SWIFT: Rapid Decentralized Federated Learning via Wait-Free Model Communication

Anonymous Author(s)
Affiliation
Address
email

Abstract

The decentralized Federated Learning (FL) setting avoids the role of a potentially unreliable or untrustworthy central host by utilizing groups of clients to collaboratively train a model via localized training and model/gradient sharing. Most existing decentralized FL algorithms require synchronization of client models where the speed of synchronization depends upon the slowest client. In this work, we propose SWIFT: a wait-free decentralized FL algorithm that allows clients to conduct training at their own speed. Theoretically, we prove that SWIFT matches the gold-standard convergence rate $\mathcal{O}(1/\sqrt{T})$ of parallel stochastic gradient descent for convex and non-convex smooth optimization (total iterations T). Furthermore, this is done in the IID and non-IID settings *without* any bounded-delay assumption for slow clients, which is required by other asynchronous decentralized FL algorithms. Although SWIFT achieves the same convergence rate with respect to T as other state-of-the-art (SOTA) parallel stochastic algorithms, it converges faster with respect to time due to its wait-free structure. Our experimental results demonstrate that communication costs between clients in SWIFT fall by an order of magnitude compared to synchronous counterparts. Furthermore, SWIFT produces loss levels for image classification, over IID and non-IID data settings, upwards of 50% faster than existing SOTA algorithms.

1 Introduction

Federated Learning (FL) is an increasingly popular setting to train powerful deep neural networks with data derived from an assortment of devices. Recent research has focused on constructing decentralized FL algorithms which overcome speed and scalability issues found within classical centralized FL [21, 19, 17, 29, 30]. Nevertheless, the speed of current decentralized algorithms is diminished by parallelization delays caused by the slowest client in the network. Few asynchronous decentralized FL algorithms have been floated to solve this issue [20], yet their communication costs remain large and their communication protocols (via gossip algorithms) do not propagate models well throughout the network. Furthermore, these asynchronous algorithms rely on a bounded-delay assumption, which assumes that the slowest client in the network performs an update at least every τ iterations. Ensuring low-communication costs is crucial for decentralized algorithms to be scalable.

To remedy these drawbacks, we propose the **S**hared **W**a**I**t-**F**ree **T**ransmission (SWIFT) algorithm: an efficient, scalable, and high-performing decentralized FL algorithm. Unlike other decentralized FL algorithms, SWIFT obtains minimal communication costs due to its wait-free structure. Furthermore, SWIFT is the first asynchronous decentralized FL algorithm to obtain an optimal $\mathcal{O}(1/\sqrt{T})$ convergence rate (aligning with parallel stochastic gradient descent) *without a bounded-delay assumption*. Our theoretical analysis shows that SWIFT converges in fewer total iterations with respect to the total number of clients than its peer asynchronous decentralized FL algorithms [20]. SWIFT improves

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

upon the lineage of feasible large-scale decentralized algorithms because it boosts convergence rates (scalability) while keeping run-times low due to its low communication costs. Experiments validate SWIFT's efficiency, showcasing a reduction in communication costs by nearly an order of magnitude and run-times of almost 50%. All the while, SWIFT remains at state-of-the-art (SOTA) global test/train loss for image classification compared to other decentralized FL algorithms.

Summary of contributions: **(1)** Construct the first wait-free decentralized FL algorithm, guaranteeing convergence without a bounded-delay assumption. **(2)** Implement the first pre-processing algorithm to ensure non-symmetric and non-doubly stochastic communication matrices are symmetric and doubly-stochastic under expectation. **(3)** Reduce the dependency of total algorithm iterations T on total clients n from $T = \mathcal{O}(n^4)$ to $T = \mathcal{O}(n^2)$ under common decentralized settings. **(4)** Provide the first theoretical error bound for non-symmetric and non-doubly stochastic communication matrices in the asynchronous setting. **(5)** Achieve a reduction in communication costs by up to a magnitude and run-times by up to 50% compared to synchronous decentralized FL baseline algorithms.

2 Problem Formulation

Decentralized FL. In the FL setting, we have n clients which are represented as the nodes of an arbitrary communication graph $\mathcal{G}$ with vertex set $\mathcal{V} = \{1, \ldots, n\}$ and edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. Each client i communicates with one-hop neighboring clients j such that $(i, j) \in \mathcal{E}$. All devices work in tandem to find the optimal model parameters for the following unconstrained optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x) := \sum_{i=1}^{n} p_i f_i(x), \quad f_i(x) := \mathbb{E}_{\xi \sim \mathcal{D}_i}\big[\ell(x, \xi)\big], \quad \sum_{i=1}^{n} p_i = 1, \quad p_i \geq 0. \tag{1}$$

The global objective function $f(x)$ is the weighted combination of all local objective functions $f_i(x)$. The client influence scores $p_i \; \forall i \in [n]$ indicate each client's influence on the global consensus model, forming the client influence vector $p = \{p_i\}_{i=1}^n$. The scores also reflect the sampling probability of each client. Each local objective function $f_i(x)$ is the same loss function ℓ evaluated with potentially different local data $\xi_i = \{\xi_{i,j}\}_{j=1}^M$ from each client i's distribution $\mathcal{D}_i$, i.e., $\xi_{i,j} \in \mathbb{R}^f \sim \mathcal{D}_i$.

Existing Inter-Client Communication in Decentralized FL. Clients balance their individual training with inter-client communication. Communication occurs between connected clients within a neighborhood constituting all connected clients of a specific client. Balancing individual training with inter-client communication ensures that individual client models are well-tailored to one's own data while remaining (i) robust to other client data and (ii) able to converge to an optimal consensus model. Foundational algorithms, including Decentralized SGD (D-SGD) [19], Periodic Averaging SGD (PA-SGD) [29] and Local Decentralized SGD (LD-SGD) [17], are reviewed in detail in Appendix E.

3 Shared WaIt-Free Transmission (SWIFT) Federated Learning

In this section, we present the **S**hared **WaI**t-**F**ree **T**ransmission (SWIFT) Algorithm. SWIFT is an asynchronous algorithm that allows clients to work at their own speed. Therefore, it removes the dependency on the slowest client found in the synchronous setting. Moreover, unlike other asynchronous algorithms, SWIFT does not require a bound on the speed of the slowest client in the network. Instead, the SWIFT design allows clients to update their model only when they are ready.

A SWIFT Overview. Each client i runs SWIFT in parallel, first receiving an initial model x_i, communication set I_1, and counter $c_i \leftarrow 0$. SWIFT is concisely summarized in the following steps:
(0) Determine client-communication weights w_i via Algorithm 2 in Appendix D.2.
(1) Broadcast the local model to all neighboring clients.
(2) Sample a random local data batch of size M.
(3) Compute the gradient update of the loss function ℓ with the sampled local data.
(4) Fetch and store neighboring local models, and average them with one's own local model if $c_i \in I_1$.
(5) Update the local model with the computed gradient update, as well as the counter $c_i \leftarrow c_i + 1$.
(6) Repeat steps **(1)**-**(5)** until convergence.

Active Clients, Asynchronous Iterations, and the Local-Model Matrix. Each time a client finishes a pass through steps **(1)**-**(5)**, one global iteration is performed. Thus, the global iteration t is increased after the completion of *any* client's averaging and local gradient update. The client that performs the t-th iteration is called the active client, and is designated as i_t. *There is only one active*

Algorithm 1: **S**hared **Wa**It-**F**ree **T**ransmission (SWIFT)

Input : Vertex set $\mathcal{V}$, Total steps T, Step-size γ, Client Influence Vector p, Distributions of local client data $\mathcal{D}_i$, Model communication set I_1, Batch size M, Loss function ℓ, and Initial model X^0

Output : Consensus model $X^T 1_n / n$

Initialize each client's local update counter $c_i \leftarrow 0,\ \forall i \in \mathcal{V}$

for $t = 1, \ldots, T$ **do**

  Randomly sample a client i_t, from probability vector p, and get its communication vector $w^t_{i_t}$

  Broadcast $x^t_{i_t}$ to all its neighbors $\{k \mid w^t_{i_t,k} \neq 0, k \neq i_t\}$

  Randomly sample a batch of client i_t's local data $\xi^t_{i_t} := \{\xi^t_{i_t,m}\}_{m=1}^M$ from distribution $\mathcal{D}_{i_t}$

  Compute the gradient update $g(x^t_{i_t}, \xi^t_{i_t}) = \frac{1}{M}\sum_{m=1}^M \nabla \ell(x^t_{i_t}, \xi^t_{i_t,m})$

  if $c_{i_t} \in I_1$ **then**

    Activate model averaging, set $W^t_{i_t} = I_n + (w^t_{i_t} - \boldsymbol{e}_{i_t})\boldsymbol{e}_{i_t}^\intercal$

    Fetch and store local models x^t_k from all neighbors $\{k \mid w^t_{i_t,k} \neq 0, k \neq i_t\}$

  else

    Do not perform any averaging, set $W^t_{i_t} = I_n$

  Average active client i_t's local model $X^{t+1/2} = X^t W^t_{i_t}$

  Update active client $x^{t+1}_{i_t} = x^{t+1/2}_{i_t} - \gamma g(x^t_{i_t}; \xi^t_{i_t})$ and others $x^{t+1}_j = x^{t+1/2}_j\ \forall j \neq i_t$

  Update active client's counter $c_{i_t} \leftarrow c_{i_t} + 1$

client per global iteration. All other client models remain unchanged during the t-th iteration. In synchronous algorithms, the global iteration t increases *only after all clients finish an update*. For *asynchronous* algorithms, however, the global iteration t increases after *any* client finishes an update. In SWIFT, we define $X^t \in \mathbb{R}^{d\times n}$ as the concatenation of all local client models at iteration t,

$$X^t := [x^t_1, \ldots, x^t_n] \in \mathbb{R}^{d\times n}. \tag{2}$$

Inspired by PA-SGD [29], SWIFT handles multiple local gradient steps before averaging models amongst neighboring clients. The number of local updates before averaging is specified from a user-inputted communication set I_1. The periodic averaging process, governed by a dynamic client-communication matrix, is described further on in Section 3.1.

Update Rule. SWIFT runs in parallel with all clients performing local gradient updates and model communication simultaneously.

$$X^{t+1} = X^t W^t_{i_t} - \gamma G(x^t_{i_t}, \xi^t_{i_t}). \tag{3}$$

The matrix $G(x^t_{i_t}, \xi^t_{i_t}) \in \mathbb{R}^{d\times n}$ is the zero-padded gradient of the active model $x^t_{i_t}$. The entries of $G(x^t_{i_t}, \xi^t_{i_t})$ are zero except for the i_t-th column, which contains the active gradient $g(x^t_{i_t}, \xi^t_{i_t})$. The client-communication matrix $W^t_{i_t}$ is described in the section below and is defined in Equation (5). The parameter γ denotes the step-size. Last, we define the recursion of the SWIFT update as

$$X^{t+1} = X^0 - \gamma \sum_{j=0}^{t} G(x^j_{i_j}, \xi^j_{i_j}) \prod_{q=j+1}^{t} W^q_{i_q}. \tag{4}$$

3.1 SWIFT Client-Communication Matrix

The backbone of decentralized FL algorithms is the client-communication matrix W (also known as the weighting matrix). To remove all forms of synchronization and to become wait-free, SWIFT relies upon a novel client-communication matrix $W^t_{i_t}$ that is neither symmetric nor doubly-stochastic. This is unlike any other algorithm in FL [29, 20, 17, 14]. The result of a non-symmetric and non-doubly stochastic client-communication matrix, is that averaging occurs for a single active client i_t and not over a pair or neighborhood of clients. This curbs superfluous communication costs.

Defining the Active Client-Communication Matrix. The active client-communication matrix in SWIFT is notated as $W^t_{i_t}$, where i_t is the active client which performs the t-th global iteration. As mentioned in the SWIFT overview, a dynamic client-communication matrix is implemented to allow

for periodic averaging. Thus, the active client-communication matrix $W_{i_t}^t$ can be one of two forms: (1) an identity matrix $W_{i_t}^t = I_n$ if $c_{i_t} \notin I_1$ or (2) a communication matrix if $c_{i_t} \in I_1$ with structure

$$W_{i_t}^t := I_n + (w_{i_t}^t - \boldsymbol{e}_{i_t})\boldsymbol{e}_{i_t}^\intercal,\ w_{i_t}^t := [w_{1,i_t}^t, \ldots, w_{n,i_t}^t]^\intercal \in \mathbb{R}^n,\ \sum_{j=1}^n w_{j,i} = 1,\ w_{i,i} \geq 1/n\ \forall i. \quad (5)$$

The vector $w_{i_t}^t \in \mathbb{R}^n$ denotes *the active client-communication vector* at iteration t. The active client-communication vector contains the communication coefficients between client i_t and all clients (including itself) in the graph at iteration t. The client-communication coefficients form a weighted average of local neighboring models: the coefficients sum to 1. This vector is often sparse, as clients are connected to few other clients relative to the total number of clients in most decentralized settings.

3.2 Novel Client-Communication Weight Selection

While utilizing a non-symmetric and non-doubly-stochastic client-communication matrix decreases communication costs, there are technical difficulties when it comes to guaranteeing the convergence. One of the novelties of our work is that we carefully design a client-communication matrix $W_{i_t}^t$ such that it is symmetric and doubly-stochastic *under expectation of all potential active clients* i_t and has diagonal values greater than or equal to $1/n$.

$$\mathbb{E}_{i_t}\left[W_{i_t}^t\right] = \sum_{i=1}^n p_i\left[I_n + w_i^t \boldsymbol{e}_i^\intercal\right] = I_n + \sum_{i=1}^n p_i(w_i^t - \boldsymbol{e}_i)\boldsymbol{e}_i^\intercal = \bar{W}^t. \quad (6)$$

We denote $\bar{W}^t$ as the *expected client-communication matrix* with the following form

$$\bar{W}^t = \begin{pmatrix} 1 + p_1(w_{1,1}^t - 1) & p_2 w_{1,2}^t & \cdots & p_n w_{1,n}^t \\ p_1 w_{2,1}^t & 1 + p_2(w_{2,2}^t - 1) & & \\ \vdots & & \ddots & \\ p_1 w_{n,1}^t & & & 1 + p_n(w_{n,n}^t - 1) \end{pmatrix} \quad (7)$$

As one can see from the expanded form of $\bar{W}^t$, its columns sum to one. If we ensure that $\bar{W}^t$ is symmetric, then it will become doubly-stochastic. By Equation (7), $\bar{W}^t$ becomes symmetric if

$$p_j w_{i,j}^t = p_i w_{j,i}^t\ \forall i, j \in \mathcal{V}. \quad (8)$$

To achieve the symmetry of Equation (8), SWIFT deploys a novel pre-processing algorithm. Given any client-influence vector p, our Communication Coefficient Selection (CCS) Algorithm allows all clients to obtain deterministic client-communication coefficients such that Equations (5) and (8) hold for all global iterations t. Algorithm 2, shown in Appendix D.2, is run by each client i in the graph.

