DSGD-CECA: Decentralized SGD with Communication-Optimal Exact Consensus Algorithm

Lisang Ding [1] **Kexin Jin** [2] **Bicheng Ying** [3] **Kun Yuan** [4 5 6] **Wotao Yin** [7]

Abstract

Decentralized Stochastic Gradient Descent (SGD) is an emerging neural network training approach that enables multiple agents to train a model collaboratively and simultaneously. Rather than using a central parameter server to collect gradients from all the agents, each agent keeps a copy of the model parameters and communicates with a small number of other agents to exchange model updates. Their communication, governed by the communication topology and gossip weight matrices, facilitates the exchange of model updates. The state-of-the-art approach uses the dynamic one-peer exponential-2 topology, achieving faster training times and improved scalability than the ring, grid, torus, and hypercube topologies. However, this approach requires a power-of-2 number of agents, which is impractical at scale. In this paper, we remove this restriction and propose Decentralized SGD with Communication-optimal Exact Consensus Algorithm (DSGD-CECA), which works for any number of agents while still achieving state-of-the-art properties. In particular, DSGD-CECA incurs a unit per-iteration communication overhead and an $\tilde{O}(n^3)$ transient iteration complexity. Our proof is based on newly discovered properties of gossip weight matrices and a novel approach to combine them with DSGD's convergence analysis. Numerical experiments show the efficiency of DSGD-CECA.

[1]Department of Mathematics, University of California, Los Angeles, CA, USA [2]Department of Mathematics, Princeton University, Princeton, NJ, USA [3]Google Inc., Los Angeles, CA, USA [4]Center for Machine Learning Research, Peking University, Beijing, P. R. China. [5]AI for Science Institute, Beijing, P. R. China [6]National Engineering Labratory for Big Data Analytics and Applications, Beijing, P. R. China [7]Decision Intelligence Lab, Alibaba US, Bellevue, WA, USA. Correspondence to: Lisang Ding <lsding@math.ucla.edu>, Wotao Yin <wotao.yin@alibaba-inc.com>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

Decentralized computing (Tsitsiklis et al., 1986; Lopes & Sayed, 2008; Nedic & Ozdaglar, 2009; Dimakis et al., 2010) is an essential subclass of distributed computing with no data fusion center. In scenarios where data and computational resources are distributed, decentralized computing enables each agent to process its local data and communicate with a selected group of other agents. This approach helps avoid the formation of central-agent-induced communication bottlenecks. Without a central agent, however, the decentralized algorithm must achieve a global result through peer-to-peer interactions of the agents. Hence, the algorithm performance heavily depends on how effectively and efficiently the agents exchange their information.

In scenarios where the global goal is to compute an average across all agents, this challenge is identified as *average consensus* or *allreduce averaging*. Various optimal methods are established for a range of prevalent communication settings to address this problem. The goal of this paper, however, is to accelerate *decentralized SGD (DSGD)* (Chen & Sayed, 2012; Lian et al., 2017; Koloskova et al., 2020), which is widely used in large-scale deep neural network training, by applying average consensus methods judiciously.

When a distributed SGD algorithm relies on a parameter server, the distributed agents have the same model parameters. However, when the scale of training requires us to use a large number of distributed agents, the parameter server becomes the bottleneck. Without a parameter server, the agents in decentralized SGD algorithms maintain the similarity among their copies of model parameters through message passing. The cost to make them the same among n agents is at least $\lceil \log_2(n) \rceil$ rounds of message passing with each agent sending and receiving one message at each round, but this cost is unnecessary.

Performing only one round of message passing after each mini-batch SGD step saves time though it causes the convergence of SGD to take more steps. It is shown in (Lian et al., 2017; Pu et al., 2019; Koloskova et al., 2020; Ying et al., 2021a) that, for distributed smooth nonconvex objectives, a decentralized approach with one communication round per SGD step is slower than centralized SGD *only* during

Table 1. Comparison between DSGD over different commonly-used topologies. "Static Exp.": static exponential graph; "O.-P. Exp.": one-peer exponential graph; "DSGD-CECA-1P": DSGD-CECA that supports 1-port communication model; "DSGD-CECA-2P": DSGD-CECA that supports 2-port communication model. Undirected graphs can admit symmetric gossip matrices. If some graph has a dynamic pattern, its associated communication matrix will vary at each iteration. Notation $\tilde{O}(\cdot)$ ignores all polylogarithmic factors.

Topology	Connection	Pattern	Per-iter Comm.	Trans. Iters.	size n
Ring (Nedić et al., 2018)	undirected	static	$\Theta(1)$	$O(n^7)$	arbitrary
Grid (Nedić et al., 2018)	undirected	static	$\Theta(1)$	$\tilde{O}(n^5)$	arbitrary
Torus (Nedić et al., 2018)	undirected	static	$\Theta(1)$	$O(n^5)$	arbitrary
Hypercube (Trevisan, 2017)	undirected	static	$\Theta(\ln(n))$	$\tilde{O}(n^3)$	power of 2
Static Exp.(Ying et al., 2021a)	directed	static	$\Theta(\ln(n))$	$\tilde{O}(n^3)$	arbitrary
O.-P. Exp.(Ying et al., 2021a)	directed	dynamic (2-port)	1	$\tilde{O}(n^3)$	power of 2
DSGD-CECA-1P (Ours)	undirected	dynamic (1-port)	1	$\tilde{O}(n^3)$	even
DSGD-CECA-2P (Ours)	directed	dynamic (2-port)	1	$\tilde{O}(n^3)$	arbitrary

an initial period of iterations, called the transient period. Afterward, SGDs with or without decentralization tend to show similar performance. Given the expense and time-intensive nature of large-scale training, a practical DSGD should aim to minimize its transient period to enhance competitiveness. Recent SGD methods based on various communication topologies, each leading to different transient iterations, are proposed. We provide a summary in Table 1.

Among different decentralized SGD algorithms, dynamic exponential-2 (also known as one-peer exponential-2) message passing (Assran et al., 2019; Ying et al., 2021a) is currently state-of-the-art. For n that is a power of 2, every agent sends messages to *one single* designated neighbor at each SGD iteration according to a subgraph taken from a cyclic sequence of $\log_2(n)$ base subgraphs. This dynamic exponential-2 message passing can reach *exact* global averaging in $\log_2(n)$ rounds of communication. Furthermore, decentralized SGD based on dynamic exponential-2 graph can obtain the state-of-the-art balance between per-iteration communication and transient iteration complexity (Ying et al., 2021a); it only incurs a unit communication overhead per iteration and $O(n^3 \log_2^4(n))$ transient iterations, both of which are nearly the best among DSGDs implemented with other commonly-used topologies.

Unfortunately, some excellent results of dynamic exponential-2 message passing *no longer* hold when n is not a power of 2, e.g., $n = 10$ or 100, including finite-time convergence for average consensus and DSGD convergence guarantees. As n increases, the number of instances where n is a power of 2 becomes increasingly scarce. For example, if one has a sizeable deep-learning training task that fits well into 20 GPU nodes at hand, the current choice is to either utilize a less-efficient DSGD algorithm or scale up to 32 nodes to run the most efficient DSGD algorithm.

Furthermore, dynamic exponential-2 message passing operates exclusively within the 2-port communication model over *directed* topologies. In this model, each agent sends information to an agent and receives information from another agent simultaneously in each round. Contrarily, it is not applicable to 1-port model over *undirected* topologies where each agent sends and receives information to/from the *same* agent during each communication round. 1-port model is typically more efficient in full-duplex communication systems, and it admits symmetric gossip communication matrices, which are required by many important decentralized optimization algorithms such as decentralized ADMM (Shi et al., 2014), EXTRA (Shi et al., 2015), and Exact-Diffusion/D^2 (Yuan et al., 2019; Li et al., 2019; Tang et al., 2018).

Since dynamic exponential-2 suffers from the above limitations, we ask the following question. *Can we develop new DSGD algorithms that work for any n (or at least any even n), support both 1-port and 2-port communication models, and inherit the nice properties of dynamic exponential-2?* This paper provides affirmative answers.

1.1. Contributions

This paper introduces a novel DSGD algorithm that works for any n (or any even n under the 1-port communication model) and achieves state-of-the-art balance between per-iteration communication and transient iteration complexity. Our main contributions are listed as follows.

- We revisit a less well-known but communication-optimal exact consensus algorithm (CECA) proposed in (Bar-Noy et al., 1993). CECA requires $\lceil \log_2(n) \rceil$ rounds of message passing (which is optimal and cannot be further reduced) to achieve global averaging. The original CECA is restricted to 2-port communication. We improve this algorithm to 1-port communication for any even n and show that it achieves exact average consensus in $\lceil \log_2(n) \rceil$ rounds of message passing.

- We next judiciously apply CECA into decentralized learning and propose DSGD-CECA. To save communications, our algorithm only conducts one single round of CECA message passing after each mini-batch SGD step. To guarantee convergence, our algorithm introduces a new strategy that maintains copies of local models, thereby inheriting the periodic global averaging property from CECA. Besides, DSGD-CECA works for any n under the 2-port communication model and any even n under the 1-port model. Importantly, our DSGD-CECA supports both directed graphs and undirected graphs.

- We further establish that DSGD-CECA incurs a $\Theta(1)$ per-iteration communication overhead and $\tilde{O}(n^3)$ transient iteration complexity; both of which are optimal compared to the baselines; see Table 1. The convergence analysis of DSGD-CECA is non-trivial because the gossip weight matrix of CECA is not doubly-stochastic. However, our analysis leverages newly discovered properties of this matrix, which helps resolve analysis challenges significantly.

Notes. This paper considers deep neural network training within high-performance data-center clusters, in which the network topology can be fully controlled and any two GPUs can communication (through network switches) when necessary. The proposed algorithms may not work well in scenarios (e.g., wireless sensor networks, internet of vehicles, etc.) where connection constraints exist. In addition, this paper studies deterministic message passing listed in Table 1. There are recent works that study DSGD with stochastic message passing (reviewed below). However, stochastic message passing is less easy to control and implement. Moreover, it can cause DSGD to be arbitrarily slow with non-zero probability.

2. Preliminary and Related Work

2.1. Preliminary

Problem. Consider n computing agents working collaboratively to solve the distributed optimization problem.

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}), \tag{1}$$

where $f_i(\boldsymbol{x}) = \mathbb{E}_{\xi_i\sim\mathcal{D}_i} F(\boldsymbol{x};\xi_i)$. In the above problem, ξ_i denotes random local data kept at agent i, and it is sampled from distribution $\mathcal{D}_i$. It is common that $\mathcal{D}_i \neq \mathcal{D}_j$ when $i \neq j$, which causes the data heterogeneity issue in distributed learning problems.

Network topology and communication matrix. Decentralized optimization depends on partial averaging among connected agents, the relationships of which are dictated by the network topology—either directed or undirected—that links all the agents. We let $\mathbf{P} \in \mathbb{R}^{n\times n}$ denote a communication matrix that characterizes the sparsity and connectivity of the network topology. To this end, we let $\mathbf{P}_{i,j} = 1$ if agent j can send information to agent i otherwise $\mathbf{P}_{i,j} = 0$.

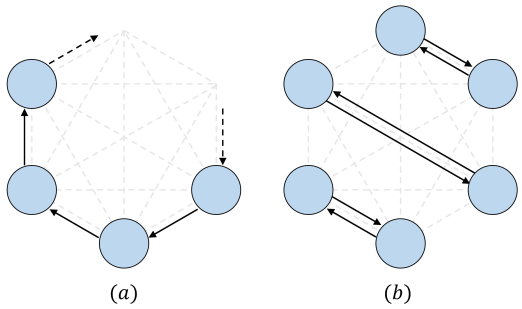

Figure 1. (a) The 2-port communication model. Each agent sends information to one neighbor and receives information from another different neighbor per communication round. (b) The 1-port communication model. Each agent is paired with one single neighbor and exchanges information with it.

