tag{20}$$

Building upon the above lower and upper bounds, we utilize the binary search method to find the optimal T_t^* and determine the optimal bandwidth allocation policy, the specific steps of which are detailed in Algorithm 2. The algorithm iteratively halves the search region, terminating when the precision criterion ε is met. Thus, the complexity of Algorithm 2 is $\mathcal{O}(\log_2 \frac{T_t^{\mathrm{up}} - T_t^{\mathrm{low}}}{\varepsilon})$.

C. Synchronous Degree Control

Based on the preceding analysis, the optimal bandwidth allocation for any client set S_t can be solved. Note that under the semi-asynchronous aggregation mechanism, for round t with a given synchronous degree K_t $(1 \le K_t \le N)$, the first K_t clients completed their local training will participate in the current-round global model update. That is, the optimal latency and bandwidth allocation policy can be obtained for any given K_t in round t. Problem (15) is to select an appropriate synchronous degree to minimize the drift-plus-penalty, i.e., $-\mu\lambda_t \sum_{n=1}^{N} s_{n,t}(\tau_{n,t-1}+1)^2 + q_t T_t$. Therefore, in each round t, we sort the clients in ascending order based on their remaining computation time. Here various sorting algorithms, such as Heapsort or Mergesort, can be adopted with a worst-case complexity of $\mathcal{O}(N \log N)$. Let $\widetilde{\mathcal{N}}$ denote the sorted client set. Subsequently, we solve the synchronous

Algorithm 2 Bandwidth Allocation

Input S_t, the precision factor $\varepsilon > 0$.
Compute the lower bound (T_t^{low}) and upper bound (T_t^{up}) based on (19) and (20), respectively.
repeat
  Set $T_{\text{mid}} = (T_t^{\text{low}} + T_t^{\text{up}})/2$.
  For each client $n \in \boldsymbol{S}_t$, compute the required bandwidth allocation ratio $\vartheta_{n,t}(T_{\text{mid}})$ based on (17).
  Compute $\sum_{n\in S_t} \vartheta_{n,t}(T_{\text{mid}})$.
  if $\sum_{n\in S_t} \vartheta_{n,t}(T_{\text{mid}}) > 1$ **then**
    Set $T_t^{\text{low}} = T_{\text{mid}}$ and $T_t^{\text{up}} = T_t^{\text{up}}$.
  else if $0 < \sum_{n\in S_t} \vartheta_{n,t}(T_{\text{mid}}) < 1 - \varepsilon$ **then**
    Set $T_t^{\text{low}} = T_t^{\text{low}}$ and $T_t^{\text{up}} = T_{\text{mid}}$.
  else
    Break the circulation.
until $|T_t^{\text{up}} - T_t^{\text{low}}| < \varepsilon$
return The optimal latency $T_t^* = T_{\text{mid}}$ and the optimal bandwidth allocation policy $\boldsymbol{\vartheta}_t$.

Algorithm 3 Adaptive Synchronous Degree Control

Input the virtual queue length q_t and λ_t, initialize μ.
Sort clients according to $T_{n,t}^{\text{rem}}$ ascendingly to acquire the sorted client set $\widetilde{N}$.
for $K_t = 1, 2, \cdots, N$ **do**
  $S = \widetilde{N}[1 : K_t]$
  Solve the optimal latency and bandwidth allocation policy of S by Algorithm 2, i.e., $T_t^*(S)$, $\boldsymbol{\vartheta}_t^*$.
  Calculate the drift-plus-penalty of K_t, i.e., $\mathcal{Y}(K_t) = -\mu\lambda_t \sum_{n=1}^{N} s_{n,t}(\tau_{n,t-1}+1)^2 + q_t T_t^*(S)$.
Find the synchronous degree $K_t^* = \arg\min_{1\le K_t\le N} \mathcal{Y}(K_t)$, the participating client set $S = \widetilde{N}[1 : K_t]$, and the corresponding bandwidth allocation policy $\boldsymbol{\vartheta}_t^*$
return The synchronous degree control policy K_t^*, bandwidth allocation policy $\boldsymbol{\vartheta}_t^*$

degree control policy by gradually increasing K_t from 1 to N. For each potential K_t, the first K_t clients in $\widetilde{\mathcal{N}}$ are selected to form the participating client set, and Algorithm 2 is executed to compute the optimal round latency. Following that, we substitute the optimal round latency into the objective function (15) to compute the drift-plus-penalty value, denoted as $\mathcal{Y}(K_t)$. The optimal synchronous control policy is derived by comparing the drift-plus-penalty value among all potential K_t, i.e., $K_t^* = \arg\min_{1\le K_t\le N} \mathcal{Y}(K_t)$. We formally describe the proposed synchronous degree control approach in Algorithm 3. Algorithm 3 performs N times Algorithm 2 to calculate the synchronous degree control policy and has a polynomial time complexity $\mathcal{O}(N\log N + N\log_2 \frac{T_t^{\text{up}} - T_t^{\text{low}}}{\varepsilon})$.

D. Optimality and Implementation

This subsection discusses the optimality and implementation of the proposed approach. To characterize the performance of the proposed adaptive synchronous degree control algorithm, we compare it with its optimal offline counterpart which is in fact problem (13), regarding the computation and communication time in the entire learning process as a predefined hyperparameter sequence. Let $s_{n,t}^*$ be the optimal client participation indicators obtained by solving the above problem. The performance guarantee of the proposed adaptive synchronous degree control algorithm is shown in the following proposition.

Proposition 1. *Compared to the offline counterpart optimal solution, the cumulative loss of the proposed adaptive asynchronous control algorithm is bounded by*

$$\sum_{t=0}^{T-1} \lambda_t \sum_{n=1}^{N} s_{n,t}(\tau_{n,t-1}+1)^2 \ge -\frac{T^2\chi^2}{2\mu} + \sum_{t=0}^{T-1} \lambda_t \sum_{n=1}^{N} s_{n,t}^*(\tau_{n,t-1}+1)^2, \quad (21)$$

and the latency of the proposed algorithm is bounded by

$$\sum_{t=0}^{T-1} T_t \le T_{\max} + \sqrt{T\chi^2 + 2\mu \sum_{t=0}^{T-1} \lambda_t \sum_{n=1}^{N} (\tau_{n,t-1}+1)^2}. \quad (22)$$

where $\chi = \max_t \left\{ \left| T_t - \frac{T_{\max}}{T} \right| \right\}$.

Proof. See Appendix E. □

According to Proposition 1, we have the following remark:

Remark 2. For the proposed adaptive synchronous degree control approach, we have: 1) the latency constraint is approximately met with the $\mathcal{O}(\sqrt{\mu})$-bounded factor, and 2) the proposed adaptive synchronous degree control algorithm achieves $\mathcal{O}(1/\mu)$-optimality concerning its optimal offline counterpart solution. Therefore, the proposed adaptive synchronous degree control algorithm exhibits an $\mathcal{O}(\sqrt{\mu}, 1/\mu)$ latency-learning trade-off. In particular, when μ is increased, greater priority is given to decreasing the gradients' staleness for the purpose of enhancing learning performance, even if it results in higher latency, and conversely.

In wireless networks, enabling the deployment of the proposed ASAFL necessitates the server to retain all clients' historical gradient information. Consequently, the server's memory size demand is contingent upon the model size and the client number. This poses limitations on the scalability of ASAFL, as the growing number of clients may lead to an exhaustion of the edge server's memory space. To address this challenge, we devise a variant ASAFL which is equivalent to the proposed ASAFL. In the variant ASAFL, the edge server only retains one gradient array $\bar{\mathcal{G}}_t$ for caching the aggregated gradient, and each client n ($n \in \mathcal{N}$) independently retains a gradient array, denoted as $\mathcal{G}_{n,t}$, to cache its latest gradient information. Following that, we make two following changes in Algorithm 1 as follows: 1) At the client side, each client n upload the difference between its current and prior gradients, i.e., $\Delta_{n,t} = \mathbf{g}_{n,t} - \mathcal{G}_{n,t-1}$, to the server, as differing from uploading current local gradient $\mathbf{g}_{n,t}$ in Algorithm 1. 2) At the server side, after receiving the client gradient information, it updates the retained gradient array according to $\bar{\mathcal{G}}_t = \bar{\mathcal{G}}_{t-1} + \frac{1}{N}\sum_{n\in S_t} \Delta_{n,t}$. Then, the global model is updated as $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\bar{\mathcal{G}}_t$. Through the above changes, we mitigate the server's memory bottleneck and obtain an equivalent implementation of the proposed ASAFL in Algorithm 1, which is proved in the following theorem.