The CCS Algorithm is a waterfall method: clients wait for their larger-degree neighbors to finish.
(1) Wait to receive coefficients from larger-degree neighbors. If the largest, or tied, skip to **(2)**.
(2) Calculate the total coefficients already assigned s_w as well as the sum of the client influence scores for the unassigned clients s_p.
(3) Assign the leftover coefficients $1 - s_w$ to the remaining unassigned neighbors (and self) in a manner proportional to each unassigned client i's percentage of the leftover influence scores p_i/s_p.
(4) If tied with neighbors in degree size, ensure assigned coefficients won't sum to larger than one.
(5) Send coefficients to waiting neighbors.

In Algorithm 2, the terms s_w and s_p play a pivotal role in satisfying Equations (5) and (8) respectively. Equation (8) is satisfied by assigning weights amongst client i and its neighbors j as follows

$$p_j \frac{(1 - s_w)p_i}{s_p} = p_i \frac{(1 - s_w)p_j}{s_p}. \quad (9)$$

Assigning weights proportionally with respect to neighboring client influence scores and total neighbors ensures higher influence clients receive higher weighting during averaging.

4 SWIFT Theoretical Analysis

Before presenting our analysis, we provide some standard assumptions that are commonly used in the analysis of stochastic gradient descent in non-convex optimization.

Assumptions. *All objective functions are L-gradient Lipschitz* $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$. *The gradient is unbiased* $\mathbb{E}_{\xi\sim\mathcal{D}}\left[\nabla \ell(x;\xi)\right] = \nabla f(x)$. *The variance of the stochastic gradient is bounded* $\mathbb{E}_{\xi\sim\mathcal{D}_i}\|\nabla f_i(x) - \nabla \ell(x;\xi)\|^2 \leq \sigma^2$, *and* $\mathbb{E}_i\|\nabla f(x) - \nabla f_i(x)\|^2 \leq \zeta^2\ \forall i$.

Since we use a symmetric and doubly stochastic communication matrix $\bar{W}^t$, it is proven in Lemma 2 that there exists a $\rho \in [0, 1)$ such that $\max\{|\lambda_2\big((\bar{W}^t)^\intercal \bar{W}^t\big)|, |\lambda_n\big((\bar{W}^t)^\intercal \bar{W}^t\big)|\} \leq \rho, \ \forall t$. This parameter ρ reflects the connectivity of the underlying graph topology. The value of ρ is inversely proportionate to how fast information spreads in the client network. A small value of ρ results in information spreading faster ($\rho = 0$ in the centralized setting). Furthermore, since we use a column stochastic communication matrix $W_{i_t}^t$, it is proven in Lemma 3 that the product of stochastic matrices converges exponentially to a stochastic vector with common ratio ν. The parameter ρ_ν, defined in Equation (114), consists of ρ and ν. Finally, for ease of notation, we define $\Delta_f := f(\bar{x}^0) - f(\bar{x}^*)$.

As mentioned in Section 3, the use of a non-symmetric, non-doubly-stochastic matrix $W_{i_t}^t$ causes issues in analysis. To alleviate this issue, our novel pre-processing algorithm, Algorithm 2, selects client weights such that $\bar{W}^t$ is symmetric and doubly-stochastic for every global iteration t. Now, we first rewrite Equation (3) adding and subtracting out $\bar{W}^t$,

$$X^{t+1} = X^t \bar{W}^t + X^t(W_{i_t}^t - \bar{W}^t) - \gamma G(x_{i_t}^t, \xi_{i_t}^t). \tag{10}$$

Lemma 1 (Client-Communication Error Bound)**.** *Given the use of Algorithm 2, the product of the difference between the expected and actual client communication matrices is bounded by:*

$$\mathbb{E}\sum_{j=0}^{t}\left\|G(x_{i_j}^j, \xi_{*,j}^{i_j}) \prod_{q=j+1}^{t}(W_{i_q}^q - \bar{W}^q)(\frac{1_n}{n} - \boldsymbol{e}_i)\right\|^2 \leq \mathcal{O}(\frac{\sigma^2}{M} + \mathbb{E}\sum_{j=0}^{t}\left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2). \tag{11}$$

Remark. One novelty of our work is that we are the first to bound the client-communication error in the asynchronous decentralized FL setting. Our theoretical analysis is the first to tackle this error, because other decentralized works [19, 17, 29, 20] do not incorporate wait-free communication.

Theorem 1. *Given the use of Algorithm 2, the assumptions above, $I_1 = 0$, the step-size γ, total iteration T, and average model $\bar{x}^t$ for iteration t are defined as*

$$\gamma := \frac{\sqrt{Mn^2\Delta_f}}{\sqrt{TL} + \sqrt{M}} \leq \sqrt{\frac{Mn^2\Delta_f}{TL}}, \quad T := 193^2 LM\Delta_f \rho_\nu^2 n^4 p_{max}^2, \quad \bar{x}^t := \frac{1}{n}\sum_{i=1}^{n} x_i^t,$$

the expected output of Algorithm 1 obtains the following convergence result

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(\bar{x}^t)\right\|^2 \leq \frac{2\sqrt{\Delta_f}}{T} + \frac{2\sqrt{L\Delta_f}\left(1 + \frac{1921}{20}\rho_\nu\big(\sigma^2 + 6\zeta^2\sqrt{M}\big)\right)}{\sqrt{TM}}. \tag{12}$$

Remarks. (1) We prove that wait-free decentralized algorithms achieve optimal convergence results. SWIFT obtains the same convergence result $\mathcal{O}(1/T + 1/\sqrt{T})$ as parallel stochastic gradient descent (serial, decentralized, and asynchronous) in convex and non-convex smooth optimization [6, 7, 19, 20]. **(2)** SWIFT allows for entire neighborhood averaging, and not just one-to-one gossip averaging. This increases neighborhood synchronization, improving model robustness and reducing model divergence. **(3)** SWIFT's convergence rate does not rely upon a bounded delay term. **(4)** SWIFT converges in fewer total iterations T with respect to the number of total clients n compared to to other asynchronous methods [20] ($T = \mathcal{O}(n^4 p_{max}^2)$ in SWIFT versus $T = \mathcal{O}(n^4)$ in AD-PSGD).

Corollary 1 (Uniform Client Influence)**.** *In the common scenario where client influences are uniform, $p_i = 1/n \ \forall i$ and thus $p_{max} = 1/n$, SWIFT obtains the following convergence improvements:*
(i) Total iterations T with respect to the number of total clients n improves to $T = \mathcal{O}(n^2)$.
(ii) The step-size γ mirrors the optimal step-size in analysis of gradient descent convergence to first-order stationary points $\mathcal{O}(1/L)$ [24].

5 Experiments

Below, we perform image classification experiments for a range of decentralized FL algorithms. We compare the results of SWIFT to D-SGD [19], LD-SGD [17], PD-SGD [29], and AD-PSGD[20]. Clients are assigned equal partitions of data to perform computations on, thereby we set $p_i = 1/n$ for every client i as is done in [20, 30, 31, 17]. Details of the experimental setup are in Appendix C.

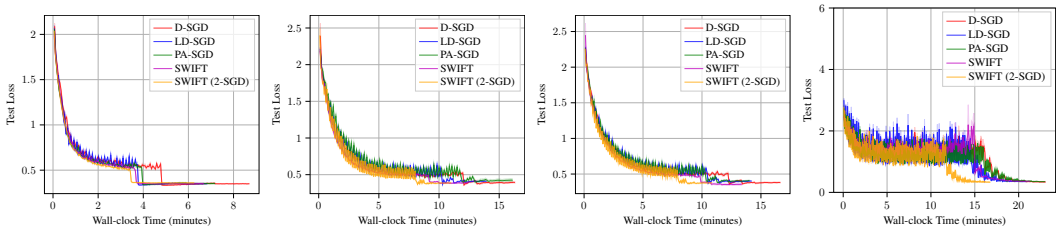
IID Experiments. To test the performance of SWIFT in the IID setting, we conduct three separate experiments with differing hyper-parameters and network topologies. The **first** experiment is identical to an experiment run in [20]. As no working code for AD-PSGD is available to run out of the box,

we chose to recreate the experiment to compare the relative performance of SWIFT to AD-PSGD. In the experiment, a ResNet-18 model is trained by 16 clients for 200 epochs in a ring network topology with the following hyper-parameters: mini-batch size of 32, learning rate of 0.1 that decays by a factor of 10 at the 81st and 122nd epoch, weight decay of 10^{-4}, and 0.9 momentum.
The **second** and **third** experiments train a ResNet-50 model using 16 clients for 200 epochs in a ring of cliques network topology shown in Figure 3. This setting is reflective of a realistic network topology, and the two experiments provide results for both 2-cluster and 4-cluster ring of cliques (ROC) topologies. The hyper-parameters used are: mini-batch size of 64, learning rate of 0.1 that decays by half every 10 epochs after the 100th epoch, weight decay of 10^{-4}, and 0.9 momentum.

Non-IID Experiments. We test SWIFT in the non-IID setting by partitioning half of each client's data as coming from one label and the other half coming from randomly generated (IID) data. A ResNet-18 model is trained by 10 clients for 300 epochs in a 3-cluster ROC network topology. We chose 10 clients to make label distribution in the non-IID setting more uniform because CIFAR-10 has 10 labels. The hyper-parameters used are: mini-batch size of 32, learning rate of 0.8 that decays by half every 10 epochs after the 200th epoch, weight decay of 10^{-4}, and 0.9 momentum. More studies on different level of data non-IIDness will be demonstrated in Appendix C.

Decentralized FL Algorithms	16 Client Ring		16 Client ROC-2C		16 Client ROC-4C	
	Epoch (s)	Comm. (s)	Epoch (s)	Comm. (s)	Epoch (s)	Comm. (s)
SWIFT ($I_1 = 0$)	**1.019**	**0.086**	**1.793**	**0.416**	**1.291**	**0.124**
D-SGD ($I_1 = 0$)	1.558	0.627	2.799	1.479	2.813	1.464
SWIFT ($I_1 = 1$)	**1.016**	**0.064**	**1.611**	**0.295**	**1.494**	**0.174**
LD-SGD ($I_1 = 1$)	1.320	0.428	2.408	0.987	2.525	1.105
PA-SGD ($I_1 = 1$)	1.281	0.358	2.639	0.765	2.216	0.708

Table 1: Baseline Comparisons (IID) of Average Epoch and Communication Times on CIFAR-10.



(a) IID ($n = 16$) [20]. (b) IID ROC-2C ($n = 16$). (c) IID ROC-4C ($n = 16$). (d) Non-IID ($n = 16$).

Figure 1: Dec. FL Algorithm Test Loss for CIFAR-10. **Figure 1(a)** displays experiment one [20]: average test loss across clients in ring topology on IID data for 200 epochs. **Figures 1(b) and 1(c)** show average test loss across clients in 2 and 4 cluster ROC topology on IID data for 200 epochs. **Figure 1(d)** depicts average test loss in 3 cluster ROC topology on non-IID data for 300 epochs.

Table 1 indicates that SWIFT obtains a 53% ($I_1 = 0$ and $I_1 = 1$) faster average epoch time than D-SGD for the first experiment. This outpaces AD-PSGD which, in the same setting (no slowdown), achieves only a 19% faster average epoch time (Table 4, [20]). SWIFT outperforms all other algorithm communication times at a rate ranging from 2-12x faster. Figure 1 reveals that SWIFT converges faster to SOTA test loss than its predecessors (IID & Non-IID). Run-times for SWIFT reach up to 66% faster than D-SGD in the IID setting. Performing experiments with no slow-down is to SWIFTs *disadvantage*, as wait-free algorithms perform best relative to synchronous ones under slowdowns. In Appendix C, we investigate performance of SWIFT under varying numbers of clients and levels of heterogeneity (causing different delays).

6 Conclusion

SWIFT is a SOTA wait-free decentralized FL algorithm that achieves reductions in communication cost and run-times while attaining near-SOTA train/test loss. SWIFT is theoretically guaranteed to converge at the same rate as its decentralized predecessors without use of a bounded-delay assumption. Given its lower communication costs and linear convergence rate reduction as clients increase, SWIFT is a scalable decentralized FL algorithm uniquely suited to solve large-scale optimization problems.

References

[1] Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. *Advances in neural information processing systems*, 24, 2011.

[2] Tuncer Can Aysal, Mehmet Ercan Yildiz, Anand D Sarwate, and Anna Scaglione. Broadcast gossip algorithms for consensus. *IEEE Transactions on Signal processing*, 57(7):2748–2761, 2009.

[3] Abhinav Bhatele, Kathryn Mohror, Steven H Langer, and Katherine E Isaacs. There goes the neighborhood: performance degradation due to nearby jobs. In *SC'13: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1–12. IEEE, 2013.

[4] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Gossip algorithms: Design, analysis and applications. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, volume 3, pages 1653–1664. IEEE, 2005.

[5] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE transactions on information theory*, 52(6):2508–2530, 2006.

[6] Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13(1), 2012.

[7] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[8] Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1):267–305, 2016.

[9] István Hegedűs, Gábor Danner, and Márk Jelasity. Decentralized learning works: An empirical comparison of gossip learning and federated learning. *Journal of Parallel and Distributed Computing*, 148:109–124, 2021.

[10] Torsten Hoefler, Timo Schneider, and Andrew Lumsdaine. Characterizing the influence of system noise on large-scale applications by simulation. In *SC'10: Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–11. IEEE, 2010.

[11] Nikita Ivkin, Daniel Rothchild, Enayat Ullah, Ion Stoica, Raman Arora, et al. Communication-efficient distributed sgd with sketching. *Advances in Neural Information Processing Systems*, 32, 2019.

[12] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[13] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*, pages 3252–3261. PMLR, 2019.

[14] Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian Stich. A unified theory of decentralized sgd with changing topology and local updates. In *International Conference on Machine Learning*, pages 5381–5393. PMLR, 2020.

[15] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[16] He Li, Kaoru Ota, and Mianxiong Dong. Learning iot in edge: Deep learning for the internet of things with edge computing. *IEEE network*, 32(1):96–101, 2018.

[17] Xiang Li, Wenhao Yang, Shusen Wang, and Zhihua Zhang. Communication-efficient local decentralized sgd methods. *arXiv preprint arXiv:1910.09126*, 2019.

[18] Xiangru Lian, Yijun Huang, Yuncheng Li, and Ji Liu. Asynchronous parallel stochastic gradient for nonconvex optimization. *Advances in Neural Information Processing Systems*, 28, 2015.

[19] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in Neural Information Processing Systems*, 30, 2017.

[20] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. Asynchronous decentralized parallel stochastic gradient descent. In *International Conference on Machine Learning*, pages 3043–3052. PMLR, 2018.

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

[22] Angelia Nedić and Alex Olshevsky. Distributed optimization over time-varying directed graphs. *IEEE Transactions on Automatic Control*, 60(3):601–615, 2014.