Communication models. This paper will develop decentralized SGD algorithms based on the following communication models.

- **1-port model**. This model applies to undirected network topologies. In this model, during each communication, each agent communicates bidirectionally, both sending and receiving information to and from the same agent, as shown in Fig. 1(b). 1-port model admits symmetric communication matrix which are required in many popular decentralized optimization methods, and it is typically more efficient than 2-port model in full-duplex communication systems. OU-EquiDyn (Song et al., 2022) adheres to the 1-port communication model.

- **2-port model**. This model operates over directed network topologies. Each agent in this model sends information to an agent and receives information from another agent simultaneously in each round, as illustrated in Fig. 1(a). Both dynamic exponential-2 (Ying et al., 2021a) and OD-EquiDyn (Song et al., 2022) follow the 2-port model.

It is worth noting that both 1-port and 2-port models are efficient in communication. They only incur $\Theta(1)$ communication overhead per iteration since each agent in these models only talks with one single neighbor.

Decenralized SGD. Decentralized SGD (DSGD), an emerging training technique for large-scale deep learning, relaxes the global averaging step in traditional parallel SGD to inexact partial averaging within neighborhood. It is characterized by its substantially less (and thus faster) communication every iteration. The less neighbors each agent needs to talk with (i.e., the sparser the network topology is), the faster the per-iteration communication is. However,

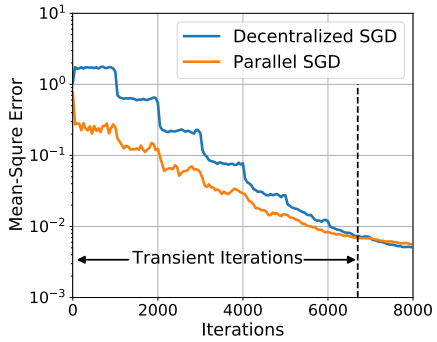

Figure 2. Illustration of the transient iterations (Ying et al., 2021a).

the communication efficiency in DSGD comes at a cost – slower convergence since partial averaging is less effective to aggregate information. It is found in (Lian et al., 2017; Pu et al., 2019; Koloskova et al., 2020) that DSGD can achieve the same convergence rate as parallel SGD after some *transient iterations*, see Fig. 2 for an illustration. The longer the transient period is, the slower the algorithm converges. This paper targets to develop new algorithms that attain minimal transient iteration complexity with little communication overhead per iteration.

Assumptions. We introduce several standard assumptions for problem (1).

Assumption 2.1 (LIPSCHITZ SMOOTHNESS)**.** Each local function f_i is $L-$smooth, i.e., $\|\nabla f_i(\boldsymbol{x}) - \nabla f_i(\boldsymbol{y})\| \leq L\|\boldsymbol{x}-\boldsymbol{y}\|$ for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$.

Assumption 2.2 (GRADIENT NOISE)**.** Random data variable $\xi_i^{(k)}$ is independent of each other for any k and i. The gradient noise satisfies $\mathbb{E}_{\xi_i \sim \mathcal{D}_i}[\nabla F(\boldsymbol{x};\xi_i)] = \nabla f_i(\boldsymbol{x})$, $\mathbb{E}_{\xi_i \sim \mathcal{D}_i}\|\nabla F(\boldsymbol{x};\xi_i) - \nabla f_i(\boldsymbol{x})\|^2 \leq \sigma^2$, for any $\boldsymbol{x} \in \mathbb{R}^d$.

Assumption 2.3 (DATA HETEROGENEITY)**.** The local functions satisfies $\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\boldsymbol{x}) - \nabla f(\boldsymbol{x})\|^2 \leq b^2$ for any $\boldsymbol{x} \in \mathbb{R}^d$ and i.

Notations. Throughout the paper, we let $[n] = \{1, \cdots, n\}$ and define a mod operation that returns a value in $[n]$ as

$$i \bmod n = \begin{cases} \ell & \text{if } i = kn + \ell \text{ for some } \ell \in [n-1], \\ n & \text{if } i = kn. \end{cases}$$

where k is an integer. When i is the agent index, we will simplify $(i-\ell) \bmod n$ as $i - \ell$ for any $\ell \in [n]$. For example, suppose $n = 6$ and $i = 1$, it holds that $i - 1 = 6$ and $i - 2 = 5$.

2.2. Related work

Decentralized deep training. Decentralized SGD algorithms (Lopes & Sayed, 2008; Yuan et al., 2016; Lian et al., 2017; Koloskova et al., 2019) are widely used to accelerate large-scale deep training. These algorithms have been extended to various practical settings, including those with directed (Assran et al., 2019) and time-varying (Kong et al., 2021; Ying et al., 2021a; Koloskova et al., 2020) network topologies, asynchronous model updating (Lian et al., 2018; Niwa et al., 2021), and momentum acceleration (Lin et al., 2021; Yuan et al., 2021). However, DSGD suffers from data heterogeneity issues (Koloskova et al., 2020; Yuan et al., 2020) in the meanwhile. Various advanced techniques such as EXTRA (Shi et al., 2015), Exact-Diffusion/D^2 (Yuan et al., 2019; Li et al., 2019; Yuan & Alghunaim, 2021; Tang et al., 2018), and gradient-tracking (Di Lorenzo & Scutari, 2016; Xu et al., 2015; Nedic et al., 2017; Qu & Li, 2018; Xin et al., 2020; Alghunaim & Yuan, 2021) are proposed to mitigate the impact of data heterogeneity and thereby accelerating the DSGD convergence.

Message passing with asymptotic consensus. Decentralized learning methods are typically based on gossip averaging. While gossip averaging allows for quick per-iteration communication when running over topologies such as rings, grids, and torus, the rate of convergence towards the average consensus slows as n increases (Nedić et al., 2018). The hypercube graph (Trevisan, 2017) maintains a nice balance between communication efficiency and consensus rate. But a hypercube cannot be formed when network size n is not a power of 2. The static exponential graph (Ying et al., 2021a), on the other hand, works for any n, but only admits directed weight matrices. Stochastic message passing is also widely used in decentralized learning. The Erdos-Renyi graph (Nachmias & Peres, 2008; Benjamini et al., 2014; Nedić et al., 2018) and the geometric random graph (Beveridge & Youngblood, 2016; Boyd et al., 2005) are two representatives. A recent work (Song et al., 2022) proposes a state-of-the-art family of EquiTopo graphs that incur $\Theta(1)$ communication overhead per iteration and enjoy a network-size independent consensus rate. However, these stochastic message passing protocols can be difficult to control. Moreover, some realizations of these random protocols can be arbitrarily slow to achieve asymptotic consensus with non-zero probabilities.

Message passing with exact consensus. The concept of exact consensus (also known as allreduce averaging) is extensively studied within the high-performance computing community. This approach can achieve exact global averaging with a finite number of communication rounds. Well-known methods include tree-allreduce (Ben-Nun & Hoefler, 2019), ring-allreduce (Patarasuk & Yuan, 2009) and BytePS (Jiang et al., 2020). Recent works start integrating exact consensus techniques to decentralized optimization to boost performance. For example, (Ying et al., 2021a) utilizes dynamic exponential-2 to balance communication efficiency and aggregation effectiveness in DSGD. Generally speaking, it is non-trivial to develop new decentralized algorithms with exact consensus techniques, mainly because they do not contribute doubly stochastic weight matrices.

3. Communication-Optimal Exact Consensus

3.1. 2-port optimal exact consensus

This section revisits the optimal message passing algorithm CECA (Bar-Noy et al., 1993) for a 2-port communication system with n agents, where n can be any positive integer.

Problem statement. Letting each agent i hold a local variable u_i, our target is to let each agent obtain $\bar{u} = \frac{1}{n}\sum_{i=1}^{n} u_i$ after $\tau = \lceil \log_2 n \rceil$ rounds of communication.

Auxiliary variables. We construct several auxiliary variables utilized in CECA.

- We convert $n-1$ to a binary number as

$$n - 1 = (\delta_0\, \delta_1\, \cdots\, \delta_{\tau-1})_2 \tag{2}$$

  where $\delta_0 \neq 0$ is the most significant bit and $\delta_{\tau-1}$ is the least significant bit, e.g., $n-1 = (101)_2$ when $n = 6$.

- We set $n_0 = 0$ and calculate $\{n_{r+1}\}_{r=0}^{\tau-1}$ by

$$n_{r+1} = 2n_r + \delta_r, \quad r = 0, 1, 2, \ldots, \tau - 1. \tag{3}$$

  It is easy to verify that $n_\tau = n-1$. For example, if $n = 6$, then $\tau = \lceil \log_2 6 \rceil = 3$ and $n_1 = 1, n_2 = 2, n_3 = 5$.

- Let each agent i maintain variables $I_i^{(r)}$ and $J_i^{(r)}$ at iteration r, and initialize them as $I_i^{(0)} = u_i$ and $J_i^{(0)} = 0$.

Main idea. For any $r = 0, 1, \cdots, \tau-1$, CECA will always guarantee that

$$I_i^{(r)} = \frac{1}{n_r+1}\sum_{j=0}^{n_r} u_{i-j}, \quad J_i^{(r)} = \begin{cases} \dfrac{1}{n_r}\displaystyle\sum_{j=1}^{n_r} u_{i-j},\ r \geq 1 \\ 0,\ r = 0 \end{cases}. \tag{4}$$

It is observed that $I_i^{(r)}$ always keeps the average of agent i and its n_r previous neighbors, while $J_i^{(r)}$ keeps the average of agent i's n_r previous neighbors (but not including u_i). When $r = \tau - 1$, it holds that $I_i^{(r)} = \frac{1}{n}\sum_{i=1}^{n} u_i$ and hence each agent will reach the average consensus.

Main recursions. To guarantee (4), CECA will conduct the following recursions for each $r = 0, 1, \cdots, \tau-1$.

$$\text{If } \delta_r = 1 \text{ update } \begin{cases} I_i^{(r+1)} = \frac{1}{2}I_i^{(r)} + \frac{1}{2}I_{i-n_r-1}^{(r)} \\ J_i^{(r+1)} = \frac{n_r}{2n_r+1}J_i^{(r)} + \frac{n_r+1}{2n_r+1}I_{i-n_r-1}^{(r)} \end{cases}$$

$$\text{If } \delta_r = 0 \text{ update } \begin{cases} I_i^{(r+1)} = \frac{n_r+1}{2n_r+1}I_i^{(r)} + \frac{n_r}{2n_r+1}J_{i-n_r}^{(r)} \\ J_i^{(r+1)} = \frac{1}{2}J_i^{(r)} + \frac{1}{2}J_{i-n_r}^{(r)} \end{cases}$$

More details on CECA as well as illustrating examples can be referred to Appendix B.1.1.

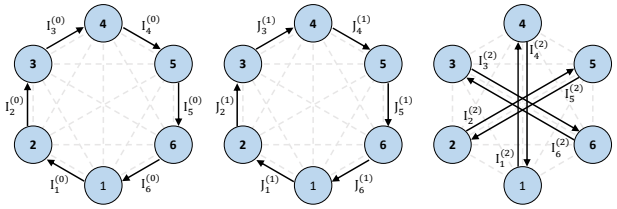

Figure 3. An example of CECA for $n = 6$ agents. The method conducts 3 communication rounds to reach average consensus. The arrows and their labels indicate the information flows.

