Anarchic Federated learning with Delayed Gradient Averaging

Dongsheng Li
dzl0093@auburn.edu
Department of Electrical and Computer Engineering,
Auburn University
Auburn, Alabama, USA

Xiaowen Gong
xgong@auburn.edu
Department of Electrical and Computer Engineering,
Auburn University
Auburn, Alabama, USA

ABSTRACT

The rapid advances in federated learning (FL) in the past few years have recently inspired a great deal of research on this emerging topic. Existing work on FL often assume that clients participate in the learning process with some particular pattern (such as balanced participation), and/or in a synchronous manner, and/or with the same number of local iterations, while these assumptions can be hard to hold in practice. In this paper, we propose AFL-DGA, an Anarchic Federated Learning algorithm with Delayed Gradient Averaging, which gives maximum freedom to clients. In particular, AFL-DGA allows clients to 1) participate in any rounds; 2) participate asynchronously; 3) participate with any number of local iterations; 4) perform gradient computations and gradient communications in parallel. The proposed AFL-DGA algorithm enables clients to participate in FL flexibly according to their heterogeneous and time-varying computation and communication capabilities, and also efficiently by improving utilization of their computation and communication resources. We characterize performance bounds on the learning loss of AFL-DGA as a function of clients' local iteration numbers, local computation delays, and global gradient delays. Our results show that the AFL-DGA algorithm can achieve a convergence rate of $O(\frac{1}{\sqrt{NT}})$ and also a linear convergence speedup, which matches that of existing benchmarks. The results also characterize the impacts of clients' various parameters on the learning loss, which provide useful insights. Numerical results demonstrate the efficiency of the proposed algorithm.

KEYWORDS

federated learning, delayed gradient, asynchronous

ACM Reference Format:
Dongsheng Li and Xiaowen Gong. 2023. Anarchic Federated learning with Delayed Gradient Averaging. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

As an emerging paradigm of machine learning (ML), *federated learning* (FL) carries out model training in a distributed manner [15]: Instead of collecting data from a possibly large number of devices to a central server in the cloud for training, FL trains a global ML model by aggregating local ML models computed distributedly across edge devices based on their local data. One significant advantage of FL is to preserve the privacy of individual devices' data. Moreover, since only local ML models rather than local data are sent to the server, the communication costs can be greatly reduced. Furthermore, FL can exploit substantial computation capabilities of ubiquitous smart devices.

In order to fully realize the potential of FL, several challenges need to be addressed due to heterogeneous and time-varying computation and communication capabilities of clients' devices. First of all, clients may not be able to participate in every round of the entire learning process. This is particularly the case for cross-device FL where many clients only have resource-constrained mobile devices which are sometimes not available to perform local computations and/or communications with the FL server. Moreover, due to heterogeneity in computation and communication capabilities, even a client is able to participate in learning, it may be impossible or inefficient for all clients to complete their local computations and also communications of their local models in every round of the learning process in a synchronous manner. As a result, clients may need to compute and communicate their local models asynchronously. Furthermore, even clients can participate in FL synchronously in a round, they may perform different numbers of local iterations of computation, based on their computation capabilities. Such heterogeneous computation configuration can improve the efficiency of clients in FL, especially when there are stragglers. However, existing work on FL only considered some of the issues discussed above, but not all the issues at the same time.

Besides the heterogeneity in clients' computation and communication capabilities, FL also faces the challenge of heterogeneous local data across clients, so that local gradients computed by clients from their local data can be diverse. In contrast to conventional distributed ML where nodes typically communicate after every local computation iteration, clients in FL can perform multiple local iterations of computation before communicating their local models. While this feature can reduce communication costs of FL, it may slow down the convergence of the global model due to local model drifts (which has been studied in some recent works such as [9, 21]).

In a typical FL algorithm, each client needs to receive the (average) global gradient of all participating clients, before starting the next round of local computation iterations. However, a client may have to wait for a long time before receiving the average global gradient, as it involves communications of the local and global gradients to and from the FL server, respectively. This is especially the case when gradient communication times are in the same order of magnitude as gradient computation times. Therefore,

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

to improve utilization of clients' computation resources, it is efficient for a client to start the next round of local computations right after the previous round, without receiving the global gradient of the previous round. In other words, the client performs gradient computations for the next round and gradient communications for the previous round in parallel.

While such concurrent gradient computations and communications improves resource utilization and thus can reduce the training time, it results in a delay of the global gradient when received by a client, in relative to the client's current local model. To mitigate the issue of local model drifts due to this global gradient delay, an effective approach is to correct a client's current local model according to the delayed global gradient for the past local computation iterations that have already been carried out in the current round. Intuitively, such delayed (global) gradient averaging (DGA) can reduce local model drifts and thus accelerate the convergence of the FL algorithm.

In this paper, we explore Anarchic Federated Learning with Delayed Gradient Averaging (AFL-DGA), which addresses all the challenges of FL as discussed above. In particular, AFL-DGA imposes minimum control on how clients participate in FL by allowing clients to 1) participate in arbitrary rounds; 2) participate asynchronously; 3) participate with arbitrary numbers of local iterations; 4) perform gradient computations and gradient communications in parallel. By giving maximum freedom to clients, AFL enables clients to participate in FL flexibly according to their needs, e.g., based on their heterogeneous and time-varying computation and communication capabilities. Moreover, AFL-DGA allows clients to pipeline gradient computations with gradient communications, which improves utilization of clients computation and communication resources. In the meanwhile, AFL-DGA can effectively reduce local model drifts due to clients' heterogeneous data, by correcting clients' local models with delayed global gradients. Although AFL has been studied very recently in [28], it does not take into account DGA (see detailed comparison in Section 4.2). While DGA has been studied in a recent work [31], it did not consider features of AFL (see detailed comparison in Section 4.2).

The main contributions of this paper are summarized as follows:

- We propose AFL-DGA, an Anarchic Federated Learning with Delayed Gradient Averaging (AFL-DGA), which allows clients to participate in any rounds, in an asynchronous manner, and with any numbers of local iterations. The proposed AFL-DGA algorithm enables clients to participate in FL efficiently and flexibly according to their needs, e.g., based on their heterogeneous and time-varying computation and communication capabilities. Moreover, the proposed algorithm allows clients to pipeline gradient computations with gradient communications, while correcting clients' local models with delayed global gradients to reduce the client's local model drifts.
- We conduct convergence analysis for the AFF-DGA algorithm by characterizing performance bounds on the learning loss as a function of various parameters of clients. Our results show that the AFL-DGA algorithm can achieve a convergence rate of $O(\frac{1}{\sqrt{NT}})$ and also a linear convergence speedup, which matches that of existing benchmarks. The results also characterize the impacts of clients' local iteration numbers, local computation delays, and global gradient delays on the learning loss, which provide useful insights.
- We evaluate the performance of the proposed AFL-DGA algorithm by conducting numerical experiments for FL benchmarks. The experimental results demonstrate the efficiency of the proposed algorithms.

The remainder of this paper is organized as follows. Section II reviews related work. In Section III, we present AFL-DGA algorithm. In Section IV, we analyze the convergence of the proposed AFL-DGA algorithm. Numerical results based on experiments are provided in Section VI.

2 RELATED WORK

FL has emerged as a disruptive computing paradigm for ML by democratizing the learning process to potentially many individual users using their end devices [2, 4, 7–9, 13, 15, 16, 18, 19, 23, 25, 26, 29, 30]. The past few years have seen tremendous research on FL. In the following, we discuss recent work on FL from three different aspects that are mostly related to this paper.