[23] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

[24] Yurii Nesterov. Introductory lectures on convex programming volume i: Basic course, 1998.

[25] Fabrizio Petrini, Darren J Kerbyson, and Scott Pakin. The case of the missing supercomputer performance: Achieving optimal performance on the 8,192 processors of asci q. In *SC'03: Proceedings of the 2003 ACM/IEEE conference on Supercomputing*, pages 55–55. IEEE, 2003.

[26] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. *Advances in neural information processing systems*, 24, 2011.

[27] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*, pages 8253–8265. PMLR, 2020.

[28] Anit Kumar Sahu, Tian Li, Maziar Sanjabi, Manzil Zaheer, Ameet Talwalkar, and Virginia Smith. On the convergence of federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 3:3, 2018.

[29] Jianyu Wang and Gauri Joshi. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms. *arXiv preprint arXiv:1808.07576*, 2018.

[30] Jianyu Wang, Anit Kumar Sahu, Zhouyi Yang, Gauri Joshi, and Soummya Kar. Matcha: Speeding up decentralized sgd via matching decomposition sampling. In *2019 Sixth Indian Control Conference (ICC)*, pages 299–300. IEEE, 2019.

[31] Hao Ye, Le Liang, and Geoffrey Ye Li. Decentralized federated learning with unreliable communications. *IEEE Journal of Selected Topics in Signal Processing*, 2022.

Supplementary Material
SWIFT: Rapid Decentralized Federated Learning via Wait-Free Model Communication

A Related Works

Decentralized Stochastic Gradient Descent (SGD). Pioneering models of FL, such as `FedAvg`[21] and `FedProx` [28], require a central controller to collect, aggregate, and distribute client parameters. The predecessor to decentralized FL is gossip learning [5, 9], introduced by the control community to assist with mean estimation of decentrally hosted data distributions [2, 4]. A key feature of gossip learning is the presence of a globally shared oracle/memory with whom clients exchange parameters at the end of training rounds [5]. SGD-based gossip algorithms successfully solve large-scale machine learning tasks [18, 20, 8, 23, 26, 1]. While read/write-accessible shared memory is well-suited for a single-organization ecosystem (i.e. all clients are controllable and trusted), this is unrealistic for more general edge-based paradigms. Neighborhood-based communication and aggregation algorithms, such as D-PSGD [19] and PD-SGD [17], can theoretically and empirically outperform their centralized counterparts, especially under heterogeneous client data distributions. Unfortunately, these algorithms suffer from synchronization slowdowns. SWIFT is asynchronous (avoiding slowdowns), utilizes neighborhood averaging, and does not require shared memory.

Asynchronous Learning. AsySG-Con [18], AD-PSGD [19], HOGWILD! [26] are seminal examples of asynchronous algorithms which allow clients to proceed at their own pace. However, these methods require shared memory/oracle from which clients grab the most up-to-date global parameters (e.g. the current graph-averaged gradient). By contrast, SWIFT relies on a message passing interface (MPI) to exchange parameters between neighbors, rather than interfacing with a shared memory structure. Algorithmically, each client relies on local memory to store current neighbor weights. To circumvent local memory blow-up in the context of an IoT cluster [16], each client relies on its edge provider to maintain a mailbox containing neighbor models: the client would pull out each neighbor model one at time to sequentially compute the desired aggregated statistics.

Communication Uncertainty and Efficiency. Few works in FL center on communication uncertainty. In [31], a lightweight, yet unreliable, transmission protocol is constructed in lieu of heavyweight protocols. A synchronous algorithm is developed to converge under expectation of a communication matrix (probabilistic link reliability). SWIFT also convergences under expectation of a communication matrix, yet in a different and asynchronous setting. SWIFT is already lightweight and reliable, and our use of expectation does not regard link reliability. Minimization of expected communication cost C per round over a graph of N clients is a challenge in FL, as the radius of information exchange can be large and take place over unreliable channels [12]. MATCHA [30] requires only a random subgraph of clients exchange parameters in each round. This requires $C = \mathcal{O}(T \sum_{i=1}^{N} p_i)$ per round, where p_i is the fixed independent activation probability of client i. LD-SGD [17] explores the trade-off between the number of local updates I_1 (i.e., steps of SGD) and neighborhood parameter exchanges I_2. A communication complexity $C = \mathcal{O}(T^{\frac{3}{4}} N^{\frac{3}{4}})$ is achievable, comparable to centralized FL, but only by increasing I_1, which delays the total iteration complexity of convergence. However, both of these methods are synchronous and consequently suffer from speed reductions. A wide class of graph-agnostic methods seek to reduce communication complexity via gradient/parameter compression [27, 11, 15] by sketching and projection methods, but the associated guarantees are probabilistic and require sub-routines to control the error associated with such dimensionality reductions [13]. SWIFT, as a wait-free algorithm, has a significantly reduced communication complexity. After each mini-batch, clients receive and store the available neighboring models sent to them in a non-blocking manner. Unavailable models signify that a neighboring client has not updated since the previous mini-batch, and the local memory model for that neighbor is current and can be used. While existing approaches assume globally shared memory and a bound on the speed of the slowest client, SWIFT uses local memory and assumes client homogeneity in convergence analysis. In practice, SWIFT allows for heterogeneous clients with differing computational speeds. A full derivation and description of SWIFT's communication complexity is found in Appendix D.3.

B Notation Table

Table 2: Notation Table

Definition	Notation	Definition	Notation
Global Iteration	t	Number of Clients	n
Active Client at t	i_t	Parameter Dimension	d
Client (i, j) Communication Coefficient at t	$w_{i,j}^t$	Data Dimension	f
Local Model of Client i at t	$x_i^t \in \mathbb{R}^d$	Step-Size	γ
Local Model of Active Client at t	$x_{i_t}^t \in \mathbb{R}^d$	Total SWIFT Iterations	T
Mini-Batch Data from Active Client at t	$\xi_{i_t}^t \in \mathbb{R}^f$	Mini-Batch Size (Uniform)	M
Gradient of the Active Model at t	$g(x_{i_t}^t, \xi_{i_t}^t) \in \mathbb{R}^d$	Variance of Stochastic Gradient	σ^2
Client i Communication Vector at t	$w_i^t \in \mathbb{R}^n$	Global Data Distribution	$\mathcal{D}$
Local Model Matrix at t	$X^t \in \mathbb{R}^{d \times n}$	Global Objective	$f(x)$
Zero-Padded Gradient of the Active Model at t	$G(x_{i_t}^t, \xi_{i_t}^t) \in \mathbb{R}^{d \times n}$	Client Influence Score	p_i
Expected Local Model Gradients at t	$\bar{G}(X^t, \xi^t) \in \mathbb{R}^{d \times n}$	Client Influence Vector	$p \in \mathbb{R}^n$
Active Client Communication Matrix at t	$W_{i_t}^t \in \mathbb{R}^{n \times n}$	One-Hot Vector	$\boldsymbol{e_i} \in \mathbb{R}^n$
Expected Client Communication Matrix at t	$\bar{W}^t \in \mathbb{R}^{n \times n}$	Identity Matrix	$I_n \in \mathbb{R}^{n \times n}$

C Additional Experimental Results and Details

C.1 Computational Specifications

We train our consensus model on a network of nodes. Each node has an NVIDIA GeForce RTX 2080 Ti GPU. All algorithms are built in Python and communicate via Open MPI, using MPI4Py (Python bindings for MPI). The training is also done in Python, leveraging Pytorch.

C.2 Data Partitioning

For all experiments, the training set is evenly partitioned amongst the number of clients training the consensus model. While the size of each client's partition is equal, we perform testing with data that is both (1) independent and identically distributed (iid) among all clients and (2) sorted by class and is thus non-iid. For the iid setting, each client is assigned data uniformly at random over all classes. In the non-iid setting, each client is assigned a subset of classes from which it will receive data exclusively. The c classes are assigned in the following steps:
(1) The class subset size n_c, for all clients, is determined as the ceiling of the number of classes per client $n_c = \lceil \frac{c}{n} \rceil$. Each class k within the class subset will take up $1/n_c$ of the client's total data partition if possible.
(2) The classes within each client's class subset are assigned cyclically, starting with the first client. The first client selects the first n_c classes, the second client selects the next n_c classes, and so on. Classes can, in some cases, be assigned to multiple clients. If the final class has been assigned, and more clients have yet to be assigned any classes, classes can be re-assigned starting at the first class.
(3) Each client is assigned data from the classes in their class subset cyclically ($1/n_c$ of its partition for each class), starting with the first client. If no more data is available from a specific class, the required data to fill its fraction of the partition is replaced by data from the next class.

C.3 Network Topologies

Ring Topology.

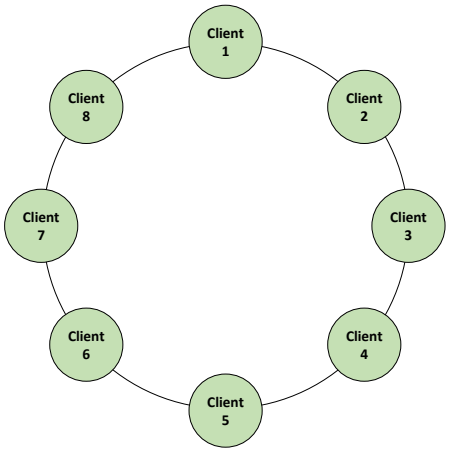


Figure 2: 8 Client Ring.

Ring of Cliques Topology.

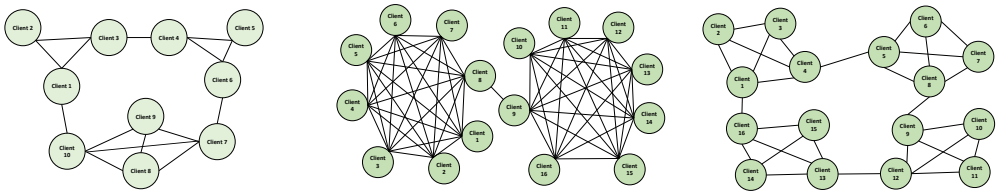


Figure 3: **(Left)** 10 Client, 3-Cluster. **(Middle)** 16 Client, 2-Cluster. **(Right)** 16 Client, 4-Cluster.

C.4 Training Loss Results for Experiments (1), (2), (3), and Non-IID

Below we display the corresponding training loss versus wall-clock time for the experiments ran in Section 5. We also provide the average epoch and communication times for the non-IID experiment described in Section 5.

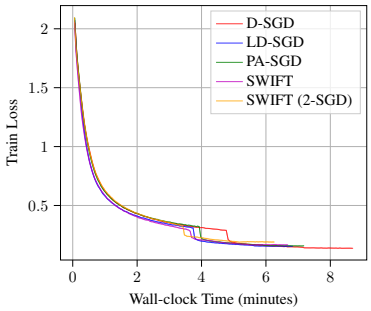
Decentralized FL Algorithms	10 Client Ring of Cliques - 3 Cluster Topology	
	Epoch Time (s)	Communication Time (s)
SWIFT ($I_1 = 0$)	**1.709**	**0.197**
D-SGD ($I_1 = 0$)	2.116	0.705
SWIFT ($I_1 = 1$)	**1.517**	**0.110**
LD-SGD ($I_1 = 1$)	1.973	0.575
PA-SGD ($I_1 = 1$)	1.929	0.421

Table 3: Average Epoch and Communication Times of Baselines for Non-IID Setting on CIFAR-10. ROC-xC signifies that a ring of cliques with x clusters is the base communication topology.

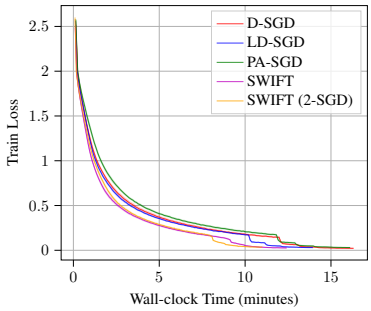
C.5 Additional Experiments and Ablation Studies

Varying Number of Clients.

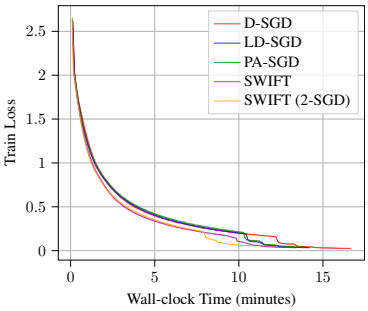
In this additional experiment, we determine how SWIFT performs versus other baseline algorithms as we vary the number of clients. The results of this experiment are shown in Table 4 as well as



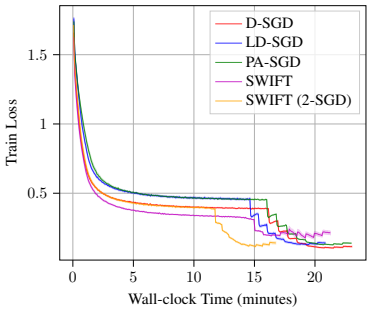
(a) IID Experiment [20] (16 Clients).



(b) IID ROC-2C (16 Clients).



(c) IID ROC-4C (16 Clients).



(d) Non-IID Experiment (10 Clients).

Figure 4: Dec. FL Algorithm Train Loss in IID, Non-IID Settings for CIFAR-10.

Figure 5. The setting and hyper-parameters are set as equivalent to those in the first IID experiment in Section 5. As we have already tested the performance of 16 clients in Section 5, we perform additional testing with 2, 4, and 8 clients.

Decentralized FL Algorithms	16 Client Ring		8 Client Ring		4 Client Ring		2 Client Ring	
	Epoch (s)	Comm. (s)	Epoch (s)	Comm. (s)	Epoch (s)	Comm. (s)	Epoch (s)	Comm. (s)
SWIFT ($I_1 = 0$)	**1.019**	**0.086**	**2.003**	**0.172**	**3.964**	**0.314**	**6.971**	**0.251**
D-SGD ($I_1 = 0$)	1.558	0.627	2.459	0.525	3.970	0.439	7.62	0.393
SWIFT ($I_1 = 1$)	**1.016**	**0.064**	**1.970**	**0.112**	**3.862**	0.232	6.83	**0.176**
LD-SGD ($I_1 = 1$)	1.320	0.428	2.217	0.345	3.946	0.300	**6.712**	0.179
PA-SGD ($I_1 = 1$)	1.281	0.358	2.093	0.285	3.871	**0.220**	7.303	0.180

Table 4: Epoch and Communication Times on CIFAR-10 with Varying Numbers of Clients in Ring Topology.

As one can see in Table 4, the time per epoch for SWIFT drops by nearly the optimal factor of 2 as the number of clients is doubled. For all algorithms, there is a bit of parallel overhead when the number of clients is small, however this becomes minimal as the number of clients grow to be large (greater than 4 clients). In comparison to the synchronous algorithms, SWIFT actually *decreases* its communication costs as the number of clients increases. This allows the parallel performance to be quite efficient, as shown in Figure 5. The parallel speed-up of SWIFT is much greater than those of the synchronous algorithm, and approaches the optimal speed-up.