Communication patterns. From the main CECA recursion listed above, it is observed that each agent will follow a 2-port communication model. To better capture the communication pattern, we let $\mathbf{Q}^{(r)}$ denote the communication matrix employed at CECA round r. If agent j sends information to agent i, we set $\mathbf{Q}_{i,j}^{(r)} = 1$; otherwise, $\mathbf{Q}_{i,j}^{(r)} = 0$.

$$\text{If } \delta_r = 1 \text{ then } \mathbf{Q}_{i,j}^{(r)} = \begin{cases} 1, & \text{if } i - j \equiv n_r + 1 \pmod n \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

$$\text{If } \delta_r = 0 \text{ then } \mathbf{Q}_{i,j}^{(r)} = \begin{cases} 1, & \text{if } i - j \equiv n_r \pmod n \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

The matrix $\mathbf{Q}^{(r)}$ is a permutation matrix that reflects the dynamic topology for message exchanging, as illustrated in Fig. 3 for the case when $n = 6$. Additionally, the matrix $\mathbf{Q}^{(r)}$ will facilitate the DSGD-CECA development.

3.2. 1-port optimal exact consensus

The vanilla CECA introduced in (Bar-Noy et al., 1993), referred to here as CECA-2P, exclusively supports the 2-port communication model. In this section, we will develop a new variant, CECA-1P, that supports 1-port communication model over undirected topology. CECA-1P enables each agent to reach average consensus after $\tau = \lceil \log_2 n \rceil$ rounds of communication when n is even.

Main recursions. To achieve average consensus, CECA-1P introduces the same auxiliary variables as CECA-2P. The main idea of CECA-1P is similar to CECA-2P. In round r, agent i pairs with agent $i + 2n_r + 1$ if i is odd, otherwise, it pairs with agent $i - 2n_r - 1$. Following this, each pair of agents exchanges information with each other. If $\delta_r = 1$, they exchange $I^{(r)}$; otherwise they exchange $J^{(r)}$ instead. We let $\omega_i^{(r)}$ denote the agent sending a message to agent i in the rth round. CECA-1P conducts the following recursions for each $r = 0, \cdots, \tau-1$.

$$\text{If } \delta_r = 1 \text{ update } \begin{cases} I_i^{(r+1)} = \frac{1}{2}I_i^{(r)} + \frac{1}{2}I_{\omega_i^{(r)}}^{(r)} \\ J_i^{(r+1)} = \frac{n_r}{2n_r+1}J_i^{(r)} + \frac{n_r+1}{2n_r+1}I_{\omega_i^{(r)}}^{(r)} \end{cases}$$

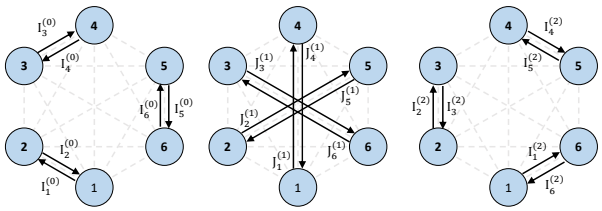

Figure 4. An example of CECA-1P for $n = 6$ agents. It is observed that the resulting topology is undirected per iteration.

$$\text{If } \delta_r = 0 \text{ update } \begin{cases} I_i^{(r+1)} = \frac{n_r+1}{2n_r+1} I_i^{(r)} + \frac{n_r}{2n_r+1} J_{\omega_i^{(r)}}^{(r)} \\ J_i^{(r+1)} = \frac{1}{2} J_i^{(r)} + \frac{1}{2} J_{\omega_i^{(r)}}^{(r)} \end{cases}$$

More details on CECA as well as illustrating examples can be referred to Appendix B.1.2.

Communication patterns. The communication matrix $\mathbf{Q}^{(r)}$ in CECA-1P is given by

$$\mathbf{Q}_{i,j}^{(r)} = \begin{cases} 1, & i \text{ odd}, \ j = i + 2n_r + 1 \\ 1, & i \text{ even}, \ j = i - 2n_r - 1 \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

Note that $\mathbf{Q}^{(r)}$ is a symmetric matrix. Fig. 4 illustrates the communication pattern for CECA-1P when $n = 6$.

4. DSGD-CECA Algorithm

This section develops a novel DSGD algorithm based on CECA, as discussed in §3.1. The resulting CECA-DSGD works with any n in the 2-port communication model and any even n in the 1-port model. In either scenario, DSGD-CECA incurs $\Theta(1)$ per-iteration communication overhead and $\tilde{O}(n^3)$ transient iteration complexity; both of which are nearly the best compared to baselines, see Table 1.

Challenges. It is highly non-trivial to integrate CECA to DSGD due to the following challenges. First, DSGD-CECA splits CECA to a sequence of separate communication rounds, and it only performs one single CECA message passing after each mini-batch SGD steps. It is unknown whether this strategy will deteriorate the optimal communication efficiency of CECA. Second, CECA requires to maintain two sets of variables, i.e., $I^{(r)}$ and $J^{(r)}$, to enforce average consensus. It is unknown what auxiliary variables shall be introduced to DSGD to facilitate the integration with CECA. Third, the introduction of CECA to DSGD will crash the doubly-stochastic property of the gossip weight matrix in DSGD. For this reason, traditional theories in (Koloskova et al., 2020; Alghunaim & Yuan, 2021; Ying et al., 2021a) cannot be utilized to analyze DSGD-CECA. This section will resolve all these challenges.

Algorithm 1 DSGD-CECA

Initialize: Randomly initialize $\boldsymbol{x}_i^{(0)}$; set $\boldsymbol{y}_i^{(0)} = \boldsymbol{x}_i^{(0)}$; set learning rate γ; compute $\tau = \lceil \log_2 n \rceil$; convert $n - 1 = (\delta_0 \cdots \delta_{\tau-1})_2$;
for $k = 0, 1, 2, \ldots, T$ **do**
  $r = k \pmod{\tau}$;
  if $\delta_r = 1$ **then**
    Compute $\mathbf{g}^{(k)} = \nabla F(\mathbf{x}^{(k)}; \boldsymbol{\xi}^{(k)})$;
    Let $\mathbf{z}^{(k)} = \mathbf{x}^{(k)}$ and $\mathbf{e}^{(k)} = \mathbf{g}^{(k)}$;
    Sample communication matrix $\mathbf{P}^{(k)}$ as $\mathbf{Q}^{(r)}$ in (5);
    Let $a^{(k)} = \frac{1}{2}$ and $b^{(k)} = \frac{n_r}{2n_r+1}$;
  else if $\delta_r = 0$ **then**
    Compute $\mathbf{h}^{(k)} = \nabla F(\mathbf{y}^{(k)}; \boldsymbol{\xi}^{(k)})$;
    Let $\mathbf{z}^{(k)} = \mathbf{y}^{(k)}$ and $\mathbf{e}^{(k)} = \mathbf{h}^{(k)}$;
    Sample communication matrix $\mathbf{P}^{(k)}$ as $\mathbf{Q}^{(r)}$ in (6);
    Let $a^{(k)} = \frac{n_r+1}{2n_r+1}$ and $b^{(k)} = \frac{1}{2}$;
  end if
  Communicate $\mathbf{z}^{(k)}, \mathbf{e}^{(k)}$ among agents following $\mathbf{P}^{(k)}$;
  Update $\mathbf{x}^{(k+1)}$ as in (8);
  Update $\mathbf{y}^{(k+1)}$ as in (9);
end for

4.1. Algorithm development

Algorithm description. We first introduce DSGD-CECA that supports 2-port communication system. To this end, we let each agent i maintain a local model $\boldsymbol{x}_i$ and an additional auxiliary model $\boldsymbol{y}_i$ to facilitate the integration with CECA. To better present the algorithm, we introduce the following notations

$$\begin{aligned} \mathbf{x}^{(k)} &= [(\boldsymbol{x}_1^{(k)})^\top; (\boldsymbol{x}_2^{(k)})^\top; \cdots; (\boldsymbol{x}_n^{(k)})^\top], \\ \mathbf{y}^{(k)} &= [(\boldsymbol{y}_1^{(k)})^\top; (\boldsymbol{y}_2^{(k)})^\top; \cdots; (\boldsymbol{y}_n^{(k)})^\top], \\ \nabla F(\mathbf{x}^{(k)}; \boldsymbol{\xi}^{(k)}) &= [(\nabla f(\boldsymbol{x}_1^{(k)}; \xi_1^{(k)})^\top; \cdots; (\nabla f(\boldsymbol{x}_n^{(k)}; \xi_n^{(k)})^\top], \\ \nabla F(\mathbf{y}^{(k)}; \boldsymbol{\xi}^{(k)}) &= [(\nabla f(\boldsymbol{y}_1^{(k)}; \xi_1^{(k)})^\top; \cdots; (\nabla f(\boldsymbol{y}_n^{(k)}; \xi_n^{(k)})^\top]. \end{aligned}$$

With these notations, DSGD-CECA is listed in Algorithm 1.

DSGD-CECA follows a similar communication protocol as CECA. The communicated information varies with iterations, and it is determined by δ_r. Agent i sends information $(\boldsymbol{z}_i^{(k)} - \gamma \boldsymbol{e}_i^{(k)})$ to agent j when $\mathbf{P}_{i,j}^{(k)} = 1$ and receives information $(\boldsymbol{z}_l^{(k)} - \gamma \boldsymbol{e}_l^{(k)})$ from agent l where $\mathbf{P}_{l,i}^{(k)} = 1$. After the information communication, $\mathbf{x}$ and $\mathbf{y}$ will update as follows

$$\begin{aligned} \mathbf{x}^{(k+1)} = a^{(k)}(\mathbf{x}^{(k)} - \gamma \mathbf{e}^{(k)}) + \\ (1 - a^{(k)}) \mathbf{P}^{(k)} (\mathbf{z}^{(k)} - \gamma \mathbf{e}^{(k)}), \end{aligned} \tag{8}$$

$$\begin{aligned} \mathbf{y}^{(k+1)} = b^{(k)}(\mathbf{y}^{(k)} - \gamma \mathbf{e}^{(k)}) + \\ (1 - b^{(k)}) \mathbf{P}^{(k)} (\mathbf{z}^{(k)} - \gamma \mathbf{e}^{(k)}). \end{aligned} \tag{9}$$

Extension to 1-port system. DSGD-CECA listed in Algorithm 1 is designed for 2-port communication system. But it can be easily extended to 1-port system with even n. The algorithm recursions are almost the same, except that in each iteration, the communication matrix $\mathbf{P}^{(k)}$ is sampled as $\mathbf{Q}^{(r)}$ in (7).

Implementation. The computation cost that incurs the highest expense in DSGD-CECA, primarily lies in the calculation of gradients. It is worth noting that, in each iteration, every agent i simply needs to compute a single stochastic gradient either at the current iterate $\boldsymbol{x}_i^{(k)}$ or $\boldsymbol{y}_i^{(k)}$. Furthermore, the iteration updates given by equations (8) and (9) involve only additions and scaling operations. Consequently, the additional computational burden introduced by DSGD-CECA is relatively minor when compared to vanilla DSGD.