Federated Learning with Partial Client Participation. One major challenge for FL is that clients may not always participate throughout the entire learning process. This is especially true for cross-device FL where many clients have resource-constrained mobile devices which are sometimes not possible or too costly to perform local computations and/or communicate local/global models with the server. Many recent works [13, 23, 28, 31] studied FL where only some of all clients participate in learning in a round. Most of these studies [13, 31] assumed that clients' participation is balanced (e.g., the set of participating clients are randomly selected from all clients), such that each client has the same probability of participation. Under this assumption, it has been shown that FL algorithms can achieve a vanishing convergence error. However, in the general case where clients' participation can be arbitrary, there is a non-vanishing convergence error due to the worst-case client participation. This paper not only considers arbitrary client participation, but also asynchronous participation and heterogeneous local iteration numbers of clients. A recent work [28] has proposed and studied the AFL algorithm, where clients can participate in a round or not in an asynchronous manner with different local iteration number. Compared to AFL, the AFL-DGA proposed in this paper integrates the DGA algorithm, which is a major algorithmic difference compared to AFL (see detailed discussion in Section 4.2).

Asynchronous Federated Learning. Many existing work [13, 23, 31] on FL studied synchronous algorithms where participating clients perform local computations and exchange local models in the same round (note that synchronous FL can also have partial client participation). However, synchronous algorithms can be inefficient as some clients may have to wait for other clients to complete their computations and/or communications, especially when there are stragglers due to heterogeneous computation and communication capabilities of clients. In this case, asynchronous algorithms [14, 28] are more efficient where a client can start its local computations in one round while completing the communication of its local model in another round. In this paper, besides asynchronous learning, we also consider arbitrary client participation and heterogeneous local

iteration numbers. A recent work [31] studied the DGA algorithm under the simplified settings where clients participate in each round in a synchronous manner with the same local iteration number. Compared to DGA, the AFL-DGA algorithm proposed in this paper includes DGA as a special case and is much more non-trivial (see detailed discussion in Section 4.2).

Federated Learning with Heterogeneous Computations. One salient feature of FL is that clients can have heterogeneous computation capabilities. As a result, it is more efficient and flexible to allow clients to use different computation configurations. Some existing work on FL [20, 28] considered clients who use different mini-batch sizes, different local iteration numbers, and/or different learning model structures, etc. This paper considers clients with different local iteration numbers as well as arbitrary client participation and asynchronous algorithms.

3 ANARCHIC FEDERATED LEARNING WITH DELAYED GRADIENT AVERAGING

In this section, we first present the settings and the problem formulation of the FL problem we study. Then we describe the algorithm design of the Anarchic Federated Learning with Delayed Gradient Averaging (AFL-DGA), and explain its rationale.

3.1 System Setting and Problem Formulation

Consider a FL system with an FL server and N clients in set $\mathcal{N}$ who collaboratively train a ML model with distributed local data in an asynchronous manner. The goal of the FL system is to minimize the training loss, which is given by the following optimization problem:

$$\min_{\boldsymbol{w}} F(\boldsymbol{w}) \triangleq \sum_{k\in\mathcal{N}} p_k F_k(\boldsymbol{w}),$$

where $F(\boldsymbol{w})$ is the global loss function, $\boldsymbol{w}$ is the model parameter p_k is the coefficient of client k's local loss function, and $\sum_{k\in\mathcal{N}} p_k = 1$. $F_k(\boldsymbol{w})$ is the local loss function determined by client k's local dataset and $f_i(\boldsymbol{w}) = E_{\xi_i}[F_i(\boldsymbol{w};\xi_i)]$. In the setting of empirical risk minimization, f_i could be further expressed as finite sums and the random variable ξ_i corresponds to a mini-batch sample. Let $\boldsymbol{w}_{i,j}^t$ denote the local model of client i in the j-th iteration of round t, and $\boldsymbol{g}_{i,j}^t$ the corresponding stochastic gradient.

A client i is considered as participating in round t if it sends its local gradient to the server in round t, which is then used by the server to compute the global gradient for round t. If a client i participates in round t, it uses its local model to perform one or multiple local iterations of stochastic gradient descent (SGD), each of which is given by

$$\boldsymbol{w}_{i,j+1}^t \triangleq \boldsymbol{w}_{i,j}^t - \eta \nabla F_i(\boldsymbol{w}_{i,j}^t, \xi_{i,j}^t),\ j = 0, 1, ..., K_i^t - 1,$$

where h is the local iteration index, $\xi_{t,h}^k$ is a sample uniformly chosen from the client k's local dataset, and K_i^t is the number of local iterations in round t.

Let τ_i^t be the delay of the client i's most recent participation with respect to round t (i.e., the difference between t and the index of the round in which client i last participate before round t). In particular, if $\tau_k^t = 1, \forall k \in \mathcal{N}$, then the FL algorithm is synchronous; otherwise, it is asynchronous.

We use $\Gamma = \sum_{k\in\mathcal{N}} p_k(F^* - F_k^*)$ to quantify the non independent and identically distributed (non-i.i.d) degree of the local data among all clients [13]. If $\Gamma = 0$, then the local data are i.i.d., otherwise, they are non-i.i.d case. The larger Γ is, the higher non-i.i.d. degree is. In addition, we do not allow a client to never update her model to the server, which means there exists a maximum delay constraint.

3.2 Algorithm Design of AFL-DGA

In the following, we present the design of the AFL-DGA algorithm.

0. In the beginning, each client i uses the initial global model $\boldsymbol{w}^0$.

1.In each round t, if client i participates in round t, it performs a K_i^t number of local iterations of SGD from its local model $\boldsymbol{w}_{i,0}^t$. If client i receives the global gradient of a previous round $t' < t$ in any local iteration k of round t, it corrects its local model according to $\boldsymbol{w}_{i,k}^t = \boldsymbol{w}_{i,k}^t - \eta K_i^{t'} \overline{\boldsymbol{g}^{t'}} + \eta \sum_{j=1}^{K_i^{t'}} \boldsymbol{g}_{i,j}^{t'}$. After this local model correction, client i continues its local SGD iterations for round t until K_i^t iterations are completed. Then client i starts a new round of local SGD iterations based on its local model, for the next round that it participates in (which can be any round $t' > t$).

2. At the end of each round t, each client participating in round t computes its local gradient averaged over its local iterations in round t, which is $\overline{\boldsymbol{g}_i^t} = \sum_{j=1}^{K_i^t} \boldsymbol{g}_{i,j}^t / K_i^t$, and then sends it to the server.

3. After receiving the average local gradient from each participating client t, the server computes the global gradient for round t as the average local gradient across all clients, which is $\overline{\boldsymbol{g}^t} = \sum_{i=1}^{N} \overline{\boldsymbol{g}_i^t}/N$. If a client i does not participate in round t, the server uses the most recent local gradient received from client i to compute the global gradient for round t. Then the server sends the global gradient for round t to all clients.