Theorem 2. *The variant ASAFL is equivalent to the proposed ASAFL in Algorithm 1.*

Proof. Firstly, for the retained $\bar{\mathcal{G}}_t$ at the server, we have

$$\begin{aligned}\bar{\mathcal{G}}_t &= \bar{\mathcal{G}}_{t-1} + \frac{1}{N}\sum\nolimits_{n=1}^{N} s_{n,t}\Delta_{n,t} \\ &= \bar{\mathcal{G}}_{t-1} + \frac{1}{N}\sum\nolimits_{n=1}^{N} s_{n,t}(\mathbf{g}_{n,t} - \mathcal{G}_{n,t-1}) \\ &= \bar{\mathcal{G}}_{t-1} + \frac{1}{N}\sum\nolimits_{n=1}^{N} (\mathcal{G}_{n,t} - \mathcal{G}_{n,t-1}). \end{aligned} \tag{23}$$

Note that, at the start of FL, we initialize the retained gradient arrays at the edge server and clients as $\mathbf{0}$, i.e., $\bar{\mathcal{G}}_{-1} = \mathbf{0}$ and $\mathcal{G}_{n,-1} = \mathbf{0} (n \in \mathcal{N})$. When $t = 0$, we have

$$\bar{\mathcal{G}}_0 = \bar{\mathcal{G}}_{-1} + \frac{1}{N}\sum_{n=1}^{N}(\mathcal{G}_{n,0} - \mathcal{G}_{n,-1}) = \frac{1}{N}\sum_{n=1}^{N}\mathcal{G}_{n,0}. \tag{24}$$

When $t = 1$, we have

$$\begin{aligned}\bar{\mathcal{G}}_1 &= \bar{\mathcal{G}}_0 + \frac{1}{N}\sum\nolimits_{n=1}^{N}(\mathcal{G}_{n,1} - \mathcal{G}_{n,0}) \\ &= \frac{1}{N}\sum\nolimits_{n=1}^{N}\mathcal{G}_{n,1} + \bar{\mathcal{G}}_0 - \frac{1}{N}\sum\nolimits_{n=1}^{N}\mathcal{G}_{n,0} \\ &= \frac{1}{N}\sum\nolimits_{n=1}^{N}\mathcal{G}_{n,1}. \end{aligned} \tag{25}$$

Similarly, for $t > 1$, we can derive that $\bar{\mathcal{G}}_t = \frac{1}{N}\sum_{n=1}^{N}\mathcal{G}_{n,t}$. Thus, the updated global model through the variant ASAFL is $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\bar{\mathcal{G}}_t = \mathbf{x}_t - \eta\frac{1}{N}\sum_{n=1}^{N}\mathcal{G}_{n,t}$, which equals to the updated global model by the ASAFL in Algorithm 1. Thus, the variant ASAFL is equivalent to ASAFL. □

According to Theorem 2, one can implement the above variant ASAFL to achieve an equivalent learning process with the proposed ASAFL in Algorithm 1 in practical wireless networks. It is worth mentioning that the computation costs, communication costs, and the learned global model of the variant ASAFL are the same as the ASAFL approach in Algorithm 1. The variant ASAFL only requires the edge server to retain a single gradient array, whereas ASAFL necessitates the retention of all K users' gradient arrays. Thus, compared to Algorithm 1, the variant ASAFL design can save K times memory size. For clarity, we summarize the detailed steps of the variant ASAFL in Algorithm 4.

Algorithm 4 Variant ASAFL algorithm

Initialization: The server initials its gradient array $\bar{\mathcal{G}}_t = \mathbf{0}$ and the global model $\mathbf{x}_0$, each client n ($n \in \mathcal{N}$) initializes its gradient array $\mathcal{G}_t = \mathbf{0}$.
Server side:
for $t = 0, 1, \cdots, T-1$ **do**
  Initialize $S_t = \emptyset$, decide the synchronous degree K_t.
  while $|S_t| < K_t$ **do**
    if Receive the gradient difference $\Delta_{n,t}$ from client n **then**
      Updating the gradient array according to $\bar{\mathcal{G}}_t = \bar{\mathcal{G}}_{t-1} + \frac{1}{N}\sum_{n\in S_t}\Delta_{n,t}$
      $S_t = S_t \cup n$
  Update the global model as $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\bar{\mathcal{G}}_t$
  Broadcast $\mathbf{x}_{t+1}$ to the clients in S_t.
Client side:
if Client n received the global model $\mathbf{x}_t$ **then**
  for $i = 0, 1, \cdots, I-1$ **do**
    Execute local training based on (3);
  Calculate local gradient as $\widetilde{\mathbf{g}}_{n,t} = -\frac{1}{\eta}(\mathbf{x}_{n,t}^I - \mathbf{x}_t)$
  Upload $\Delta_{n,t} = \mathbf{g}_{n,t} - \mathcal{G}_{n,t-1}$ to the edge server.

V. Numerical Results

This section evaluates the efficacy of the proposed ASAFL. Unless otherwise identified, the defaulting experimental parameters are summarised in Table II. The parameters chosen in the simulation are based on the parameter settings of a typical wireless FL system such as [12], [16], [17], [24]–[26], [33]. We consider a circular region with a radius of 500m, where the edge server is positioned at the centre, and N clients are uniformly distributed. The channel gain of each client n ($n \in \mathcal{N}$) is modelled as $h_{n,t} = h_0\rho_n(t)d_n^{-2}$, where h_0 is the path loss constant, $\rho_{n,t} \sim \exp(1)$ is the small-scale fading channel power gain of client n in round t, d_n is the distance from client n to the edge server. The CPU frequency of each client n ($n \in \mathcal{N}$) is randomly selected from $[0.1, 0.8]$ GHz with interval 0.1 GHz. We evaluate the learning performance of the proposed ASAFL using the MNIST and CIFAR-10 datasets. The network architectures used for the learning tasks on these three datasets are summarized in Table III, where 'F' denotes the fully connected module, 'C' denotes the convolution module, 'M' denotes the 2×2 max-pooling layer, and the number indicates the number of neurons in fully connected layers or filters in convolution layers. For MNIST, we train a LeNet-5 [34], which requires 497098 FLOPs to process one data sample. For CIFAR-10, we train a VGG-11 model [10], which requires 305662730 FLOPs to process one data sample. Each CPU cycle is able to process 4 FLOPs. For the dataset splitting, we follow prior arts [35] to model the non-IID data distribution using a Dirichlet distribution Dir(α), in which a smaller α indicates higher data heterogeneity.

TABLE II
Parameter Settings

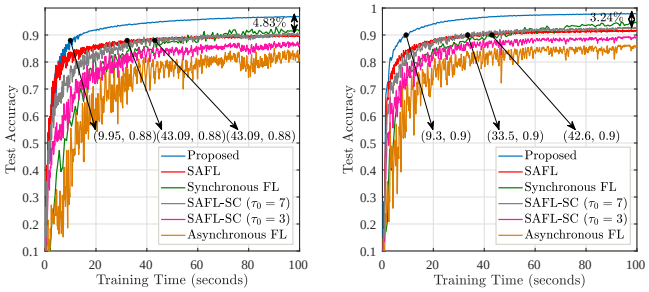
Parameter	Value	Parameter	Value
N	50	B	10 MHz
p_n ($\forall n \in \mathcal{N}$)	10 dBm	σ^2	10^{-12} W
q	32 bits	h_0	-30 dBm
I	8	A_b	128
η	0.01	α	0.01
Q(LeNet-5)	19670	C_n(LeNet-5)	124,274.5
Q(VGG-11)	9354378	C_n(VGG-11)	76,415,682.5
$T_{\max}$(LeNet-5)	$0.1T$ s	$T_{\max}$(VGG-11)	$10T$ s

TABLE III
Network Architecture for the Classification Model

Dataset	Model Name	Model Architecture
MNIST	LeNet-5 [34]	C: [6, M, 16, M]
		F: [256, 64, 10]
CIFAR-100	VGG-11 [10]	C: VGG-11 feature extractor
		F: [512, 256, 10]

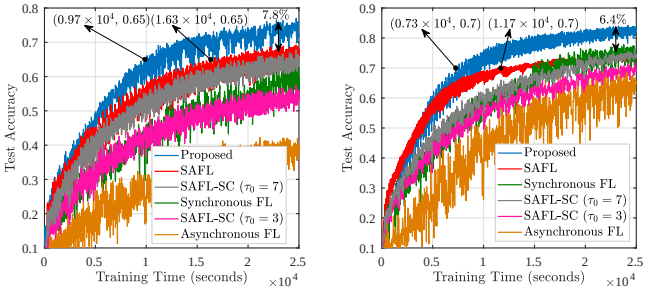
A. Effectiveness of Gradient Calibration

This subsection demonstrates the advantages of the proposed gradient calibration method through contrasting it to the following algorithms under different data heterogeneous



(a) (b)

Fig. 2. Comparison of various FL algorithms on MNIST dataset: (a) $\alpha = 0.01$ (b) $\alpha = 0.1$.