4.2. Convergence analysis

CECA breaks double-stochasticity. To analyze DSGD-CECA for both 1-port and 2-port communication models, we introduce the following model mixing matrix $\mathbf{W} \in \mathbb{R}^{2n\times 2n}$ and gradient mixing matrix $\mathbf{W}_g \in \mathbb{R}^{2n\times 2n}$. In iteration k, we calculate $r = k \pmod{\tau}$. If $\delta_r = 1$, we have

$$\begin{aligned}\mathbf{W}^{(k)} &= \begin{bmatrix} \frac{1}{2}\mathbf{I}_n + \frac{1}{2}\mathbf{P}^{(k)} & \mathbf{0}_n \\ \frac{n_r+1}{2n_r+1}\mathbf{P}^{(k)} & \frac{n_r}{2n_r+1}\mathbf{I}_n \end{bmatrix}, \\ \mathbf{W}_g^{(k)} &= \begin{bmatrix} \frac{1}{2}\mathbf{I}_n + \frac{1}{2}\mathbf{P}^{(k)} & \mathbf{0}_n \\ \frac{n_r}{2n_r+1}\mathbf{I}_n + \frac{n_r+1}{2n_r+1}\mathbf{P}^{(k)} & \mathbf{0}_n \end{bmatrix}.\end{aligned} \tag{10}$$

If $\delta_r = 0$, we have

$$\begin{aligned}\mathbf{W}^{(k)} &= \begin{bmatrix} \frac{n_r+1}{2n_r+1}\mathbf{I}_n & \frac{n_r}{2n_r+1}\mathbf{P}^{(k)} \\ \mathbf{0}_n & \frac{1}{2}\mathbf{I}_n + \frac{1}{2}\mathbf{P}^{(k)} \end{bmatrix}, \\ \mathbf{W}_g^{(k)} &= \begin{bmatrix} \mathbf{0}_n & \frac{n_r+1}{2n_r+1}\mathbf{I}_n + \frac{n_r}{2n_r+1}\mathbf{P}^{(k)} \\ \mathbf{0}_n & \frac{1}{2}\mathbf{I}_n + \frac{1}{2}\mathbf{P}^{(k)} \end{bmatrix}.\end{aligned} \tag{11}$$

With mixing matrices $\mathbf{W}$ and $\mathbf{W}_g$, the update in (8) and (9) can be simply written as

$$\begin{bmatrix} \mathbf{x}^{(k+1)} \\ \mathbf{y}^{(k+1)} \end{bmatrix} = \mathbf{W}^{(k)} \begin{bmatrix} \mathbf{x}^{(k)} \\ \mathbf{y}^{(k)} \end{bmatrix} - \gamma \mathbf{W}_g^{(k)} \begin{bmatrix} \mathbf{g}^{(k)} \\ \mathbf{h}^{(k)} \end{bmatrix}. \tag{12}$$

Vanilla DSGD typically requires mixing matrix be doubly stochastic, i.e., each row and column sum equals 1. This adorable property will enable the algorithm to converge to a consensus and correct solution. However, it is observed in (10) or (11) that neither $\mathbf{W}$ nor $\mathbf{W}_g$ is column-stochastic. This will bring fundamental challenges to establish convergence guarantees for DSGD-CECA.

Favorable properties of $\mathbf{W}$ and $\mathbf{W}_g$. We next establish several fundamental properties of $\mathbf{W}$ and $\mathbf{W}_g$. To this end, we introduce the *mixing matrix family* $\mathscr{F}$ as follows,

$$\mathscr{F} = \Big\{ \mathbf{W} = \begin{bmatrix} c\mathbf{W}_{1,1} & (1-c)\mathbf{W}_{1,2} \\ d\mathbf{W}_{2,1} & (1-d)\mathbf{W}_{2,2} \end{bmatrix} \in \mathbb{R}^{2n\times 2n}, \\ \text{where } 0 \le c, d \le 1, \mathbf{W}_{i,j} \in \mathbb{R}^{n\times n} \text{ doubly stochastic}, \\ i, j \in \{1, 2\} \Big\} \tag{13}$$

We summarize several properties for any $\mathbf{W} \in \mathscr{F}$. The proofs are in Appendix B.2.

Lemma 4.1. *The matrix family $\mathscr{F}$ satisfies the following properties:*

- *The matrix family $\mathscr{F}$ is a convex subset of the row stochastic matrices.*
- *For any $\mathbf{W} \in \mathscr{F}$, it holds that $\|\mathbf{W}\| \le \sqrt{2}$.*
- *For any $\mathbf{W} \in \mathscr{F}$, it holds that*

$$\mathbf{W} \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \mathbf{0}_n & \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \end{bmatrix} = \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \mathbf{0}_n & \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \end{bmatrix} \mathbf{W},$$

where $\mathbb{1}_n$ is the all-ones vector in $\mathbb{R}^n$.

Lemma 4.2. *In Algorithm 1, when the communication matrix $\mathbf{P}^{(k)}$ is sampled from (5) or (6) according to δ_r, and the mixing matrix $\mathbf{W}^{(k)}$ is sampled from (10) or (11) according to δ_r. The product of matrices $\mathbf{W}^{(k)}$ satisfies*

$$\prod_{k=0}^{t} \mathbf{W}^{(k)} = \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \end{bmatrix}, \quad \forall t \ge \tau. \tag{14}$$

Remark 4.3. Lemma 4.1 and Lemma 4.2 are fundamental to establish the convergence property of DSGD-CECA. Lemma 4.1 implies that the gossip matrix in DSGD-CECA, while not doubly stochastic, belongs to a family that shares many similarities with the doubly stochastic matrix family. These properties help break the doubly stochastic constraint in the standard DSGD analysis framework. Lemma 4.2 essentially states that, while not being column-stochastic, the special structure of the gossip matrix as constructed in (10) or (11) can still enable global average after multiplying with more than $(\tau + 1)$ consecutive $\mathbf{W}^{(k)}$. Both models $\boldsymbol{x}$ and $\boldsymbol{y}$ can achieve the global average, which extends beyond the findings of (Bar-Noy et al., 1993) that focused solely on the consensus of the $\boldsymbol{x}$ model.

Convergence property. We finally establish the convergence theorem of Algorithm 1. We let $\bar{\mathbf{x}}^{(k)} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i^{(k)} \in \mathbb{R}^d$ be the average of all local model.

Theorem 4.4 (CONVERGENCE PROPERTY)**.** *Suppose Assumptions 2.1-2.3 hold, and we conduct global averaging in the first τ steps so that $\boldsymbol{x}_i^{(k)} = \bar{\mathbf{x}}^{(k)}$, $\boldsymbol{y}_i^{(k)} = \bar{\mathbf{y}}^{(k)}$ for $0 \le k < \tau$. Starting from the $(\tau+1)$th step, we perform DSGD-CECA iterations (8), (9). If γ satisfies*

$$\gamma = \frac{1}{\left(\frac{2n\Delta}{L\sigma^2(T+1)}\right)^{-\frac{1}{2}} + \left(\frac{\Delta}{24L^2\tau^2(\sigma^2+2b^2)(T+1)}\right)^{-\frac{1}{3}} + 8\tau L}$$

where $\Delta = \mathbb{E}f(\bar{\mathbf{x}}^0) - f^\star$, then DSGD-CECA converges at

$$\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\nabla f(\bar{\mathbf{x}}^{(k)})\|^2 \leq 16\left(\frac{\Delta L\sigma^2}{n(T+1)}\right)^{\frac{1}{2}} + 24\left(\frac{\Delta^2 L^2\tau^2(\sigma^2+2b^2)}{(T+1)^2}\right)^{\frac{1}{3}} + \frac{32\tau\Delta L}{T+1}. \quad (15)$$

with any n when utilizing the 2-port communication model, or with any even n when utilizing the 1-port model.

The proof of Theorem 4.4 can be referred to Appendix B.4.

Remark 4.5. Based on the convergence rate in (15), we can derive that when $T = O(n^3\log_2^4(n))$, the linear speedup term $O(1/\sqrt{nT})$ dominates the other two terms $O(\tau^{\frac{2}{3}}/T^{\frac{2}{3}})$ and $O(\tau/T)$ up to a constant scalar. This linear speedup term dominates the convergence rate. This implies DSGD-CECA has $O(n^3\log_2^4(n))$ transient iterations.

5. Numerical Experiments

In this section, we validate the previous theoretical results via numerical experiments. First, we show CECA-1P indeed achieves the global consensus in finite iterations over a variety of choices of the number of nodes. Next, we examine the performance of DSGD-CECA and compare it with many other popular SOTA algorithms on a standard convex task. Lastly, we apply the DSGD-CECA on the deep learning setting to show it still achieves good performance in train loss and test accuracy with respect to the iterations and communicated data. The codes used to generate the figures in this section are available in the github[1].

Finite-time exact consensus convergence. We examine the convergence rate of CECA-1P over different network sizes n. In each experiment, we initialize a random vector $\boldsymbol{x}_i^{(0)}, \boldsymbol{y}_i^{(0)}$ on each node i, and obtain $\boldsymbol{x}_i^{(k)}, \boldsymbol{y}_i^{(k)}$ by applying the communication topology. The residue $\sum_i \|\boldsymbol{x}_i^{(k)} - \boldsymbol{x}^\star\|$ is calculated at each iteration k, where $\boldsymbol{x}^\star$ is the global average of all initial $\boldsymbol{x}_i^{(0)}$. From Fig. 5, we observe the results coincide with the proved theorem. Especially, the number of iterations of CECA-1P to achieve exact global average is $\lceil\log_2(n)\rceil$ as the theorem predicted.

Next, we compare CECA with other popular topologies. We set the network size to $n = 130$ and $n = 1026$, respectively (both are not the power of 2 but close to it). Results are averaged over 3 independent random runs. In Fig. 6, we observe CECA achieves global average with a finite number of iterations, whereas the others do not.

DSGD: least-squares problem. We examine DSGD-CECA by solving the distributed least square problem in which each $f_i(\boldsymbol{x}) := (1/2)\|\mathbf{A}_i\boldsymbol{x} - \boldsymbol{b}_i\|^2$ where $i \in \{1, 2, \ldots, n\}$,

[1] https://github.com/kexinjinnn/DSGD-CECA

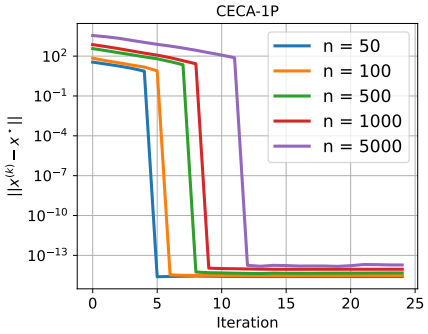

Figure 5. The CECA-1P can achieve finite-time consensus convergence for different network sizes.

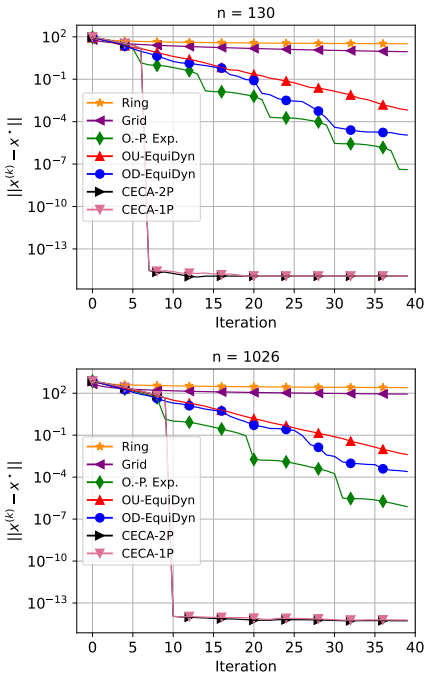

Figure 6. The consensus rates of CECA-2P and CECA-1P are faster than those of the other $\Theta(1)$-neighbor topologies.

$\boldsymbol{x} \in \mathbb{R}^d$ and $\mathbf{A}_i \in \mathbb{R}^{N\times d}$. We generate $\mathbf{A}_i$ from $\mathcal{N}(0, I)$. The measurement $\boldsymbol{b}_i$ is generated by $\boldsymbol{b}_i = \mathbf{A}_i\boldsymbol{x}^\star + \boldsymbol{v}_i$ with a given $\boldsymbol{x}^\star \in \mathbb{R}^d$ where $\boldsymbol{v}_i \sim \mathcal{N}(0, \sigma_s^2\mathbf{I})$. Each node will generate a stochastic gradient via $\widehat{\nabla f_i}(\boldsymbol{x}) = \nabla f_i(\boldsymbol{x}) + \boldsymbol{\epsilon}_i$ at each iteration, where $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \sigma_n^2\mathbf{I})$ is the noise level of SGD. In the simulation, we set the size of the network $n = 258$, $d = 10$, $N = 50$, $\sigma_s = 0.1$, and $\sigma_n = 5$. We set the initial learning rate to be 0.02 to all algorithms. Then, every 20 iterations the learning rate decays by a factor 1.5. The results are averaged over 5 independent random experiments. Fig. 7 depicts the performance of each algorithm. It is observed that DSGD-CECA achieves the best convergence performance.

