OPT2024: 16th Annual Workshop on Optimization for Machine Learning

Pseudo-Asynchronous Local SGD: Robust and Efficient Data-Parallel Training

Hiroki Naganuma* NAGANUMA.HIROKI@MILA.QUEBEC
Mila, University of Montreal, Canada
Xinzhi Zhang* XINZHI20@UW.EDU
University of Washington, United States
Man-Chung Yue XYZ@SAMPLE.COM
The University of Hong Kong, Hong Kong
Ioannis Mitliagkas IOANNIS@MILA.QUEBEC
Mila, University of Montreal, Canada
Philipp Andre Witte† PWITTE@MICROSOFT.COM
Russell J. Hewett† RHEWETT@MICROSOFT.COM
Yin Tat Lee† YINTATLEE@MICROSOFT.COM
Microsoft, United States

Abstract

Recent trends of larger model and larger datasets require huge amounts of computational resources, making distributed deep learning essential. Data parallelism is a common approach to speed up training, but it often involves frequent communication between workers, which can be a bottleneck. In this work, we propose a method called Pseudo-Asynchronous Local SGD (PALSGD) to improve the efficiency of data-parallel training. PALSGD is a novel extension of LocalSGD [21], designed to further reduce communication frequency by introducing a pseudo-synchronization mechanism. PALSGD allows the use of longer synchronization intervals compared to standard LocalSGD. Despite the reduced communication frequency, the pseudo-synchronization approach ensures that model consistency is maintained, leading to performance results comparable to those achieved with more frequent synchronization. Furthermore, we provide a theoretical analysis of PALSGD, establishing its convergence and deriving its convergence rate. This analysis offers insights into the algorithm's behavior and performance guarantees. We evaluated PALSGD on CIFAR-10 using a CNN and GPT-NEO on TinyStories. Our results show that PALSGD achieves better performance in less time compared to existing methods like distributed data parallel (DDP), Local SGD and DiLoCo [2].

1. Introduction

Training neural networks has become more computationally expensive, requiring distributed deep learning techniques to handle the growing data and model sizes. Standard approaches to distributed training typically rely on data parallelism [11], where a batch of training samples is further split into multiple micro batches that are assigned to different workers. These workers perform forward and backward passes on their local data shards and synchronize their model updates through operations like ALL-REDUCE. However, synchronization at every step introduces significant communication overhead, especially as the number of workers increases, because all model gradients have to be

*. Alphabetical order, –These authors contributed equally to this work. This work was performed when H.Naganuma and X.Zhang were Microsoft Research interns
†. Alphabetical order

synchronized between workers [14]. In addition, increasing the batch size to improve throughput can negatively impact model generalization, resulting in suboptimal performance [7].

To address these issues, we propose *Pseudo-Asynchronous Local SGD* (PALSGD), a novel extension of the Local SGD [21] framework that incorporates a pseudo-asynchronous model update mechanism. In PALSGD, workers perform local updates for extended periods and synchronize with local copies of central models probablistically, allowing them to avoid the strict synchronization required in traditional methods. This pseudo-synchronization reduces the frequency of ALL-REDUCE operations, mitigating communication overhead while maintaining model consistency. By introducing probabilistic updates, workers operate more independently between synchronization points, leading to better training efficiency. Our approach is particularly suited for large-scale distributed training scenarios, where communication delays and worker idling due to speed variations are common bottlenecks.

Our contributions are as follows:

- **Pseudo Synchronization**: We introduce a probabilistic pseudo-synchronization mechanism to allow workers to loosely synchronize with the global model, reducing the need for frequent full synchronization. This approach balances communication efficiency and model consistency.
- **Theoretical Analysis**: We provide a theoretical analysis of PALSGD, proving its convergence and deriving its convergence rate. This analysis provides insights into the algorithm's behavior and its performance guarantees.
- **Empirical Validation**: We demonstrate the effectiveness of PALSGD through experiments on CIFAR-10 and TinyStories [3] datasets. We show that it achieves better validation/train loss performance in shorter training times compared to existing methods like Distributed Data Parallel (DDP), Local SGD [21] and DiLoCo [2].

Our work builds upon previous research on Local SGD and asynchronous methods. We address their limitations and advance the field of efficient distributed deep learning.

2. Preliminaries

We consider the following stochastic optimization problem:

$$\min_{x \in \mathbb{R}^d} F(x), F(x) = \mathbb{E}_{\xi \sim \mathcal{D}} f(x, \xi). \tag{1}$$

which aims to minimize the expected value of our cost function f with samples ξ drawn from the universal data distribution $\mathcal{D}$. In distributed training, consider K workers running in parallel and they are initialized as the same parameter $x^{(0)}$. The training data is uniformly randomly partitioned into K data shards $\mathcal{D}_1, \cdots, \mathcal{D}_K$, with each worker performs local computations based on its own data shard, independent of the other workers.

In the *pseudo-asynchronous* setting, workers operate mostly asynchronously between synchronization points. The workers are only synchronized through the pre-scheduled ALL-REDUCE operations, which aggregates model parameters (or other states) from all workers. During ALL-REDUCE, each worker must reach the same point in its local computation, meaning that faster workers need to wait for slower ones before ALL-REDUCE can occur. However, between two consecutive synchronizations, the workers operate independently, performing local updates without having to wait for

Algorithm 1: Pseudo-Asynchronous Local SGD with Decoupled Optimizers

Data: $x^{(0)}$ (initial model), $K > 0$ (number of workers), $p \in [0, 1]$ (probability of mixing step), $\eta_t > 0$ (mixing rate), $H > 0$ (sync interval), optimizers INNEROPT and OUTEROPT, α_t (learning rate scheduler for INNEROPT)

for *worker* $k = 1, \cdots, K$ **do**
  $x_k^{(0)} \leftarrow x^{(0)}$;
  for $t = 0, \cdots, T-1$ **do**
    $b \sim U[0, 1]$;
    if $b \leq p$ **then**
      $x_k^{(t)} \leftarrow x_k^{(t)} - \frac{\alpha_t \eta_t}{p} \cdot (x_k^{(t)} - x^{(t)})$; `pseudo-synchronization step`
    else
      Sample data $\xi \sim \mathcal{D}_k$;
      $g_k^{(t)} \leftarrow \nabla f(x_k^{(t)}, \xi)$;
      $x_k^{(t+1)} \leftarrow \text{INNEROPT}(x_k^{(t)}, g_k^{(t)}, \frac{\alpha_t}{1-p})$; `gradient step`
    end
    if $(t+1) \mod H = 0$ **then**
      $\Delta^{(t)} \leftarrow \text{ALL-REDUCE}(x^{(t-1)} - x_k^{(t)})$; `aggregate outer gradient`
      $x^{(t+1)} \leftarrow \text{OUTEROPT}(x^{(t)}, \Delta^{(t)})$; `update global model`
    else
      $x^{(t+1)} \leftarrow x^{(t)}$
    end
  end
end

each other. This allows workers to progress at their own pace during most of the training process, reducing total idle time over training by decreasing number of communication operations.

3. Proposed Method: PALSGD

Our PALSGD algorithm is outlined in Algorithm 1. It extends the Local SGD method by introducing a pseudo-synchronous step that incorporates probabilistic synchronization with the "global" model $x^{(t)}$, which is stored locally on each worker. In this approach, each worker updates its local model independently for H steps, and with probability p, performs a pseudo-synchronization step that partially aligns the local model $x_k^{(t)}$ with the global model $x^{(t)}$. After H inner steps, $x^{(t)}$ is updated through an ALL-REDUCE operation, aggregating the differences between the local and global models. This probabilistic synchronization reduces the frequency of full synchronization and significantly lowering the communication overhead while maintaining sufficient alignment between workers' models.

To further improve the empirical performance, we apply several practical techniques. First, similar to Post-Local SGD [14], we initialize the model $x^{(0)}$ from a model pretrained by DDP. This addresses the instability issues often observed in the initial phases of training. Additionally, we employ a decoupled optimizer strategy. Following the DiLoCo framework [2], we employed AdamW [8] as INNEROPT that handles local updates and Nesterov momentum [22] as OUTEROPT that updates the global models. Together, these modifications ensure that PALSGD not only reduces communication costs but also achieves faster convergence and better model performance across various deep learning tasks.