(a) (b)

Fig. 3. Comparison of various FL algorithms on CIFAR-10 dataset: (a) $\alpha = 0.01$ (b) $\alpha = 0.1$.

degrees on MNIST and CIFAR-10 datasets. 1) Asynchronous FL: The edge server immediately updates the global model once it receives one local update from clients. 2) Synchronous FL [12], [17]: The server aggregates K local gradients in a synchronous way for the global model updating per round. 3) Semi-Asynchronous FL (SAFL) [24]: The edge server updates the global model when it receives K local gradients from clients, and clients perform their local training asynchronously. 4) Semi-Asynchronous FL with staleness control (SAFL-SC) [25]: The training process of SAFA-SC closely resembles that of SAFA, with one notable distinction being the introduction of a staleness threshold hyperparameter τ_0. In the training process, after the per-round global model updating, the edge server distributes the updated global model to clients exceeding the staleness threshold τ_0. Subsequently, these clients halt their local training and restart local training based on the latest received global model. Due to page limits, we only present the results of $K = 10$ [12], [17], [25], and the results on other values of K show similar conclusion with $K = 10$.

Fig. 2 compares the above FL schemes on MNIST dataset. In Fig. 2(a), we configure the data heterogeneity parameter as $\alpha = 0.01$. The results show that the semi-asynchronous approach with the proposed gradient calibration attains superior accuracy and convergence speed than the benchmarks. Specifically, our proposed approach improves at least 4.83% accuracy compared to the baseline schemes. When aiming for an accuracy of 88%, our approach requires 9.95s to achieve the target, while the Semi-Asynchronous FL requires 43.09s. That is, the proposed approach can provide a 4.3x speed up for the convergence. The performance gain of the proposed approach comes from the gradient calibration mechanism, which adopts the absent clients' historical gradients to calibrate the global model update and mitigate the bias in the global aggregation. However, the three benchmarks update the global model only based on partial clients' gradients, which may induce biased global model updating under the data heterogeneity scenarios. Fig. 2(b) evaluates the proposed approach and the benchmarks by configuring $\alpha = 0.1$, which induced smaller data heterogeneity than Fig. 2(a). It is observed that all the FL algorithms under $a = 0.1$ achieve better learning performance than that under $\alpha = 0.01$. The reason is that increased data heterogeneity weakens the generalization capability of the global model, thereby worsening the degradation in accuracy. Moreover, the proposed approach improves accuracy by 3.24% compared to the benchmark schemes, and it is able to provide 3.6x speed up to achieve 90% test accuracy.

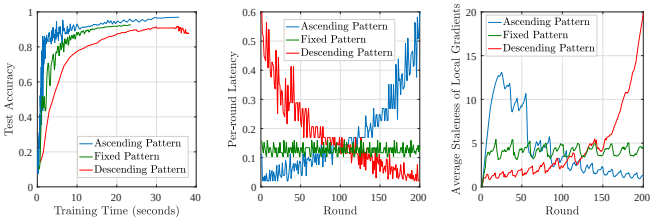
One interesting phenomenon in Fig. 2 is that ASFL-SC with a large staleness threshold ($\tau_0 = 7$) performs better than that with a small staleness threshold ($\tau_0 = 3$). This may be because that small τ_0 forces clients with weak computation and communication capabilities frequency halt their current local training and restart training based on the latest received global model. Thus, these weak clients are less involved in the global model updating and induce learning performance degradation. In addition, ASFL-SC with a large staleness threshold ($\tau_0 = 7$) slightly outperforms ASFL under the data heterogeneity setting of $\alpha = 0.1$ while performing worse than ASFL under higher data heterogeneity setting of $\alpha = 0.01$. This reveals that the staleness control may not significantly impact the learning accuracy when the data distribution is highly non-IID. Therefore, in real-world systems, it becomes imperative for the staleness control policy to thoroughly account for factors such as the degree of data heterogeneity and other system variables as we discussed in Section II-A. Due to page limits, we leave this promising direction as our future works.

A similar comparison is conducted using the CIFAR-10 dataset, as depicted in Fig. 3. The results show that the proposed approach with gradient calibration surpasses the baselines in both accuracy and convergence speed. Notably, a more pronounced performance gain is observed for the proposed approach on this intricate dataset compared to the MNIST dataset in Fig. 2. In Fig. 3(a), we set $\alpha = 0.01$, where the proposed approach achieves a notable 7.8% accuracy boosting compared to the benchmarks. In Fig. 3(b), α is set to 0.1, leading to a 6.4% accuracy improvement for the proposed approach. In addition, similar to the results on the MNIST data set, it is observed that the proposed approach performs better than the ASFL-SC approach and the effectiveness of the staleness control policy is highly related to the data heterogeneity degree.

In Table IV, we present the required time necessary to achieve a target accuracy for both the proposed approach and the top-performing baseline algorithm. Specifically, on the CIFAR-10 dataset, when $\alpha = 0.01$, the proposed approach spends only 0.97×10^4s to achieve 65% accuracy, while semi-asynchronous FL (the best benchmark) requires 1.63×10^4s.

TABLE IV
REQUIRED TIME TO REACH A TARGET ACCURACY

Dataset	α	Target	Proposed	Best Baseline	Speed Up
MNIST	0.01	88%	9.95s	43.09s	4.3x
	0.1	90%	9.3s	33.5s	3.6x
CIFAR-10	0.01	65%	0.97×10^4s	1.63×10^4s	1.7x
	0.1	70%	0.73×10^4s	1.17×10^4s	1.6x



(a) (b) (c)

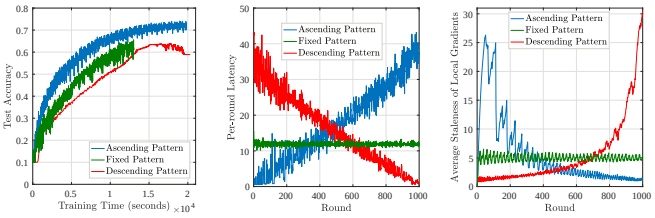
Fig. 4. Comparison of synchronous degree control patterns on MNIST dataset: (a) Test accuracy. (b) Latency. (c) Average staleness of local gradients.

That is, the proposed approach achieves 1.7x speed up in comparison to the benchmarks. When $\alpha = 0.1$, the proposed approach attains 1.6x speed up to reach 70% accuracy compared to the benchmarks.

B. Impacts of Synchronous Degree Control Policy

This subsection validates the theoretical results presented in Remark 1 by examining the effects of the following temporal synchronous degree control patterns on the learning performance of ASAFL: 1) Fixed Synchronous Degree Pattern: The synchronous degree in each round is consistently set to 10, indicating that the edge server waits for ten clients' local gradients to update the global model. 2) Ascending Synchronous Degree Pattern: This pattern gradually increases the synchronous degree from 1 to 20, resulting in an average synchronous degree of 10 in each round. 3) Descending Synchronous Degree Pattern: Conversely, in this pattern, the synchronous degree declines from 20 to 1, maintaining the average synchronous degree in each round to 10.