DSGD: deep learning. We apply DSGD-CECA-2P to solve the image classification task with CNN over **MNIST** dataset

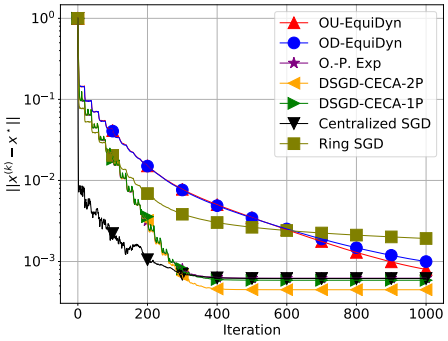

Figure 7. Performance comparison between stochastic decentralized algorithms using various effective topologies.

(LeCun et al., 2010). As for the implementation of decentralized parallel training, we utilize BlueFog library (Ying et al., 2021b) in a cluster of 17 NVIDIA GeForce RTX 2080 GPUs. The network contains two convolutional layers with max pooling and ReLu and two feed-forward layers. The local batch size is set to 64. The learning rate is 0.3 for DSGD-CECA-2P with no momentum and 0.1 for other algorithms with momentum 0.5. The results are obtained by averaging over 3 independent random experiments. Fig. 8 illustrates the training loss and test accuracy curves, while Table 2 provides the corresponding numerical values. The results indicate that DSGD-CECA-2P outperforms other decentralized algorithms, exhibiting slightly better training loss and test accuracy. Fig. 9 depicts the performance of different algorithms in terms of data communicated. The data communicated is calculated based on the total length of the vectors that one node sends and receives. If different nodes have different values, we choose the one with the largest value since it is synchronized style algorithm. The figure implies that one-peer decentralized algorithms, including DSGD-CECA and O.-P.-Exp., outperform centralized SGD significantly.

We also provide additional experiments on **CIFAR-10** dataset (Krizhevsky & Hinton, 2009) in Appendix C.

Table 2. Comparison of train loss and test accuracy(%) with different topologies over MNIST dataset.

Topology	Train Loss	Test Acc.
Centralized SGD	**2.079**	98.34
Ring	2.090	98.32
O.-P. Exp.	2.091	98.33
OD-EquiDyn	2.090	98.36
OU-EquiDyn	2.091	98.03
DSGD-CECA-2P	**2.083**	**98.50**

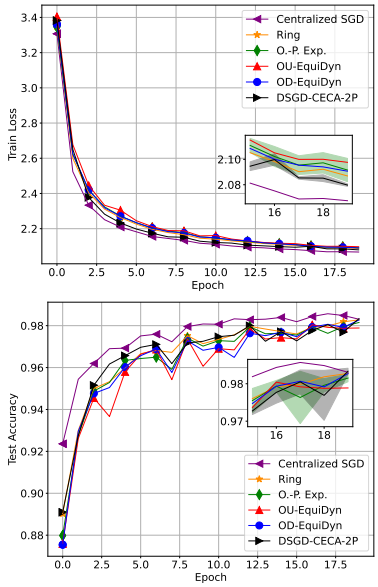

Figure 8. Train loss and test accuracy of different DSGD algorithms for CNN on MNIST. The solid curve indicates the average, while the shaded area indicates the deviation.

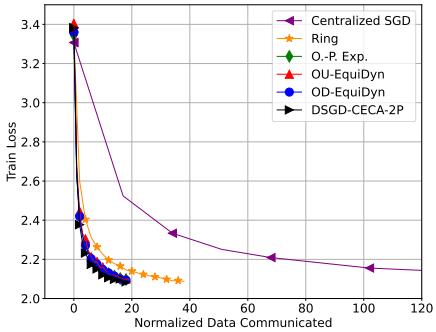

Figure 9. Performance of different algorithms over MNIST dataset in terms of data communicated.

6. Conclusion

In this paper, we propose a novel decentralized stochastic gradient descent algorithm, named DSGD-CECA. This algorithm consists of two versions: DSGD-CECA-1P and DSGD-CECA-2P, designed for the 1-port and 2-port message-passing models, respectively. The convergence rates of both versions are theoretically analyzed for non-convex stochastic optimization. The results demonstrate that, even at the minimal communication cost per iteration, the total number of iterations and the transient iterations are comparable to the state-of-the-art methods. Notably, the proposed methods are applicable to any number of agents, significantly relaxing the previous restriction of the power of two. Furthermore, empirical experiments validate the efficiency of DSGD-CECA in comparison to other DSGD algorithms.

References

Alghunaim, S. A. and Yuan, K. A unified and refined convergence analysis for non-convex decentralized learning. *arXiv preprint arXiv:2110.09993*, 2021.

Assran, M., Loizou, N., Ballas, N., and Rabbat, M. Stochastic gradient push for distributed deep learning. In *International Conference on Machine Learning (ICML)*, pp. 344–353, 2019.

Bar-Noy, A., Kipnis, S., and Schieber, B. An optimal algorithm for computing census functions in message-passing systems. *Parallel Processing Letters*, 3(01):19–23, 1993.

Ben-Nun, T. and Hoefler, T. Demystifying parallel and distributed deep learning: An in-depth concurrency analysis. *ACM Computing Surveys (CSUR)*, 52(4):1–43, 2019.

Benjamini, I., Kozma, G., and Wormald, N. The mixing time of the giant component of a random graph. *Random Structures & Algorithms*, 45(3):383–407, 2014.

Beveridge, A. and Youngblood, J. The best mixing time for random walks on trees. *Graphs and Combinatorics*, 32 (6):2211–2239, 2016.

Boyd, S. P., Ghosh, A., Prabhakar, B., and Shah, D. Mixing times for random walks on geometric random graphs. In *ALENEX/ANALCO*, pp. 240–249, 2005.

Chen, J. and Sayed, A. H. Diffusion adaptation strategies for distributed optimization and learning over networks. *IEEE Transactions on Signal Processing*, 60(8):4289–4305, 2012.

Di Lorenzo, P. and Scutari, G. Next: In-network nonconvex optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 2(2):120–136, 2016.

Dimakis, A. G., Kar, S., Moura, J. M., Rabbat, M. G., and Scaglione, A. Gossip algorithms for distributed signal processing. *Proceedings of the IEEE*, 98(11):1847–1864, 2010.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Jiang, Y., Zhu, Y., Lan, C., Yi, B., Cui, Y., and Guo, C. A unified architecture for accelerating distributed dnn training in heterogeneous gpu/cpu clusters. In *Proceedings of the 14th USENIX Conference on Operating Systems Design and Implementation*, pp. 463–479, 2020.

Koloskova, A., Stich, S., and Jaggi, M. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, pp. 3478–3487, 2019.

Koloskova, A., Loizou, N., Boreiri, S., Jaggi, M., and Stich, S. U. A unified theory of decentralized sgd with changing topology and local updates. In *International Conference on Machine Learning (ICML)*, pp. 1–12, 2020.

Kong, L., Lin, T., Koloskova, A., Jaggi, M., and Stich, S. U. Consensus control for decentralized deep learning. In *International Conference on Machine Learning*, 2021.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

LeCun, Y., Cortes, C., and Burges, C. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Li, Z., Shi, W., and Yan, M. A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates. *IEEE Transactions on Signal Processing*, July 2019. early acces. Also available on arXiv:1704.07807.

Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pp. 5330–5340, 2017.

Lian, X., Zhang, W., Zhang, C., and Liu, J. Asynchronous decentralized parallel stochastic gradient descent. In *International Conference on Machine Learning*, pp. 3043–3052, 2018.

Lin, T., Karimireddy, S. P., Stich, S. U., and Jaggi, M. Quasi-global momentum: Accelerating decentralized deep learning on heterogeneous data. In *International Conference on Machine Learning*, 2021.

Liu, K. Train CIFAR10 with PyTorch. `https://github.com/kuangliu/pytorch-cifar`, 2021. Accessed: 2023-01.

Lopes, C. G. and Sayed, A. H. Diffusion least-mean squares over adaptive networks: Formulation and performance analysis. *IEEE Transactions on Signal Processing*, 56(7): 3122–3136, 2008.

Nachmias, A. and Peres, Y. Critical random graphs: diameter and mixing time. *The Annals of Probability*, 36(4): 1267–1286, 2008.

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

Nedic, A., Olshevsky, A., and Shi, W. Achieving geometric convergence for distributed optimization over time-varying graphs. *SIAM Journal on Optimization*, 27(4): 2597–2633, 2017.

Nedić, A., Olshevsky, A., and Rabbat, M. G. Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE*, 106 (5):953–976, 2018.

Niwa, K., Zhang, G., Kleijn, W. B., Harada, N., Sawada, H., and Fujino, A. Asynchronous decentralized optimization with implicit stochastic variance reduction. In *International Conference on Machine Learning*, pp. 8195–8204. PMLR, 2021.

Patarasuk, P. and Yuan, X. Bandwidth optimal all-reduce algorithms for clusters of workstations. *Journal of Parallel and Distributed Computing*, 69(2):117–124, 2009.

Pu, S., Olshevsky, A., and Paschalidis, I. C. A sharp estimate on the transient time of distributed stochastic gradient descent. *arXiv preprint arXiv:1906.02702*, 2019.

Qu, G. and Li, N. Harnessing smoothness to accelerate distributed optimization. *IEEE Transactions on Control of Network Systems*, 5(3):1245–1260, 2018.

Shi, W., Ling, Q., Yuan, K., Wu, G., and Yin, W. On the linear convergence of the admm in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 62(7):1750–1761, 2014.

Shi, W., Ling, Q., Wu, G., and Yin, W. EXTRA: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

Song, Z., Li, W., Jin, K., Shi, L., Yan, M., Yin, W., and Yuan, K. Communication-efficient topologies for decentralized learning with $o(1)$ consensus rate. *arXiv preprint arXiv:2210.07881*, 2022.

Tang, H., Lian, X., Yan, M., Zhang, C., and Liu, J. d^2: Decentralized training over decentralized data. In *International Conference on Machine Learning*, pp. 4848–4856, 2018.

Trevisan, L. Lecture notes on graph partitioning, expanders and spectral methods. *University of California, Berkeley, https://people. eecs. berkeley. edu/˜ luca/books/expanders-2016. pdf*, 2017.

Tsitsiklis, J., Bertsekas, D., and Athans, M. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE transactions on automatic control*, 31(9):803–812, 1986.

Xin, R., Khan, U. A., and Kar, S. An improved convergence analysis for decentralized online stochastic non-convex optimization. *arXiv preprint arXiv:2008.04195*, 2020.

Xu, J., Zhu, S., Soh, Y. C., and Xie, L. Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes. In *IEEE Conference on Decision and Control (CDC)*, pp. 2055–2060, Osaka, Japan, 2015.

Ying, B., Yuan, K., Chen, Y., Hu, H., Pan, P., and Yin, W. Exponential graph is provably efficient for decentralized deep training. *Advances in Neural Information Processing Systems*, 34:13975–13987, 2021a.

Ying, B., Yuan, K., Hu, H., Chen, Y., and Yin, W. Bluefog: Make decentralized algorithms practical for optimization and deep learning. *arXiv preprint arXiv:2111.04287*, 2021b.

Yuan, K. and Alghunaim, S. A. Removing data heterogeneity influence enhances network topology dependence of decentralized sgd. *arXiv preprint arXiv:2105.08023*, 2021.

Yuan, K., Ling, Q., and Yin, W. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.

Yuan, K., Ying, B., Zhao, X., and Sayed, A. H. Exact dffusion for distributed optimization and learning – Part I: Algorithm development. *IEEE Transactions on Signal Processing*, 67(3):708 – 723, 2019.

Yuan, K., Alghunaim, S. A., Ying, B., and Sayed, A. H. On the influence of bias-correction on distributed stochastic optimization. *IEEE Transactions on Signal Processing*, 2020.