Building on this framework, we showed the following theoretical convergence bound for a simplified version of our algorithm, where the inner optimizer is standard SGD and the outer model is updated by taking the average across inner models. We include the proof in Appendix A.

Theorem 1 (Convergence of PALSGD, Informal) *Let $x^{(0)}, \cdots, x^{(T-1)}$ be the sequence generated by Algorithm 1 with* INNEROPT *as* SGD *and* OUTEROPT *as* SGD *with step size* 1. *Under Assumptions 1, 2, 3, and 4, let $\kappa = \frac{L}{\mu}$. For any $0 < p \leq \frac{1}{2}$, and for any $T > 0$, there exists a sequence of inner step size $\{\alpha_t\}_{t=0}^{T-1}$, a sequence of mixing rate $\{\eta_t\}_{t=0}^{T-1}$, and a weight sequence $\{w_t\}_{t=0}^{T-1}$ such that for $\hat{x}_T = \frac{1}{Z_T}\sum_{t=0}^{T-1} w_t x^{(t)}$ where $Z_T = \sum_{t=0}^{T-1} w_t$, and ignorining the logrithmic and exponentially decaying terms, we have*

$$\mathbb{E}[F(\hat{x}_T)] - F(x^*) \leq \tilde{O}\left(\frac{\sigma^2}{\mu K T} + \frac{\kappa H^2 \sigma^2}{\mu T^2}\right). \tag{2}$$

4. Experiments

Experimental Setup

We conducted experiments using two datasets: CIFAR-10 for image classification and TinyStories for language modeling. For CIFAR-10, a small CNN architecture was used, for TinyStories, we employed GPT-NEO [1] with 8 million parameters to evaluate the performance of PALSGD in a distributed training environment.

The CIFAR-10 experiments simulated distributed training to measure the achievable accuracy. We compared PALSGD with DDP, LocalSGD, and DiLoCo across varying numbers of workers and synchronization intervals. For the TinyStories experiments, we trained the models in a real distributed environment using 4 to 8 workers. Further details are provided in Appendix C.

Simulation Experiments: CIFAR-10 on Small CNN

In the CIFAR-10 experiments, we observed that both DDP and LocalSGD significantly degrade in accuracy as the synchronization interval (H) or the number of workers (K) increased. In contrast, DiLoCo and PALSGD demonstrated more stable performance across these variables, with PALSGD showing the least sensitivity. Specifically, PALSGD outperformed LocalSGD by 10% when H=256 and K=4, and it surpassed DiLoCo by 1.0% when H=32 and K=4. These results highlight PALSGD's ability to maintain accuracy while reducing communication overhead.

1. https://huggingface.co/docs/transformers/en/model_doc/gpt_neo

Figure 1: Simulation Experiments: (Left) Comparison of K (Number of workers) with H=32. (Left) Comparison of H (Sync Interval) with K=4. PALSGD has low sensitivity to K and H, and achieves high accuracy consistently.

Practical Experiments: TinyStories on GPT-NEO

In the TinyStories experiments, PALSGD demonstrated significant reductions in communication overhead by minimizing the frequency of synchronization steps. Specifically, PALSGD reduced the total number of synchronization steps by 93.75% compared to DDP [2]. As a result, PALSGD shortened the total training time by 20-23% while achieving the target loss. DiLoCo, however, converged more slowly and was unable to reach the target loss within the same training time frame. More details can be found in Appendix C.

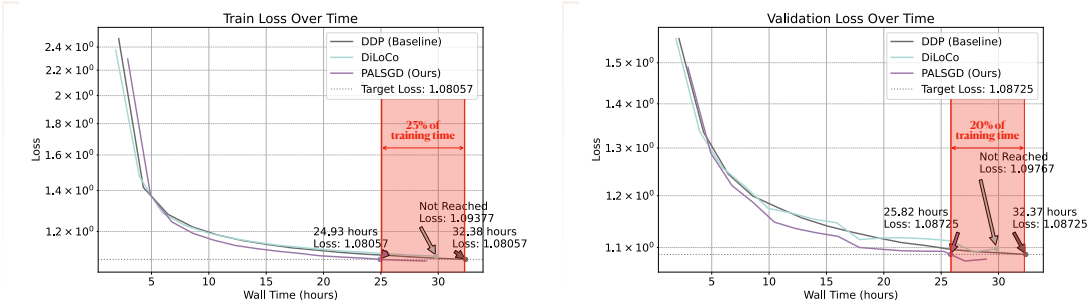

Figure 2: GPT-NEO Experiments (K=4 / H=16): Training time comparison across distributed algorithm to achieve target loss. While PALSGD achieve fastest and lowest loss, DDP is slowest and DiLoCo did not achieve target loss.

Our experiments demonstrate that PALSGD effectively balances communication efficiency and model performance. The probabilistic pseudo-synchronization mechanism allows workers to update their local models independently, leading to faster convergence and reduced communication overhead. Compared to existing methods, PALSGD achieves significant improvements in both training speed and model loss, particularly in large-scale distributed environments.

5. Discussion and Conclusion

We introduced Pseudo-Asynchronous Local SGD (PALSGD), which reduces communication overhead in large-scale distributed learning through probabilistic pseudo-synchronization. Extending

2. This is the theoretical value when the communication frequency is reduced to one-sixteenth.

Local SGD, PALSGD decreases the frequency of ALL-REDUCE operations, allowing for extended local updates. This method is particularly effective in high-latency environments, such as intercontinental data centers, where it enables more efficient, scalable training.

Our key contributions are as follows: i) We introduced PALSGD, a novel extension of Local SGD that incorporates probabilistic pseudo-synchronization, significantly reducing the cost of synchronization without sacrificing model performance. ii) We provided theoretical convergence bounds for a simplified version of PALSGD. iii) We empirically validated PALSGD on image classification and language modeling tasks, demonstrating its effectiveness in reducing training time and improving model performance compared to baseline methods such as DDP, Local SGD and DiLoCo.

The limitations of our work include several factors. First, our current approach assumes homogeneous hardware and network configuration across all workers. Future research could explore adaptive methods to address heterogeneous environments, where worker speeds or network latencies vary. Second, our theoretical analysis was simplified, focusing on PALSGD with SGD as the inner optimizer and assuming strongly convex functions, whereas training deep models is inherently non-convex. Extending this theoretical framework to more complex optimizers like Adam or other adaptive methods may offer deeper insights into the algorithm's performance. Finally, while PALSGD enhances communication efficiency, future studies could investigate further reducing synchronization costs, for example, by employing gradient compression techniques or decentralized communication patterns.

References

[1] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. Large scale distributed deep networks. *Advances in neural information processing systems*, 25, 2012.

[2] Arthur Douillard, Qixuan Feng, Andrei A Rusu, Rachita Chhaparia, Yani Donchev, Adhiguna Kuncoro, Marc'Aurelio Ranzato, Arthur Szlam, and Jiajun Shen. Diloco: Distributed low-communication training of language models. *arXiv preprint arXiv:2311.08105*, 2023.

[3] Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*, 2023.

[4] Gauthier Gidel, Reyhane Askari Hemmat, Mohammad Pezeshki, Rémi Le Priol, Gabriel Huang, Simon Lacoste-Julien, and Ioannis Mitliagkas. Negative momentum for improved game dynamics. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1802–1811. PMLR, 2019.

[5] Xinran Gu, Kaifeng Lyu, Longbo Huang, and Sanjeev Arora. Why (and when) does local sgd generalize better than sgd? *arXiv preprint arXiv:2303.01215*, 2023.

[6] Farzin Haddadpour, Mohammad Mahdi Kamani, Mehrdad Mahdavi, and Viveck Cadambe. Local sgd with periodic averaging: Tighter analysis and adaptive synchronization. *Advances in Neural Information Processing Systems*, 32, 2019.

[7] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

[8] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[9] Byung-Il Koh, Alan D George, Raphael T Haftka, and Benjamin J Fregly. Parallel asynchronous particle swarm optimization. *International journal for numerical methods in engineering*, 67(4):578–595, 2006.

[10] 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.

[11] Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, et al. Pytorch distributed: Experiences on accelerating data parallel training. *arXiv preprint arXiv:2006.15704*, 2020.