4. Repeat the step $1-3$ until the training process converges or reaching the maximum training round.

To illustrate the main ideas of the AFL-DGA algorithm, we analyze its dynamics as follows. In round t, client i first performs K_i^t local iterations such that

$$\boldsymbol{w}_{i,K_i^t}^t = \boldsymbol{w}_{i,K_i^t-1}^t - \eta \boldsymbol{g}_{i,K_i^t-1}^t = ... = \boldsymbol{w}_{i,0}^t - \eta \sum_{j=0}^{K_i^t-1} \boldsymbol{g}_{i,j}^t$$

When client i completes its local iterations for round t, it sends its average local gradient $\overline{\boldsymbol{g}_i^t}$ to the server. Right after the local gradient is sent, client i immediately continues to perform local iterations for round $t+1$ (assuming that it participates in round $t+1$), leaving the local gradient of round $t$in transmission. When client i receives the global gradient for round t from the server, it has already performed another D_i^t local iterations for round $t+1$, such that

$$\boldsymbol{w}_{i,D_i^t}^{t+1} = \boldsymbol{w}_{i,D_i^t-1}^{t+1} - \eta \boldsymbol{g}_{i,D_i^t-1}^{t+1} = ... = \boldsymbol{w}_{i,K_i^t}^t - \underbrace{\eta \sum_{j=0}^{D_i^t-1} \boldsymbol{g}_{i,j}^{t+1}}_{\text{round } t+1}$$

At this point, the global gradient $\overline{\boldsymbol{g}^t}$ of round t arrives, Then we correct client i's local model by replacing all the local gradients computed for round t in (1) by the global gradient of round t, given

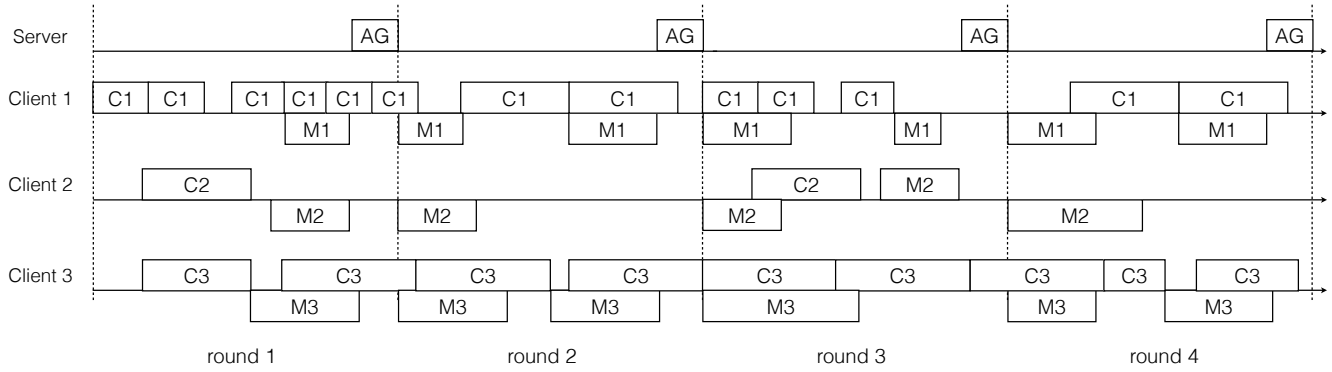

Figure 1: Schedule of AFL-DGA: AG is local gradient aggregation, each C block represents an iteration of local gradient Computation, each M block represents a local gradient coMmunication (uplink or downlink communication).

by

$$\boldsymbol{w}_{i,D_i^t}^{t+1} = \boldsymbol{w}_{i,K_i^t}^{t} - \eta \sum_{j=0}^{D_i^t-1} \boldsymbol{g}_{i,j}^{t+1} - \eta K_i^t (\overline{\boldsymbol{g}^t} - \overline{\boldsymbol{g}_i^t})$$

Compared to the time spent on computing the gradient, correcting the gradient takes almost no time, due to the fact that correcting the gradient only involves a small number of adding and subtraction operations. The detailed process is shown in Algorithm 1.

The schedule of the proposed AFL-DGA algorithm is described in Fig 1. As an example, the evolution trajectory of client 3's local model is given as below:

$$\begin{aligned}&\boldsymbol{w}_0 - \eta \boldsymbol{g}_{3,1}^1 - \eta \boldsymbol{g}_{3,1}^2 - \eta \boldsymbol{g}_{3,2}^2 - \eta(\overline{\boldsymbol{g}^1} - \boldsymbol{g}_{3,1}^1) - \eta \boldsymbol{g}_{3,1}^4 - \eta \boldsymbol{g}_{3,2}^4 \\ &- \eta(2\overline{\boldsymbol{g}^2} - \boldsymbol{g}_{3,1}^2 - \boldsymbol{g}_{3,2}^2) - \eta \boldsymbol{g}_{3,3}^4 - \eta \boldsymbol{g}_{3,4}^4 - \eta \overline{\boldsymbol{g}^3} - \eta \boldsymbol{g}_{3,5}^4 - \eta \boldsymbol{g}_{3,1}^5.\end{aligned}$$

Note that client 3 participates in rounds 1, 2, and 4, but not round 3.

3.3 Rationale of AFL-DGA

Next we discuss the rationale behind the algorithm design of AFL-DGA.

Global gradient averaging replaces clients' local gradients with the averaged global gradient of all clients, which reduces clients' local model drifts and thus can accelerate the convergence of the FL algorithm. However, global gradient averaging involves gradient communications which can incur substantial delays. By parallelizing gradient computations with gradient communications, clients' computation and communication resources are better utilized while the running time of the FL algorithm can be reduced. However, such parallelization results in delayed gradient averaging, which comes at the cost that the extra local gradients computed before averaging a client's local model with the delayed global gradient is more biased than that computed after the averaging.

We observe from the algorithm design of AFL-DGA that each clients sends its local gradient averaged over its local iterations to the server, which is then further averaged over all clients to find the global gradient. Intuitively, the averaged local gradient (rather than the accumulative local gradient) over the local iterations should be used, since all clients have the same weight in the global loss function of the FL problem.

We should also note from the algorithm design of AFL-DGA that, instead of replacing a client's local model entirely by the delayed global model, the local model is corrected by the delayed global gradient by replacing only the local gradient components of the local model that are computed in the round of the global gradient. Intuitively, the extra local gradients computed before correcting the local model, although they are biased, are still useful components of the local model to direct the progress of local computations. This is a major difference of DGA algorithms compared to standard asynchronous FL algorithms,

Moreover, it is worth noting that the delayed global gradient is always used by each client to correct its local model, as soon as it is received by the client, no matter whether the client participate in a round or not. Intuitively, even when a client does not participate in a round, after correcting the client's local model with the delayed global model of that round, its local model (and thus its local gradient computed from the local model) would be less biased.

4 CONVERGENCE ANALYSIS OF AFL-DGA

In this section, we first introduce some assumptions, followed with the convergence analysis of our algorithm.

Assumption 1. ***(Smoothness)**. Each local objective function is L-smooth, that is, $\forall x, y$*

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L \|x - y\|$$

Assumption 2. ***(Unbiased Local Stochastic Variance)**. The stochastic gradient at each client is an unbiased estimator of the local gradient: $E_{\xi_i}[g_i(\boldsymbol{x}|\xi)] = \nabla F_i(\boldsymbol{x})$, and has bounded variance*

$$\mathrm{E}_{\xi}\left[\|g_i(\boldsymbol{w}|\xi_i) - \nabla f_i(\boldsymbol{w})\|^2\right] \leq \sigma^2, \ \ \forall i \in \{1,\ 2,\ ,\ ...,\ m\},\ \sigma^2 \geq 0$$

Assumption 3. ***(Bounded Gradients)**. We assume that the unbiased gradients has bounded second moment: $\mathrm{E}\,\|g_i(\boldsymbol{w})\|^2 \leq G^2$.*

Assumption 4. ***(Bounded Asynchronous Delay)**. We assume that there exists a maximum delay t_d, which means a client must communicate with the server within t_d rounds ($t_d \geq \max\{\tau_i^t | i \in \mathcal{N},\ t \in [1, T]\}$).*

Assumption 1 and 4 are standard and commonly used in the literature on learning and optimization [6, 21, 28]. For Assumption 2, the boundedness of local stochastic gradients' variances is also a