Varying Degrees of Non-IIDness.

This experiment is an expansion of the Non-IID experiment in Section 5. In Section 5, we test the performance of SWIFT versus synchronous algorithms when clients have half of their data coming from one label and the rest of their data being evenly distributed (0.5 degree non-IIDness). In this experiment, we vary the levels of non-IIDness from 0.25 to 0.9 (25% to 90% of each client's local data coming from one label). The results of this experiment are shown in Figure 6, where the average test loss of all clients is plotted versus the run-time.

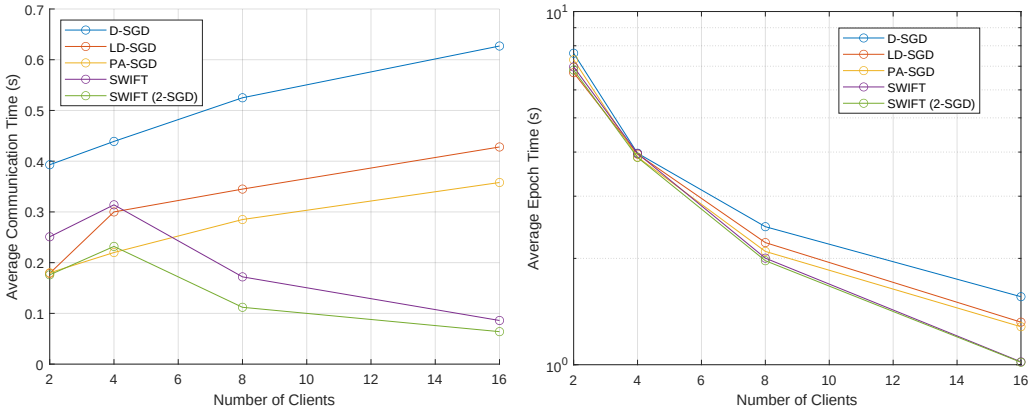
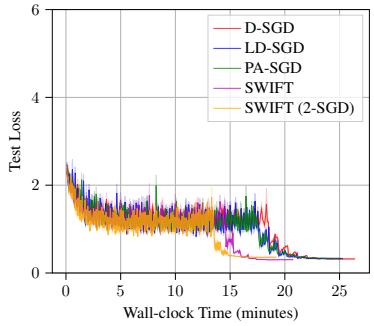
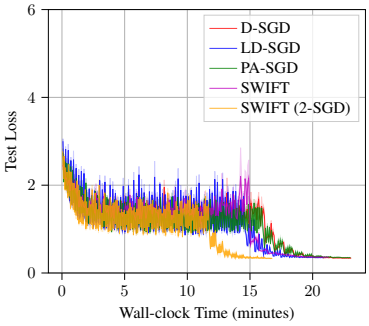


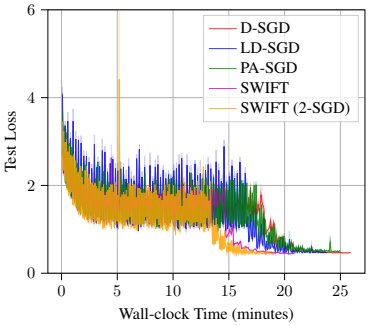
Figure 5: Average Communication and Epoch Times for Increasing Number of Clients



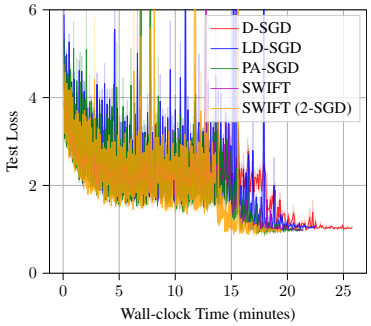
(a) 0.25 Degree Non-IID Data.



(b) 0.5 Degree Non-IID Data.



(c) 0.7 Degree Non-IID Data.



(d) 0.9 Degree Non-IID Data.

Figure 6: Dec. FL Algorithm Average Test Loss in Varying Degrees of Non-IIDness for CIFAR-10 in a 3-Cluster Ring of Cliques Topology with 10 Clients.

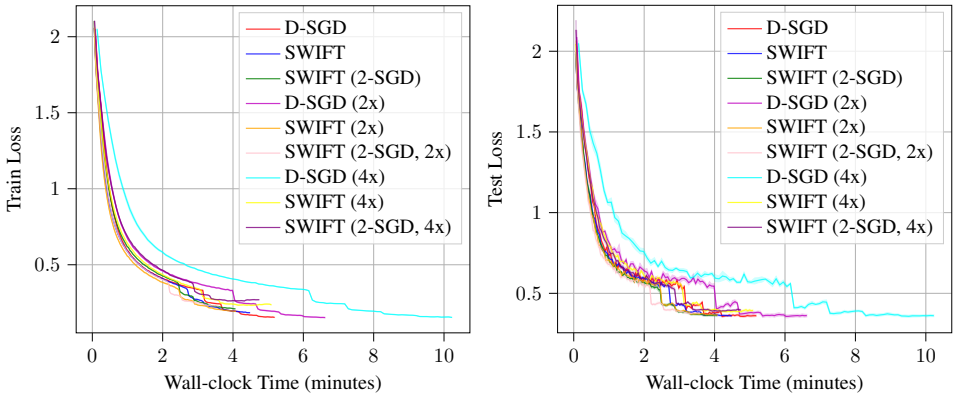
As expected, as data becomes more non-IID, the test loss will become higher and the overall accuracy lower. We do see, however, that SWIFT converges faster, and to a lower average loss, than all other synchronous baselines. In fact, SWIFT with $I_1 = 1$ blows the other synchronous algorithms out of the water in terms of how fast it converges. This is an impressive result, showing that SWIFT converges both quicker and to a smaller test loss than synchronous algorithms in the non-IID setting.

Varying Heterogeneity of Clients.

Our final additional experiment is to vary the heterogeneity, or speed, of our clients. To do so, we add an artificial slowdown to one of the clients such that it takes a certain amount of time longer (slowdown) to perform the computational portions of the training process. We perform tests in the case where a client is two times (2x) and four times (4x) as slow as usual. We then compare how the decentralized algorithms fare under these circumstances compared to the normal setting with no added slowdown. It is important to note that we run this experiment in an IID setting with a ring topology and 16 clients. A ResNet-18 model is trained for 100 epochs. The hyper-parameters used are: mini-batch size of 32, learning rate of 0.1 that decays by half every 10 epochs after the 50th epoch, weight decay of 10^{-4}, and 0.9 momentum. The results of this experiment are shown in Table 5 as well as Figure 7.

Decentralized FL Algorithms	No Slowdown			2x Slowdown			4x Slowdown		
	Total (s)	Epoch (s)	Comm. (s)	Total (s)	Epoch (s)	Comm. (s)	Total (s)	Epoch (s)	Comm. (s)
SWIFT ($I_1 = 0$)	**2.69**	**1.033**	**0.087**	**2.451**	**0.964**	**0.064**	**3.054**	**1.117**	**0.091**
D-SGD ($I_1 = 0$)	3.110	1.45	0.564	3.972	1.571	0.616	6.137	1.666	0.651
SWIFT ($I_1 = 1$)	**2.44**	**0.996**	**0.061**	**2.152**	**0.928**	**0.040**	**2.847**	**1.074**	**0.065**
LD-SGD ($I_1 = 1$)	3.323	1.353	0.413	3.762	1.389	0.428	5.917	1.412	0.448
PA-SGD ($I_1 = 1$)	3.183	1.270	0.331	3.557	1.262	0.309	5.743	1.270	0.318

Table 5: Epoch and Communication Times on CIFAR-10 in Ring Topology with Single Client Slowdown.



(a) IID Train Loss Slowdown Comparison. (b) IID Test Loss Slowdown Comparison.

Figure 7: Dec. FL Algorithm Train and Test Loss in IID with Slowdown for CIFAR-10 in a Ring Topology with 16 Clients.

In Table 5, the average epoch, communication, and total time is displayed. The average total time is the average total time that each client performs its computations and communication, including any slowdown (wait time) during computation or communication. As one can see from Table 5, SWIFT avoids these large average total times as the slowdown grows larger. The reason is the wait-free structure of SWIFT, which allows all other clients to finish their work at their own speed. All other algorithms require all clients to wait for the slowest client to finish a mini-batch before proceeding. As one can see, at a large slowdown (4x) the average total time for clients in SWIFT is nearly *half* the average total time for clients in synchronous algorithms. This shows how effective SWIFT is at reducing the run-time when clients are slow within the network.

Figure 7 shows how SWIFT is able to converge faster to an equivalent, or smaller, test loss than D-SGD for all slowdowns. In the case of large slowdowns (4x), SWIFT massively outperforms

D-SGD, finishing in better than half the run-time. We do not include the other baseline algorithms as it made the plots too clustered, however SWIFT also performs much better than PA-SGD and LD-SGD as shown in Table 5. These results show that the wait-free structure of SWIFT allows it to be efficient under client slowdown.

D Algorithm Details and Communication Complexity

D.1 SWIFT Diagram

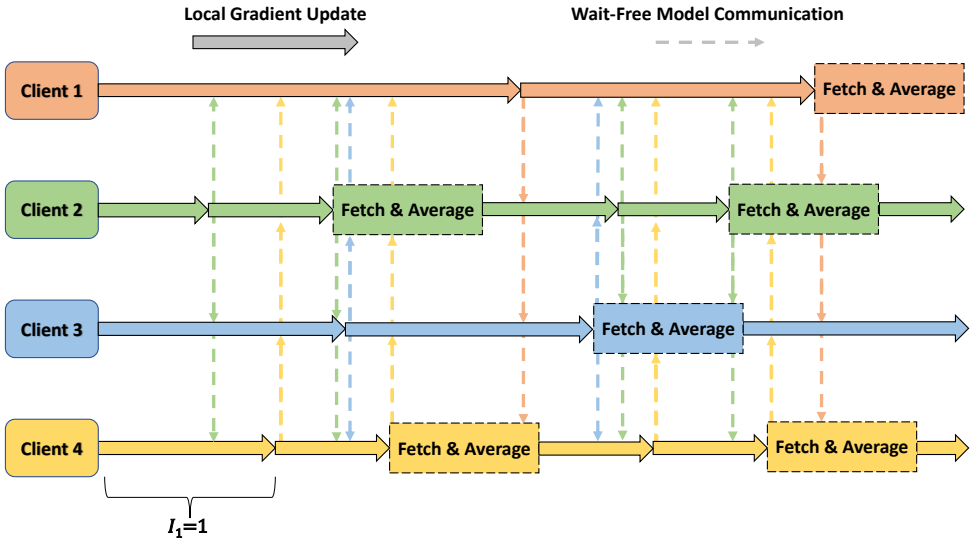


Figure 8: Overview of SWIFT with 4 Clients and $I_1 = 1$

D.2 SWIFT Pre-processing: Setting Client-Communication Weights

Below we present the the algorithmic pseudocode here for our novel client-communication selection algorithm in Algorithm 2. As a note, we include different client-communication vector initializations if the client influence scores are uniform versus non-uniform. The reason for this is to ensure that the selfweight for each client i, $w_{i,i}$, has a value greater than $1/n$. This naturally occurs when the CIS are uniform, however is not so when they are non-uniform.

D.3 SWIFT Communication Complexity

Communication Complexity. The wait-free mechanism allows clients to move through epochs as fast as they are able to train, using locally stored neighbor weights in the absence of new updates. Once a client has completed all their epochs, they cease to transmit model updates. Furthermore, all of their neighbors will reciprocally cease communicating updates to them. This completion disparity is where we empirically observe savings in communication complexity. To mathematically formulate this situation, we assign each client i a continuous client-specific, time-dependent velocity function $v_i(\theta_i, t)$, such that $M_i(t) = \int_{t_0}^{t} v_i(\theta_i, t)\, dt$ returns the number of mini-batches completed over the time interval $[t_0, t]$, where t_0 is the initial time. Here, θ_i is an abstract set of controllable and exogenous parameters specific to the client and cluster environment. Examples include: computational resources available, priority, number of nearby jobs (physically on the hardware), background processes concurrently running, and more. All of these factors have been shown to affect execution time [3, 25, 10]. If we let m_T denote the desired number of total mini-batches, then we are interested

Algorithm 2: Communication Coefficient Selection (CCS)

Input : Client Influence Score (CIS) $p_i \in \mathbb{R}$, Client Degree $d_i \in \mathbb{R}$,
Client Neighbor Set $J_i = \{\forall j : \text{client } j \text{ is a one-hop neighbor of client } i\}, \forall i$

Output : Client-Communication Vector $w_i \in \mathbb{R}^n, \forall i \in [n]$

for $i = 1 : n$ *in parallel* **do**
  if *CIS are non-uniform* **then**
    Initialize Client-Communication Vector $w_i = [w_{1,i}, w_{2,i}, \cdots, w_{n,i}] \leftarrow (1/n)\boldsymbol{e}_i$
  else
    Initialize Client-Communication Vector $w_i = [w_{1,i}, w_{2,i}, \cdots, w_{n,i}] \leftarrow \mathbf{0}$
  Exchange CIS and degree with all neighbors
  Store *Neighbor CIS Vector* $P^J \leftarrow [\{p_j\}_{j \in J_i}] \in \mathbb{R}^{d_i}$
  Construct Neighbor Subsets J^{L}, J^{SE}, $J^{\mathsf{E}} \subset J_i$ as subsets of i's neighbors with degree larger than, no larger than and equal to d_i respectively
  for $\forall j \in J^{\mathsf{L}}$ **do**
    Wait to fetch $w_{j,i}$ from neighbor client j with a degree larger than d_i
  Determine the sum of the *total coefficients assigned (TCA)* $s_w^i \leftarrow \sum_{m=1}^n w_{m,i}$
  if $|J^{\mathsf{SE}}| > 0$ **then**
    Determine the sum of all remaining neighbors' CIS $s_p^i \leftarrow \sum_{j \in J^{\mathsf{SE}}} P_j^J$
    if $|J^{\mathsf{E}}| > 0$ **then**
      Exchange s_w^i and s_p^i with all neighbors $j \in J^{\mathsf{E}}$, storing all exchanged s_w^j, s_p^j
      Store $s_w^* \leftarrow \max\{s_w^i, s_w^j\}$ and $s_p^* \leftarrow \max\{s_p^i, s_p^j\} \;\; \forall j \in J^{\mathsf{E}}$
      Set $w_{j,i} \leftarrow \frac{(1-s_w^*)P_j^J}{s_p^*} \; \forall j \in J^{\mathsf{E}}$
      Recompute $s_w^i = \sum_{m=1}^n w_{m,i}$ and $s_p^i = \sum_{j \in \{J^{\mathsf{SE}} \cup i\} \setminus J^{\mathsf{E}}} P_j^J$
    Set $w_{j,i} \leftarrow w_{j,i} + \frac{(1-s_w^i)P_j^J}{s_p^i} \; \forall j \in \{J^{\mathsf{SE}} \cup i\} \setminus J^{\mathsf{E}}$ for all remaining neighbors
    Send $w_{i,j} = \frac{(1-s_w^i)P_i^J}{s_p^i}$ to all waiting neighbors $j \in J^{\mathsf{SE}} \setminus J^{\mathsf{E}}$
  else
    $w_{i,i} = 1 - s_w^i$

in $t_i^* > t_0$ such that $M_i(t_i^*) = m_T$. With this model in hand, the total communication complexity of SWIFT over N clients is

$$C = \sum_{i=1}^{N} \sum_{j \in \mathcal{N}(i)} W \cdot M_i(t_j^*), \tag{13}$$

where W, a constant, is the total bits to be communicated (i.e., model size) at the end of each batch and the term $\mathcal{N}(i)$ denotes the set of clients connected to client i (neighborhood of client i).