[12] Xiangru Lian, Yijun Huang, Yuncheng Li, and Ji Liu. Asynchronous parallel stochastic gradient for nonconvex optimization. *Advances in neural information processing systems*, 28, 2015.

[13] 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.

[14] Tao Lin, Sebastian U Stich, Kumar Kshitij Patel, and Martin Jaggi. Don't use large mini-batches, use local sgd. *arXiv preprint arXiv:1808.07217*, 2018.

[15] Bo Liu, Rachita Chhaparia, Arthur Douillard, Satyen Kale, Andrei A Rusu, Jiajun Shen, Arthur Szlam, and Marc'Aurelio Ranzato. Asynchronous local-sgd training for language modeling. *arXiv preprint arXiv:2401.09135*, 2024.

[16] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[17] 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.

[18] Jose Javier Gonzalez Ortiz, Jonathan Frankle, Mike Rabbat, Ari Morcos, and Nicolas Ballas. Trade-offs of local sgd at scale: An empirical study. *arXiv preprint arXiv:2110.08133*, 2021.

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

[20] Max Ryabinin, Eduard Gorbunov, Vsevolod Plokhotnyuk, and Gennady Pekhimenko. Moshpit sgd: Communication-efficient decentralized training on heterogeneous unreliable devices. *Advances in Neural Information Processing Systems*, 34:18195–18211, 2021.

[21] Sebastian U Stich. Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767*, 2018.

[22] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147. PMLR, 2013.

[23] Jianyu Wang and Gauri Joshi. Cooperative sgd: A unified framework for the design and analysis of local-update sgd algorithms. *Journal of Machine Learning Research*, 22(213): 1–50, 2021.

[24] Jianyu Wang, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat. Slowmo: Improving communication-efficient distributed sgd with slow momentum. *arXiv preprint arXiv:1910.00643*, 2019.

[25] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*, 2019.

[26] Sixin Zhang, Anna E Choromanska, and Yann LeCun. Deep learning with elastic averaging sgd. *Advances in neural information processing systems*, 28, 2015.

[27] Tuo Zhang, Lei Gao, Sunwoo Lee, Mi Zhang, and Salman Avestimehr. Timelyfl: Heterogeneity-aware asynchronous federated learning with adaptive partial training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5064–5073, 2023.

[28] Shuxin Zheng, Qi Meng, Taifeng Wang, Wei Chen, Nenghai Yu, Zhi-Ming Ma, and Tie-Yan Liu. Asynchronous stochastic gradient descent with delay compensation. In *International conference on machine learning*, pages 4120–4129. PMLR, 2017.

Appendix A. Proof of Theorem 1

A.1. Assumptions and Main Result

We make the following assumptions on F for our theoretical analysis:

Assumption 1 (L-Smoothness) *There exists a constant $L > 0$ such that for each ξ in the support of $\mathcal{D}$, and for each $x, y \in \mathbb{R}^d$,*

$$\|\nabla f(x,\xi) - \nabla f(y,\xi)\| \leq L\|x - y\|.$$

Assumption 2 (μ-Strongly Convex) *There exists a constant $\mu > 0$ such that for each ξ in the support of $\mathcal{D}$, $f(x,\xi)$ is μ-strongly convex. Moreover, write $x^* = \arg\min_{x\in\mathbb{R}^d} F(x)$ as the global minimal solution.*

Assumption 3 (Identical Data Distributions among Workers) *Let $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K$ be the data distributions for K workers. Assume that these distributions are identical and independent, denoted by $\mathcal{D}_1 = \mathcal{D}_2 = \cdots = \mathcal{D}_K = \mathcal{D}$.*

Assumption 4 (Bounded Variance) *There exists $\sigma \geq 0$ such that for any $x \in \mathbb{R}^d$,*

$$\mathbb{E}_{\xi\sim\mathcal{D}}[\|\nabla f(x,\xi)\|^2] \leq \sigma^2.$$

Theorem 2 (Convergence of PALSGD) *Let $x^{(0)}, \cdots, x^{(T-1)}$ be the sequence generated by Algorithm 1 with* INNEROPT *as* SGD *and* OUTEROPT *as* SGD *with step size* 1. *Under Assumptions 1, 2, 3, and 4, for any $0 < p \leq \frac{1}{2}$, let $\alpha_t = \alpha$, $\eta_t = \eta = \frac{p}{2H\alpha}$, and $w_t = (1-\mu\alpha)^{-(t+1)}$ where*

$$\alpha = \min\left(\frac{p}{48LH}, \frac{\ln(\mu^2 d_0 T^2 K/\sigma^2)}{\mu T}\right)$$

For any $T > 0$, let $\hat{x}_T = \frac{1}{Z_T}\sum_{t=0}^{T-1} w_t x^{(t)}$ where $Z_T = \sum_{t=0}^{T-1} w_t$, we have

$$\mathbb{E}[F(\hat{x}_T)] - F(x^*) \leq \tilde{O}\left(\frac{LHR_0^2}{p}\cdot\exp(-\frac{pT}{\kappa H}) + \frac{\sigma^2}{\mu KT} + \frac{\kappa H^2\sigma^2}{\mu T^2}\right). \tag{3}$$

where $R_0 = \|x^{(0)} - x^*\|^2$ and $\kappa = \frac{L}{\mu}$. Specifically, when $p \leq \frac{\mu}{LH}$, we have

$$\mathbb{E}[F(\hat{x}_T)] - F(x^*) \leq \tilde{O}\left(\frac{LHR_0^2}{p}\cdot\exp(-\frac{pT}{\kappa H}) + \frac{\sigma^2}{\mu KT} + \frac{(\kappa + p^{-1}H)\sigma^2}{\mu T^2}\right). \tag{4}$$

The roadmap for the remainder of this section is as follows: Section A.2 introduces the basic definitions used in the proof. Section A.3 provides a proof sketch and presents the main technical lemmas. Section A.4 contains the full proof of Theorem 2. Finally, Section A.5 proves the technical lemmas stated in Section A.3.

A.2. Basic Definitions

For any $k \in [K], t = 0, \cdots, T-1$, let $b_k^{(t)}$ denote a Bernoulli random variable with parameter p (i.e., $b_k^{(t)} = 1$ with probability p and $b_k^{(t)} = 0$ with probability $1-p$). Let

$$g_k^{(t)} = \frac{1 - b_k^{(t)}}{1-p} \nabla f(x_k^{(t)}, \xi_k^{(t)}) + \frac{\eta_t b_k^{(t)}}{pK}(x_k^{(t)} - x^{(t)}).$$

The for any t such that $(t+1) \mod H \neq 0$, we can rewrite the inner step as

$$x_k^{(t+1)} = x_k^{(t)} - \alpha_t \, g_k^{(t)}.$$

Let

$$g^{(t)} = \frac{1}{K} \sum_{k \in [K]} g_k^{(t)}. \tag{5}$$

Let $\bar{x}^{(t)} = \frac{1}{K} \sum_{k \in [K]} x_k^{(t)}$ as the current mean of the client servers at step t. Then we have

$$\bar{x}^{(t+1)} = \bar{x}^{(t)} - \alpha_t \, g^{(t)}.$$

Let $\xi_k^{(t)}$ denote the data sampled by the k-th server at step t. Let $\mathcal{F}_t = \{\xi_k^{(s)}\}_{s=0,\cdots,t-1,k\in[K]} \cup \{b_k^{(s)}\}_{s=0,\cdots,t-1,k\in[K]}$ for $t \geq 1$ and $\mathcal{F}_0 = \emptyset$. Define $\bar{g}^{(t)}$ as the expectation of $g^{(t)}$ over the randomness at step t, i.e.

$$\bar{g}^{(t)} = \mathbb{E}[g^{(t)} \mid \mathcal{F}_t] = \frac{1}{K} \sum_{k \in [K]} \left(\nabla F(x_k^{(t)}) + \eta_t (x_k^{(t)} - x^{(t)}) \right). \tag{6}$$

For any $t \geq 0$, let $t^- \leq t$ be the largest integral multiples of H that is at most t, i.e. t^- is the last iteration at or before t such that $x^{(t)}$ is updated. Similarly, let $t^+ \geq t$ be smallest integral multiples of H that is at least t, i.e. the next round at or after t such that $x^{(t)}$ is updated.