Algorithm 1 Anarchic Federated Learning with Delayed Gradient Averaging (AFL-DGA),

input: local iteration numbers $\{K_i^t | i \in \mathcal{N},\ t \in [1,T]\}$, global gradient delays $\{D_i^t | i \in \mathcal{N},\ t \in [1,T]\}$, local computation delays $\{\tau_i^t | i \in \mathcal{N},\ t \in [1,T]\}$, initial global model $\boldsymbol{w}^0$;
for Round $t = 1$ to T **do**
  Sever:
  Collect the updates $\overline{\boldsymbol{g}_i^t}$ from communicating clients;
  if Client i communicates with the server **then**
    Set $\tau_i^t = 1$ and store $\overline{\boldsymbol{g}_i^t}$ on the server;
  end if
  if Client i does not communicate with the server **then**
    Set $\tau_i^t = \tau_i^{t-1} + 1$;
  end if
  $\overline{\boldsymbol{g}^t} = \frac{1}{N}\sum_{i=1}^N \overline{\boldsymbol{g}_i^{t-\tau_i^t}}$ and $\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \eta\overline{\boldsymbol{g}^t}$;
  Broadcast $\overline{\boldsymbol{g}^t}$;
  Client:
  for Client $i = 1$ to N **do**
    for round $s = t - \tau_i^t$ to t **do**
      if Receive the average global gradient $\overline{\boldsymbol{g}^s}$ **then**
        $\boldsymbol{w}_i^s = \boldsymbol{w}_i^{s-1} + \eta\overline{\boldsymbol{g}_i^{s-1}} - \eta\overline{\boldsymbol{g}^{s-1}}$, where $\boldsymbol{w}_i^0 = \boldsymbol{w}^0$;
      end if
      for Local iteration $k = 1$ to K_i^s **do**
        Sample the stochastic gradient $\boldsymbol{g}_{i,k}^s$ at the previous iterate $\boldsymbol{w}_{i,k}^s$;
        $\boldsymbol{w}_{i,k+1}^s = \boldsymbol{w}_{i,k}^s - \eta\boldsymbol{g}_{i,k}^s$;
      end for
    end for
    Send the t-th round accumulated local updates $\overline{\boldsymbol{g}_i^t} := \frac{1}{K_i^t}\sum_{h=1}^{K_i^t}\boldsymbol{g}_{i,h}^t$ to the server;
  end for
end for
Return: $\boldsymbol{w}^T$.

common assumption for prior work on FL with non-IID datasets [5, 17, 24]. Assumption 3 is used in some works [13].

4.1 Main Results

Next, we present theoretical performance guarantee for the AFL-DGA algorithm via convergence analysis.

Theorem 1. *Under Assumption 1, 2, 3 and 4. The sequence generated by delayed gradient averaging with stepsize $\eta \le \frac{1}{L}$ satisfies*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})\right\|^2\right] \le \frac{2}{\eta T}\left(\mathrm{E}\left[f(\overline{\boldsymbol{w}^0})\right] - \mathrm{E}\left[f(\overline{\boldsymbol{w}^T})\right]\right) + \frac{\eta L}{N}(\sigma^2+G^2)$$
$$+ L^2\eta^2(K_m+D_m)^2G^2 + \frac{L^2\eta^2}{N^2}(\sigma^2+G^2)\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i=1}^{N}(\tau_i^t + \tau_i^{t-\tau_i^t} - 1)$$

where $D_m = \max\{D_i^t | i \in [1,N],\ t \in [1,T]\}$ and $K_m = \max\{K_i^t | i \in [1,N],\ t \in [1,T]\}$.

Remark: We note that the convergence error bound consists of two parts: a vanishing term that decreases and goes to 0 as the number of rounds T increases, and a non-vanishing (constant) term which depends on the parameters of the problem instance and is independent of T. The decay rate of the vanishing term matches that of the typical SGD methods. The first part of the non-vanishing term (i.e., $\frac{\eta L}{N}(\sigma^2+G^2)$) is due to the local stochastic gradients used by each client, which shrinks at rate $1/N$ as the number of clients N increases.

We observe that the first part of the non-vanishing term involves the local gradient variance σ^2 and depends on the number of clients N. This error term is due to the variance of stochastic gradients, and it reduces at the rate of $1/N$. Intuitively, although clients' data are heterogeneous, the variance of the aggregated local stochastic gradients across clients is lower than that of a single client, which results in the variance reduction. The second part of the non-vanishing term (i.e., $L^2\eta^2(K_m+D_m)^2G^2$) depends on clients' local iteration numbers K_i^t, and it increases with K_i^t. Intuitively, due to clients' heterogeneous data, more local computation iterations drives each client's local model more towards its local optimal model and possibly away from the global optimal model (also known as "local drifts" in existing works on FL [13, 18]). As a result, the error bound increases as the local iteration numbers go up. To make the non-vanishing terms small, a sufficiently small learning rate η should be chosen.

Remark: We also observe that the error bound increases as clients' local model delays increase. This is because, as the local model delay increases, there is more error in the most recent local model used in the proposed algorithm compared to the actual local model without any delay. Therefore, the error increases when the delay is higher.

We observe that the convergence error bound also depends on clients' global gradient delays.

Remark: Since gradient computations can be performed simultaneously with gradient communications, the total time span of a round t can be reduced from $\max_i\{(K_i^t + D_i^t)d\}$ to $\max_i\{K_i^t d\}$ where d is the delay of one local computation iteration. Therefore, DGA can reduce the training time by a fold of $(K_i^t + D_i^t)/K_i^t$. This performance gain is substantial when communication delays are large.

Based on Theorem 1, we obtain the following convergence rate for the proposed AFL-DGA algorithm with a proper choice of the learning rate.

Corollary 1. *Let the stepsize $\eta = \frac{\sqrt{N}}{L\sqrt{T}}$ and $\Delta = \mathrm{E}\left[f(\overline{\boldsymbol{w}^0})\right] - \mathrm{E}\left[f(\overline{\boldsymbol{w}^T})\right]$ yields*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})\right\|^2\right] = O\left(\frac{1}{\sqrt{NT}}(L\Delta+\sigma^2+G^2)\right)$$
$$+ O\left(\frac{N}{T}(K_m+D_m)^2G^2\right) + O\left(\frac{N}{T}(2t_d-1)(\sigma^2+G^2)\right)$$

Remark: The result above shows that our AFL-DGA algorithm achieves a convergence rate of $O(\frac{1}{\sqrt{T}})$. It has been shown that asynchronous FL algorithms under the non-convex setting can achieve a convergence rate of $O(\frac{1}{\sqrt{T}})$ (e.g., AsyncCommSGD [3], AFA-CD [28]). As our algorithm which is asynchronous can reach

a convergence rate of $O(\frac{1}{\sqrt{T}})$, it matches that of the existing asynchronous algorithms.

We observe that when T is larger than some threshold (which depends on $(K_m + D_m)^2$), the first term of the bound is dominant, so that the error bound is $O(\frac{1}{\sqrt{NT}})$. This shows that AFL-DGA achieves a linear speedup despite non-IID datas of clients, which matches many existing algorithms.

Remark: Corollary 1 shows that the proposed Algorithm 1 can converge to the optimal value (rather than an error neighborhood) in the sense that the convergence error can be made arbitrarily small if the number of rounds t is large enough. It has been shown in prior work [22] that FL with arbitrary client participation results in a non-vanishing convergence error. This is due to an objective function drift under the worst-case scenario of client participation, regardless of the choices of learning rates and local iteration numbers. In our proposed algorithm, we use the most recent local model from a client in a round if the server does not receive a local model update from that client in that round. In this way, we show that the objective function drift can be addressed, despite of using the most recent local model rather than the actual local model from the client if the server would receive a local model update from that client in that round. In fact, the error between the most recent local model and the actual local model can be properly controlled by choosing an appropriate learning rate.