In Fig. 4, we compare the performance of the above three synchronous degree control policies on MNIST dataset. In Fig. 4(a), we can see that the ascending pattern outperforms the other two patterns in both accuracy and convergence speed. The latent reason for this phenomenon can be found in Fig. 4(b) and Fig. 4(c). Fig. 4(b) shows that the per-round latency of the Ascending pattern exhibits a gradual increase since the ascending in synchronous degree compels the edge server to await more clients for global model updating. This facilitates fewer clients to be involved in the early global model updating stages to mitigate the waiting time and accelerate the convergence. As the learning process progresses, more clients engage in the global model updating process to reduce the convergence error and improve the final learning accuracy. In Fig. 4(c), the average staleness of local gradients in the ascending pattern exhibits an initial increase followed by a subsequent decrease during the learning process. It is worth



(a) (b) (c)

Fig. 5. Comparison of synchronous degree control patterns on CIFAR-10 dataset: (a) Test accuracy. (b) Latency. (c) Average staleness of local gradients.

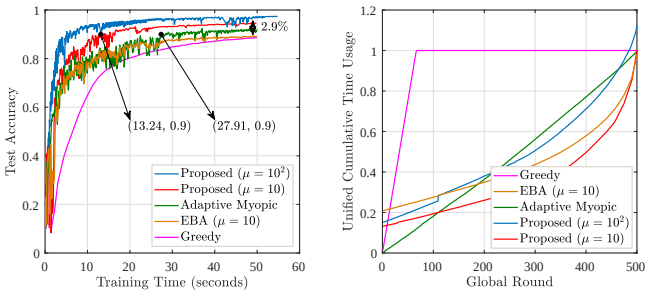
noting that the rise in staleness in the early learning stages is because the initial staleness is equal to zero. According to Theorem 1, the gradient staleness in the later learning rounds has a more distinct impact on the learning performance than that in the early rounds. Consequently, the proposed approach, which assigns a higher synchronous degree in later rounds to mitigate corresponding staleness, achieves superior convergence error reduction and enhanced learning performance compared to the benchmarks. This aligns with our theoretical findings illustrated in Remark 1.

Similarly, the evaluations based on CIFAR-10 in Fig. 5 provide additional validation of our theoretical insights outlined in Remark 1. It is observed that the ascending synchronous degree control pattern achieves better performance than the other two patterns. Thus, in practical FL systems with a given learning time budget, it is recommended to incrementally increase the synchronous degree to attain fast learning convergence and high model accuracy.

C. Overall Effectiveness

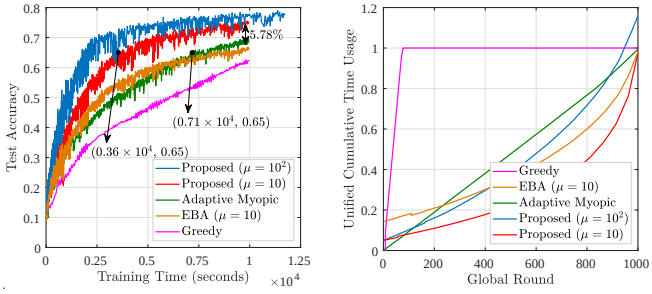
This subsection assesses the proposed ASAFL by comparing it with the following benchmarks: 1) Greedy synchronous degree control: In each round t, the available learning time is the remaining time, i.e., $T_{\max}-\sum_{x=1}^{t-1} T_t$. 2) Adaptive Myopic synchronous degree control: In each round t, the available time is calculated as the residual time divided by the residual round number, i.e., $\frac{T_{\max}-\sum_{x=1}^{t-1} T_t}{T-t+1}$. 3) Equal bandwidth allocation policy (EBA): In each round, the synchronous degree is determined by the proposed approach, while the bandwidth is equally allocated to each selected client. For the proposed approach, we set $\lambda_t = \frac{1}{T-t}$ $(\forall t \in \{0, 1, T-1\})$.

Fig. 6 compares these synchronous degree control approaches on the MNIST dataset. Fig. 6(a) shows that the proposed algorithm surpasses the benchmarks in terms of convergence speed and learning accuracy. Fig. 6(b) depicts the cumulative time usage of various synchronous degree control algorithms. It is observed that the growth speed of the time usage of the proposed approach under $\mu = 10$ and $\mu = 100$ increases along with the learning process. This indicates that the proposed approach has higher time usage in the later rounds than in the early rounds. Thus, the proposed approach acquires superior learning accuracy. Particularly, the proposed approach with $\mu = 10$ exhibits an equivalent time usage to the Adaptive Myopic algorithm. Both of them adhere to the time constraint, with the unified time usage being less than 1 at the end of the



(a) (b)

Fig. 6. Comparison of different synchronous degree control algorithms on MNIST dataset: (a) Test accuracy (b) Unified cumulative time usage.

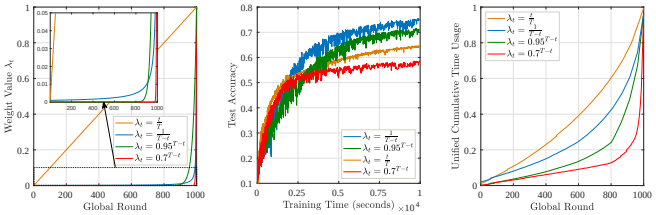


(a) (b)

Fig. 7. Comparison of different synchronous degree control algorithms on CIFAR-10 dataset: (a) Test accuracy (b) Unified cumulative time usage.

training process. However, the proposed approach with $\mu = 10$ significantly outperforms the Adaptive Myopic method, i.e., enhancing accuracy by 2.9%, as well as achieving 2.1x speed up when aiming for an accuracy of 90%. In addition, the proposed approach significantly surpasses the equal bandwidth allocation approach. This verifies the importance of optimizing the bandwidth allocation policy, which is able to save training time and involve more clients to participate in the learning process.

Fig. 7 further juxtaposes the performance of the synchronous degree control algorithms on the CIFAR-10 dataset. Similarly, the proposed approach consistently outperforms the benchmarks. Fig. 7(b) illustrates that the proposed scheme allocates more time in the later rounds than the benchmarks. This means that the proposed approach has a larger synchronous degree in the later rounds. Following Remark 1, this leads to the superior performance of the proposed approach compared to the benchmarks. In particular, the proposed approach with $\mu = 10$ consumes the same learning time as Myopic, while achieving better performance. Specifically, the proposed approach boosts accuracy by 5.78% compared to the Myopic method, and it provides 1.9x speed up to attain 65% accuracy. Similar to the results on the MNIST dataset, the proposed approach outperforms the EBA approach. This further illustrates the necessity to optimize the bandwidth resources to save training time and encourage more clients to join the learning process.



(a) (b) (c)

Fig. 8. Comparison of different weighting policies of λ_t: (a) Weight value λ_t over global rounds. (b) Test accuracy. (c) Unified cumulative time usage.

D. Weighting Parameter Tuning

This subsection compares the following weighting policies on the CIFAR-10 dataset to verify the guidelines for selecting the weighting parameter λ in Section IV-A. 1) $\lambda_t = \frac{1}{T-t}$, 2) $\lambda_t = 0.95^{T-t-1}$, 3) $\lambda_t = 0.7^{T-t-1}$, 4) $\lambda_t = \frac{t}{T}$.