Yuan, K., Chen, Y., Huang, X., Zhang, Y., Pan, P., Xu, Y., and Yin, W. DecentLaM: Decentralized momentum SGD for large-batch deep training. *arXiv preprint arXiv:2104.11981*, 2021.

A. Notations

We introduce the following notations to simplify analysis.

- $\mathbf{x}^{(k)} = [(\boldsymbol{x}_1^{(k)})^\top; (\boldsymbol{x}_2^{(k)})^\top; \cdots; (\boldsymbol{x}_n^{(k)})^\top] \in \mathbb{R}^{n\times d}$ and $\bar{\mathbf{x}}^{(k)} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i^{(k)} \in \mathbb{R}^d$.
- $\mathbf{y}^{(k)} = [(\boldsymbol{y}_1^{(k)})^\top; (\boldsymbol{y}_2^{(k)})^\top; \cdots; (\boldsymbol{y}_n^{(k)})^\top] \in \mathbb{R}^{n\times d}$ and $\bar{\mathbf{y}}^{(k)} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{y}_i^{(k)} \in \mathbb{R}^d$.
- $\mathbf{z}^{(k)} = [(\boldsymbol{z}_1^{(k)})^\top; (\boldsymbol{z}_2^{(k)})^\top; \cdots; (\boldsymbol{z}_n^{(k)})^\top] \in \mathbb{R}^{n\times d}$ and $\bar{\mathbf{z}}^{(k)} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{z}_i^{(k)} \in \mathbb{R}^d$.
- $\mathbf{W}^{(k)} = [w_{i,j}^{(k)}] \in \mathbb{R}^{2n\times 2n}$.
- $\mathbb{1}_n = [1; 1; \cdots; 1] \in \mathbb{R}^n$.
- $\nabla F(\mathbf{x}) = [(\nabla f_1(\boldsymbol{x}_1))^\top; (\nabla f_2(\boldsymbol{x}_2))^\top; \cdots; (\nabla f_n(\boldsymbol{x}_n))^\top] \in \mathbb{R}^{n\times d}$ for any $\mathbf{x} = [\boldsymbol{x}_1^\top; \boldsymbol{x}_2^\top; \cdots; \boldsymbol{x}_n^\top] \in \mathbb{R}^{n\times d}$.
- $\nabla F(\mathbf{x}; \boldsymbol{\xi}) = [(\nabla F_1(\boldsymbol{x}_1; \xi_1))^\top; (\nabla F_2(\boldsymbol{x}_2; \xi_2))^\top; \cdots; (\nabla F_n(\boldsymbol{x}_n; \xi_n))^\top] \in \mathbb{R}^{n\times d}$.
- $\mathbf{g}^{(k)} = \nabla F(\mathbf{x}^{(k)}; \boldsymbol{\xi}^{(k)})$ and $\bar{\mathbf{g}}^{(k)} = \frac{1}{n}\sum_{i=1}^n \nabla F(\boldsymbol{x}_i^{(k)}; \boldsymbol{\xi}_i^{(k)})$.
- $\mathbf{h}^{(k)} = \nabla F(\mathbf{y}^{(k)}; \boldsymbol{\xi}^{(k)})$ and $\bar{\mathbf{h}}^{(k)} = \frac{1}{n}\sum_{i=1}^n \nabla F(\boldsymbol{y}_i^{(k)}; \boldsymbol{\xi}_i^{(k)})$.
- $\mathbf{e}^{(k)} = \nabla F(\mathbf{z}^{(k)}; \boldsymbol{\xi}^{(k)})$ and $\bar{\mathbf{e}}^{(k)} = \frac{1}{n}\sum_{i=1}^n \nabla F(\boldsymbol{z}_i^{(k)}; \boldsymbol{\xi}_i^{(k)})$.
- Given two matrices $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{n\times d}$, we define the inner product $\langle \mathbf{x}, \mathbf{y}\rangle = \mathrm{tr}(\mathbf{x}^\top \mathbf{y})$, the Frobenius norm $\|\mathbf{x}\|_F^2 = \langle \mathbf{x}, \mathbf{x}\rangle$.
- Given any vector $\boldsymbol{x} \in \mathbb{R}^d$, we let $\|\boldsymbol{x}\|$ be its ℓ_2 norm.
- Given a sequence of matrices $\{\mathbf{A}^{(k)}\}_{k=i}^j$ with $j \geq i$, we let $\prod_{k=i}^j \mathbf{A}^{(k)} = \mathbf{A}^{(j)}\mathbf{A}^{(j-1)}\cdots\mathbf{A}^{(i+1)}\mathbf{A}^{(i)}$. If $j < i$, we let $\prod_{k=i}^j \mathbf{A}^{(k)} = \mathbf{I}$.

B. Optimal dynamic topology DSGD

B.1. Supplementary materials on CECA

B.1.1. CECA FOR THE 2-PORT MESSAGE PASSING SYSTEM

In this section, we present the CECA in a detailed way. The pseudo code of the CECA as in Algorithm 2.

Before the iterations start, we calculate $\tau = \lceil \log_2 n \rceil$. We compute $\delta_r, n_r, r = 0, 1, 2, \ldots, \tau - 1$ as in (2) and (3).

The method is initialized before the round 0 with each agent i holding information $I_i^{(0)} = u_i, J_i^{(0)} = 0$. We use $I_i^{(r)}$ and $J_i^{(r)}$ to denote agent i's information right before the rth round, $r = 0, 1, \ldots, \tau - 1$. In the rth round, if $\delta_r = 1$, agent i sends information $I_i^{(r)}$ to agent $i + n_r + 1$ and receives information $I_{i-n_r-1}^{(r)}$ from agent $i - n_r - 1$; if $\delta_r = 0$, agent i sends information $J_i^{(r)}$ to agent $i + n_r$ and receives information $J_{i-n_r}^{(r)}$ from agent $i - n_r$.

After receiving the information, each agent i updates information $I^{(r)}$ and $J^{(r)}$ with the received information. We switch over cases where δ_r is either 0 or 1. If $\delta_r = 1$,

$$
\begin{aligned}
I_i^{(r+1)} &= \frac{1}{2} I_i^{(r)} + \frac{1}{2} I_{i-n_r-1}^{(r)}, \\
J_i^{(r+1)} &= \frac{n_r}{2n_r+1} J_i^{(r)} + \frac{n_r+1}{2n_r+1} I_{i-n_r-1}^{(r)}.
\end{aligned}
$$

If $\delta_r = 0$,

$$
I_i^{(r+1)} = \frac{n_r+1}{2n_r+1} I_i^{(r)} + \frac{n_r}{2n_r+1} J_{i-n_r}^{(r)}, \quad J_i^{(r+1)} = \frac{1}{2} J_i^{(r)} + \frac{1}{2} J_{i-n_r}^{(r)}.
$$

Algorithm 2 CECA for the 2-port system

Input: n agents: each agent i has data $u_i, i = 1, 2, \ldots, n$.
Set $\tau = \lceil \log_2 n \rceil$ (total rounds of message passing);
Represent $n-1$ with the binary sequence $(\delta_0\, \delta_1\, \cdots\, \delta_{\tau-1})_2$;
Initialize $n_0 = 0$ and, for $i = 1, \ldots, n$, $I_i^{(0)} = u_i$ and $J_i^{(0)} = 0$;
for $r = 0, 1, 2, \ldots, \tau - 1$ **do**
  if $\delta_r = 1$ **then**
    Agent i sends information $I_i^{(r)}$ to agent $i + n_r + 1$ and receives information $I_{i-n_r-1}^{(r)}$ from agent $i - n_r - 1$;
    $I_i^{(r+1)} = \frac{1}{2} I_i^{(r)} + \frac{1}{2} I_{i-n_r-1}^{(r)}$;
    $J_i^{(r+1)} = \frac{n_r}{2n_r+1} J_i^{(r)} + \frac{n_r+1}{2n_r+1} I_{i-n_r-1}^{(r)}$;
  else if $\delta_r = 0$ **then**
    Agent i sends information $J_i^{(r)}$ to agent $i + n_r$ and receives information $J_{i-n_r}^{(r)}$ from agent $i - n_r$;
    $I_i^{(r+1)} = \frac{n_r+1}{2n_r+1} I_i^{(r)} + \frac{n_r}{2n_r+1} J_{i-n_r}^{(r)}$;
    $J_i^{(r+1)} = \frac{1}{2} J_i^{(r)} + \frac{1}{2} J_{i-n_r}^{(r)}$;
  end if
  $n_{r+1} = 2n_r + \delta_r$;
end for
return $I_i^{(\tau)}, i = 1, 2, \ldots, n$**, which equal** $\frac{1}{n}(u_1 + \cdots + u_n)$**.**

In either case $\delta_r = 0$ or $\delta_r = 1$, we have

$$I_i^{(r+1)} = \frac{1}{n_{r+1}+1} \sum_{j=0}^{n_{r+1}} u_{i-j}, \quad J_i^{(r+1)} = \frac{1}{n_{r+1}} \sum_{j=1}^{n_{r+1}} u_{i-j}.$$

by induction. After τ rounds of communication,

$$I_i^{(\tau)} = \frac{1}{n} \sum_{j=1}^{n} u_j, \; J_i^{(\tau)} = \frac{1}{n-1} \sum_{j \neq i} u_j.$$

Example: We consider the case where the number of agents $n = 6$. To make consensus among agents, we need $\tau = \lceil \log_2(6) \rceil = 3$ rounds. The binary representation of $n - 1 = 5$ is

$$5 = (1\,0\,1)_2.$$

According to (2), we assign $\delta_0 = 1$ as the most significant (left most) digit of the binary representation. Besides, we assign $\delta_1 = 0, \delta_2 = 1$. We calculate $n_0 = 0, n_1 = 1, n_2 = 2$. Besides, we let $\mathbf{I}^{(r)} = [I_1^{(r)}; I_2^{(r)}; \cdots; I_6^{(r)}], \mathbf{J}^{(r)} = [J_1^{(r)}; J_2^{(r)}; \cdots; J_6^{(r)}]$ be the formal column vectors of $I_i^{(r)}$ and $J_i^{(r)}$. Both $I_i^{(r)}$ and $J_i^{(r)}$ are linear combinations of $u_1, u_2, \ldots, u_6$. We formally use the matrix-vector product

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,6} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,6} \\ \vdots & \vdots & \ddots & \vdots \\ a_{6,1} & a_{6,2} & \cdots & a_{6,6} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_6 \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{6} a_{1,j} u_j \\ \sum_{j=1}^{6} a_{2,j} u_j \\ \vdots \\ \sum_{j=1}^{6} a_{6,j} u_j \end{bmatrix}$$

to be the representation of $\mathbf{I}^{(r)}$ or $\mathbf{J}^{(r)}$. At the very beginning, agent i has information $I_i^{(0)} = u_i, J_i^{(0)} = 0$. So,

$$\mathbf{I}^{(0)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}, \quad \mathbf{J}^{(0)} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}.$$

In the 0th round, $\delta_0 = 1$, agent i sends $I_i^{(0)} = u_i$ to agent $i+1$ and receives $I_{i-1}^{(0)} = u_{i-1}$ from agent $i-1$. Agent i averages the received information with $I_i^{(0)}, J_i^{(0)}$, respectively, and get $I_i^{(1)}, J_i^{(1)}$ as follows,

$$\mathbf{I}^{(1)} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}, \quad \mathbf{J}^{(1)} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}.$$

In the 1st round, $\delta_1 = 0$, agent i sends $J_i^{(1)} = u_{i-1}$ to agent $i+1$ and receives $J_{i-1}^{(1)} = u_{i-2}$ from agent $i-1$. After averaging, we get

$$\mathbf{I}^{(2)} = \begin{bmatrix} \frac{1}{3} & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 \\ 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}, \quad \mathbf{J}^{(2)} = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}.$$

In the 2nd round, $\delta_2 = 1$, agent i sends $I_i^{(2)} = \frac{1}{3}u_{i-2} + \frac{1}{3}u_{i-1} + \frac{1}{3}u_i$ to agent $i+3$, and receives $I_{i-3}^{(2)} = \frac{1}{3}u_{i-5} + \frac{1}{3}u_{i-4} + \frac{1}{3}u_{i-3}$ from agent $i-3$. After averaging the information, we get

$$\mathbf{I}^{(3)} = \begin{bmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}, \quad \mathbf{J}^{(3)} = \begin{bmatrix} 0 & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & 0 & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & 0 & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & 0 & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & 0 & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}.$$

In the upper part of Table B.1.1, we give the update process of $I_i^{(r)}, J_i^{(r)}$ in the 3 rounds when the initial u_1–u_6 are assigned to be values 1–6.