4.2 Discussions

Comparison with DGA. The DGA algorithm has been proposed and studied in [31] under ideal and simplified settings where all clients participate in each round in a synchronous manner with the same local iteration number. In this paper, we propose AFL-DGA under much more general settings which are practical but complex. Compared to DGA, AFL-DGA includes DGA as a special case and is much more non-trivial. In particular, the algorithm design of AFL-DGA has several major differences: 1) if a client does not participate in a round, the server uses the most recent averaged local gradient received from the client to compute the global gradient of the round; 2) a client computes the averaged (normalized) local gradient over its multiple local iterations in a round, which is used by the server to compute the global gradient for the round; 3) a client uses the delayed global gradient of a round to correct its local model, regardless of whether the client participates in that round or not. Due to these algorithmic differences of AFL-DGA, its convergence analysis is also different from that of DGA in non-trivial ways. In particular, the use of the most recent averaged local gradient of each client allows us to decompose an error term involving the global loss function's gradient into multiple error terms, each involving the gradient of only one client's local loss function. Also, using the delayed global gradient in every round to correct a client's local model allows us to quantify the error between the client's local model and the server's global model in a round. Moreover, in the convergence analysis of [31], as all clients use the same local iteration number, a bound is found on the error between a client's local model and the average local model of all clients in each local iteration. However, as heterogeneous local iteration numbers are considered in this paper, we need to bound the error between a client's local model and the global model in a round, which results in substantial differences in the analysis.

Comparison with AFL. A recent work [28] has proposed and studied the AFL algorithm, where clients can participate in a round or not in an asynchronous manner with different local iteration number. Compared to AFL, AFL-DGA integrates the key idea of the DGA algorithm, which is using delayed global gradients to correct a client's local model. This major algorithmic difference of AFL-DGA compared to AFL results in a non-trivial challenge in the convergence analysis, due to the coupling among asynchronous local gradient delays, global gradient delays, and local iteration numbers. In particular, the error between a client's local model and the server's global model (which results in the non-vanishing term in the convergence error) depends on all these three parameters.

5 NUMERICAL EXPERIEMENTS

In this section, we conduct experiments to verify our theoretical results.

5.1 Simulation Setup

We use i) logistic regression (LR) on manually partitioned non-i.i.d. MNIST dataset [11] ii) convolutional neural network (CNN) for image classification using CIFAR-10 [10]. To impose data heterogeneity, we distribute the data evenly to each worker in a label-based partition following the same process in the literature [12, 15, 27]. We use the code Federated-Learning-Master [1] and the above real datasets to verify our theoretical results.

We first compare the accuracy of 4 algorithms in i.i.d. and non-i.i.d data settings. Then we further simulate the relationship between communication rounds and test accuracy, and training loss, respectively.

5.2 Simulation Results

5.2.1 Training accuracy. : We conduct the following experiments with different algorithms. We choose 3 different algorithms, which are FedAvg, DGA [32], and our proposed AFL-DGA. We choose the stepsize at 0.01, local iteration is 10. From Table 1 and Fig 2, it can be found that the algorithm with DGA can maintain accuracy compared with FedAvg. As our algorithm allows clients to run heterogeneous local iterations and delayed iterations which better make use of training time while DGA [32] only allows homogeneous local iteration and delayed iterations, thus our AFL-DGA algorithm can reach a slightly higher test accuracy than DGA.

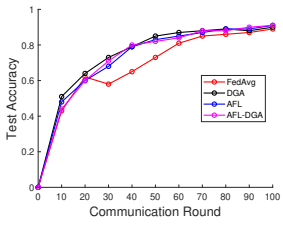

Figure 2: Test accuracy vs communication round for 4 algorithms.

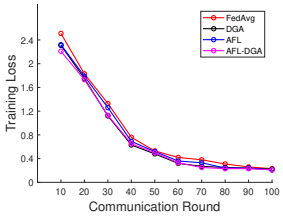

Figure 3: Training Loss vs communication round for 4 algorithms.

Table 1: Comparison of FedAvg, DGA's and our algorithm's accuracy on 2 datasets with both i.i.d and non-i.i.d partitions in synchronous federated learning.

Datasets	Partition	FedAvg (N=5)	FedAvg (N=10)	DGA [32] (N=5, D=20)	Our Algorithm (N=5, D=20, $t_d = 1$)
MNIST	i.i.d	89.1	90.1	90.2	90.7
	non-i.i.d	62.3	61.7	61.9	62.0
CIFAR-10	i.i.d	87.8	89.3	89.5	90.2
	non-i.i.d	68.6	67.2	69.3	69.1

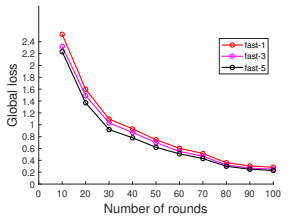

Figure 4: Training loss vs communication round under different schemes for AFL-DGA.

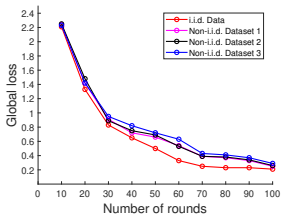

Figure 5: Training loss vs communication round under different non-i.i.d degree for AFL-DGA.

5.2.2 Impact of the scheme of AFL-DGA. In the previous experiments, we have proved that our AFL-DGA algorithm can work well in most settings, and we next test the effect of different communication schemes on the convergence speed. We choose the stepsize at 0.01, local iteration is 10, the number of clients is 30, the asynchronous delay $t_d = 2$, and the non-i.i.d. degree $\Gamma = 0$. As shown in Fig 4, fast-k refers to the server collecting the fast-s clients who finish their local computation under the maximum delay constraint in the IID setting. It can be found that more clients computing in a round (a larger s) can speed up the training process because more participating clients can let clients compute with a smaller delay model. When there are only a few clients can update their model in each round, then there must exist a straggler with a much higher maximum delay, which can make the aggregation result degradation.

5.2.3 Impact of the non-i.i.d. degree. In the previous experiments, we test the effect of the non-i.i.d. degree on the convergence speed. We choose the stepsize at 0.01, local iteration is 10, the number of clients is 30, the asynchronous delay $t_d = 2$, and the communication scheme is top-1. As shown in Fig 5, the non-i.i.d. degree of 4 datasets is increasing (Γ(i.i.d) < Γ(non-i.i.d dataset 1) < Γ(non-i.i.d dataset 2) < Γ(non-i.i.d dataset 3)). It is shown that the degree of non-IID affects the convergence rate, where a slower convergence speed with a higher non-i.i.d. degree, but it does not affect the final results, which meet our analyses.

5.2.4 Impact of the asynchronous delay. In this experiment, we test the effect of the asynchronous delay on the convergence speed. We choose the stepsize at 0.01, local iteration is 10, the number of clients is 30, the non-i.i.d. degree $\Gamma = 0$, and the communication scheme is top-1. As shown in Fig 6, the global loss increases with increasing maximum delay. For a synchronous FL ($t_d = 0$), which obviously has the lowest global loss among all situations. At the beginning of the FL training, the difference between the 4 different maximum delays is relatively small, since we only require the maximum delay, resulting in a small difference in clients' choices at the beginning of the training. When the training is nearly finished, the difference becomes larger and larger due to the effect of straggler clients.

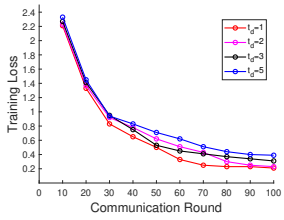

Figure 6: Training loss vs communication round with varies maximum delay for AFL-DGA.

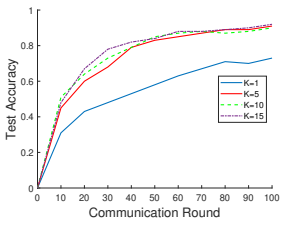

Figure 7: Test accuracy vs communication round under different local iterations for AFL-DGA.