A.3. Main Technical Lemmas

We now sketch the proof of Proof of Theorem 2. Our analysis employs the framework of Local SGD [21]. The main technical lemmas are as follows:

Lemma 1 *Under Assumption 1 with L and Assumption 2 with $\mu \geq 0$, for any $t \geq 0$ with $\alpha_t \leq \frac{1}{4L}$, it holds that*

$$\begin{aligned}\mathbb{E}\left[\|\bar{x}^{(t+1)} - x^*\|^2\right] \leq (1 - \mu\alpha_t)\, \mathbb{E}\left[\|\bar{x}^{(t)} - x^*\|^2\right] + \alpha_t^2\, \mathbb{E}\left[\|g^{(t)} - \bar{g}^{(t)}\|^2\right] \\ - \frac{\alpha_t}{2}\, \mathbb{E}\left[F(\bar{x}^{(t)}) - F(x^*)\right] + \frac{2\alpha_t L}{K} \sum_{k \in [K]} \mathbb{E}\left[\|x_k^{(t)} - \bar{x}^{(t)}\|^2\right].\end{aligned}$$

Lemma 1 can be proved almost verbatim to [21, Lemma 3.1].

Lemma 2 *Under Assumption 1 with $L \geq 0$, Assumption 3 and Assumption 4. Suppose that $p \leq \frac{1}{2}$, $12L^2 \leq \eta_t^2/p$, and $\mathbb{E}_{\xi\sim\mathcal{D}}[\|\nabla f(x^*,\xi)\|^2] \leq \sigma^2$. Let $g^{(t)}$ and $\bar{g}^{(t)}$ be defined as (5) and (6) respectively. Then for any $t \geq 0$, it holds that*

$$\mathbb{E}\left[\|g^{(t)} - \bar{g}^{(t)}\|^2\right] \leq \frac{3\eta_t^2(1-p)}{p} \cdot \frac{1}{K^2}\sum_{k=1}^{K}\mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2] + \frac{24L}{K^2}\sum_{k=1}^{K}\mathbb{E}[F(\bar{x}^{(t)}) - F(x^*)] + \frac{12\sigma^2}{K}.$$

The proof of Lemma 2 can be found in Section A.5.1.

We then show the following lemma that upper-bounds $\frac{1}{K}\sum_{k\in[K]}\mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2]$ by a recursion:

Lemma 3 *Suppose $0 < p \leq \frac{1}{2}$ and the sequences $\{\alpha_t\}_{t\geq 0}, \{\eta_t\}_{t\geq 0}$ satisfy (1) $\alpha_t\eta_t = \frac{p}{2H}$ and (2) $\alpha_t \leq \frac{p}{6LH}$. Suppose that $\mathbb{E}_{\xi\sim\mathcal{D}}[\|\nabla f(x^*,\xi)\|^2] \leq \sigma^2$. Then for any $t \geq 0$, if $(t+1) \mod H = 0$ we have $\Xi_t = 0$, if $(t+1) \mod H \neq 0$ we have*

$$\begin{aligned}
&\frac{1}{K}\sum_{k\in[K]}\mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2] \\
&\leq 12LH \cdot \sum_{s=t^-}^{t-1}\alpha_s^2\frac{1}{K}\sum_{k\in[K]}\mathbb{E}[F(x^{(s)}) - F(x^*)] + 6H^2\alpha_{t^-}^2\sigma^2 + \frac{p^2}{2H}\cdot\sum_{s=t^-}^{t-1}\frac{1}{K}\sum_{k\in[K]}\mathbb{E}[\|x_k^{(s)} - x^{(t^-)}\|^2].
\end{aligned}$$

The proof of Lemma 3 can be found in Section A.5.2.

We further simplify the recurrence of $\frac{1}{K}\sum_{k\in[K]}\mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2]$ by taking weighted sum from $t = 0$ to T.

Lemma 4 *Suppose the sequences $\{\Xi_t\}_{t\geq 0}, \{e_t\}_{t\geq 0}$ satisfy (1) for all $(t+1) \mod H = 0$ $\Xi_t = 0$, and (2) for all $(t+1) \mod H \neq 0$,*

$$\Xi_t \leq \frac{p}{2H}\cdot\sum_{s=t^-}^{t-1}\Xi_s + 6H\alpha_{t^-}^2\sum_{s=t^-}^{t-1}\sigma^2 + 12LH\cdot\sum_{s=t^-}^{t-1}\alpha_s^2 e_s. \tag{7}$$

Suppose that for all $t \geq 0$, $\alpha_t = \alpha \leq \frac{p}{6LH}$ and $w_t \leq w_{t+1} \leq (1+\frac{p}{H})w_t$. Then for all $T > 0$ we have

$$\sum_{s=0}^{T} w_s\Xi_s \leq 9\alpha^2H^2\sum_{s=0}^{T}w_s\sigma^2 + \frac{p^2}{48L}\sum_{s=0}^{T}w_s e_s.$$

The proof of Lemma 4 can be found in Section A.5.3.

A.4. Proof of Theorem 2

We are now able to show Theorem 2 using the previous lemmas.

Proof Combining with Lemma 1 and Lemma 2, we have

$$\mathbb{E}\left[\|\bar{x}^{(t+1)} - x^*\|^2\right] \leq (1-\mu\alpha_t)\,\mathbb{E}\left[\|\bar{x}^{(t)} - x^*\|^2\right] + \left(\frac{24L\alpha_t^2}{K^2} - \frac{\alpha_t}{2}\right)\mathbb{E}\left[F(\bar{x}^{(t)}) - F(x^*)\right]$$

$$
\begin{aligned}
&+\left(\frac{2\alpha_t L}{K}+\frac{3\eta_t^2\alpha_t^2(1-p)}{pK^2}\right)\sum_{k\in[K]}\mathbb{E}\left[\|x_k^{(t)}-\bar{x}^{(t)}\|^2\right]+\frac{12\alpha_t^2\sigma^2}{K}\\
&\le(1-\mu\alpha_t)\,\mathbb{E}\left[\|\bar{x}^{(t)}-x^*\|^2\right]+\left(\frac{24L\alpha_t^2}{K^2}-\frac{\alpha_t}{2}\right)\mathbb{E}\left[F(\bar{x}^{(t)})-F(x^*)\right]\\
&+\left(\frac{8\alpha_t L}{K}+\frac{12\eta_t^2\alpha_t^2(1-p)}{pK^2}\right)\sum_{k\in[K]}\mathbb{E}[\|x_k^{(t)}-x^{(t)}\|^2]+\frac{12\alpha_t^2\sigma^2}{K} \qquad (8)
\end{aligned}
$$

where the second inequality comes from

$$
\frac{1}{K}\sum_{k\in[K]}\mathbb{E}[\|\bar{x}^{(t)}-x^{(t)}\|^2]\le\frac{1}{K}\sum_{k\in[K]}\mathbb{E}[2\|x_k^{(t)}-\bar{x}^{(t)}\|^2+2\|\bar{x}^{(t)}-x^{(t)}\|^2]
$$

and

$$
\mathbb{E}[\|\bar{x}^{(t)}-x^{(t)}\|^2]=\frac{1}{K^2}\mathbb{E}[\|\sum_{k\in[K]}(x_k^{(t)}-x^{(t)})\|^2]\le\frac{1}{K}\cdot\sum_{k\in[K]}\mathbb{E}[\|x_k^{(t)}-x^{(t)}\|^2].
$$

For simplicity, write $d_t=\mathbb{E}\left[\|\bar{x}^{(t)}-x^*\|^2\right]$, $e_t=\mathbb{E}\left[F(\bar{x}^{(t)})-F(x^*)\right]$ and $\Xi_t=\frac{1}{K}\sum_{k\in[K]}\mathbb{E}[\|\bar{x}^{(t)}-x^{(t)}\|^2]$. Then we can rewrite (8) as

$$
d_{t+1}\le(1-\mu\alpha_t)\,d_t+\left(\frac{24L\alpha_t^2}{K^2}-\frac{\alpha_t}{2}\right)e_t+\left(8\alpha_t L+\frac{12\eta_t^2\alpha_t^2(1-p)}{pK}\right)\Xi_t+\frac{12\alpha_t^2\sigma^2}{K}.
$$