B.1.2. Optimal allreduce algorithm for the 1-port message passing system

Here, we give a more detailed description of the optimal allreduce algorithm introduced in Section 3.2

We calculate that $\tau = \lceil \log_2 n \rceil$ rounds are needed to reach consensus. We compute the necessary parameters $\delta_r, n_r, r = 0, 1, 2, \ldots, \tau - 1$ as in (2) and (3).

Before the communication starts, we initialize $I_i^{(0)} = u_i$, $J_i^{(0)} = 0$. In the rth round, $r = 0, 1, 2, \ldots, \tau - 1$, if i odd, we pair up agents i and $i + 2n_r + 1$; if i even, we pair up agents i and $i - 2n_r - 1$. Each agent exchanges information with its peer. If $\delta_r = 1$, the peers exchange $I^{(r)}$; if $\delta_r = 0$, they exchange $J^{(r)}$ instead.

Let $\omega_i^{(r)}$ denote the index of the agent who sends message to agent i in the rth round. If $\delta_r = 1$,

$$\begin{aligned} I_i^{(r+1)} &= \frac{1}{2} I_i^{(r)} + \frac{1}{2} I_{\omega_i^{(r)}}^{(r)} \\ J_i^{(r+1)} &= \frac{n_r}{2n_r + 1} J_i^{(r)} + \frac{n_r + 1}{2n_r + 1} I_{\omega_i^{(r)}}^{(r)}. \end{aligned}$$

If $\delta_r = 0$,

$$I_i^{(r+1)} = \frac{n_r + 1}{2n_r + 1} I_i^{(r)} + \frac{n_r}{2n_r + 1} J_{\omega_i^{(r)}}^{(r)},$$

2-port CECA												
round r	$I_1^{(r)}$	$J_1^{(r)}$	$I_2^{(r)}$	$J_2^{(r)}$	$I_3^{(r)}$	$J_3^{(r)}$	$I_4^{(r)}$	$J_4^{(r)}$	$I_5^{(r)}$	$J_5^{(r)}$	$I_6^{(r)}$	$J_6^{(r)}$
0	1	0	2	0	3	0	4	0	5	0	6	0
	$n_0 = 0$, $\delta_0 = 1$											
1	3.5	6	1.5	1	2.5	2	3.5	3	4.5	4	5.5	5
	$n_1 = 1$, $\delta_1 = 0$											
2	4	5.5	3	3.5	2	1.5	3	2.5	4	3.5	5	4.5
	$n_2 = 2$, $\delta_2 = 1$											
3	3.5	4	3.5	3.8	3.5	3.6	3.5	3.4	3.5	3.2	3.5	3

New algorithm (1-port allreduce averaging)												
round r	$I_1^{(r)}$	$J_1^{(r)}$	$I_2^{(r)}$	$J_2^{(r)}$	$I_3^{(r)}$	$J_3^{(r)}$	$I_4^{(r)}$	$J_4^{(r)}$	$I_5^{(r)}$	$J_5^{(r)}$	$I_6^{(r)}$	$J_6^{(r)}$
0	1	0	2	0	3	0	4	0	5	0	6	0
	$n_0 = 1$, $\delta_0 = 1$, agents $(1\ 2)\ (3\ 4)\ (5\ 6)$ exchange I											
1	1.5	2	1.5	1	3.5	4	3.5	3	5.5	6	5.5	5
	$n_1 = 1$, $\delta_1 = 0$, agents $(1\ 4)\ (2\ 5)\ (3\ 6)$ exchange J											
2	2	2.5	3	3.5	4	4.5	3	2.5	4	3.5	5	4.5
	$n_2 = 2$, $\delta_2 = 1$, agents $(1\ 6)\ (2\ 3)\ (4\ 5)$ exchange I											
3	3.5	4	3.5	3.8	3.5	3.6	3.5	3.4	3.5	3.2	3.5	3

Table 3. Illustration of the two allreduce averaging algorithms applied to 6 agents with agents 1–6 having initial numbers 1–6. Both algorithms take $\lceil \log_2(6) \rceil = 3$ rounds to achieve $I_i^{(r)} \equiv 3.5$ for all $i = 1, \ldots, 6$.

$$J_i^{(r+1)} = \frac{1}{2} J_i^{(r)} + \frac{1}{2} J_{\omega_i^{(r)}}^{(r)}.$$

After the above averaging process, each odd agent i has information

$$I_i^{(r+1)} = \frac{1}{n_{r+1}+1} \sum_{j=0}^{n_{r+1}} u_{i+j}, \quad J_i^{(r+1)} = \frac{1}{n_{r+1}} \sum_{j=1}^{n_{r+1}} u_{i+j}.$$

Each even agent i has information

$$I_i^{(r+1)} = \frac{1}{n_{r+1}+1} \sum_{j=0}^{n_{r+1}} u_{i-j}, \quad J_i^{(r+1)} = \frac{1}{n_{r+1}} \sum_{j=1}^{n_{r+1}} u_{i-j}.$$

After τ rounds, each agent has the consensus information $I_i^{(\tau)} = \frac{1}{n} \sum_{j=1}^{n} u_j$.

Example: We give an example when $n = 6$. The calculation of δ_r and n_r is the same as in the example of Section 3.1. Before the communications start, each agent i has information $I_i^{(0)} = u_i$, $J_i^{(0)} = 0$. In the 0th round, we pair up agents as follows: 1 with 2, 3 with 4, and 5 with 6, in a peer-to-peer manner. Each agent sends its $I^{(0)}$ information to its peer and receives the same type of $I^{(0)}$ information in return. After averaging, I_i, J_i are updated as

$$\mathbf{I}^{(1)} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}, \quad \mathbf{J}^{(1)} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}.$$

In the 1st round, $\delta_1 = 0$. We pair up agents $(1, 4), (2, 5), (3, 6)$. Each agent sends $J^{(1)}$ information to its peer and also

receives $J^{(1)}$ information in return. After averaging, we get

$$\mathbf{I}^{(2)} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 & \frac{1}{3} \\ 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 \\ \frac{1}{3} & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}, \quad \mathbf{J}^{(2)} = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}.$$

In the 2nd round, $\delta_2 = 1$. The pairing mode for agents is $(1, 6), (2, 3), (4, 5)$. Each agent exchanges $J^{(2)}$ information with its peer. After merging the received information into the previous information, we have

$$\mathbf{I}^{(3)} = \begin{bmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}, \quad \mathbf{J}^{(3)} = \begin{bmatrix} 0 & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & 0 & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & 0 & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & 0 & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & 0 & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}.$$

In the lower part of Table B.1.1, we give the changes of $I_i^{(r)}, J_i^{(r)}$ when the initial values u_1–u_6 are set to be 1–6.

B.2. Transfer matrix family

In this section, we prove the properties of transfer matrix family $\mathscr{F}$ defined in (13). The transfer matrices $\mathbf{W}, \mathbf{W}_g$ defined in (10) and (11) are in this family. We show the useful properties of the matrix family in the following lemmas.

Lemma B.1. *The matrix family $\mathscr{F}$ is closed under matrix multiplication. If $\mathbf{W}, \mathbf{V} \in \mathscr{F}$, then $\mathbf{WV} \in \mathscr{F}$. Thus the family $\mathscr{F}$ is a semigroup under matrix multiplication.*

Proof. Let

$$\mathbf{W} = \begin{bmatrix} c_w \mathbf{W}_{1,1} & (1-c_w)\mathbf{W}_{1,2} \\ d_w \mathbf{W}_{2,1} & (1-d_w)\mathbf{W}_{2,2} \end{bmatrix}, \quad \mathbf{V} = \begin{bmatrix} c_v \mathbf{V}_{1,1} & (1-c_v)\mathbf{V}_{1,2} \\ d_v \mathbf{V}_{2,1} & (1-d_v)\mathbf{V}_{2,2} \end{bmatrix}.$$

Here $0 \le c_w, c_v, d_w, d_v \le 1$, $\mathbf{W}_{i,j}, \mathbf{V}_{i,j}$ are doubly stochastic, $i, j \in \{1, 2\}$.

$$\mathbf{WV} = \begin{bmatrix} c_w c_v \mathbf{W}_{1,1}\mathbf{V}_{1,1} + (1-c_w)d_v \mathbf{W}_{1,2}\mathbf{V}_{2,1} & c_w(1-c_v)\mathbf{W}_{1,1}\mathbf{V}_{1,2} + (1-c_w)(1-d_v)\mathbf{W}_{1,2}\mathbf{V}_{2,2} \\ d_w c_v \mathbf{W}_{2,1}\mathbf{V}_{1,1} + (1-d_w)d_v \mathbf{W}_{2,2}\mathbf{V}_{2,1} & d_w(1-c_v)\mathbf{W}_{2,1}\mathbf{V}_{1,2} + (1-d_w)(1-d_v)\mathbf{W}_{2,2}\mathbf{V}_{2,2} \end{bmatrix}$$

Consider the first block, if $c_w c_v + (1-c_w)d_w = 0$,

$$c_w c_v \mathbf{W}_{1,1}\mathbf{V}_{1,1} + (1-c_w)d_v \mathbf{W}_{1,2}\mathbf{V}_{2,1} = \mathbf{0}_n = (c_w c_v + (1-c_w)d_w)\mathbf{I}_n$$

If $c_w c_v + (1-c_w)d_w > 0$,

$$c_w c_v \mathbf{W}_{1,1}\mathbf{V}_{1,1} + (1-c_w)d_v \mathbf{W}_{1,2}\mathbf{V}_{2,1} = (c_w c_v + (1-c_w)d_v)\left(\frac{c_w c_v}{c_w c_v + (1-c_w)d_v}\mathbf{W}_{1,1}\mathbf{V}_{1,1} + \frac{(1-c_w)d_v}{c_w c_v + (1-c_w)d_v}\mathbf{W}_{1,2}\mathbf{V}_{2,1}\right).$$

The product of two doubly stochastic matrix is doubly stochastic. The convex combination of doubly stochastic matrices is still doubly stochastic. So, we can always find a doubly stochastic matrix $\mathbf{U}_{1,1}$ such that

$$c_w c_v \mathbf{W}_{1,1}\mathbf{V}_{1,1} + (1-c_w)d_v \mathbf{W}_{1,2}\mathbf{V}_{2,1} = (c_w c_v + (1-c_w)d_w)\mathbf{U}_{1,1}.$$

Similarly, we can find doubly stochastic matrices $\mathbf{U}_{i,j}$, $i, j \in \{1, 2\}$, such that

$$\mathbf{WV} = \begin{bmatrix} (c_w c_v + (1-c_w)d_v)\mathbf{U}_{1,1} & (c_w(1-c_v) + (1-c_w)(1-d_v))\mathbf{U}_{1,2} \\ (d_w c_v + (1-d_w)d_v)\mathbf{U}_{2,1} & (d_w(1-c_v) + (1-d_w)(1-d_v))\mathbf{U}_{2,2} \end{bmatrix} \in \mathscr{F}.$$

□

Lemma B.2. *For any* $\mathbf{W} \in \mathscr{F}$, $\|\mathbf{W}\| \leq \sqrt{2}$.