Fig. 8(a) shows the weight values over rounds for these weighting policies. Specifically, the weight values of $\lambda_t = \frac{t}{T}$ are linear increasing with respect to t. For the other three weighting policies, the weight values in the later rounds are far bigger than in the early rounds. Compared to $\lambda_t = 0.95^{T-t}$ and $\lambda_t = 0.7^{T-t}$, $\lambda_t = \frac{1}{T-t}$ has large weight values in the early rounds. Fig. 8(b) and Fig. 8(c) show the learning performance and the cumulative time usage of all four weighting policies. Since changing the weighting policy may break the learning time constraint without changing the drift-plus-penalty hyperparameter μ, this subsection sets different drift-plus-penalty hyperparameters for these weighting policies such that the consumed time does not exceed the time constraint. Specifically, we set $\mu = 10$ for $\lambda_t = \frac{1}{T-t}$, $\mu = 18$ for $\lambda_t = 0.95^{T-t}$, $\mu = 30$ for $\lambda_t = 0.7^{T-t}$ and $\mu = 0.5$ for $\lambda_t = \frac{t}{T}$. From Fig. 8(c), all these policies adhere to the time constraint, with the unified time usage being less than 1 at the end of the training process. It is observed from Fig. 8(b) that the weighting policy $\lambda_t = \frac{1}{T-t}$ achieves the highest learning accuracy. The reason comes from the following aspects: 1) Compared to $\lambda_t = \frac{t}{T}$, $\lambda_t = \frac{1}{T-t}$ is able to emphasise more importance in the later learning rounds. 2) Compared to $\lambda_t = 0.95^{T-t}$ and $\lambda_t = 0.7^{T-t}$, $\lambda_t = \frac{1}{T-t}$ allows more clients to participate in the early rounds to provide a more accurate model for the training in later rounds.

VI. Conclusion

This work proposed a novel ASAFL approach to dynamically adjust the synchronous degree to strike a balance between learning latency and accuracy. To mitigate the biased model updating resulting from partial client aggregation in the presence of data heterogeneity, we have proposed calibrating the global updates by incorporating the historical local gradients of absent clients in each round. We have theoretically analyzed the convergence behaviour of the proposed ASAFL, revealing that allocating more learning time to the later rounds helps improve the learning performance when the total learning time budget is fixed. Inspired by this, we develop an online synchronous degree control and resource optimization scheme to improve the learning performance of ASAFL. Simulation

results demonstrated the efficacy of ASAFL in improving convergence speed and learning accuracy.

APPENDIX

A. Proof of Lemma 1

Note that Lemma 1 is satisfied when $i = 0$ due to $\mathbf{x}_{n,t}^0 = \mathbf{x}_t$. For the case of $i \geq 1$, we have

$$\begin{aligned}
&\mathbb{E}\left\|\mathbf{x}_{n,t}^i - \mathbf{x}_t\right\|^2 = \mathbb{E}\left\|\mathbf{x}_{n,t}^{i-1} - \eta\nabla\tilde{\mathcal{L}}_n(\mathbf{x}_{n,t}^{i-1}) - \mathbf{x}_t\right\|^2 \\
&= \mathbb{E}\left\|\mathbf{x}_{n,t}^{i-1} - \eta\nabla\mathcal{L}_n(\mathbf{x}_{n,t}^{i-1}) - \mathbf{x}_t\right\|^2 \\
&+ \eta^2\mathbb{E}\left\|\nabla\tilde{\mathcal{L}}_n(\mathbf{x}_{n,t}^{i-1}) - \nabla\mathcal{L}_n(\mathbf{x}_{n,t}^{i-1})\right\|^2 \\
&\leq \mathbb{E}\left\|\mathbf{x}_{n,t}^{i-1} - \eta\nabla\mathcal{L}_n(\mathbf{x}_{n,t}^{i-1}) - \mathbf{x}_t\right\|^2 + \eta^2\varsigma^2, \qquad (26)
\end{aligned}$$

where the second equation is due to the unbiased stochastic gradient assumption. The last inequation is due to Assumption 2. For the first term on the RHS of (26), we have

$$\begin{aligned}
&\mathbb{E}\left\|\mathbf{x}_{n,t}^{i-1} - \eta\nabla\mathcal{L}_n(\mathbf{x}_{n,t}^{i-1}) - \mathbf{x}_t\right\|^2 \\
&= \mathbb{E}\left\|\mathbf{x}_{n,t}^{i-1} - \mathbf{x}_t\right\|^2 + \eta^2\mathbb{E}\left\|\nabla\mathcal{L}_n(\mathbf{x}_{n,t}^{i-1})\right\|^2 \\
&- 2\mathbb{E}\Big\langle\frac{1}{\sqrt{I-1}}(\mathbf{x}_{n,t}^{i-1} - \mathbf{x}_t), \sqrt{I-1}\eta\nabla\mathcal{L}_n(\mathbf{x}_{n,t}^{i-1})\Big\rangle \\
&\leq (1 + \frac{1}{I-1})\mathbb{E}\left\|\mathbf{x}_{n,t}^{i-1} - \mathbf{x}_t\right\|^2 + \eta^2 I\mathbb{E}\left\|\nabla\mathcal{L}_n(\mathbf{x}_{n,t}^{i-1})\right\|^2 \\
&\leq (1 + \frac{1}{I-1})\mathbb{E}\left\|\mathbf{x}_{n,t}^{i-1} - \mathbf{x}_t\right\|^2 \\
&+ 2\eta^2 I\mathbb{E}\left\|\nabla\mathcal{L}_n(\mathbf{x}_{n,t}^{i-1}) - \nabla\mathcal{L}_n(\mathbf{x}_t)\right\|^2 + 2\eta^2 I\mathbb{E}\left\|\nabla\mathcal{L}_n(\mathbf{x}_t)\right\|^2 \\
&\leq (1 + \frac{3}{2(I-1)})\mathbb{E}\left\|\mathbf{x}_{n,t}^{i-1} - \mathbf{x}_t\right\|^2 + 2\eta^2 IG^2, \qquad (27)
\end{aligned}$$

where the first and second inequation holds by the triangle inequality. The last inequation comes from Assumption 1. Substituting (27) into (26) obtains:

$$\begin{aligned}
\mathbb{E}\left\|\mathbf{x}_{n,t}^i - \mathbf{x}_t\right\|^2 &\leq (1 + \frac{3}{2(I-1)})\mathbb{E}\left\|\mathbf{x}_{n,t}^{i-1} - \mathbf{x}_t\right\|^2 \\
&+ 2\eta^2 IG^2 + \eta^2\varsigma^2. \qquad (28)
\end{aligned}$$

Telescoping the above inequation obtains

$$\begin{aligned}
\mathbb{E}\left\|\mathbf{x}_{n,t}^i - \mathbf{x}_t\right\|^2 &\leq (2\eta^2 IG^2 + \eta^2\varsigma^2)\frac{(1 + \frac{3}{2(I-1)})^{I-1} - 1}{\frac{3}{2(I-1)}} \\
&\leq 4\eta^2(I-1)(2IG^2 + \varsigma^2), \qquad (29)
\end{aligned}$$

where the last inequation is due to $(1 + \frac{3}{2(I-1)})^{I-1} = (1 + \frac{3}{2(I-1)})^{\frac{3}{2}\frac{2(I-1)}{3}} \leq \exp(\frac{3}{2}) < 5$ and $\frac{2(I-1)}{3} < I - 1$.

B. Proof of Lemma 2

For two different rounds, i.e., t_1 and t_2 ($t_1 > t_2$), we have

$$\begin{aligned}
&\mathbb{E}\left\|\mathbf{x}_{t_1} - \mathbf{x}_{t_2}\right\|^2 = \mathbb{E}\Big\|\sum_{t=t_2}^{t_1-1}(\mathbf{x}_{t+1} - \mathbf{x}_t)\Big\|^2 \\
&= \eta^2 I^2\mathbb{E}\Big\|\sum_{t=t_2}^{t_1-1}\frac{1}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\nabla\tilde{\mathcal{L}}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)\Big\|^2 \\
&\leq \frac{3\eta^2 I}{N}(t_1 - t_2)\sum_{t=t_2}^{t_1-1}\sum_{n=1}^{N}\sum_{i=0}^{I-1}L^2\mathbb{E}\left\|\mathbf{x}_{n,t-\tau_{n,t}}^i - \mathbf{x}_{t-\tau_{n,t}}\right\|^2 \\
&+ 3\eta^2 I^2(t_1 - t_2)^2(\varsigma^2 + G^2) \\
&\leq 3c_1\eta^2(t_1 - t_2)^2, \qquad (30)
\end{aligned}$$

where $c_1 = (I^2 + I - 1)\varsigma^2 + (I^2 + 2I(I-1))G^2$. The first inequation comes from adding and subtracting $\nabla\mathcal{L}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i) + \nabla\mathcal{L}_n(\mathbf{x}_{t-\tau_{n,t}})$, then using the triangle inequality. The second inequation is due to Lemma 1.