Multiplying both sides by $\frac{1}{\alpha_t}$ and rearranging, we have

$$
\left(\frac{1}{2}-\frac{24L\alpha_t}{K^2}\right)e_t\le\frac{1-\mu\alpha_t}{\alpha_t}\,d_t-\frac{1}{\alpha_t}d_{t+1}+\left(8L+\frac{12\eta_t^2\alpha_t(1-p)}{pK}\right)\Xi_t+\frac{12\alpha_t\sigma^2}{K}.
$$

From $\alpha_t=\alpha\le\frac{p}{48HL}\le\frac{1}{4L}$ and $\eta_t=\eta=\frac{p}{2H\alpha}$, we can further simplify the above inequality as

$$
\frac{1}{4}e_t\le\frac{1-\mu\alpha}{\alpha}\,d_t-\frac{1}{\alpha}d_{t+1}+9L\Xi_t+\frac{12\alpha\sigma^2}{K}.
$$

Next, by taking the weighted average from $t=0$ to $T-1$ with weight $w_t=(1-\mu\alpha)^{-(t+1)}$ and normalizing factor $W_T=\sum_{t=0}^{t-1}w_t$, we have

$$
\begin{aligned}
\frac{1}{4W_T}\sum_{t=0}^{T-1}w_t e_t&\le\frac{1}{\alpha}\cdot\frac{1}{W_T}\sum_{t=0}^{T-1}\left((1-\mu\alpha)w_t d_t-w_t d_{t+1}\right)\\
&\quad+\frac{9L}{W_T}\sum_{t=0}^{T-1}w_t\Xi_t+\frac{12\alpha\sigma^2}{K}\cdot\frac{1}{W_T}\sum_{t=0}^{T-1}w_t\\
&\le\frac{1}{\alpha}\cdot\frac{1}{W_T}\sum_{t=0}^{T-1}\left(w_{t-1}d_t-w_t d_{t+1}\right)\\
&\quad+\frac{9L}{W_T}\sum_{t=0}^{T-1}w_t\Xi_t+\frac{12\alpha\sigma^2}{K}\cdot\frac{1}{W_T}\sum_{t=0}^{T-1}w_t \qquad \text{(By } (1-\mu\alpha)w_t=w_{t-1}\text{)}
\end{aligned}
$$

$$
\begin{aligned}
&\leq \frac{1}{\alpha} \cdot \frac{1}{W_T} \sum_{t=0}^{T-1} \left(w_{t-1} d_t - w_t d_{t+1}\right) \\
&\quad + \frac{9L}{W_T} \cdot \left(9\alpha^2 H^2 \sum_{t=0}^{T-1} w_t \sigma^2 + \frac{p^2}{2L} \sum_{t=0}^{T-1} w_t e_t\right) + \frac{12\alpha\sigma^2}{K} \qquad \text{(By Lemma 4)} \\
&\leq \frac{1}{\alpha} \cdot \frac{1}{W_T} \left(d_0 - w_T d_{t+1}\right) \\
&\quad + \frac{9L}{W_T} \cdot \left(9\alpha^2 H^2 \sum_{t=0}^{T-1} w_t \sigma^2 + \frac{p^2}{48L} \sum_{t=0}^{T-1} w_t e_t\right) + \frac{12\alpha\sigma^2}{K} \\
&\qquad\qquad \text{(Taking telescoping sum on the first term)}
\end{aligned}
$$

Rearranging and using $p \leq \frac{1}{2}$, we get that

$$
\begin{aligned}
\frac{1}{5} \cdot \frac{1}{W_T} \sum_{t=0}^{T-1} w_t e_t &\leq \frac{1}{\alpha} \cdot \frac{1}{W_T} \left(d_0 - w_T d_{t+1}\right) + \left(81LH^2\alpha^2 + \frac{12\alpha}{K}\right)\sigma^2 \\
&\leq \frac{d_0}{\alpha} \left(1 - \mu\alpha\right)^T + \left(81LH^2\alpha^2 + \frac{12\alpha}{K}\right)\sigma^2 \\
&\leq \frac{d_0}{\alpha} \exp(-\mu\alpha T) + \left(81LH^2\alpha^2 + \frac{12\alpha}{K}\right)\sigma^2
\end{aligned}
$$

Let

$$
\alpha = \min\left(\frac{p}{48LH}, \frac{\ln(\mu^2 d_0 T^2 K/\sigma^2)}{\mu T}\right)
$$

If $\frac{p}{48LH} \leq \frac{\ln(\mu^2 d_0 T^2 K/\sigma^2)}{\mu T}$, then

$$
\frac{1}{W_T} \sum_{t=0}^{T-1} w_t e_t \leq \tilde{O}\left(\frac{LHd_0}{p} \exp(-\frac{\mu p T}{LH}) + \frac{LH^2\sigma^2}{\mu^2 T^2} + \frac{\sigma^2}{\mu K T}\right)
$$

Otherwise, if $\frac{\ln(\mu^2 d_0 T^2 K/\sigma^2)}{\mu T} < \frac{p}{48LH}$, then

$$
\frac{1}{W_T} \sum_{t=0}^{T-1} w_t e_t \leq \tilde{O}\left(\frac{\sigma^2}{\mu T K} + \frac{LH^2\sigma^2}{\mu^2 T^2}\right)
$$

Therefore, assuming $p \leq \frac{1}{\kappa H}$ we have

$$
\begin{aligned}
\frac{1}{W_T} \sum_{t=0}^{T-1} w_t e_t &\leq \tilde{O}\left(\frac{LHd_0}{p} \exp(-\frac{\mu p T}{LH}) + \frac{LH^2\sigma^2}{\mu^2 T^2} + \frac{\sigma^2}{\mu K T}\right) \\
&\leq \tilde{O}\left(\frac{LHd_0}{p} \exp(-\frac{\mu p T}{LH}) + \frac{(L + \mu p^{-1} H)\sigma^2}{\mu^2 T^2} + \frac{\sigma^2}{\mu K T}\right).
\end{aligned}
$$

The proof then follows from

$$\mathbb{E}[F(\hat{x}_T)] - F(x^*) \leq \frac{1}{W_T}\sum_{t=0}^{T-1} w_t \mathbb{E}[F(x^{(t)})] - F(x^*) \leq \frac{1}{W_T}\sum_{t=0}^{T-1} w_t e_t.$$

■

A.5. Proof of Main Technical Lemmas

A.5.1. PROOF OF LEMMA 2

Proof Noting that $\{g_k^{(t)} - (x_k^{(t)} + x^{(t)})b_k^{(t)} - \nabla F(x_k^{(t)})\}_{k=1}^K$ are independent zero-mean random vectors, we have that for any $t \geq 0$,

$$\begin{aligned}\mathbb{E}\left[\|g^{(t)} - \bar{g}^{(t)}\|^2\right] &= \mathbb{E}\left[\left\|\frac{1}{K}\sum_{k=1}^K \left(g_k^{(t)} - \nabla F(x_k^{(t)}) + \eta_t x^{(t)} - \eta_t x_k^{(t)}\right)\right\|^2\right] \\ &= \frac{1}{K^2}\sum_{k=1}^K \mathbb{E}\left[\left\|g_k^{(t)} - \nabla F(x_k^{(t)}) + \eta_t x^{(t)} - \eta_t x_k^{(t)}\right\|^2\right]. \end{aligned} \tag{9}$$

where the last term comes from the independence of $\{g_t^k\}$. We then rewrite each of the summands as