E Review of Existing Inter-Client Communication in Decentralized FL

Decentralized SGD (D-SGD) [19] One of the foundational decentralized Federated Learning algorithms is Decentralized SGD. In order to minimize Equation (1), D-SGD orchestrates a local gradient step for all clients before performing synchronous neighborhood averaging. The D-SGD process for a single client i is defined as:

$$x_i^{t+1} = \sum_{j=1}^{n} W_{ij} \big[x_j^t - g(x_j^t, \xi_j^t)\big]. \tag{14}$$

The term $g(x_j^t, \xi_j^t)$ denotes the stochastic gradient of x_j^t with mini-batch data ξ_j^t sampled from the local data distribution of client j. The matrix W is a weighting matrix, where W_{ij} is the amount of x_i^{t+1} which will be made up of client j's local model after one local gradient step (e.g. if $W_{ij} = 1/2$, then half of x_i^{t+1} will have been composed of client j's model after its local gradient step). The

weighting matrix only has a zero value $W_{ij} = 0$ if clients i and j are not connected (they are not within the same neighborhood). The values of W_{ij} are generally selected ahead of time by a central host, with the usual weighting scheme being uniform. In D-SGD, model communication occurs only after all local gradient updates are finished. These gradient updates are computed in parallel.

Periodic Averaging SGD (PA-SGD) [29] The Periodic Averaging SGD algorithm is an extension of D-SGD. In order to save communication costs when the number of clients grows to be large, PA-SGD performs model averaging after an additional I_1 local gradient steps. The special case of $I_1 = 0$ reduces to D-SGD. The PA-SGD process for a single client i is defined as:

$$x_i^{t+1} = \begin{cases} \sum_{j=1}^{w} W_{ij}\left[x_j^t - g(x_j^t, \xi_j^t)\right], & t \mod (I_1 + 1) = 0 \\ x_i^t - g(x_i^t, \xi_i^t), & \text{otherwise} \end{cases} \tag{15}$$

Compared with D-SGD, PA-SGD still suffers from the inefficiency of having to wait for the slowest client for each update. However, PA-SGD saves communication costs by reducing the frequency of communication.

Local Decentralized SGD (LD-SGD) [17] Continuing to generalize the foundational decentralized Federated Learning algorithms is Local Decentralized SGD. LD-SGD generalizes PA-SGD by allowing multiple chunks of singular D-SGD updates, as described in Equation (14), in between the increased local gradient steps seen in PA-SGD. The number of D-SGD chunks is dictated by a new parameter I_2. For example, in the case $I_1 = 3, I_2 = 2$, LD-SGD will take three local gradient steps and then perform two D-SGD updates (which consists of a local gradient step and then averaging) as shown in Figure 9. The special case of $I_2 = 1$ reduces to PA-SGD. The LD-SGD process for a single client i is defined as:

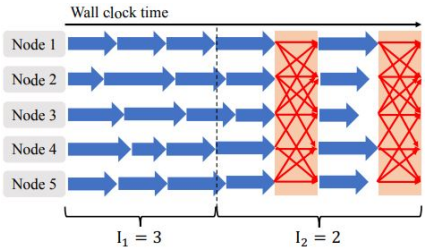


Figure 9: I_1, I_2 Depiction (from [17]).

$$x_i^{t+1} = \begin{cases} \text{Perform } \sum_{j=1}^{w} W_{ij}\left[x_j^t - g(x_j^t, \xi_j^t)\right] \text{ for } I_2 \text{ iteration(s)}, & t \mod (I_1 + 1) = 0 \\ x_i^t - g(x_i^t, \xi_i^t), & \text{otherwise} \end{cases} \tag{16}$$

F Proof of the Main Theorem

Before beginning, we quickly define the expected gradient $\mathbb{E}_{i_t}\left[G(x_{i_t}^t, \xi_{*,t}^{i_t})\right]$ as

$$\bar{G}(X^t, \xi^t) := \mathbb{E}_{i_t}\left[G(x_{i_t}^t, \xi_{i_t}^t)\right] = \sum_{i=1}^n p_i G(x_{i_t}^t, \xi_{i_t}^t). \tag{17}$$

Proof of Theorem 1. In this theorem, we characterize the convergence of the average of *all* local models.

Using the Gradient Lipschitz assumption with Equation (10) yields

$$f(\frac{X^{t+1}1_n}{n}) \leq f(\frac{X^t 1_n}{n}) + \left\langle \nabla f(\frac{X^t 1_n}{n}), \frac{X^t(W_{i_t}^t - \bar{W}^t)1_n}{n} - \gamma \frac{G(x_{i_t}^t, \xi_{*,t}^{i_t})1_n}{n} \right\rangle \tag{18}$$

$$+ \frac{L}{2}\left\| \frac{X^t(W_{i_t}^t - \bar{W}^t)1_n}{n} - \gamma \frac{G(x_{i_t}^t, \xi_{*,t}^{i_t})1_n}{n} \right\|^2 \tag{19}$$

We first denote the average over all local models as $\bar{x}^t := \frac{X^t 1_n}{n}$. Taking the expectation with respect to the updating client i_t yields

$$f(\bar{x}^{t+1}) \leq f(\bar{x}^t) + \left\langle \nabla f(\bar{x}^t), X^t(\bar{W}^t - \bar{W}^t)\frac{1_n}{n} - \gamma \bar{G}(X^t, \xi_{*,t})\frac{1_n}{n} \right\rangle \tag{20}$$

$$+ \frac{L}{2}\mathbb{E}_{i_t}\left\| X^t(W_{i_t}^t - \bar{W}^t)\frac{1_n}{n} - \gamma G(x_{i_t}^t, \xi_{*,t}^{i_t})\frac{1_n}{n} \right\|^2 \tag{21}$$

$$= f(\bar{x}^t) + \left\langle \nabla f(\bar{x}^t), -\gamma \bar{G}(X^t, \xi_{*,t})\frac{1_n}{n} \right\rangle + \frac{L}{2}\mathbb{E}_{i_t}\left\| X^t(W_{i_t}^t - \bar{W}^t)\frac{1_n}{n} - \gamma G(x_{i_t}^t, \xi_{*,t}^{i_t})\frac{1_n}{n} \right\|^2 \tag{22}$$

$$= f(\bar{x}^t) + \left\langle \nabla f(\bar{x}^t), -\frac{\gamma}{Mn}\sum_{i=1}^n \sum_{m=1}^M p_i \nabla \ell(x_i^t, \xi_{m,t}^i) \right\rangle \tag{23}$$

$$+ \frac{L}{2}\mathbb{E}_{i_t}\left\| X^t(W_{i_t}^t - \bar{W}^t)\frac{1_n}{n} - \gamma G(x_{i_t}^t, \xi_{*,t}^{i_t})\frac{1_n}{n} \right\|^2 \tag{24}$$

$$= f(\bar{x}^t) - \gamma\left\langle \nabla f(\bar{x}^t), \frac{1}{Mn}\sum_{i=1}^n \sum_{m=1}^M p_i \nabla \ell(x_i^t, \xi_{m,t}^i) \right\rangle \tag{25}$$

$$+ \frac{L}{2}\mathbb{E}_{i_t}\left\| X^t(W_{i_t}^t - \bar{W}^t)\frac{1_n}{n} - \gamma G(x_{i_t}^t, \xi_{*,t}^{i_t})\frac{1_n}{n} \right\|^2 \tag{26}$$

Taking the expectation over all local data $\mathbb{E}_{\xi\sim\mathcal{D}_i}$ yields

$$f(\bar{x}^{t+1}) - f(\bar{x}^t) \leq -\frac{\gamma}{n}\left\langle \nabla f(\bar{x}^t), \sum_{i=1}^n p_i \nabla f_i(x_i^t) \right\rangle \tag{27}$$

$$+ \frac{L}{2}\mathbb{E}_{\xi\sim\mathcal{D}_i, i_t}\left\| X^t(W_{i_t}^t - \bar{W}^t)\frac{1_n}{n} - \gamma G(x_{i_t}^t, \xi_{*,t}^{i_t})\frac{1_n}{n} \right\|^2 \tag{28}$$

By properties of the inner product

$$\leq -\frac{\gamma}{2n}\left(\left\|\nabla f(\bar{x}^t)\right\|^2 + \left\| \sum_{i=1}^n p_i \nabla f_i(x_i^t) \right\|^2 - \left\| \nabla f(\bar{x}^t) - \sum_{i=1}^n p_i \nabla f_i(x_i^t) \right\|^2 \right) \tag{29}$$

$$+ \frac{L}{2}\underbrace{\mathbb{E}_{\xi\sim\mathcal{D}_i, i_t}\left\| X^t(W_{i_t}^t - \bar{W}^t)\frac{1_n}{n} - \gamma G(x_{i_t}^t, \xi_{*,t}^{i_t})\frac{1_n}{n} \right\|^2}_{A} \tag{30}$$

Bounding Term A.

Given the update equation, term A can be transformed into

$$\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left\|\left(X^t(W^t_{i_t}-\bar{W}^t)-\gamma G(x^t_{i_t},\xi^{i_t}_{*,t})\right)\frac{1_n}{n}\right\|^2=\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left\|\left(X^{t+1}-X^t\bar{W}^t\right)\frac{1_n}{n}\right\|^2 \tag{31}$$

Due to the symmetric and doubly stochastic property of $\bar{W}^t$ this reduces to

$$\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left\|\bar{x}^{t+1}-\bar{x}^t\right\|^2=\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left\|\frac{1}{n}\sum_{i=1}^n\left(x_i^{t+1}-x_i^t\right)\right\|^2=\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left\|\frac{1}{n}\left(x_{i_t}^{t+1}-x_{i_t}^t\right)\right\|^2 \tag{32}$$

$$=\frac{1}{n^2}\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left\|x_{i_t}^{t+1}-x_{i_t}^t\right\|^2 \tag{33}$$

$$\leq\frac{3}{n^2}\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left(\left\|x_{i_t}^{t+1}-\bar{x}^{t+1}\right\|^2+\left\|\bar{x}^t-x_{i_t}^t\right\|^2+\left\|\bar{x}^{t+1}-\bar{x}^t\right\|^2\right) \tag{34}$$

Combining like terms yields

$$(1-\frac{3}{n^2})\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left\|\bar{x}^{t+1}-\bar{x}^t\right\|^2\leq\frac{3}{n^2}\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left(\left\|x_{i_t}^{t+1}-\bar{x}^{t+1}\right\|^2+\left\|\bar{x}^t-x_{i_t}^t\right\|^2\right) \tag{35}$$

Since $n\geq 2$ (we assume at least 2 devices are running the algorithm) we find the following result

$$\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left\|\bar{x}^{t+1}-\bar{x}^t\right\|^2\leq\frac{3}{(n^2-3)}\mathbb{E}_{\xi\sim\mathcal{D}_i,i_t}\left(\left\|x_{i_t}^{t+1}-\bar{x}^{t+1}\right\|^2+\left\|\bar{x}^t-x_{i_t}^t\right\|^2\right) \tag{36}$$

$$=\frac{3}{(n^2-3)}\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{i=1}^n p_i\left(\left\|\bar{x}^{t+1}-x_i^{t+1}\right\|^2+\left\|\bar{x}^t-x_i^t\right\|^2\right) \tag{37}$$

Thus, we have bounded Term A. Substituting this back into Equation (30) results in

$$\leq-\frac{\gamma}{2n}\left(\left\|\nabla f(\bar{x}^t)\right\|^2+\left\|\sum_{i=1}^n p_i\nabla f_i(x_i^t)\right\|^2-\left\|\nabla f(\bar{x}^t)-\sum_{i=1}^n p_i\nabla f_i(x_i^t)\right\|^2\right) \tag{38}$$

$$+\frac{3L}{2(n^2-3)}\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{i=1}^n p_i\left(\left\|\bar{x}^{t+1}-x_i^{t+1}\right\|^2+\left\|\bar{x}^t-x_i^t\right\|^2\right) \tag{39}$$

Taking the sum from $t=0$ to $t=T-1$ yields

$$f(\bar{x}^T)-f(\bar{x}^0)\leq-\frac{\gamma}{2n}\left(\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)\right\|^2-\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)-\sum_{i=1}^n p_i\nabla f_i(x_i^t)\right\|^2\right. \tag{40}$$

$$\left.+\sum_{t=0}^{T-1}\left\|\sum_{i=1}^n p_i\nabla f_i(x_i^t)\right\|^2\right)+\frac{3L}{2(n^2-3)}\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{t=0}^{T-1}\sum_{i=1}^n p_i\left(\left\|\bar{x}^{t+1}-x_i^{t+1}\right\|^2+\left\|\bar{x}^t-x_i^t\right\|^2\right) \tag{41}$$

Using the Lipschitz Gradient assumption, the following term is bounded as

$$\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)-\sum_{i=1}^n p_i\nabla f_i(x_i^t)\right\|^2=\sum_{t=0}^{T-1}\left\|\sum_{i=1}^n p_i\nabla f_i(\bar{x}^t)-\sum_{i=1}^n p_i\nabla f_i(x_i^t)\right\|^2 \tag{42}$$

$$\leq\sum_{t=0}^{T-1}\sum_{i=1}^n p_i^2\left\|\nabla f_i(\bar{x}^t)-\nabla f_i(x_i^t)\right\|^2\leq L^2\sum_{t=0}^{T-1}\sum_{i=1}^n p_i^2\left\|\bar{x}^t-x_i^t\right\|^2\leq L^2 p_{max}\sum_{t=0}^{T-1}\sum_{i=1}^n p_i\left\|\bar{x}^t-x_i^t\right\|^2 \tag{43}$$

Placing this back into Equation (40), and rearranging, yields

$$f(\bar{x}^T)-f(\bar{x}^0)\leq-\frac{\gamma}{2n}\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)\right\|^2-\frac{\gamma}{2n}\sum_{t=0}^{T-1}\left\|\sum_{i=1}^n p_i\nabla f_i(x_i^t)\right\|^2 \tag{44}$$

$$+\frac{3L}{2(n^2-3)}\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{t=0}^{T-1}\sum_{i=1}^n p_i\left(\left\|\bar{x}^{t+1}-x_i^{t+1}\right\|^2+\left\|\bar{x}^t-x_i^t\right\|^2\right)+\frac{\gamma L^2 p_{max}}{2n}\sum_{t=0}^{T-1}\sum_{i=1}^n p_i\left\|\bar{x}^t-x_i^t\right\|^2 \tag{45}$$