5.2.5 Impact of the local iteration numbers. In this experiment, we test the effect of the local iteration numbers on the test accuracy. We choose the stepsize at 0.01, the number of clients is 30, the non-i.i.d. degree $\Gamma = 0$, the asynchronous delay $t_d = 2$, and the communication scheme is top-1. The fact that the clients are allowed to make several local iterations is a crucial component of FL algorithms. In this experiment, we investigate how varying local update round counts affect training efficiency. As shown in Fig 7, we test the accuracy of the AFL-DGA in i.i.d. data setting with homogeneous local iteration numbers. It is shown that when the local iteration is too small e.g., $K = 1$, it needs more communication rounds to reach convergence. Moreover, too many local iterations can not speed up the convergence, due to the local training may not contribute much to the global model as the local iteration increases.

5.2.6 Impact of the number of clients. In this experiment, we test the effect of the number of clients on the test accuracy. We choose the stepsize at 0.01, local iteration is 10, the non-i.i.d. degree $\Gamma = 0$, the asynchronous delay $t_d = 2$, and the communication scheme is top-1. We conduct the following experiments with different number of clients. We use the number of clients from the set $\{10, 30, 50, 70\}$. As shown in Fig 8, test accuracys with different number of clients have nearly similar performances.

5.2.7 Impact of the stepsize. In this experiment, we test the effect of the number of clients on the test accuracy. We choose the number

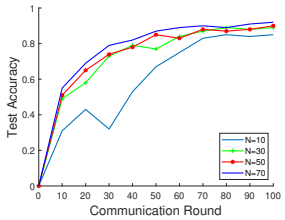

Figure 8: Training loss vs communication round with number of clients for AFL-DGA.

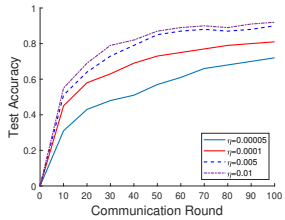

Figure 9: Test accuracy vs communication round under different stepsizes for AFL-DGA.

of clients is 30, local iteration is 10, the non-i.i.d. degree $\Gamma = 0$, the asynchronous delay $t_d = 2$, and the communication scheme is top-1. We use the stepsize from the set $\{0.0005, 0.001, 0.005, 0.01\}$. As shown in Fig. 9, larger local step-sizes lead to faster convergence rates.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose Anarchic Federated Learning with Delay Gradient Averaging (AFL-DGA) to deal with high communication latency in both synchronous and asynchronous federated learning. We have justified that the theoretical convergence is no slower than FedAvg in non-convex settings for both situations. We also loosen the restriction on local loss function gradients being bounded. Next, we demonstrate that our algorithm is capable of enjoying high scalability under poor network conditions while preserving accuracy, especially on non-i.i.id partitions. Finally, using realistic datasets, we run simulations. We think that a variety of applications in high latency networks among heterogeneous federated learning contexts will be made possible by our work.

For future work, we will explore AFL-DGA in other settings of FL, such as for decentralized networks of clients. These cases will be more challenging to study due to the complex communication structure.

REFERENCES

[1] [n. d.]. Federated-Learning-master. https://github.com/mysun95/Federated-Learning-master

[2] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. 2017. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*. 1709–1720.

[3] Dmitrii Avdiukhin and Shiva Kasiviswanathan. 2021. Federated learning under arbitrary communication patterns. In *International Conference on Machine Learning*. PMLR, 425–435.

[4] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konecny, Stefano Mazzocchi, H Brendan McMahan, et al. 2019. Towards federated learning at scale: System design. In *SysML Conference*.

[5] Léon Bottou, Frank E Curtis, and Jorge Nocedal. 2018. Optimization methods for large-scale machine learning. *Siam Review* 60, 2 (2018), 223–311.

[6] Zheng Chai, Yujing Chen, Liang Zhao, Yue Cheng, and Huzefa Rangwala. 2020. Fedat: A communication-efficient federated learning method with asynchronous tiers under non-iid data. *ArXiv* (2020).

[7] Yuanxiong Guo, Ying Sun, Rui Hu, and Yanmin Gong. 2022. Hybrid local SGD for federated learning with heterogeneous communications. In *International Conference on Learning Representations (ICLR)*.

[8] Yan Huang, Ying Sun, Zehan Zhu, Changzhi Yan, and Jinming Xu. 2022. Tackling data heterogeneity: A new unified framework for decentralized sgd with sample-induced topology. *International Conferene on Machine Learning (ICML)* (2022).

[9] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning (ICML)*.

[10] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[11] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[12] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2019. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189* (2019).

[13] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020. On the convergence of FedAvg on non-IID data. In *International Conference on Learning Representations (ICLR)*.

[14] Xiangru Lian, Yijun Huang, Yuncheng Li, and Ji Liu. 2015. Asynchronous parallel stochastic gradient for nonconvex optimization. In *Advances in Neural Information Processing Systems (NIPS)*. 2737–2745.

[15] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[16] Aritra Mitra, Rayana Jaafar, George J Pappas, and Hamed Hassani. 2021. Linear convergence in federated learning: Tackling client heterogeneity and sparse gradients. *Advances in Neural Information Processing Systems* 34 (2021), 14606–14619.

[17] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. 2020. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295* (2020).

[18] Sebastian U Stich. 2019. Local SGD converges fast and communicates little. In *International Conference on Learning Representations (ICLR)*.

[19] Ananda Theertha Suresh, Felix X Yu, Sanjiv Kumar, and H Brendan McMahan. 2017. Distributed mean estimation with limited communication. In *International Conference on Machine Learning*. 3329–3337.

[20] Nguyen H Tran, Wei Bao, Albert Zomaya, Nguyen Minh NH, and Choong Seon Hong. 2019. Federated learning over wireless networks: Optimization model design and analysis. In *International Conference on Computer Communications (INFOCOM)*. IEEE.

[21] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. 2020. Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. *Advances in Neural Information Processing Systems (NIPS)* (2020).

[22] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. 2021. A novel framework for the analysis and design of heterogeneous federated learning. *IEEE Transactions on Signal Processing* 69 (2021), 5234–5249.

[23] Shiqiang Wang and Mingyue Ji. 2022. A Unified Analysis of Federated Learning with Arbitrary Client Participation. In *Advances in Neural Information Processing Systems (NIPS)*.

[24] Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K Leung, Christian Makaya, Ting He, and Kevin Chan. 2019. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications* 37, 6 (2019), 1205–1221.

[25] Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang. 2018. Gradient sparsification for communication-efficient distributed optimization, In Advances in Neural Information Processing Systems (NIPS). *Advances in Neural Information Processing Systems*.

[26] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2017. Terngrad: Ternary gradients to reduce communication in distributed deep learning. In *Advances in neural information processing systems*. 1509–1519.

[27] Haibo Yang, Minghong Fang, and Jia Liu. 2021. Achieving linear speedup with partial worker participation in non-iid federated learning. *arXiv preprint arXiv:2101.11203* (2021).

[28] Haibo Yang, Xin Zhang, Prashant Khanduri, and Jia Liu. 2022. Anarchic federated learning. In *International Conference on Machine Learning (ICML)*. PMLR.

[29] Xin Zhang, Minghong Fang, Zhuqing Liu, Haibo Yang, Jia Liu, and Zhengyuan Zhu. 2022. Net-fleet: Achieving linear convergence speedup for fully decentralized federated learning with heterogeneous data. In *International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MobiHoc)*.