$$\begin{aligned}&\mathbb{E}\left[\left\|g_k^{(t)} - \eta_t x_k^{(t)} + \eta_t x^{(t)} - \nabla F(x_k^{(t)})\right\|^2\right] \\ =&\mathbb{E}_{\mathcal{F}_t}\left[(1-p)\cdot \mathbb{E}_{\xi_k^{(t)}}\left[\left\|\frac{1}{1-p}\nabla f(x_k^{(t)}, \xi_k^{(t)}) - \nabla F(x_k^{(t)}) - \eta_t x_k^{(t)} + \eta_t x^{(t)}\right\|^2 \,\middle|\, \mathcal{F}_{t-1}\right]\right. \\ &\left. + p\cdot\left\|\left(\frac{1}{p} - 1\right)\eta_t(x_k^{(t)} - x^{(t)}) - \nabla F(x_k^{(t)})\right\|^2\right] \\ =&\mathbb{E}_{\mathcal{F}_t}\left[\frac{1}{1-p}\mathbb{E}_{\xi_k^{(t)}}\left[\|\nabla f(x_k^{(t)}, \xi_k^{(t)}) - \nabla F(x_k^{(t)})\|^2 \mid \mathcal{F}_t\right] + (1-p)\left\|\eta_t x_k^{(t)} - \eta_t x^{(t)} - \frac{p}{1-p}\nabla F(x_k^{(t)})\right\|^2\right. \\ &\left. + \frac{(1-p)^2}{p}\left\|\eta_t x_k^{(t)} - \eta_t x^{(t)} - \frac{p}{1-p}\nabla F(x_k^{(t)})\right\|^2\right] \\ =&\mathbb{E}_{\mathcal{F}_t}\left[\frac{1}{1-p}\mathbb{E}_{\xi_k^{(t)}}\left[\|\nabla f(x_k^{(t)}, \xi_k^{(t)}) - \nabla F(x_k^{(t)})\|^2 \mid \mathcal{F}_t\right] + \frac{1-p}{p}\left\|\eta_t(x_k^{(t)} - x^{(t)}) - \frac{p}{1-p}\nabla F(x_k^{(t)})\right\|^2\right]\end{aligned}$$

Notice that for all $k \in [K]$,

$$\begin{aligned}&\mathbb{E}_{\xi_k^{(t)}}\left[\|\nabla f(x_k^{(t)}, \xi_k^{(t)}) - \nabla f(x_k^{(t)})\|^2\right] \\ \leq&\mathbb{E}_{\xi_k^{(t)}}[\|\nabla f(x_k^{(t)}, \xi_k^{(t)})\|^2] \\ \leq&\mathbb{E}_{\xi_k^{(t)}}[\|\nabla f(x_k^{(t)}, \xi_k^{(t)}) - \nabla f(x^*, \xi_k^{(t)}) + \nabla f(x^*, \xi_k^{(t)})\|^2]\end{aligned}$$

$$
\begin{aligned}
&\leq \mathbb{E}_{\xi_k^{(t)}}[\|\nabla f(x_k^{(t)}, \xi_k^{(t)}) - \nabla f_{\xi_k^{(t)}}(x^{(t)}) + \nabla f_{\xi_k^{(t)}}(x^{(t)}) - \nabla f(x^*, \xi_k^{(t)}) + \nabla f(x^*, \xi_k^{(t)})\|^2] \\
&\leq 3 \cdot \mathbb{E}_{\xi_k^{(t)}}[\|\nabla f(x_k^{(t)}, \xi_k^{(t)}) - \nabla f_{\xi_k^{(t)}}(x^{(t)})\|^2 + \|\nabla f_{\xi_k^{(t)}}(x^{(t)}) - \nabla f(x^*, \xi_k^{(t)})\|^2 + \|\nabla f(x^*, \xi_k^{(t)})\|^2] \\
&\leq 3L^2 \mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2] + 6L \mathbb{E}_{\xi_k^{(t)}}[f(x^{(t)}, \xi_k^{(t)}) - f(x^*, \xi_k^{(t)})] + 3\sigma^2 \\
&= 3L^2 \mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2] + 6L \mathbb{E}[F(x^{(t)}) - F(x^*)] + 3\sigma^2 \qquad (10)
\end{aligned}
$$

and from Jensen's inequality,

$$
\|\nabla f(x_k^{(t)})\|^2 \leq \mathbb{E}_{\xi_k^{(t)}}\left[\|\nabla f(x_k^{(t)}, \xi_k^{(t)})\|^2 \mid \mathcal{F}_t\right] \leq 3L^2 \mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2] + 6L(F(x^{(t)}) - F(x^*)) + 3\sigma^2,
$$

we then have

$$
\begin{aligned}
&\mathbb{E}_{\mathcal{F}_t}\left[\frac{1}{1-p} + \frac{1-p}{p}\left\|\eta_t(x_k^{(t)} - x^{(t)}) - \frac{p}{1-p}\nabla F(x_k^{(t)})\right\|^2\right] \\
&\leq \frac{1}{1-p}\mathbb{E}_{\xi_k^{(t)}}\left[\|\nabla f(x_k^{(t)}, \xi_k^{(t)}) - \nabla F(x_k^{(t)})\|^2 \mid \mathcal{F}_t\right] + \frac{2\eta_t^2(1-p)}{p}\mathbb{E}\left[\|x_k^{(t)} - x^{(t)}\|^2\right] + \frac{2p}{1-p}\mathbb{E}\left[\|\nabla F(x_k^{(t)})\|^2\right] \\
&\leq \frac{(1+2p)}{1-p} \cdot (3L^2 \mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2] + 6L\,\mathbb{E}[F(x^{(t)}) - F(x^*)] + 3\sigma^2) + \frac{2\eta_t^2(1-p)}{p}\mathbb{E}\left[\|x_k^{(t)} - x^{(t)}\|^2\right] \\
&\leq \frac{3\eta_t^2(1-p)}{p}\,\mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2] + 24L\,\mathbb{E}[F(x^{(t)}) - F(x^*)] + 12\sigma^2.
\end{aligned}
$$

Where the last step comes from $p \leq \frac{1}{2}$ and $12L^2 \leq \frac{\eta_t^2}{p}$. Substituting the last inequality into (9) yields the desired inequality. ■

A.5.2. PROOF OF LEMMA 3

Proof Note that $x^{(t)} = \frac{1}{K}\sum_{k\in[K]} x_k^{(t^-)} = \bar{x}^{(t^-)} = x_k^{(t^-)}$. Here the last equality comes from that the client model synchronizes with the average model at step t^-. Therefore, we can write $\frac{1}{K}\sum_{k\in[K]}\mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2]$ as

$$
\frac{1}{K}\sum_{k\in[K]}\mathbb{E}[\|x_k^{(t)} - x^{(t)}\|^2] = \frac{1}{K}\sum_{k\in[K]}\mathbb{E}[\|x_k^{(t)} - x_k^{(t^-)}\|^2] \qquad (11)
$$

Using $\|\sum_{i=1}^H x_i\|^2 \leq H \cdot \sum_{i=1}^H \|x_i\|^2$ we have

$$
\begin{aligned}
&\frac{1}{K} \cdot \sum_{k\in[K]} \mathbb{E}\left[\|x_k^{(t)} - x_k^{(t^-)}\|^2\right] \\
&\leq \frac{H}{K} \cdot \sum_{k\in[K]} \sum_{s=t^-}^{t-1} \mathbb{E}\left[\|x_k^{(t+1)} - x_t^k\|^2\right] \\
&= \frac{H}{K} \cdot \sum_{k\in[K]} \sum_{s=t^-}^{t-1} \left(\frac{\alpha_s^2}{(1-p)} \cdot \mathbb{E}_{\xi_k^{(s)}}[\|\nabla f(x_k^{(s)}, \xi_k^{(s)})\|^2] + \frac{\alpha_s^2\eta_s^2}{p} \cdot \mathbb{E}[\|x_k^{(s)} - x^{(t^-)}\|^2]\right)
\end{aligned}
$$

$$\leq \frac{H}{K} \cdot \sum_{k\in[K]} \sum_{s=t^-}^{t-1} \left(2\alpha_s^2 \cdot \left(6L\mathbb{E}[F(x^{(s)}) - F(x^*)] + 3\sigma^2\right) + \left(\frac{\alpha_s^2\eta_s^2}{p} + 6\alpha_s^2 L^2\right) \cdot \mathbb{E}[\|x_k^{(s)} - x^{(t^-)}\|^2] \right)$$

$$\leq \frac{H}{K} \cdot \sum_{k\in[K]} \sum_{s=t^-}^{t-1} \left(6\alpha_s^2 \cdot \left(2L\mathbb{E}[F(x^{(s)}) - F(x^*)] + \sigma^2\right) + \frac{p^2}{2H^2} \cdot \mathbb{E}[\|x_k^{(s)} - x^{(t^-)}\|^2] \right) d$$

$$\leq 12LH \cdot \sum_{s=t^-}^{t-1} \alpha_s^2 \frac{1}{K} \sum_{k\in[K]} \mathbb{E}[F(x^{(s)}) - F(x^*)] + 6H^2\alpha_{t^-}^2\sigma^2 + \frac{p^2}{2H} \cdot \sum_{s=t^-}^{t-1} \frac{1}{K} \sum_{k\in[K]} \mathbb{E}[\|x_k^{(s)} - x^{(t^-)}\|^2]. \tag{12}$$

where the third step comes from (10) and $p \leq \frac{1}{2}$, and the fourth step comes from $\alpha_s^2\eta_s^2 = \frac{p^2}{4H^2}$ and $\alpha_s^2 \leq \frac{p^2}{24L^2H^2}$. The lemma then follows.