Given that $\bar{x}^0 = x_i^0$ for all clients i, one can see that

$$\sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \left\|\bar{x}^{t+1} - x_i^{t+1}\right\|^2 = \sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \left\|\bar{x}^{t+1} - x_i^{t+1}\right\|^2 + \sum_{i=1}^{n} p_i \left\|\bar{x}^{0} - x_i^{0}\right\|^2 \tag{46}$$

$$= \sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \left\|\bar{x}^{t} - x_i^{t}\right\|^2 + \sum_{i=1}^{n} p_i \left\|\bar{x}^{T} - x_i^{T}\right\|^2 \geq \sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \left\|\bar{x}^{t} - x_i^{t}\right\|^2. \tag{47}$$

Using the result of Equation (46) condenses Equation (44) into

$$f(\bar{x}^T) - f(\bar{x}^0) \leq -\frac{\gamma}{2n}\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)\right\|^2 - \frac{\gamma}{2n}\sum_{t=0}^{T-1}\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^t)\right\|^2 \tag{48}$$

$$+ \left(\frac{3L}{(n^2-3)} + \frac{\gamma L^2 p_{max}}{2n}\right)\underbrace{\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \left\|\bar{x}^{t+1} - x_i^{t+1}\right\|^2}_{B} \tag{49}$$

Bounding Term B. As detailed in Equation (4), the recursion of our update rule can be slightly rearranged to become

$$X^{t+1} = X^0 - \gamma\sum_{j=0}^{t} G(x_{i_j}^j, \xi_{*,j}^{i_j})\prod_{q=j+1}^{t}\bar{W}^q - \gamma\sum_{j=0}^{t} G(x_{i_j}^j, \xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q - \bar{W}^q) \tag{50}$$

The recursion equation for the expected consensus model $\bar{x}^t$ and expected local model for client i can be computed by multiplying by $\frac{1_n}{n}$ and $\boldsymbol{e}_i$ respectively

$$\bar{x}^{t+1} = \bar{x}^0 - \gamma\sum_{j=0}^{t} G(x_{i_j}^j, \xi_{*,j}^{i_j})\frac{1_n}{n} - \gamma\sum_{j=0}^{t} G(x_{i_j}^j, \xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q - \bar{W}^q)\frac{1_n}{n} \tag{51}$$

$$x_i^{t+1} = x_i^0 - \gamma\sum_{j=0}^{t} G(x_{i_j}^j, \xi_{*,j}^{i_j})\prod_{q=j+1}^{t}\bar{W}^q\boldsymbol{e}_i - \gamma\sum_{j=0}^{t} G(x_{i_j}^j, \xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q - \bar{W}^q)\boldsymbol{e}_i \tag{52}$$

Using the recursive equations above transforms the bound on term B to become

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \left\|\bar{x}^{t+1} - x_i^{t+1}\right\|^2 = \gamma^2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i\Bigg\| - \sum_{j=0}^{t} G(x_{i_j}^j, \xi_{*,j}^{i_j})\left(\frac{1_n}{n} - \prod_{q=j+1}^{t}\bar{W}^q\boldsymbol{e}_i\right) \tag{53}$$

$$- \sum_{j=0}^{t} G(x_{i_j}^j, \xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q - \bar{W}^q)(\frac{1_n}{n} - \boldsymbol{e}_i)\Bigg\|^2 \tag{54}$$

$$\leq 2\gamma^2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i\Bigg(\left\|\sum_{j=0}^{t} G(x_{i_j}^j, \xi_{*,j}^{i_j})\left(\frac{1_n}{n} - \prod_{q=j+1}^{t}\bar{W}^q\boldsymbol{e}_i\right)\right\|^2 \tag{55}$$

$$+ \left\|\sum_{j=0}^{t} G(x_{i_j}^j, \xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q - \bar{W}^q)(\frac{1_n}{n} - \boldsymbol{e}_i)\right\|^2\Bigg) \tag{56}$$

$$= 2\gamma^2 \mathbb{E}_{\xi\sim\mathcal{D}_i} \sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \Bigg(\underbrace{\sum_{j=0}^{t} \left\| G(x_{i_j}^j, \xi_{*,j}^{i_j}) \Big(\frac{1_n}{n} - \prod_{q=j+1}^{t} \bar{W}^q \boldsymbol{e}_i \Big) \right\|^2}_{B_1} \tag{57}$$

$$+ \underbrace{\sum_{j=0}^{t} \left\| G(x_{i_j}^j, \xi_{*,j}^{i_j}) \prod_{q=j+1}^{t} (W_{i_q}^q - \bar{W}^q)(\frac{1_n}{n} - \boldsymbol{e}_i) \right\|^2}_{B_3} \Bigg) \tag{58}$$

$$+ \underbrace{2\sum_{j=0}^{t}\sum_{j'=j+1}^{t} \langle G(x_{i_j}^j, \xi_{*,j}^{i_j})(\frac{1_n}{n} - \prod_{q=j+1}^{t} \bar{W}^q \boldsymbol{e}_i), G(x_{i_{j'}}^{j'}, \xi_{*,j'}^{i_{j'}})(\frac{1_n}{n} - \prod_{q=j'+1}^{t} \bar{W}^q \boldsymbol{e}_i) \rangle}_{B_2} \Bigg) \tag{59}$$

$$+ \underbrace{2\sum_{j=0}^{t}\sum_{j'=j+1}^{t} \langle G(x_{i_j}^j, \xi_{*,j}^{i_j}) \prod_{q=j+1}^{t} (W_{i_q}^q - \bar{W}^q)(\frac{1_n}{n} - \boldsymbol{e}_i), G(x_{i_{j'}}^{j'}, \xi_{*,j'}^{i_{j'}}) \prod_{q=j'+1}^{t} (W_{i_q}^q - \bar{W}^q)(\frac{1_n}{n} - \boldsymbol{e}_i) \rangle}_{B_4} \Bigg) \tag{60}$$

Bounding Term B_1. Using Lemma 2,

$$\mathbb{E}_{\xi\sim\mathcal{D}_i} \sum_{j=0}^{t} \left\| G(x_{i_j}^j, \xi_{*,j}^{i_j}) \Big(\frac{1_n}{n} - \prod_{q=j+1}^{t} \bar{W}^q \boldsymbol{e}_i \Big) \right\|^2 \tag{61}$$

$$\leq \mathbb{E}_{\xi\sim\mathcal{D}_i} \sum_{j=0}^{t} \left\| G(x_{i_j}^j, \xi_{*,j}^{i_j}) \right\|^2 \left\| \Big(\frac{1_n}{n} - \prod_{q=j+1}^{t} \bar{W}^q \boldsymbol{e}_i \Big) \right\|^2 \tag{62}$$

$$= \mathbb{E}_{\xi\sim\mathcal{D}_i} \sum_{j=0}^{t} \left\| \frac{1}{M} \sum_{m=1}^{M} \nabla \ell(x_{i_j}^j; \xi_{m,j}^{i_j}) \right\|^2 \left\| \Big(\frac{1_n}{n} - \prod_{q=j+1}^{t} \bar{W}^q \boldsymbol{e}_i \Big) \right\|^2 \tag{63}$$

$$\leq \mathbb{E}_{\xi\sim\mathcal{D}_i} \sum_{j=0}^{t} \left\| \frac{1}{M} \sum_{m=1}^{M} \nabla \ell(x_{i_j}^j; \xi_{m,j}^{i_j}) \right\|^2 (\frac{n-1}{n}) \rho^{t-j} \tag{64}$$

Using Lemma 4, the equation above becomes

$$= 2\sum_{j=0}^{t} \left(\frac{\sigma^2}{M} + \left\| \nabla f_{i_j}(x_{i_j}^j) \right\|^2 \right) (\frac{n-1}{n}) \rho^{t-j} \tag{65}$$

$$= 2\Bigg(\sum_{j=0}^{t} \frac{\sigma^2}{M} (\frac{n-1}{n}) \rho^{t-j} + \sum_{j=0}^{t} \left\| \nabla f_{i_j}(x_{i_j}^j) \right\|^2 (\frac{n-1}{n}) \rho^{t-j} \Bigg) \tag{66}$$

$$\leq \frac{2(n-1)\sigma^2}{(1-\rho)Mn} + 2\sum_{j=0}^{t} \left\| \nabla f_{i_j}(x_{i_j}^j) \right\|^2 (\frac{n-1}{n}) \rho^{t-j} \tag{67}$$

Taking the expectation over worker i_j yields the desired bound

$$\mathbb{E}_{i_j} \left[\frac{2(n-1)\sigma^2}{(1-\rho)Mn} + 2\sum_{j=0}^{t} \left\| \nabla f_{i_j}(x_{i_j}^j) \right\|^2 (\frac{n-1}{n}) \rho^{t-j} \right] \tag{68}$$

$$= \frac{2(n-1)\sigma^2}{(1-\rho)Mn} + \frac{2(n-1)}{n} \sum_{j=0}^{t} \mathbb{E}_{i_j} \left\| \nabla f_{i_j}(x_{i_j}^j) \right\|^2 \rho^{t-j} \tag{69}$$

Bounding Term B_2. Using Lemma 2,

$$2\sum_{j=0}^{t}\sum_{j'=j+1}^{t}\langle G(x_{i_j}^j,\xi_{*,j}^{i_j})(\frac{1_n}{n}-\prod_{q=j+1}^{t}\bar{W}^q\boldsymbol{e}_i),G(x_{i_{j'}}^{j'},\xi_{*,j'}^{i_{j'}})(\frac{1_n}{n}-\prod_{q=j'+1}^{t}\bar{W}^q\boldsymbol{e}_i)\rangle \tag{70}$$

$$=2\sum_{j=0}^{t}\sum_{j'=j+1}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\right\|\left\|(\frac{1_n}{n}-\prod_{q=j+1}^{t}\bar{W}^q\boldsymbol{e}_i)\right\|\left\|G(x_{i_{j'}}^{j'},\xi_{*,j'}^{i_{j'}})\right\|\left\|(\frac{1_n}{n}-\prod_{q=j'+1}^{t}\bar{W}^q\boldsymbol{e}_i)\right\| \tag{71}$$

For any $\alpha_{j,j'}>0$ we find

$$\leq 2\sum_{j=0}^{t}\sum_{j'=j+1}^{t}\Bigg(\frac{\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\right\|^2\left\|G(x_{i_{j'}}^{j'},\xi_{*,j'}^{i_{j'}})\right\|^2}{2\alpha_{j,j'}} \tag{72}$$

$$+\frac{\alpha_{j,j'}\left\|(\frac{1_n}{n}-\prod_{q=j+1}^{t}\bar{W}^q\boldsymbol{e}_i)\right\|^2\left\|(\frac{1_n}{n}-\prod_{q=j'+1}^{t}\bar{W}^q\boldsymbol{e}_i)\right\|^2}{2}\Bigg) \tag{73}$$

$$\leq\sum_{j\neq j'}^{t}\Bigg(\frac{\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\right\|^2\left\|G(x_{i_{j'}}^{j'},\xi_{*,j'}^{i_{j'}})\right\|^2}{2\alpha_{j,j'}}+\frac{\alpha_{j,j'}\rho^{t-\min\{j,j'\}}}{2}(\frac{n-1}{n})^2\Bigg),\alpha_{j,j'}=\alpha_{j',j} \tag{74}$$

By applying inequality of arithmetic and geometric means to the term in the last step, we can choose $\alpha_{j,j'}>0$ s.t.

$$\leq\frac{n-1}{n}\sum_{j\neq j'}^{t}\left(\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\right\|\left\|G(x_{i_{j'}}^{j'},\xi_{*,j'}^{i_{j'}})\right\|\rho^{\frac{t-\min\{j,j'\}}{2}}\right) \tag{75}$$

$$\leq\frac{n-1}{n}\sum_{j\neq j'}^{t}\left(\frac{\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\right\|^2+\left\|G(x_{i_{j'}}^{j'},\xi_{*,j'}^{i_{j'}})\right\|^2}{2}\rho^{\frac{t-\min\{j,j'\}}{2}}\right) \tag{76}$$

$$=\frac{n-1}{n}\sum_{j\neq j'}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\right\|^2\rho^{\frac{t-\min\{j,j'\}}{2}} \tag{77}$$

$$=\frac{n-1}{n}\sum_{j=0}^{t}\sum_{j'=j+1}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\right\|^2\rho^{\frac{t-j}{2}} \tag{78}$$

$$=\frac{n-1}{n}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\right\|^2 2(t-j)\rho^{\frac{t-j}{2}} \tag{79}$$

$$=\frac{n-1}{n}\sum_{j=0}^{t}2(t-j)\rho^{\frac{t-j}{2}}\left\|\frac{1}{M}\sum_{m=1}^{M}\nabla\ell(x_{i_j}^j;\xi_{m,j}^{i_j})\right\|^2 \tag{80}$$

Using Lemma 4 (and the expectation $\mathbb{E}_{\xi\sim\mathcal{D}_i}$ that was omitted above but is present) yields

$$=\frac{2(n-1)}{n}\sum_{j=0}^{t}2(t-j)\rho^{\frac{t-j}{2}}\left(\frac{\sigma^2}{M}+\left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2\right) \tag{81}$$

$$\leq\frac{4(n-1)\sqrt{\rho}\sigma^2}{Mn(1-\sqrt{\rho})^2}+\frac{2(n-1)}{n}\sum_{j=0}^{t}\left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2 2(t-j)\rho^{\frac{t-j}{2}} \tag{82}$$

Taking the expectation over worker i_j yields the desired bound

$$\mathbb{E}_{i_j}\left[\frac{4(n-1)\sqrt{\rho}\sigma^2}{Mn(1-\sqrt{\rho})^2}+\frac{2(n-1)}{n}\sum_{j=0}^{t}\left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2 2(t-j)\rho^{\frac{t-j}{2}}\right] \tag{83}$$

$$=\frac{4(n-1)\sqrt{\rho}\sigma^2}{Mn(1-\sqrt{\rho})^2}+\frac{2(n-1)}{n}\sum_{j=0}^{t}\mathbb{E}_{i_j}\left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2 2(t-j)\rho^{\frac{t-j}{2}} \tag{84}$$

Bounding Term B_3.