[30] Xin Zhang, Jia Liu, Zhengyuan Zhu, and Elizabeth Serena Bentley. 2021. GT-storm: taming sample, communication, and memory complexities in decentralized non-convex learning. In *International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MobiHoc)*.

[31] Ligeng Zhu, Hongzhou Lin, Yao Lu, Yujun Lin, and Song Han. 2021. Delayed gradient averaging: Tolerate the communication latency for federated learning. In *Advances in Neural Information Processing Systems (NIPS)*.

[32] Ligeng Zhu, Hongzhou Lin, Yao Lu, Yujun Lin, and Song Han. 2021. Delayed gradient averaging: Tolerate the communication latency for federated learning. *Advances in Neural Information Processing Systems* 34 (2021), 29995–30007.

A LEMMA 1 AND PROOF

Lemma 1. *The difference between the i-th client at j-th local iteration in round $t-\tau_i^t$ and the average parameter across all clients is uniformly bounded:*

$$\mathrm{E}\left[\left\|\overline{\boldsymbol{w}^t}-\boldsymbol{w}_{i,j}^{t-\tau_i^t}\right\|^2\right]\leq\sum_{j=t-\tau_i^t-\tau_i^{t-\tau_i^t}}^{t-1}\mathrm{E}\left[\left\|\eta\overline{\boldsymbol{g}^j}\right\|^2\right]+\eta^2(K_m+D_m)^2G^2$$

Proof: For parameter $\overline{\boldsymbol{w}^t}$ and $\boldsymbol{w}_{i,j}^{t-\tau_i^t}$, we need to decide when client i communicates with the server. If $D^{t-\tau_i^t-1}<K^{t-\tau_i^t}$, then we can find that client i communicates with the server in round $t-\tau_i^t-1$. Otherwise, if $D^{t-\tau_i^t-1}>K^{t-\tau_i^t}$, and $D^{t-\tau_i^t-2}<K^{t-\tau_i^t-1}+K^{t-\tau_i^t}$, then we can find that client i communicates with the server in round $t-\tau_i^t-2$. This process can continue to the time when $D^{x-1}<K^x$, thus we can get that $j\leq D_m+K_m$, where $D_m=\max\{D_i^t|i\in[1,N],\ t\in[1,T]\}$ and $K_m=\max\{K_i^t|i\in[1,N],\ t\in[1,T]\}$.

In addition, we can obtain

$$\begin{aligned}
&\mathrm{E}\left[\left\|\overline{\boldsymbol{w}^t}-\boldsymbol{w}_{i,j}^{t-\tau_i^t}\right\|^2\right]\\
&=\mathrm{E}\left[\left\|(\overline{\boldsymbol{w}^0}-\eta\sum_{h=1}^{t-1}\overline{\boldsymbol{g}^h})-(\overline{\boldsymbol{w}^0}-\eta\sum_{h=1}^{t-\tau_i^t-\tau_i^{t-\tau_i^t}-1}\overline{\boldsymbol{g}^h}\right.\right.\\
&\left.\left.-\eta\sum_{h=t-\tau_i^t-\tau_i^{t-\tau_i^t}}^{t-1}\overline{\boldsymbol{g}_i^h}-\eta\sum_{h=1}^{j-1}\boldsymbol{g}_{i,h}^{t-1})\right\|^2\right]\\
&\overset{d_1}{\leq}\sum_{j=t-\tau_i^t-\tau_i^{t-\tau_i^t}}^{t-1}\mathrm{E}\left[\left\|\eta\overline{\boldsymbol{g}^j}\right\|^2\right]+\eta^2(K_m+D_m)^2G^2 \qquad (1)
\end{aligned}$$

where d_1 follows Assumption 3.

From (3), we can get

$$\begin{aligned}
&\mathrm{E}\left[\left\|-\eta\overline{\boldsymbol{g}^t}\right\|^2\right]=\eta^2\mathrm{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\boldsymbol{g}_{i,j}^{t-\tau_i^t}\right\|^2\right]\\
&\overset{b_1}{\leq}\eta^2\frac{1}{N^2}\sum_{i=1}^N\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\mathrm{E}\left[\left\|\boldsymbol{g}_{i,j}^{t-\tau_i^t}\right\|^2\right]\\
&=\eta^2\frac{1}{N^2}\sum_{i=1}^N\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\mathrm{E}\left[\left\|\boldsymbol{g}_{i,j}^{t-\tau_i^t}-\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right.\right.\\
&\left.\left.+\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]\\
&\leq\eta^2\frac{1}{N^2}\sum_{i=1}^N\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\left(\mathrm{E}\left[\left\|\boldsymbol{g}_{i,j}^{t-\tau_i^t}-\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]\right.\\
&\left.+\mathrm{E}\left[\left\|\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]\right)\\
&\overset{b_2}{\leq}\eta^2\frac{1}{N^2}\sum_{i=1}^N\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\left(\sigma^2+\mathrm{E}\left[\left\|\nabla f_i(\boldsymbol{w}_i^{t-\tau_i^t})\right\|^2\right]\right)\\
&=\frac{\eta^2}{N}\left(\sigma^2+G^2\right) \qquad (2)
\end{aligned}$$

where b_1 follows Jensen's inequality, and b_2 is from Assumption 2.

Thus, we can get

$$\begin{aligned}
\sum_{j=t-\tau_i^t-\tau_i^{t-\tau_i^t}}^{t-1}\mathrm{E}\left[\left\|\eta\overline{\boldsymbol{g}^j}\right\|^2\right]&\leq\sum_{j=t-\tau_i^t-\tau_i^{t-\tau_i^t}}^{t-1}\frac{\eta^2}{N}\left(\sigma^2+G^2\right)\\
&=\frac{\eta^2}{N}(\tau_i^t+\tau_i^{t-\tau_i^t}-1)(\sigma^2+G^2)
\end{aligned}$$

Thus,

$$\mathrm{E}\left[\left\|\overline{\boldsymbol{w}^t}-\boldsymbol{w}_{i,j}^{t-\tau_i^t}\right\|^2\right]\leq\sum_{j=t-\tau_i^t-\tau_i^{t-\tau_i^t}}^{t-1}\mathrm{E}\left[\left\|\eta\overline{\boldsymbol{g}^j}\right\|^2\right]+\eta^2(K_m+D_m)^2G^2$$

This completes the proof.

B PROOF OF THEOREM 1

Proof:

By the smoothness of f, we have

$$\mathrm{E}\left[f(\overline{\boldsymbol{w}^{t+1}})\right]-\mathrm{E}\left[f(\overline{\boldsymbol{w}^t})\right]\leq\mathrm{E}\left[\left\langle\nabla f(\overline{\boldsymbol{w}^{t+1}}),\ \overline{\boldsymbol{w}^{t+1}}-\overline{\boldsymbol{w}^t}\right\rangle+\frac{L}{2}\left\|\overline{\boldsymbol{w}^{t+1}}-\overline{\boldsymbol{w}^t}\right\|^2\right]$$