■

A.5.3. PROOF OF LEMMA 4

Proof Substituting $\Xi_{t^-}, \cdots, \Xi_{t-1}$ on the right-hand-side of (7) by (7), we get

$$\begin{aligned}
\Xi_t &\leq \frac{p}{2H} \cdot \sum_{s=t^-}^{t-1} \Xi_s + 6H\alpha^2 \sum_{s=t^-}^{t-1} \sigma^2 + 12LH \cdot \sum_{s=t^-}^{t-1} \alpha^2 e_s \\
&\leq \frac{p}{2H} \cdot \sum_{s=t^-}^{t-1} \left(\frac{p}{2H} \cdot \sum_{r=t^-}^{s-1} \Xi_r + 6H\alpha^2 \sum_{r=t^-}^{s-1} \sigma^2 + 12LH \cdot \sum_{r=t^-}^{s-1} \alpha^2 e_r \right) \\
&\quad + 6H\alpha^2 \sum_{s=t^-}^{t-1} \sigma^2 + 12LH \cdot \sum_{s=t^-}^{t-1} \alpha^2 e_s \\
&\leq \frac{p^2}{4H} \cdot \sum_{s=t^-}^{t-2} \Xi_s + \sum_{s=t^-}^{t-2} (3Hp\alpha^2 + 6H\alpha^2)\sigma^2 + 6H\alpha^2\sigma^2 + \sum_{s=t^-}^{t-2} (6LHp\alpha^2 + 12LH\alpha^2)e_s + 12LH\alpha^2 e_{t-1}.
\end{aligned}$$

Repeat the above step for $t - 1 - t^-$ times and using $\Xi_{t^-} = 0$ gets

$$\begin{aligned}
\Xi_t &\leq \sum_{s=t^-}^{t-1} 6H\alpha^2(1+p)(1 - \frac{p^{s-t^-}}{2^{s-t^-}})\sigma^2 + \sum_{s=t^-}^{t-1} 12LH\alpha^2(1+p)(1 - \frac{p^{s-t^-}}{2^{s-t^-}})e_s \\
&\leq 9H^2\alpha^2\sigma^2 + \sum_{s=t^-}^{t-1} 18LH\alpha^2 e_s
\end{aligned} \tag{13}$$

Taking average of Ξ_t from $t = 0$ to T with weight $w_t = (1 - c\alpha)^t$, we get

$$\begin{aligned}
\sum_{t=0}^{T} w_t\Xi_t &\leq 9H^2\alpha^2 \sum_{t=0}^{T} w_t\sigma^2 + 18LH\alpha^2 \sum_{t=0}^{T} w_t \sum_{s=t^-}^{t-1} e_s \\
&\leq 9H^2\alpha^2 \sum_{t=0}^{T} w_t\sigma^2 + 18LH\alpha^2 \sum_{t=0}^{T} e_t \sum_{s=t^-}^{t-1} w_s
\end{aligned}$$

$$\leq 9H^2\alpha^2 \sum_{t=0}^{T} w_t\sigma^2 + 18LH\alpha^2 \sum_{t=0}^{T} w_t e_t \sum_{s=t^-}^{t-1} (1+\frac{p}{H})^{s-t^-}$$
$$\leq 9H^2\alpha^2 \sum_{t=0}^{T} w_t\sigma^2 + \frac{p}{16L} \sum_{t=0}^{T} w_t e_t$$

where the third step comes from $w_{t+1} \leq (1+\frac{p}{H})w_t$, and the last step comes from $\alpha_t \leq \frac{p}{48HL}$. The lemma then follows. ■

Appendix B. Related Work

Local SGD and Variants Local SGD is a widely adopted technique for reducing communication overhead in distributed optimization. It allows workers to perform multiple local updates before synchronizing with a central server, minimizing the need for frequent gradient exchanges and improving communication efficiency. Introduced in federated learning by [17], Local SGD has been extensively studied for its convergence properties. [21] demonstrated that it converges at the same rate as mini-batch SGD, particularly in smooth and strongly convex settings. Subsequent works, such as [6] and [10], extended these results by analyzing Local SGD under more generalized frameworks, including varying network topologies, different convexity settings, and data heterogeneity. The key insight of our algorithm is that it allows workers to mix with the "central model" more frequently without incurring extra communication overhead, leading to better alignment of local models compared to Local SGD.

Several variants of Local SGD have been proposed to further enhance scalability. Elastic Averaging SGD (EASGD) [26] allows local models to diverge from the global model within a bounded range, improving convergence in non-convex settings. While our proposed algorithm similarly introduces slackness through proximal terms in the loss function, it further reduces communication by employing stochastic synchronization rather than stepwise synchronization. Moreover, our algorithm reduces variance between local models more effectively by ensuring that all workers start from the same global model every H local steps, leading to improved consistency across workers. Cooperative SGD [23] expands on Local SGD by enabling direct communication between workers, reducing reliance on a central server and improving robustness in decentralized systems. Additionally, Post-Local SGD, a combination of mini-batch SGD and Local SGD, was introduced by [14] and shown to strike a better balance between communication efficiency and generalization performance in deep learning tasks [5]. In our proposed method, we adopt a similar strategy to Post-Local SGD by using mini-batch SGD in the warmup phase, followed by the application of our PALSGD algorithm in the second phase. This approach allows us to leverage the fast initial convergence of mini-batch SGD before transitioning to our more communication-efficient method.

Negative Momentum Momentum-based optimization methods are widely employed to accelerate the convergence of gradient-based algorithms. Recent study [4], particularly in the context of adversarial learning such as Generative Adversarial Networks (GANs), have emphasized the role of negative momentum in improving game dynamics. In their study, negative momentum was introduced as a stabilizing mechanism to address oscillatory behavior in adversarial settings. Their results demonstrated that alternating gradient updates with a negative momentum term achieve more

efficient convergence, both theoretically and empirically, especially in bilinear games and challenging scenarios like saturating GANs.

Our proposed method, PALSGD, shares conceptual similarities with negative momentum in the context of distributed learning, despite the focus being on single-objective function optimization rather than adversarial learning. The pseudo-synchronization introduced in PALSGD can be interpreted as a form of regularization, akin to how negative momentum explicitly modifies the update direction in adversarial games. While previous study [4] applied negative momentum to mitigate instability in adversarial games, our method regularizes model divergence in large-scale data-parallel SGD, reducing the instability often observed in such setups. Thus, negative momentum and our pseudo-asynchronous approach provide complementary insights into enhancing the stability and efficiency of gradient-based methods, albeit in distinct settings: adversarial games versus large-scale distributed learning.

Robust Aggregation through Decoupled Method While Local SGD is theoretically fast-converging and communication-efficient, it faces empirical limitations in large-scale optimization tasks [18]. One of the challenges is that simple averaging of local models, as used in Local SGD, struggles in scenarios involving adaptive optimizers like SGD momentum and AdamW, which are common in large-scale training. Recent works such as Slomo [24] and FedOpt [19] have focused on more robust aggregation techniques by decoupling the inner optimizer for local training and the outer optimizer for model aggregation. More recent approaches such as DiLoCo [2] and Asynchronous Local SGD [15] have validated the effectiveness of using AdamW for local updates and Nesterov momentum for outer optimization in large-scale language modeling tasks, offering improved performance and robustness. Our proposed algorithm intergrates the decoupled method from the DiLoCo framework with the pseudo-synchronization process. We showed that our method significantly outperforms DiLoCo on image classification and language modeling tasks.