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \tag{85}$$

$$=\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal+\phi 1^\intercal-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \tag{86}$$

$$=\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\Big[\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)-\prod_{q=j+1}^{t}(\bar{W}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\Big]\right\|^2 \tag{87}$$

$$\leq 2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \tag{88}$$

$$+2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(\bar{W}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \tag{89}$$

Due to the structure of $\phi 1^\intercal$, multiplying this matrix by $\frac{1_n}{n}$ or $\boldsymbol{e}_i$ yields the same result. Using this, as well as the double stochasticity of $\bar{W}$, we find

$$=2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\Bigg[\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2+\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}\bar{W}^q(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2\Bigg] \tag{90}$$

$$=2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\Bigg[\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2+\underbrace{\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})(\frac{1_n}{n}-\prod_{q=j+1}^{t}\bar{W}^q\boldsymbol{e}_i)\right\|^2}_{B_1}\Bigg] \tag{91}$$

Using the result of Lemma 3, as our communication graph $\mathcal{G}$ is uniformly strongly connected and $[W_{i_t}^t]_{i,i}\geq 1/n$ by construction, we see that

$$|\big[\prod_{q=j+1}^{t}\bar{W}^q-\phi 1^\intercal\big]_{h,k}|\leq 4\nu^{t-j-1}\ \forall\ h,k. \tag{92}$$

Using this result, we find that

$$|\big[\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\big]_h|\leq 8(\frac{n-1}{n})\nu^{t-j-1}\ \forall\ h. \tag{93}$$

We can remove the -1 exponent by doubling the constant out front

$$|\big[\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\big]_h|\leq 16(\frac{n-1}{n})\nu^{t-j}\ \forall\ h. \tag{94}$$

Finally, using the fact that $G(x_{i_j}^j, \xi_{*,j}^{i_j})$ is all zeros except for one column, the i_j-th column, yields the desired result

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \tag{95}$$

$$\leq \sum_{j=0}^{t}256(\frac{n-1}{n})^2\nu^{2(t-j)}\mathbb{E}_{\xi\sim\mathcal{D}_i}\left\|\frac{1}{M}\sum_{m=1}^{M}\nabla\ell(x_{i_j}^j,\xi_{*,j}^{i_j})\right\|^2. \tag{96}$$

Utilizing Lemma 4 yields

$$\leq 256(\frac{n-1}{n})^2\sum_{j=0}^{t}\nu^{2(t-j)}\left(\frac{\sigma^2}{M}+\left\|\nabla f_{i_j}(x_{i_j}^t)\right\|^2\right). \tag{97}$$

By properties of geometric series, and taking the expectation over worker i_j, we find

$$\leq \frac{256\sigma^2}{(1-\nu^2)M}(\frac{n-1}{n})^2+256(\frac{n-1}{n})^2\sum_{j=0}^{t}\mathbb{E}_{i_j}\left\|\nabla f_{i_j}(x_{i_j}^t)\right\|^2\nu^{2(t-j)}. \tag{98}$$

Using the bound of B_1 in the main proof above, we arrive at the final bound of B_3

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \leq \frac{512\sigma^2}{(1-\nu^2)M}(\frac{n-1}{n})^2 \tag{99}$$

$$+\frac{4(n-1)\sigma^2}{(1-\rho)Mn}+512(\frac{n-1}{n})^2\sum_{j=0}^{t}\mathbb{E}_{i_j}\left\|\nabla f_{i_j}(x_{i_j}^t)\right\|^2\nu^{2(t-j)} \tag{100}$$

$$+\frac{4(n-1)}{n}\sum_{j=0}^{t}\mathbb{E}_{i_j}\left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2\rho^{t-j} \tag{101}$$

Simplifying this yields

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \leq \frac{4(n-1)\sigma^2}{Mn}\left(\frac{1}{(1-\rho)}+\frac{128}{(1-\nu^2)}\right) \tag{102}$$

$$+\frac{4(n-1)}{n}\sum_{j=0}^{t}\mathbb{E}_{i_j}\left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2\left(\rho^{t-j}+128\nu^{2(t-j)}\right) \tag{103}$$

Bounding Term B_4. Following similar steps as bounding Term B_2 we find

$$2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\sum_{j'=j+1}^{t}\langle G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i), \tag{104}$$

$$G(x_{i_{j'}}^{j'},\xi_{*,j'}^{i_{j'}})\prod_{q=j'+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\rangle \tag{105}$$

$$\leq 2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\sum_{j'=j+1}^{t}\left(\frac{\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2}{2}\right. \tag{106}$$

$$\left.+\frac{\left\|G(x_{i_{j'}}^{j'},\xi_{*,j'}^{i_{j'}})\prod_{q=j'+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2}{2}\right) \tag{107}$$

$$\leq 2\underbrace{\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2}_{B_3} \tag{108}$$

Once again, we can use the proof of Lemma 1 to bound this result.

Finishing Bound of Term B. Putting all terms together, we find that Term B is bounded above by

$$\sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \left\|\bar{x}^{t+1} - x_i^{t+1}\right\|^2 \leq 2\gamma^2 \sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \Bigg(\frac{2(n-1)\sigma^2}{(1-\rho)Mn} + \frac{2(n-1)}{n}\sum_{j=0}^{t} \mathbb{E}_{i_j} \left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2 \rho^{t-j} \tag{109}$$

$$+\frac{4(n-1)\sqrt{\rho}\sigma^2}{Mn(1-\sqrt{\rho})^2} + \frac{2(n-1)}{n}\sum_{j=0}^{t} \mathbb{E}_{i_j} \left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2 2(t-j)\rho^{\frac{t-j}{2}} + \frac{12(n-1)\sigma^2}{Mn}\left(\frac{1}{(1-\rho)} + \frac{128}{(1-\nu^2)}\right) \tag{110}$$

$$+\frac{12(n-1)}{n}\sum_{j=0}^{t} \mathbb{E}_{i_j} \left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2 \left(\rho^{t-j} + 128\nu^{2(t-j)}\right)\Bigg) \tag{111}$$

Simplifying results in

$$\leq 2\gamma^2 \sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \Bigg(\frac{4(n-1)\sigma^2}{Mn}\left(\frac{7}{2(1-\rho)} + \frac{\sqrt{\rho}}{(1-\sqrt{\rho})^2} + \frac{384}{(1-\nu^2)}\right) \tag{112}$$

$$+\frac{4(n-1)}{n}\sum_{j=0}^{t} \mathbb{E}_{i_j} \left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2 \left(\frac{7}{2}\rho^{t-j} + (t-j)\rho^{\frac{t-j}{2}} + 384\nu^{2(t-j)}\right)\Bigg) \tag{113}$$

$$\leq \frac{8\gamma^2\sigma^2 T}{M}\Bigg(\underbrace{\frac{n-1}{n}\left(\frac{7}{2(1-\rho)} + \frac{\sqrt{\rho}}{(1-\sqrt{\rho})^2} + \frac{384}{(1-\nu^2)}\right)}_{\rho_\nu}\Bigg) \tag{114}$$

$$+\frac{8(n-1)\gamma^2}{n}\sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \sum_{j=0}^{t} \mathbb{E}_{i_j} \left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2 \left(\frac{7}{2}\rho^{t-j} + (t-j)\rho^{\frac{t-j}{2}} + 384\nu^{2(t-j)}\right) \tag{115}$$

Using Lemma 5 results in

$$\leq \frac{8\gamma^2\sigma^2 T\rho_\nu}{M} + \frac{8(n-1)\gamma^2}{n}\sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \sum_{j=0}^{t} \Bigg(2\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^j)\right\|^2 \tag{116}$$

$$+12L^2\sum_{i=1}^{n} p_i \left\|\bar{x}^j - x_i^j\right\|^2 + 6\zeta^2\Bigg)\left(\frac{7}{2}\rho^{t-j} + (t-j)\rho^{\frac{t-j}{2}} + 384\nu^{2(t-j)}\right) \tag{117}$$

$$\leq \frac{8\gamma^2\sigma^2 T\rho_\nu}{M} + \frac{16(n-1)\gamma^2}{n}\sum_{t=0}^{T-1}\sum_{j=0}^{t} \Bigg(\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^j)\right\|^2 \tag{118}$$

$$+6L^2\sum_{i=1}^{n} p_i \left\|\bar{x}^j - x_i^j\right\|^2 + 3\zeta^2\Bigg)\left(\frac{7}{2}\rho^{t-j} + (t-j)\rho^{\frac{t-j}{2}} + 384\nu^{2(t-j)}\right) \tag{119}$$

$$\leq \frac{8\gamma^2\sigma^2 T\rho_\nu}{M} + \frac{16(n-1)\gamma^2}{n}\sum_{j=0}^{T-1}\sum_{t=j+1}^{\infty} \Bigg(\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^j)\right\|^2 \tag{120}$$

$$+6L^2\sum_{i=1}^{n} p_i \left\|\bar{x}^j - x_i^j\right\|^2 + 3\zeta^2\Bigg)\left(\frac{7}{2}\rho^{t-j} + (t-j)\rho^{\frac{t-j}{2}} + 384\nu^{2(t-j)}\right) \tag{121}$$

$$\leq \frac{8\gamma^2\sigma^2 T\rho_\nu}{M} + \frac{16(n-1)\gamma^2}{n}\sum_{j=0}^{T-1} \Bigg(\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^j)\right\|^2 \tag{122}$$

$$+6L^2\sum_{i=1}^{n} p_i \left\|\bar{x}^j - x_i^j\right\|^2 + 3\zeta^2\Bigg)\left(\sum_{h=0}^{\infty}\frac{7}{2}\rho^h + h\rho^{\frac{h}{2}} + 384\nu^{2h}\right) \tag{123}$$

$$\leq \frac{8\gamma^2\sigma^2 T\rho_\nu}{M} + 16\rho_\nu\gamma^2\sum_{j=0}^{T-1}\Bigg(\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^j)\right\|^2 + 6L^2\sum_{i=1}^{n} p_i \left\|\bar{x}^j - x_i^j\right\|^2 + 3\zeta^2\Bigg) \tag{124}$$

Using Equation (46) ends with

$$\leq \frac{8\gamma^2\sigma^2 T\rho_\nu}{M} + 48T\rho_\nu\gamma^2\zeta^2 + 16\rho_\nu\gamma^2 \sum_{t=0}^{T-1}\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^t)\right\|^2 + 96L^2\rho_\nu\gamma^2 \sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \left\|\bar{x}^{t+1} - x_i^{t+1}\right\|^2 \tag{125}$$

Now subtract the final term on the right hand side from both sides

$$\sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \left\|\bar{x}^{t+1} - x_i^{t+1}\right\|^2 \Big(\underbrace{1 - 96L^2\rho_\nu\gamma^2}_{z}\Big) \leq \frac{8\gamma^2\sigma^2 T\rho_\nu}{M} + 48T\rho_\nu\gamma^2\zeta^2 + 16\rho_\nu\gamma^2 \sum_{t=0}^{T-1}\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^t)\right\|^2 \tag{126}$$

By Lemma 6, we can divide z from both sides

$$\sum_{t=0}^{T-1}\sum_{i=1}^{n} p_i \left\|\bar{x}^{t+1} - x_i^{t+1}\right\|^2 \leq \frac{8\gamma^2\sigma^2 T\rho_\nu}{Mz} + \frac{48T\rho_\nu\gamma^2\zeta^2}{z} + \frac{16\rho_\nu\gamma^2}{z} \sum_{t=0}^{T-1}\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^t)\right\|^2 \tag{127}$$

Finishing Bound of Term A. Substituting the bound of B above into Equation (48) yields

$$f(\bar{x}^T) - f(\bar{x}^0) \leq -\frac{\gamma}{2n}\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)\right\|^2 - \frac{\gamma}{2n}\sum_{t=0}^{T-1}\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^t)\right\|^2 \tag{128}$$

$$+ \Bigg(\underbrace{\frac{3L}{(n^2-3)} + \frac{\gamma L^2 p_{max}}{2n}}_{\phi}\Bigg)\Bigg(\frac{8\gamma^2\sigma^2 T\rho_\nu}{Mz} + \frac{48T\rho_\nu\gamma^2\zeta^2}{z} + \frac{16\rho_\nu\gamma^2}{z}\sum_{t=0}^{T-1}\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^t)\right\|^2\Bigg) \tag{129}$$

Rearranging terms simplifies the inequality above to

$$f(\bar{x}^T) - f(\bar{x}^0) \leq -\frac{\gamma}{2n}\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)\right\|^2 + \frac{\gamma}{2n}\Big(\frac{32\rho_\nu\phi\gamma n}{z} - 1\Big)\sum_{t=0}^{T-1}\left\|\sum_{i=1}^{n} p_i \nabla f_i(x_i^t)\right\|^2 \tag{130}$$

$$+ \frac{8\phi\gamma^2\sigma^2 T\rho_\nu}{Mz} + \frac{48\phi T\rho_\nu\gamma^2\zeta^2}{z} \tag{131}$$

From Lemma 8, we find that $(1 - \frac{32\rho_\nu\phi\gamma n}{z}) \geq 0$. Therefore, the second term of the right hand side above can be removed.

$$f(\bar{x}^T) - f(\bar{x}^0) \leq -\frac{\gamma}{2n}\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)\right\|^2 + \frac{8\phi\gamma^2\sigma^2 T\rho_\nu}{Mz} + \frac{48\phi T\rho_\nu\gamma^2\zeta^2}{z} \tag{132}$$

Rearranging the inequality above and dividing by T yields

$$\frac{1}{T}\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)\right\|^2 \leq \frac{2n\big(f(\bar{x}^0) - f(\bar{x}^T)\big)}{T\gamma} + \frac{16n\phi\gamma\sigma^2\rho_\nu}{Mz} + \frac{96n\phi\rho_\nu\gamma\zeta^2}{z} \tag{133}$$

$$= \frac{2n\big(f(\bar{x}^0) - f(\bar{x}^T)\big)}{T\gamma} + \frac{16\gamma\phi n\rho_\nu}{z}\big(\frac{\sigma^2}{M} + 6\zeta^2\big) \tag{134}$$

From Lemmas 6 and 7, the inequality above becomes

$$\frac{1}{T}\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)\right\|^2 \leq \frac{2n\big(f(\bar{x}^0) - f(\bar{x}^T)\big)}{T\gamma} + \frac{\frac{1921}{10}L\gamma\rho_\nu}{n}\big(\frac{\sigma^2}{M} + 6\zeta^2\big) \tag{135}$$

Substituting in the defined step-size γ (as well as its bound) yields

$$\frac{1}{T}\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)\right\|^2 \leq \frac{2n\big(f(\bar{x}^0) - f(\bar{x}^T)\big)}{T}\Bigg(\frac{\sqrt{TL} + \sqrt{M}}{\sqrt{Mn^2\Delta_f}}\Bigg) \tag{136}$$

$$+ \frac{\frac{1921}{10}L}{n}\Bigg(\frac{\sqrt{Mn^2\Delta_f}}{\sqrt{TL}}\Bigg)\rho_\nu\big(\frac{\sigma^2}{M} + 6\zeta^2\big) \tag{137}$$

$$= \frac{2\sqrt{\Delta_f}}{T} + \frac{2\sqrt{L\Delta_f}}{\sqrt{TM}} + \frac{\frac{1921}{10}\sqrt{L\Delta_f}\rho_\nu\big(\sigma^2 + 6\zeta^2\sqrt{M}\big)}{\sqrt{TM}} \tag{138}$$