C. Proof of Theorem 1

According to the β-smooth of $\mathcal{L}(\cdot)$, we derive that

$$\begin{aligned}
&\mathbb{E}[\mathcal{L}(\mathbf{x}_{t+1}) - \mathcal{L}(\mathbf{x}_t)] \\
&\leq \mathbb{E}\left\langle\nabla\mathcal{L}(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}_t\right\rangle + \frac{\beta}{2}\mathbb{E}\left\|\mathbf{x}_{t+1} - \mathbf{x}_t\right\|^2 \\
&= -\eta\mathbb{E}\Big\langle\nabla\mathcal{L}(\mathbf{x}_t), \frac{1}{N}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\nabla\tilde{\mathcal{L}}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)\Big\rangle \\
&+ \frac{\beta\eta^2}{2}\mathbb{E}\Big\|\frac{1}{N}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\nabla\tilde{\mathcal{L}}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)\Big\|^2 \\
&= \underbrace{-\eta\mathbb{E}\Big\langle\nabla\mathcal{L}(\mathbf{x}_t), \frac{1}{N}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\nabla\mathcal{L}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)\Big\rangle}_{A_1} \\
&\underbrace{-\frac{\eta}{N}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\mathbb{E}\Big\langle\nabla\mathcal{L}(\mathbf{x}_t), \nabla\tilde{\mathcal{L}}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i) - \nabla\mathcal{L}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)\Big\rangle}_{A_2} \\
&+ \frac{\beta\eta^2 I^2}{2}\underbrace{\mathbb{E}\Big\|\frac{1}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\nabla\tilde{\mathcal{L}}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i))\Big\|^2}_{A_3}, \qquad (31)
\end{aligned}$$

where the last equation holds by adding and subtracting $\nabla\mathcal{L}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)$ into $\nabla\tilde{\mathcal{L}}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)$. Below we bound the three terms in (31). For A_1, we have

$$\begin{aligned}
A_1 &= -\eta I\mathbb{E}\left\|\nabla\mathcal{L}(\mathbf{x}_t)\right\|^2 + \eta I\mathbb{E}\Big\langle\nabla\mathcal{L}(\mathbf{x}_t), \\
&\nabla\mathcal{L}(\mathbf{x}_t) - \frac{1}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\nabla\mathcal{L}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)\Big\rangle \\
&\leq -\frac{1}{2}\eta I\mathbb{E}\left\|\nabla\mathcal{L}(\mathbf{x}_t)\right\|^2 \\
&+ \frac{\eta I}{2}\mathbb{E}\Big\|\nabla\mathcal{L}(\mathbf{x}_t) - \frac{1}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\nabla\mathcal{L}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)\Big\|^2. \qquad (32)
\end{aligned}$$

For the last term on the RHS of (32), we have

$$\begin{aligned}
&\mathbb{E}\Big\|\nabla\mathcal{L}(\mathbf{x}_t) - \frac{1}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\nabla\mathcal{L}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)\Big\|^2 \\
&\leq \frac{1}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\mathbb{E}\left\|\nabla\mathcal{L}(\mathbf{x}_t) - \nabla\mathcal{L}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)\right\|^2 \\
&\leq \frac{3}{N}\sum_{n=1}^{N}\mathbb{E}\left\|\nabla\mathcal{L}(\mathbf{x}_t) - \nabla\mathcal{L}_n(\mathbf{x}_t)\right\|^2 \\
&+ \frac{3}{N}\sum_{n=1}^{N}\mathbb{E}\left\|\nabla\mathcal{L}_n(\mathbf{x}_t) - \nabla\mathcal{L}_n(\mathbf{x}_{t-\tau_{n,t}})\right\|^2 \\
&+ \frac{3}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\mathbb{E}\left\|\nabla\mathcal{L}_n(\mathbf{x}_{t-\tau_{n,t}}) - \nabla\mathcal{L}_n(\mathbf{x}_{n,t-\tau_{n,t}}^i)\right\|^2 \\
&\leq 3\Gamma + \frac{3}{N}\sum_{n=1}^{N}\beta^2\mathbb{E}\left\|\mathbf{x}_t - \mathbf{x}_{t-\tau_{n,t}}\right\|^2 \\
&+ \frac{3}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\beta^2\mathbb{E}\|\mathbf{x}_{t-\tau_{n,t}} - \mathbf{x}_{n,t-\tau_{n,t}}^i\|^2, \qquad (33)
\end{aligned}$$

where the first inequation follows the Jensen's inequality. The second inequation is derived by adding and subtracting $\nabla\mathcal{L}_n(\mathbf{x}_t)+\nabla\mathcal{L}_n(\mathbf{x}_{t-\tau_{n,t}})$ into $\nabla\mathcal{L}(\mathbf{x}_t)$. The third inequation abides by Assumption 1 and Assumption 2. Substituting (33) into (32), we have

$$A_1 \leq -\frac{1}{2}\eta I\mathbb{E}\left\|\nabla\mathcal{L}(\mathbf{x}_t)\right\|^2+\frac{3\eta I}{2N}\sum_{n=1}^{N}\beta^2\mathbb{E}\left\|\mathbf{x}_t-\mathbf{x}_{t-\tau_{n,t}}\right\|^2 + \frac{3\eta I\Gamma}{2}+\frac{3\eta}{2N}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\beta^2\mathbb{E}\left\|\mathbf{x}_{t-\tau_{n,t}}-\mathbf{x}^i_{n,t-\tau_{n,t}}\right\|^2. \tag{34}$$

For the term A_2, we have

$$\begin{aligned}A_2 &= -\frac{\eta}{N}\sum\nolimits_{n=1}^{N}\sum\nolimits_{i=0}^{I-1}\mathbb{E}\Big\langle\nabla\mathcal{L}(\mathbf{x}_t)-\nabla\mathcal{L}(\mathbf{x}_{t-\tau_{n,t}}),\\ &\quad\nabla\tilde{\mathcal{L}}_n(\mathbf{x}^i_{n,t-\tau_{n,t}})-\nabla\mathcal{L}_n(\mathbf{x}^i_{n,t-\tau_{n,t}})\Big\rangle\\ &\leq \frac{\eta I}{2N}\sum\nolimits_{n=1}^{N}\mathbb{E}\left\|\nabla\mathcal{L}(\mathbf{x}_t)-\nabla\mathcal{L}(\mathbf{x}_{t-\tau_{n,t}})\right\|^2\\ &\quad+\frac{\eta}{2N}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\mathbb{E}\|\nabla\tilde{\mathcal{L}}_n(\mathbf{x}^i_{n,t-\tau_{n,t}})-\nabla\mathcal{L}_n(\mathbf{x}^i_{n,t-\tau_{n,t}})\|^2\\ &\leq \frac{\eta I\beta^2}{2N}\sum\nolimits_{n=1}^{N}\mathbb{E}\left\|\mathbf{x}_t-\mathbf{x}_{t-\tau_{n,t}}\right\|^2+\frac{1}{2}\eta I\varsigma^2.\end{aligned} \tag{35}$$

Now we bound the term A_3 as follows:

$$\begin{aligned}A_3 &\leq \frac{1}{NI}\sum\nolimits_{n=1}^{N}\sum\nolimits_{i=0}^{I-1}\mathbb{E}\left\|\nabla\tilde{\mathcal{L}}_n(\mathbf{x}^i_{n,t-\tau_{n,t}})\right\|^2\\ &\leq \frac{1}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\mathbb{E}\left\|\nabla\mathcal{L}_n(\mathbf{x}^i_{n,t-\tau_{n,t}})\right\|^2+\varsigma^2\\ &\leq \frac{2}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\mathbb{E}\left\|\nabla\mathcal{L}_n(\mathbf{x}^i_{n,t-\tau_{n,t}})-\nabla\mathcal{L}(\mathbf{x}_t)\right\|^2\\ &\quad+2\mathbb{E}\left\|\nabla\mathcal{L}(\mathbf{x}_t)\right\|^2+\varsigma^2,\end{aligned} \tag{36}$$

where the second inequation is due to Assumption 2. The third inequation is by adding and subtracting $\nabla\mathcal{L}(\mathbf{x}_t)$ into $\nabla\mathcal{L}_n(\mathbf{x}^i_{n,t-\tau_{n,t}})$. The first term in the above inequality is bounded as