For $\mathrm{E}\left[\frac{L}{2}\left\|\overline{\boldsymbol{w}^{t+1}}-\overline{\boldsymbol{w}^t}\right\|^2\right]$, we have

$$\begin{aligned}
&\mathrm{E}\left[\frac{L}{2}\left\|\overline{\boldsymbol{w}^{t+1}}-\overline{\boldsymbol{w}^t}\right\|^2\right]=\frac{L}{2}\mathrm{E}\left[\left\|-\eta\overline{\boldsymbol{g}^t}\right\|^2\right]\\
&=\eta^2\frac{L}{2}\mathrm{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\boldsymbol{g}_{i,j}^{t-\tau_i^t}\right\|^2\right]\\
&\overset{b_1}{\leq}\eta^2\frac{L}{2}\frac{1}{N^2}\sum_{i=1}^N\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\mathrm{E}\left[\left\|\boldsymbol{g}_{i,j}^{t-\tau_i^t}\right\|^2\right]\\
&=\eta^2\frac{L}{2}\frac{1}{N^2}\sum_{i=1}^N\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\mathrm{E}\left[\left\|\boldsymbol{g}_{i,j}^{t-\tau_i^t}-\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right.\right.\\
&\left.\left.+\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]\\
&\leq\eta^2\frac{L}{2}\frac{1}{N^2}\sum_{i=1}^N\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\left(\mathrm{E}\left[\left\|\boldsymbol{g}_{i,j}^{t-\tau_i^t}-\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]\right.\\
&\left.+\mathrm{E}\left[\left\|\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]\right)
\end{aligned}$$

$$\overset{b_2}{\leq}\eta^2\frac{L}{2}\frac{1}{N^2}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\left(\sigma^2+\mathrm{E}\left[\left\|\nabla f_i(\boldsymbol{w}_i^{t-\tau_i^t})\right\|^2\right]\right)$$

$$=\eta^2\frac{L}{2}\frac{\sigma^2}{N}+\eta^2\frac{L}{2}\frac{1}{N^2}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\mathrm{E}\left[\left\|\nabla f_i(\boldsymbol{w}_i^{t-\tau_i^t})\right\|^2\right] \quad (3)$$

where b_1 follows Jensen's inequality, and b_2 is from Assumption 2.

Moreover,

$$\mathrm{E}\left[\left\langle\nabla f(\overline{\boldsymbol{w}^t}),\ \overline{\boldsymbol{w}^{t+1}}-\overline{\boldsymbol{w}^t}\right\rangle\right]$$
$$=\mathrm{E}\left[\left\langle\nabla f(\overline{\boldsymbol{w}^t}),\ -\eta\overline{\boldsymbol{g}^t}\right\rangle\right]=-\eta\mathrm{E}\left[\left\langle\nabla f(\overline{\boldsymbol{w}^t}),\ \overline{\boldsymbol{g}^t}\right\rangle\right]$$
$$=-\eta\mathrm{E}\left[\left\langle\nabla f(\overline{\boldsymbol{w}^t}),\ \frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\rangle\right]$$
$$=\frac{\eta}{2}\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})-\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]$$
$$-\frac{\eta}{2}\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})\right\|^2\right]-\frac{\eta}{2}\mathrm{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right] \quad (4)$$

Then for $\frac{\eta}{2}\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})-\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]$, we have

$$\frac{\eta}{2}\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})-\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]$$
$$\overset{c_1}{\leq}\frac{\eta}{2}\frac{1}{N}\sum_{i=1}^{N}\mathrm{E}\left[\left\|\nabla f_i(\overline{\boldsymbol{w}^t})-\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]$$
$$=\frac{\eta}{2}\frac{1}{N}\sum_{i=1}^{N}\mathrm{E}\left[\left\|\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\left(\nabla f_i(\overline{\boldsymbol{w}^t})-\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right)\right\|^2\right]$$
$$\overset{c_2}{\leq}\frac{\eta}{2}\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\mathrm{E}\left[\left\|\nabla f_i(\overline{\boldsymbol{w}^t})-\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]$$
$$\overset{c_3}{\leq}\frac{\eta}{2}\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}L^2\mathrm{E}\left[\left\|\overline{\boldsymbol{w}^t}-\boldsymbol{w}_{i,j}^{t-\tau_i^t}\right\|^2\right] \quad (5)$$

where c_1 and c_2 follows Jensen's inequality, and c_3 follows L-smoothness.

Thus, using Lemma 1 to substitute (5), we can get

$$\frac{\eta}{2}\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})-\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]$$
$$\leq\frac{\eta}{2}\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}L^2\mathrm{E}\left[\left\|\overline{\boldsymbol{w}^t}-\boldsymbol{w}_{i,j}^{t-\tau_i^t}\right\|^2\right]$$
$$\leq\frac{\eta}{2}\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}L^2\left(\frac{\eta^2}{N}(\tau_i^t+\tau_i^{t-\tau_i^t}-1)(\sigma^2+G^2)\right.$$
$$\left.+\eta^2(K_m+D_m)^2G^2\right)$$
$$=\frac{L^2\eta^3}{2N^2}(\sigma^2+G^2)\sum_{i=1}^{N}(\tau_i^t+\tau_i^{t-\tau_i^t}-1)+\frac{L^2\eta^3}{2}(K_m+D_m)^2G^2 \quad (6)$$

Next, combining (3), (4), and (6) together, we have

$$\mathrm{E}\left[f(\overline{\boldsymbol{w}^{t+1}})\right]-\mathrm{E}\left[f(\overline{\boldsymbol{w}^t})\right]$$
$$\leq\frac{L^2\eta}{2}\left(\frac{\eta^2}{N}(\tau_i^t+\tau_i^{t-\tau_i^t}-1)(\sigma^2+G^2)+\eta^2(K_m+D_m)^2G^2\right)$$
$$-\frac{\eta}{2}\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})\right\|^2\right]-\frac{\eta}{2}\mathrm{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\nabla f_i(\boldsymbol{w}_{i,j}^{t-\tau_i^t})\right\|^2\right]$$
$$+\eta^2\frac{L}{2}\frac{\sigma^2}{N}+\eta^2\frac{L}{2}\frac{1}{N^2}\sum_{i=1}^{N}\frac{1}{K_i^{t-\tau_i^t}}\sum_{j=1}^{K_i^{t-\tau_i^t}}\mathrm{E}\left[\left\|\nabla f_i(\boldsymbol{w}_i^{t-\tau_i^t})\right\|^2\right]$$

Thus, we can get

$$\mathrm{E}\left[f(\overline{\boldsymbol{w}^{t+1}})\right]-\mathrm{E}\left[f(\overline{\boldsymbol{w}^t})\right]$$
$$\leq\frac{L^2\eta^3}{2N^2}(\sigma^2+G^2)\sum_{i=1}^{N}(\tau_i^t+\tau_i^{t-\tau_i^t}-1)+\frac{L^2\eta^3}{2}(K_m+D_m)^2G^2$$
$$-\frac{\eta}{2}\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})\right\|^2\right]+\eta^2\frac{L}{2}\frac{\sigma^2}{N}+\eta^2\frac{L}{2}\frac{1}{N}G^2$$

Moreover,

$$\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})\right\|^2\right]$$
$$\leq\frac{2}{\eta}\left(\mathrm{E}\left[f(\overline{\boldsymbol{w}^t})\right]-\mathrm{E}\left[f(\overline{\boldsymbol{w}^{t+1}})\right]\right)+\frac{L^2\eta^3}{2N^2}(\sigma^2+G^2)\sum_{i=1}^{N}(\tau_i^t+\tau_i^{t-\tau_i^t}-1)$$
$$+\frac{L^2\eta^3}{2}(K_m+D_m)^2G^2+\frac{L\eta}{N}\sigma^2+\frac{L\eta}{N}G^2$$

Telescoping from $t=1,\ldots,T$, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\left\|\nabla f(\overline{\boldsymbol{w}^t})\right\|^2\right]\leq\frac{2}{\eta T}\left(\mathrm{E}\left[f(\overline{\boldsymbol{w}^0})\right]-\mathrm{E}\left[f(\overline{\boldsymbol{w}^T})\right]\right)+\frac{\eta L}{N}(\sigma^2+G^2)$$
$$+L^2\eta^2(K_m+D_m)^2G^2++\frac{L^2\eta^2}{N^2}(\sigma^2+G^2)\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i=1}^{N}(\tau_i^t+\tau_i^{t-\tau_i^t}-1)$$

This completes the proof.