Asynchronous and Pseudo-Asynchronous Methods Asynchronous and pseudo-asynchronous methods have been developed to address inefficiencies in synchronous training, particularly the "straggler effect", where faster workers are forced to wait for slower ones. This issue has been widely observed in synchronous distributed settings [1, 9, 12, 13]. [1] introduced one of the earliest asynchronous frameworks, enabling each worker to update the global model independently, which significantly improved computational utilization. However, this approach introduced the challenge of stale gradients, where outdated updates from slower workers are applied to newer models, hindering convergence. Methods like Asynchronous SGD with Delay Compensation [28] addressed this issue by approximating fresher gradients. Other approaches such as Polyak Averaging [25] proposed downweighting stale updates to improve robustness. More recently, Asynchronous Local SGD [15] introduced a decoupled method, demonstrating that the strategic use of momentum can alleviate many of the challenges posed by staleness. In federated learning, methods like Moshpit SGD [20] and TimelyFL [27] have explored asynchronous approaches to better manage unreliable or heterogeneous devices in large-scale distributed systems.

Appendix C. Experimental Details

We conducted our experiments on two datasets: CIFAR-10 and TinyStories. The CIFAR-10 dataset was used for image classification tasks, while TinyStories was employed for language modeling

experiments. We implemented the distributed algorithm (DDP, LocalSGD, DiLoCo and PALSGD) on two different distributed systems, referred to as Cluster A and Cluster B, which have varying GPU configurations and interconnects, as detailed below.

C.1. Settings

Hardware Configurations: We utilized two types of clusters for our experiments:

Cluster A

- GPU: `NVIDIA Tesla T4 (16GB)` x 4
- GPU Bandwidth: 320.0 GB/s
- GPU Interconnect: PCIe with NUMA Node Interconnect (No NVLink)

Cluster B

- GPU: `NVIDIA Tesla V100 DGXS (32GB)` x 8
- GPU Bandwidth: 897.0 GB/s
- GPU Interconnect: NVLink, 150 GB/s per GPU

Software and Library Configurations: Both clusters used the following software environment:

- Python: 3.11.6
- PyTorch: 2.3.1+cu121
- CUDA: 12.1
- CUDNN: 8902

Workloads:

- **Small CNN on CIFAR-10**: The CIFAR-10 dataset [3] is widely used in machine learning research, especially for image recognition tasks. It consists of 60,000 color images, each measuring 32x32 pixels, evenly distributed across ten distinct classes. These classes include common objects such as airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks. Each class is represented by 6,000 images, and the dataset is divided into a training set of 50,000 images and a test set of 10,000 images.
  We used a small CNN for training on CIFAR-10. Our experiment on CIFAR-10 is preliminary, aimed at selecting algorithms for comparison and conducting ablation studies rather than measuring training speed. Therefore, to simulate multiple workers on a single GPU and examine the effect on loss and accuracy when multiple models are allocated to the GPU, we employed a small CNN with fewer parameters.
  The small CNN is defined as follows. The model consists of two convolutional layers, each followed by a ReLU activation and max-pooling operation. The first convolutional layer takes the input (which is a 3-channel CIFAR-10 image) and applies 32 filters of size 5x5 with padding to preserve spatial dimensions. The second convolutional layer increases the number of filters to 64, using the same filter size and padding. After each convolution, the image is downsampled by a 2x2 max-pooling operation. Following the convolutional layers, the feature map is flattened and passed through a fully connected layer with 1,024 units, which is activated by

3. https://www.cs.toronto.edu/~kriz/cifar.html

ReLU. The final layer maps the output to 10 classes, corresponding to the categories in the CIFAR-10 dataset.

- **GPT-NEO on TinyStories**: The TinyStories dataset [3] is designed for small-scale text generation tasks, providing a benchmark for language modeling performance on short narrative texts. We utilized the GPT-NEO model [4] with 8 million parameters to evaluate the PALSGD algorithm under distributed training conditions.
  The GPT-NEO model used in this experiment is configured with the following architecture. It includes a maximum positional embedding size of 300, which allows the model to handle input sequences of up to 300 tokens. The hidden size is set to 128, determining the dimensionality of the model's internal representations. The model has 8 attention heads, enabling it to capture diverse relationships across tokens in a sequence through its self-attention mechanism. Additionally, there are 8 hidden layers, each contributing to the depth of the model, allowing it to learn more complex hierarchical patterns in the data.

Training Configuration:

All experimental results, unless otherwise noted, refer to the configuration of the hyper parameter with the best results for that metric.

- **Image Classification on Simulation Environment**: We conducted experiments on the CIFAR-10 dataset to evaluate the accuracy performance of the distributed algorithm. The model used for this experiment was a small CNN architecture, trained for 200 epochs with 1 to 64 worker. The inner optimizer for this experiment was AdamW [16], with an initial learning rate of $\{0.0001, 0.0005, 0.001\}$. The outer learning rate was also set to 0.01, with the outer optimizer is Nesterov Momentum SGD. We applied Post-Local SGD after 250 iterations. The model and optimizer states were synchronized at initialization, but the optimizer state was not synchronized during training phase of Local SGD, DiLoCo and PALSGD. We used a batch size per worker is 64, probability of 0.25 for psuedo synchronization updates, and η_t is 1 in Algorithm 1.
  We conducted ablation studies on a small CNN architecture, as shown in Fig. 1, focusing on the effects of synchronization interval (H: 16 to 256) and the number of workers (K: 2 to 64). The experiments compared the performance of PALSGD against baseline methods such as DDP, Local SGD, and DiLoCo.
- **Languale Modeling on Distributed Environment**: We trained the model for 15 epochs with a local batch size of 512 per GPU, using 4 GPUs in A cluster. The global batch size was set to 2048. The inner optimizer's learning rate was fixed at 0.001, and we employed the AdamW [16] for inner optimizer with gradient clipping enabled. Regarding outer optimization for DiLoCo and PALSGD, we use Nesterov Momentum SGD with outer learning rate was fixed at 0.1 to 0.2. The synchronization interval (H) was set to 16. The variants of Local SGD algorithm started after 1024 iterations. For experiments of PALSGD, synchronization interval set as 16 to 64, probabilistic synchronization parameter p=0.1, and η_t is 1 to 16.

4. https://huggingface.co/docs/transformers/en/model_doc/gpt_neo

C.2. Additional Results

In the 8-GPU experiment, we used Cluster B. While it might seem that the communication bottleneck would be larger due to the doubled number of GPUs compared to the 4-GPU cluster, this is not necessarily the case. Since the GPUs in Cluster B does not share the same bandwidth and FLOPS as those in Cluster A, furhtermore, Cluster B differs in that the GPUs are connected via NVLink, which provides faster communication. This makes it an ideal computing environment for distributed deep learning. As a result, we achieved only a 20% improvement in training speed. What we want to emphasize here is that even in an optimal communication environment like NVLink, which does not span across nodes, there is still room for a 20% increase in training speed.

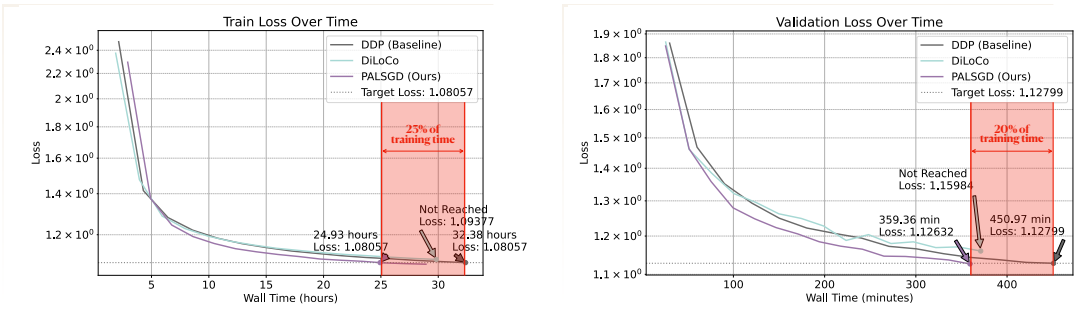

Figure 3: GPT-NEO Experiments (K=8 / H=64) on DGX-1 (8 V100 GPUs Connected by NVLINK): Training time comparison across distributed algorithm to achieve target loss. While PALSGD achieve fastest and lowest loss, DDP is slowest and DiLoCo did not achieve target loss.