$$\begin{aligned}&\frac{2}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\mathbb{E}\left\|\nabla\mathcal{L}_n(\mathbf{x}^i_{n,t-\tau_{n,t}})-\nabla\mathcal{L}(\mathbf{x}_t)\right\|^2\\ &\leq \frac{6}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\beta^2\mathbb{E}\left\|\mathbf{x}^i_{n,t-\tau_{n,t}}-\mathbf{x}_{t-\tau_{n,t}}\right\|^2\\ &\quad+\frac{6}{N}\sum_{n=1}^{N}\beta^2\mathbb{E}\left\|\mathbf{x}_{t-\tau_{n,t}}-\mathbf{x}_t\right\|^2\\ &\quad+\frac{6}{N}\sum_{n=1}^{N}\mathbb{E}\left\|\nabla\mathcal{L}_n(\mathbf{x}_t)-\nabla\mathcal{L}(\mathbf{x}_t)\right\|^2\\ &\leq \frac{6}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\beta^2\mathbb{E}\left\|\mathbf{x}^i_{n,t-\tau_{n,t}}-\mathbf{x}_{t-\tau_{n,t}}\right\|^2\\ &\quad+\frac{6}{N}\sum_{n=1}^{N}\beta^2\mathbb{E}\left\|\mathbf{x}_{t-\tau_{n,t}}-\mathbf{x}_t\right\|^2+6\Gamma^2.\end{aligned} \tag{37}$$

Substituting (37) into (36), we have

$$\begin{aligned}A_3 \leq\ & 2\mathbb{E}\left\|\nabla\mathcal{L}(\mathbf{x}_t)\right\|^2\\ &+\frac{6}{NI}\sum_{n=1}^{N}\sum_{i=0}^{I-1}\beta^2\mathbb{E}\left\|\mathbf{x}^i_{n,t-\tau_{n,t}}-\mathbf{x}_{t-\tau_{n,t}}\right\|^2\\ &+\frac{6}{N}\sum_{n=1}^{N}\beta^2\mathbb{E}\left\|\mathbf{x}_{t-\tau_{n,t}}-\mathbf{x}_t\right\|^2+\varsigma^2+6\Gamma^2.\end{aligned} \tag{38}$$

Substituting (34), (38), and (35) into (31), we have

$$\begin{aligned}\mathbb{E}[\mathcal{L}(\mathbf{x}_{t+1})-\mathcal{L}(\mathbf{x}_t)] \leq\ & (-\frac{1}{2}\eta I+\beta\eta^2I^2)\mathbb{E}\left\|\nabla\mathcal{L}(\mathbf{x}_t)\right\|^2\\ &+\frac{21\eta^2\beta}{4N}c_1\sum\nolimits_{n=1}^{N}\tau^2_{n,t}+12\eta^2\beta(I-1)IG^2\\ &+(6\eta^2\beta(I-1)+\frac{3}{4}\eta I)\varsigma^2+3\eta I\Gamma.\end{aligned} \tag{39}$$

According to Assumption 1, we have $\left\|\nabla\mathcal{L}(\mathbf{x}_t)\right\|^2 \geq 2\mu\left(\mathcal{L}(\mathbf{x}_t)-\mathcal{L}^*\right)$. By subtracting $\mathcal{L}^*$ to $\mathbf{x}_{t+1})$ and $\mathcal{L}(\mathbf{x}_t)$, we have

$$\begin{aligned}\mathbb{E}[\mathcal{L}(\mathbf{x}_{t+1})-\mathcal{L}^*] \leq\ & (1-\mu\eta I+2\mu\beta\eta^2I^2)\mathbb{E}[\mathcal{L}(\mathbf{x}_t)-\mathcal{L}^*]\\ &+\frac{21\eta^2\beta c_1}{4N}\sum_{n=1}^{N}\tau^2_{n,t}+c.\end{aligned} \tag{40}$$

Since the staleness of each client n's local gradient evolves as

$$\tau_{n,t}=\begin{cases}\tau_{n,t-1}+1, & \text{if } s_{n,t}=0,\\ 0, & \text{else},\end{cases} \tag{41}$$

we have $\tau^2_{n,t}=(1-s_{n,t})^2(\tau_{n,t-1}+1)^2=(1-s_{n,t})(\tau_{n,t-1}+1)^2$. Consequently, we have

$$\begin{aligned}\mathbb{E}[\mathcal{L}(\mathbf{x}_{t+1})-\mathcal{L}^*] \leq\ & (1-\mu\eta I+2\mu\beta\eta^2I^2)\mathbb{E}[\mathcal{L}(\mathbf{x}_t)-\mathcal{L}^*]\\ &+\frac{21\eta^2\beta c_1}{4N}\sum\nolimits_{n=1}^{N}(1-s_{n,t})(\tau_{n,t-1}+1)^2+c.\end{aligned} \tag{42}$$

By telescoping the above inequation, and let $\eta \leq \frac{1}{2I\beta}$, we complete the proof.

D. Proof of Lemma 3

In accordance with (7), the communication latency $T^{\text{comm}}_{n,t}$ exhibits a decreasing trend concerning $\vartheta_{n,t}$ monotonically. For a client who completes its gradient uploading ahead of others, we can redistribute its bandwidth to the slower clients. This reduces the latency in the current round decided by stragglers. The bandwidth redistribution continues until all clients complete local computation and transmit gradients. Hence, the optimal solution to problem (16) is conducted by allocating the overall bandwidth to all clients in S_t, ensuring they complete gradient computing and uploading simultaneously. Therefore, the optimal bandwidth allocation policy adheres

$$\begin{cases}T^{\text{rem}}_{n,t}+T^{\text{comm}}_{n,t}=T^*_t, \forall n\in S_t,\\ \sum\limits_{n\in S_t}\vartheta_{n,t}=1.\end{cases} \tag{43}$$

where T^*_t is the optimal delay of round t. By resolving the above equations, we complete the proof.

E. Proof of Proposition 1

Firstly, we define the Lyapunov function as $\Phi(t)=\frac{1}{2}q^2_t$. According to the virtual queue defined in (14), we have $q^2_{t+1}\leq(q_t+T_t-\frac{T_{\max}}{T})^2$. The Lyapunov drift in the t-th round is $\Lambda(t)=\Phi(t+1)-\Phi(t)$, i.e.,

$$\begin{aligned}\Lambda(t)&=\frac{1}{2}(q^2_{t+1}-q^2_t)\\ &\leq\frac{1}{2}(T_t-\frac{T_{\max}}{T})^2+q_t(T_t-\frac{T_{\max}}{T})\end{aligned}$$

$$\leq \frac{1}{2}\chi^2 + q_t(T_t - \frac{T_{\max}}{T}), \quad (44)$$

where $\chi = \max_t \{|T_t - \frac{T_{\max}}{T}|\}$. By adding $-\mu\lambda_t \sum_{n=1}^{N} s_{n,t}(\tau_{n,t-1}+1)^2$ into both sides of (44), the upper bound of the drift-plus-penalty in round t is

$$\Lambda(t) - \mu\lambda_t \sum\nolimits_{n=1}^{N} s_{n,t}(\tau_{n,t-1}+1)^2 \leq \frac{1}{2}\chi^2 + q_t(T_t - \frac{T_{\max}}{T}) - \mu\lambda_t \sum_{n=1}^{N} s_{n,t}(\tau_{n,t-1}+1)^2. \quad (45)$$

The classical drift-plus-penalty algorithm strives to minimize the upper bound of $\Lambda(t) - \mu\lambda_t \sum_{n=1}^{N} s_{n,t}(\tau_{n,t-1}+1)^2$. Thus, its solution makes the RHS on (45) not larger than that achieved by any feasible solution. Denoting the solution of the proposed algorithm by $s^+_{n,t}$, and considering a feasible solution with $T_t = 0$ and $s_{n,t} = 0$, we have

$$\Lambda(t) - \mu\lambda_t \sum\nolimits_{n=1}^{N} s^+_{n,t}(\tau_{n,t-1}+1)^2 \leq \frac{1}{2}\chi^2. \quad (46)$$