The final desired result is shown as

$$\frac{1}{T}\sum_{t=0}^{T-1}\left\|\nabla f(\bar{x}^t)\right\|^2 \leq \frac{2\sqrt{\Delta_f}}{T} + \frac{2\sqrt{L\Delta_f}\left(1+\frac{1921}{20}\rho_\nu\left(\sigma^2+6\zeta^2\sqrt{M}\right)\right)}{\sqrt{TM}} \tag{139}$$

□

G Additional Lemmas

Lemma 2 (From Lemma 3 in Lian et al. 2018 [20])**.** *Let W^t be a symmetric doubly stochastic matrix for each iteration t. Then*

$$\left\|\frac{\mathbf{1}_n}{n} - \prod_{t=1}^{T} W^t e_i\right\|^2 \leq \frac{n-1}{n}\rho^T,\ \forall T \geq 0.$$

Lemma 3 (From Corollary 2 in Nedic and Olshevsky 2014 [22])**.** *Let the communication graph $\mathcal{G}$ be uniformly strongly connected (otherwise known as B-strongly-connected for some integer $B > 0$), and $A(t) \in \mathbb{R}^{n\times n}$ be a column stochastic matrix with $[A(t)]_{i,i} \geq 1/n\ \forall i, t$. Define the product of matrices $A(t)$ through $A(s)$ (for $t \geq s \geq 0$) as $A(t:s) := A(t)\dots A(s)$. Then, there exists a stochastic vector $\phi(t) \in \mathbb{R}^n$ such that*

$$|[A(t:s)]_{i,j} - \phi_i(t)| \leq C\nu^{t-s}$$

will always hold for the following values of C and ν,

$$C = 4, \quad \nu = (1 - 1/n^{nB})^{1/B} < 1.$$

Lemma 4. *Under Assumption 1, the following inequality holds*

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\left\|\frac{1}{M}\sum_{m=1}^{M}\nabla\ell(x_i^t,\xi_{m,t}^i)\right\|^2 \leq \frac{\sigma^2}{M} + \left\|\nabla f_i(x_i^t)\right\|^2.$$

Lemma 5. *Under Assumption 1, the following inequality holds*

$$\mathbb{E}_i\left\|\nabla f_i(x_i^t)\right\|^2 \leq 2\left\|\sum_{i=1}^{n} p_i\nabla f_i(x_i^t)\right\|^2 + 12L^2\sum_{i=1}^{n} p_i\left\|\bar{x}^t - x_i^t\right\|^2 + 6\zeta^2.$$

Lemma 6. *Given the defined step-size γ and total iterations T in Theorem 1, the term z is and bounded by*

$$1 > z := 1 - 96L^2\rho_\nu\gamma^2 \geq 1 - \frac{384}{193^2(775)}.$$

Lemma 7. *Given the defined step-size γ and total iterations T in Theorem 1, the term ϕ is and bounded by*

$$\phi := \frac{3L}{(n^2-3)} + \frac{L^2\gamma p_{max}}{2n} \leq \frac{L}{n^2}(12 + \frac{2}{775(193)}).$$

Lemma 8. *Given the defined step-size γ and total iterations T in Theorem 1, we find the following bound*

$$(1 - \frac{32\rho_\nu\phi\gamma n}{z}) \geq 0$$

H Lemma Proofs

Proof of Lemma 1. These steps are shown in the Main Theorem proof. Due to the symetry and double stochasticity of $\bar{W}$ we find

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \tag{140}$$

$$=\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal+\phi 1^\intercal-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \tag{141}$$

$$=\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\Big[\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)-\prod_{q=j+1}^{t}(\bar{W}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\Big]\right\|^2 \tag{142}$$

$$\leq 2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \tag{143}$$

$$+2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(\bar{W}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \tag{144}$$

Due to the structure of $\phi 1^\intercal$, multiplying this matrix by $\frac{1_n}{n}$ or $\boldsymbol{e}_i$ yields the same result. Using this, as well as the double stochasticity of $\bar{W}$, we find

$$=2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\Bigg[\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2+\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}\bar{W}^q(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2\Bigg] \tag{145}$$

$$=2\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\Bigg[\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2+\underbrace{\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})(\frac{1_n}{n}-\prod_{q=j+1}^{t}\bar{W}^q\boldsymbol{e}_i)\right\|^2}_{B_1}\Bigg] \tag{146}$$

Using the result of Lemma 3, as our communication graph $\mathcal{G}$ is uniformly strongly connected and $[W_{i_t}^t]_{i,i}\geq 1/n$ by construction, we see that

$$|\Big[\prod_{q=j+1}^{t}\bar{W}^q-\phi 1^\intercal\Big]_{h,k}|\leq 4\nu^{t-j-1}\ \forall\ h,k. \tag{147}$$

Using this result, we find that

$$|\Big[\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\Big]_h|\leq 8(\frac{n-1}{n})\nu^{t-j-1}\ \forall\ h. \tag{148}$$

We can remove the -1 exponent by doubling the constant out front

$$|\Big[\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\Big]_h|\leq 16(\frac{n-1}{n})\nu^{t-j}\ \forall\ h. \tag{149}$$

Finally, using the fact that $G(x_{i_j}^j, \xi_{*,j}^{i_j})$ is all zeros except for one column, the i_j-th column, yields the desired result

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\phi 1^\intercal)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \tag{150}$$

$$\leq \sum_{j=0}^{t}256(\frac{n-1}{n})^2\nu^{2(t-j)}\mathbb{E}_{\xi\sim\mathcal{D}_i}\left\|\frac{1}{M}\sum_{m=1}^{M}\nabla\ell(x_{i_j}^j,\xi_{*,j}^{i_j})\right\|^2. \tag{151}$$

Utilizing Lemma 4 yields

$$\leq 256(\frac{n-1}{n})^2\sum_{j=0}^{t}\nu^{2(t-j)}\left(\frac{\sigma^2}{M}+\left\|\nabla f_{i_j}(x_{i_j}^t)\right\|^2\right). \tag{152}$$

By properties of geometric series, and taking the expectation over worker i_j, we find

$$\leq \frac{256\sigma^2}{(1-\nu^2)M}(\frac{n-1}{n})^2+256(\frac{n-1}{n})^2\sum_{j=0}^{t}\mathbb{E}_{i_j}\left\|\nabla f_{i_j}(x_{i_j}^t)\right\|^2\nu^{2(t-j)}. \tag{153}$$

Using the bound Equation (69) (term B_1) in the main proof above, we arrive at the final bound

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \leq \frac{512\sigma^2}{(1-\nu^2)M}(\frac{n-1}{n})^2 \tag{154}$$

$$+\frac{4(n-1)\sigma^2}{(1-\rho)Mn}+512(\frac{n-1}{n})^2\sum_{j=0}^{t}\mathbb{E}_{i_j}\left\|\nabla f_{i_j}(x_{i_j}^t)\right\|^2\nu^{2(t-j)} \tag{155}$$

$$+\frac{4(n-1)}{n}\sum_{j=0}^{t}\mathbb{E}_{i_j}\left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2\rho^{t-j} \tag{156}$$

Simplifying this yields

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \leq \frac{4(n-1)\sigma^2}{Mn}\left(\frac{1}{(1-\rho)}+\frac{128}{(1-\nu^2)}\right) \tag{157}$$

$$+\frac{4(n-1)}{n}\sum_{j=0}^{t}\mathbb{E}_{i_j}\left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2\left(\rho^{t-j}+128\nu^{2(t-j)}\right) \tag{158}$$

Thus, we have our desired result

$$\mathbb{E}\sum_{j=0}^{t}\left\|G(x_{i_j}^j,\xi_{*,j}^{i_j})\prod_{q=j+1}^{t}(W_{i_q}^q-\bar{W}^q)(\frac{1_n}{n}-\boldsymbol{e}_i)\right\|^2 \leq \mathcal{O}(\frac{\sigma^2}{M}+\mathbb{E}\sum_{j=0}^{t}\left\|\nabla f_{i_j}(x_{i_j}^j)\right\|^2). \tag{159}$$

□

Proof of Lemma 4.

$$\mathbb{E}_{\xi\sim\mathcal{D}_i}\left\|\frac{1}{M}\sum_{m=1}^{M}\nabla\ell(x_i^t,\xi_{m,t}^i)\right\|^2 \tag{160}$$

$$=\frac{1}{M^2}\mathbb{E}_{\xi\sim\mathcal{D}_i}\left\|\sum_{m=1}^{M}\nabla\ell(x_i^t,\xi_{m,t}^i)-\nabla f_i(x_i^t)+\nabla f_i(x_i^t)\right\|^2 \tag{161}$$

$$=\frac{1}{M^2}\mathbb{E}_{\xi\sim\mathcal{D}_i}\left\|\sum_{m=1}^{M}\nabla\ell(x_i^t,\xi_{m,t}^i)-\nabla f_i(x_i^t)\right\|+\mathbb{E}_{\xi\sim\mathcal{D}_i}\left\|\sum_{m=1}^{M}\nabla f_i(x_i^t)\right\|^2 \tag{162}$$

$$+\frac{2}{M^2}\mathbb{E}_{\xi\sim\mathcal{D}_i}\Big\langle\sum_{m=1}^{M}\big(\nabla\ell(x_i^t,\xi_{m,t}^i)-\nabla f_i(x_i^t)\big),\sum_{m=1}^{M}\nabla f_i(x_i^t)\Big\rangle \tag{163}$$

$$=\frac{1}{M^2}\sum_{m=1}^{M}\mathbb{E}_{\xi\sim\mathcal{D}_i}\left\|\nabla\ell(x_i^t,\xi_{m,t}^i)-\nabla f_i(x_i^t)\right\|+M^2\left\|\nabla f_i(x_i^t)\right\|^2 \tag{164}$$

$$\leq\frac{\sigma^2}{M}+\left\|\nabla f_i(x_i^t)\right\|^2 \tag{165}$$

□

Proof of Lemma 5.

$$\mathbb{E}_i\left\|\nabla f_i(x_i^t)\right\|^2=\mathbb{E}_i\left\|\nabla f_i(x_i^t)-\sum_{i'=1}^{n}p_{i'}\nabla f_i(x_{i'}^t)+\sum_{i'=1}^{n}p_{i'}\nabla f_i(x_{i'}^t)\right\|^2 \tag{166}$$

$$\leq 2\mathbb{E}_i\left(\left\|\nabla f_i(x_i^t)-\sum_{j=1}^{n}p_j\nabla f_j(x_j^t)\right\|^2+\left\|\sum_{j=1}^{n}p_j\nabla f_j(x_j^t)\right\|^2\right) \tag{167}$$

The first term on the right hand side can be bounded by

$$\mathbb{E}_i\left\|\nabla f_i(x_i^t)-\sum_{j=1}^{n}p_j\nabla f_j(x_j^t)\right\|^2=3\mathbb{E}_i\left(\left\|\nabla f_i(x_i^t)-\nabla f_i(\frac{X^t1_n}{n})\right\|^2\right. \tag{168}$$

$$\left.+\left\|\sum_{j=1}^{n}p_j\nabla f_j(\frac{X^t1_n}{n})-\sum_{j=1}^{n}p_j\nabla f_j(x_j^t)\right\|^2+\left\|\nabla f_i(\frac{X^t1_n}{n})-\sum_{j=1}^{n}p_j\nabla f_j(\frac{X^t1_n}{n})\right\|^2\right) \tag{169}$$

$$\leq 3\mathbb{E}_i\left(L^2\left\|x_i^t-\bar{x}^t\right\|^2+\sum_{j=1}^{n}p_j\left\|\nabla f_j(\bar{x}^t)-\nabla f_j(x_j^t)\right\|^2+\left\|\nabla f_i(\bar{x}^t)-\nabla f(\bar{x}^t)\right\|^2\right) \tag{170}$$

$$\leq 3\mathbb{E}_i\left(L^2\left\|x_i^t-\bar{x}^t\right\|^2+L^2\sum_{j=1}^{n}p_j\left\|\bar{x}^t-x_j^t\right\|^2\right)+3\zeta^2 \tag{171}$$

$$=6L^2\sum_{i=1}^{n}p_i\left\|\bar{x}^t-x_i^t\right\|^2+3\zeta^2 \tag{172}$$

Combining all terms yields the final result

$$\mathbb{E}_i\left\|\nabla f_i(x_i^t)\right\|^2\leq 2\left\|\sum_{i=1}^{n}p_i\nabla f_i(x_i^t)\right\|^2+12L^2\sum_{i=1}^{n}p_i\left\|\bar{x}^t-x_i^t\right\|^2+6\zeta^2 \tag{173}$$

□

Proof of Lemma 6. It is trivial to see that $z < 1$. We now determine the lower bound of z given $n \geq 2$, $p_{max} \geq 1/n$, and $\rho_\nu \geq 775/4$

$$z = 1 - 96L^2\rho_\nu\gamma^2 = 1 - 96L^2\rho_\nu(\frac{Mn^2\Delta_f}{TL}) = 1 - 96L^2\rho_\nu(\frac{1}{193^2L^2\rho_\nu^2n^2p_{max}^2}) \tag{174}$$

$$= 1 - \frac{96}{193^2\rho_\nu n^2 p_{max}^2} \leq 1 - \frac{384}{193^2(775)} \tag{175}$$

□

Proof of Lemma 7. Given $n \geq 2$ and $\rho_\nu \geq 775/4$, and the definition of γ, one can see

$$\phi = (\frac{3L}{(n^2-3)} + \frac{L^2\gamma p_{max}}{2n}) = \frac{L}{n^2}(\frac{3}{(1-3/n^2)} + \frac{L\gamma n p_{max}}{2}) \tag{176}$$

$$\leq \frac{L}{n^2}(\frac{3}{(1-3/n^2)} + \frac{Lnp_{max}}{2}(\frac{\sqrt{Mn^2\Delta_f}}{\sqrt{TL}})) \tag{177}$$

Using the definition of T it follows that

$$\phi \leq \frac{L}{n^2}(\frac{3}{(1-3/n^2)} + \frac{Lnp_{max}}{2}(\frac{1}{193L\rho_\nu np_{max}})) \leq \frac{L}{n^2}(\frac{3}{(1-3/4)} + \frac{1}{386\rho_\nu}) \tag{178}$$

$$= \frac{L}{n^2}(12 + \frac{2}{775(193)}) \tag{179}$$

□

Proof of Lemma 8. Given $n \geq 2$, $p_{max} \geq 1/n$ Lemma 6, and Lemma 7, one can see

$$1 - 1 - \frac{32\rho_\nu\phi\gamma n}{z} = 1 - \frac{32\rho_\nu\phi n}{z}(\frac{1}{193L\rho_\nu np_{max}}) = 1 - \frac{32n\phi}{193Lz} \tag{180}$$

$$\geq 1 - \frac{32(12 + \frac{2}{775(193)})}{193zn} \geq 1 - \frac{16(12 + \frac{2}{775(193)})}{193(1 - \frac{384}{193^2(775)})} \geq 0 \tag{181}$$

□