Let $\Lambda_T = \Phi(T-1) - \Phi(0) = \frac{1}{2}q^2_{k,T-1}$ be the T-round drift. Thus, we bound the T-round drift-plus-penalty as

$$\Lambda_T = \frac{1}{2}q^2_{k,T-1} \leq \frac{1}{2}T\chi^2 + \sum_{t=0}^{T-1}\mu\lambda_t \sum_{n=1}^{N}(\tau_{n,t-1}+1)^2. \quad (47)$$

Thus, we have

$$q^2_{k,T-1} \leq T\chi^2 + 2\sum\nolimits_{t=0}^{T-1}\mu\lambda_t \sum\nolimits_{n=1}^{N}(\tau_{n,t-1}+1)^2. \quad (48)$$

According to the queue defined in (14), we have $T_t - \frac{T_{\max}}{T} \leq q_{t+1} - q_t$. Summing the above inequality along T rounds obtains

$$\sum\nolimits_{t=0}^{T-1}(T_t - \frac{T_{\max}}{T}) \leq \sum\nolimits_{t=0}^{T-1}(q_{t+1} - q_t) \leq \sqrt{T\chi^2 + 2\sum\nolimits_{t=0}^{T-1}\mu\lambda_t \sum\nolimits_{n=1}^{N}(\tau_{n,t-1}+1)^2}. \quad (49)$$

By rearranging the above inequality, (22) is obtained. Below we analyze the optimality of the proposed algorithm, which minimizes the RHS in (45). Since $\Lambda(t)$ is positive, we have

$$-\sum_{t=0}^{T-1}\mu\lambda_t \sum_{n=1}^{N} s^+_{n,t}(\tau_{n,t-1}+1)^2 \leq \frac{1}{2}T\chi^2 + \sum_{t=0}^{T-1} q_t(T^*_t - \frac{T_{\max}}{T}) - \sum_{t=0}^{T-1}\mu\lambda_t \sum_{n=1}^{N} s^*_{n,t}(\tau_{n,t-1}+1)^2, \quad (50)$$

where the second term of the above inequation is bounded as

$$\sum_{t=0}^{T-1} q_t(T^*_t - \frac{T_{\max}}{T}) = \sum_{t=0}^{T-1}(q_t - q_0)(T^*_t - \frac{T_{\max}}{T}) \leq \frac{T(T-1)}{2}\chi^2, \quad (51)$$

where the inequation holds due to $q_0 = 0$ and $q_{t+1} - q_t \leq \chi$. Substituting (51) into (50), (21) is derived.

REFERENCES

[1] Z. Chen, W. Yi, S. Hyundong, and A. Nallanathan, "Fast wireless federated learning with adaptive synchronous degree control," in *Proc. IEEE Vehicular Technology Conference (VTC-Spring)*, 2024.

[2] Z. Yang, M. Chen, K.-K. Wong, H. V. Poor, and S. Cui, "Federated learning for 6g: Applications, challenges, and opportunities," *Engineering*, vol. 8, pp. 33–41, 2022.

[3] D. Wen, X. Li, Y. Zhou, Y. Shi, S. Wu, and C. Jiang, "Integrated sensing-communication-computation for edge artificial intelligence," *arXiv preprint arXiv:2306.01162*, 2023.

[4] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated learning for internet of things: Recent advances, taxonomy, and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1759–1799, 2021.

[5] D. Wen, P. Liu, G. Zhu, Y. Shi, J. Xu, Y. C. Eldar, and S. Cui, "Task-oriented sensing, computation, and communication integration for multi-device edge ai," *IEEE Trans. Wireless Commun.*, vol. 23, no. 3, pp. 2486–2502, 2024.

[6] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proc. Artificial Intelligence and Statistics (AISTATS)*, 20–22, Apr. 2017.

[7] N. Huang, M. Dai, Y. Wu, T. Q. S. Quek, and X. Shen, "Wireless federated learning with hybrid local and centralized training: A latency minimization design," *IEEE J. Sel. Topics in Signal Processing*, vol. 17, no. 1, pp. 248–263, 2023.

[8] W. Xu, Z. Yang, D. W. K. Ng, M. Levorato, Y. C. Eldar, and M. Debbah, "Edge learning for b5g networks with distributed signal processing: Semantic communication, edge computing, and wireless sensing," *IEEE J. Sel. Topics Signal Processing*, vol. 17, no. 1, pp. 9–39, 2023.

[9] P. Li, G. Cheng, X. Huang, J. Kang, R. Yu, Y. Wu, M. Pan, and D. Niyato, "Snowball: Energy efficient and accurate federated learning with coarse-to-fine compression over heterogeneous wireless edge devices," *IEEE Trans. Wireless Commun.*, vol. 22, no. 10, pp. 6778–6792, 2023.

[10] Z. Chen, W. Yi, H. Shin, and A. Nallanathan, "Adaptive model pruning for communication and computation efficient wireless federated learning," *IEEE Trans. Wireless Commun.*, pp. 1–1, 2023.

[11] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[12] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 1935–1949, 2021.

[13] E. Rizk, S. Vlaski, and A. H. Sayed, "Optimal importance sampling for federated learning," in *Proc IEEE ICASSP*, 2021, pp. 3095–3099.

[14] M. M. Amiri, D. Gndz, S. R. Kulkarni, and H. V. Poor, "Convergence of update aware device scheduling for federated learning at the wireless edge," *IEEE Trans. Wireless Commun.*, vol. 20, no. 6, pp. 3643–3658, 2021.

[15] Z. Chen, W. Yi, and A. Nallanathan, "Exploring representativity in device scheduling for wireless federated learning," *IEEE Trans. Wireless Commun.*, vol. 23, no. 1, pp. 720–735, 2024.

[16] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2457–2471, 2021.

[17] 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 Trans. Wireless Commun.*, vol. 20, no. 1, pp. 269–283, 2021.

[18] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint arXiv:1903.03934*, 2019.

[19] J. Nguyen, K. Malik, H. Zhan, A. Yousefpour, M. Rabbat, M. Malek, and D. Huba, "Federated learning with buffered asynchronous aggregation," in *Proc. Conf. Artificial Intelligence and Statistics (AISTATS)*, 28–30 Mar 2022.

[20] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Differentially private asynchronous federated learning for mobile edge computing in urban informatics," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2134–2143, 2020.

[21] S. Zheng, Q. Meng, T. Wang, W. Chen, N. Yu, Z.-M. Ma, and T.-Y. Liu, "Asynchronous stochastic gradient descent with delay compensation," in *Proc. Int. Conf. Machine Learning*, 06–11 Aug 2017.

[22] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," *Advances in neural information processing systems*, vol. 33, pp. 7611–7623, 2020.

[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," in *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, 2021, pp. 1–10.

[24] 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 Trans. Computers*, vol. 70, no. 5, pp. 655–668, 2021.

[25] 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 J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3654–3672, 2021.

[26] J. Liu, H. Xu, L. Wang, Y. Xu, C. Qian, J. Huang, and H. Huang, "Adaptive asynchronous federated learning in resource-constrained edge computing," *IEEE Trans. Mobile Computing*, vol. 22, no. 2, pp. 674–690, 2023.

[27] H. Yang, M. Fang, and J. Liu, "Achieving linear speedup with partial worker participation in non-iid federated learning," *arXiv preprint arXiv:2101.11203*, 2021.

[28] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-IID data," in *Proc. International Conference on Learning Representations (ICLR)*, 2020.

[29] Z. Chen, W. Yi, Y. Liu, and A. Nallanathan, "Knowledge-aided federated learning for energy-limited wireless networks," *IEEE Trans. Commun.*, vol. 71, no. 6, pp. 3368–3386, 2023.

[30] A. Virmaux and K. Scaman, "Lipschitz regularity of deep neural networks: analysis and efficient estimation," in *Proc. Neural Information Processing Systems*, 2018.

[31] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[32] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[33] H.-S. Lee and J.-W. Lee, "Adaptive transmission scheduling in wireless networks for asynchronous federated learning," *IEEE J. Sel. Areas in Commun.*, vol. 39, no. 12, pp. 3673–3687, 2021.

[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] Z. Zhu, J. Hong, and J. Zhou, "Data-free knowledge distillation for heterogeneous federated learning," in *Proc. Int. Conf. Mach. Learning (ICML)*, 18–24 Jul, 2021.