Proof. Let

$$\mathbf{W} = \begin{bmatrix} c\mathbf{W}_{1,1} & (1-c)\mathbf{W}_{1,2} \\ d\mathbf{W}_{2,1} & (1-d)\mathbf{W}_{2,2} \end{bmatrix}.$$

Here $0 \leq c, d \leq 1$, $\mathbf{W}_{i,j}, \mathbf{V}_{i,j}$ are doubly stochastic, $i, j \in \{1, 2\}$. For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$,

$$\begin{aligned}
\left\| \mathbf{W} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \right\|^2 &= \left\| \begin{bmatrix} c\mathbf{W}_{1,1}\mathbf{x} + (1-c)\mathbf{W}_{1,2}\mathbf{y} \\ d\mathbf{W}_{2,1}\mathbf{x} + (1-d)\mathbf{W}_{2,2}\mathbf{y} \end{bmatrix} \right\|^2 \\
&= \|c\mathbf{W}_{1,1}\mathbf{x} + (1-c)\mathbf{W}_{1,2}\mathbf{y}\|^2 + \|d\mathbf{W}_{2,1}\mathbf{x} + (1-d)\mathbf{W}_{2,2}\mathbf{y}\|^2 \\
&\overset{(a)}{\leq} c\|\mathbf{W}_{1,1}\mathbf{x}\|^2 + (1-c)\|\mathbf{W}_{1,2}\mathbf{y}\|^2 + d\|\mathbf{W}_{2,1}\mathbf{x}\|^2 + (1-d)\|\mathbf{W}_{2,2}\mathbf{y}\|^2 \\
&\overset{(b)}{\leq} c\|\mathbf{x}\|^2 + (1-c)\|\mathbf{y}\|^2 + d\|\mathbf{x}\|^2 + (1-d)\|\mathbf{y}\|^2 \\
&\leq 2(\|\mathbf{x}\|^2 + \|\mathbf{y}\|^2).
\end{aligned}$$

In the above inequalities, (a) holds because of Jensen's inequality; (b) holds because the norm of a doubly stochastic matrix is bounded by 1. □

Lemma B.3. *For any* $\mathbf{W} \in \mathscr{F}$,

$$\mathbf{W} \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \mathbf{0}_n & \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \end{bmatrix} = \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \mathbf{0}_n & \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \end{bmatrix} \mathbf{W}.$$

Proof. Let

$$\mathbf{W} = \begin{bmatrix} c\mathbf{W}_{1,1} & (1-c)\mathbf{W}_{1,2} \\ d\mathbf{W}_{2,1} & (1-d)\mathbf{W}_{2,2} \end{bmatrix}.$$

Here $0 \leq c, d \leq 1$, $\mathbf{W}_{i,j}, \mathbf{V}_{i,j}$ are doubly stochastic, $i, j \in \{1, 2\}$. We have

$$\begin{aligned}
\mathbf{W} \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \mathbf{0}_n & \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \end{bmatrix} &= \begin{bmatrix} c\mathbf{W}_{1,1}\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & (1-c)\mathbf{W}_{1,2}\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \\ d\mathbf{W}_{2,1}\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & (1-d)\mathbf{W}_{2,2}\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \end{bmatrix} \\
&= \begin{bmatrix} c\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & (1-c)\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \\ d\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & (1-d)\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \end{bmatrix} \\
&= \begin{bmatrix} c\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top\mathbf{W}_{1,1} & (1-c)\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top\mathbf{W}_{1,2} \\ d\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top\mathbf{W}_{2,1} & (1-d)\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top\mathbf{W}_{2,2} \end{bmatrix} \\
&= \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \mathbf{0}_n & \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \end{bmatrix} \mathbf{W}.
\end{aligned}$$

□

Lemma B.4 (matrix consensus)**.** *In the 2-port system. When the communication matrix* $\mathbf{P}^{(k)}$ *is sampled from* (5) *or* (6) *according to* δ_r*, and the mixing matrix* $\mathbf{W}^{(k)}$ *is sampled from* (10) *or* (11) *according to* δ_r*. The product of matrices* $\mathbf{W}^{(k)}$ *satisfies*

$$\prod_{k=0}^{\tau-1} \mathbf{W}^{(k)} = \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \frac{1}{n-1}(\mathbb{1}_n\mathbb{1}_n^\top - \mathbf{I}_n) & \mathbf{0}_n \end{bmatrix}, \tag{16}$$

$$\prod_{k=0}^{t} \mathbf{W}^{(k)} = \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \end{bmatrix}, \quad \forall t \geq \tau. \tag{17}$$

Proof. Since $\delta_0 = 1, n_0 = 0$, we have

$$\mathbf{W}^{(0)} = \begin{bmatrix} \frac{1}{2}\mathbf{I}_n + \frac{1}{2}\mathbf{P}^{(0)} & \mathbf{0}_n \\ \mathbf{P}^{(0)} & \mathbf{0}_n \end{bmatrix}.$$

So, for any $s \geq 0$, $\prod_{k=0}^{s} \mathbf{W}^{(k)}$ is of the form

$$\begin{bmatrix} * & \mathbf{0}_n \\ * & \mathbf{0}_n \end{bmatrix}.$$

Here $*$ stands for an undetermined $n \times n$ matrix block. By induction, for $s \leq \tau$,

$$\prod_{k=0}^{s} \mathbf{W}^{(k)} = \begin{bmatrix} \mathbf{A} & \mathbf{0}_n \\ \mathbf{B} & \mathbf{0}_n \end{bmatrix},$$

$$\mathbf{A}_{i,j} = \begin{cases} \frac{1}{n_{s+1}+1}, & (i-j) \in \{0, 1, 2, \ldots, n_{s+1}\} \pmod n \\ 0, & \text{otherwise} \end{cases}, \quad \mathbf{B}_{i,j} = \begin{cases} \frac{1}{n_{s+1}}, & (i-j) \in \{1, 2, \ldots, n_{s+1}\} \pmod n \\ 0, & \text{otherwise} \end{cases}.$$

When $s = \tau - 1$, we get (16). For $t \geq \tau$,

$$\begin{aligned}
\prod_{k=0}^{t} \mathbf{W}^{(k)} &= \left(\prod_{k=\tau+1}^{t} \mathbf{W}^{(k)} \right) \cdot \mathbf{W}^{(\tau)} \cdot \left(\prod_{k=0}^{\tau-1} \mathbf{W}^{(k)} \right) \\
&= \left(\prod_{k=\tau+1}^{t} \mathbf{W}^{(k)} \right) \cdot \begin{bmatrix} \frac{1}{2}\mathbf{I}_n + \frac{1}{2}\mathbf{P}^{(0)} & \mathbf{0}_n \\ \mathbf{P}^{(0)} & \mathbf{0}_n \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \frac{1}{n-1}(\mathbb{1}_n\mathbb{1}_n^\top - \mathbf{I}_n) & \mathbf{0}_n \end{bmatrix} \\
&= \left(\prod_{k=\tau+1}^{t} \mathbf{W}^{(k)} \right) \cdot \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \end{bmatrix} \\
&\overset{(a)}{=} \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \end{bmatrix}.
\end{aligned}$$

Here, $\left(\prod_{k=\tau+1}^{t} \mathbf{W}^{(k)}\right) \in \mathscr{F}$ because $\mathscr{F}$ is a semigroup according to Lemma B.1, any matrix in $\mathscr{F}$ is row stochastic, so (a) holds. We get (17). □

B.3. Convex analysis tools

In this section, we list some convex analysis concepts and inequalities useful in the algorithm convergence analysis.

Definition B.5 (L-smoothness)**.** A differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is called L-smooth if for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$, we have

$$\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|. \tag{18}$$

For an L-smooth function f, we have the following inequality,

$$f(\boldsymbol{x}) \leq f(\boldsymbol{y}) + \langle \nabla f(\boldsymbol{y}), \boldsymbol{x} - \boldsymbol{y} \rangle + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{y}\|^2, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d. \tag{19}$$

Definition B.6 (Convexity)**.** We call a function $f : \mathbb{R}^d \to \mathbb{R}$ convex if for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$, we have

$$f(\boldsymbol{x}) \geq f(\boldsymbol{y}) + \langle \nabla f(\boldsymbol{y}), \boldsymbol{x} - \boldsymbol{y} \rangle. \tag{20}$$

Given a function f, let $\mathbf{x}^*$ be the minimizer of f. If f is both L-smooth and convex, we have

$$\|\nabla f(\mathbf{x})\|^2 \leq 2L(f(\mathbf{x}) - f(\mathbf{x}^*)), \quad \forall \mathbf{x} \in \mathbb{R}^d. \tag{21}$$

B.4. Convergence property of DSGD-CECA

In this section, we provide the convergence proof for DSGD-CECA-2P, as presented in Algorithm 1. DSGD-CECA-1P differs from DSGD-CECA-2P only in the sampling of the communication matrix $\mathbf{P}^{(k)}$. Their convergence proofs follow a similar approach. For simplicity, we use DSGD-CECA to refer to DSGD-CECA-2P in this section. To establish the convergence property of DSGD-CECA, we first present several supporting lemmas.

Lemma B.7 (RELATION BETWEEN AVERAGED VARIABLES)**.** *Consider DSGD-CECA-2P recursions in Algorithm 1. It holds for any $k = 0, 1, 2, \cdots$ that $\bar{\mathbf{x}}^{(k)} = \bar{\mathbf{y}}^{(k)} = \bar{\mathbf{z}}^{(k)}$ and*

$$\bar{\mathbf{x}}^{(k+1)} = \bar{\mathbf{x}}^{(k)} - \gamma \bar{\mathbf{e}}^{(k)},$$

where $\bar{\mathbf{e}}^{(k)} = \frac{1}{n}\nabla F(\mathbf{z}^{(k)}; \boldsymbol{\xi}^{(k)})^\top \mathbb{1}_n$.

Proof. Left-multiplying $\frac{1}{n}\mathbb{1}_n^\top$ to both sides of (8), (9), we get

$$\bar{\mathbf{x}}^{(k+1)} = a^{(k)}(\bar{\mathbf{x}}^{(k)} - \gamma \bar{\mathbf{e}}^{(k)}) + (1 - a^{(k)})(\bar{\mathbf{z}}^{(k)} - \gamma \bar{\mathbf{e}}^{(k)}), \tag{22}$$

$$\bar{\mathbf{y}}^{(k+1)} = b^{(k)}(\bar{\mathbf{y}}^{(k)} - \gamma \bar{\mathbf{e}}^{(k)}) + (1 - b^{(k)})(\bar{\mathbf{z}}^{(k)} - \gamma \bar{\mathbf{e}}^{(k)}). \tag{23}$$

The difference between (22) and (23) is

$$\bar{\mathbf{x}}^{(k+1)} - \bar{\mathbf{y}}^{(k+1)} = \theta^{(k)}(\bar{\mathbf{x}}^{(k)} - \bar{\mathbf{y}}^{(k)}),$$

where $\theta^{(k)} = b^{(k)}$ if $\bar{\mathbf{z}}^{(k)} = \bar{\mathbf{x}}^{(k)}$ when $\delta_r = 1$, and $\theta^{(k)} = a^{(k)}$ if $\bar{\mathbf{z}}^{(k)} = \bar{\mathbf{y}}^{(k)}$ when $\delta_r = 0$. Since $\mathbf{x}^{(0)} = \mathbf{y}^{(0)}$ in the initial setting, we have $\bar{\mathbf{x}}^{(0)} = \bar{\mathbf{y}}^{(0)}$. Inductively, we have $\bar{\mathbf{x}}^{(k)} = \bar{\mathbf{y}}^{(k)} = \bar{\mathbf{z}}^{(k)}$, for any $k \geq 0$. Putting this result to (22), we have

$$\bar{\mathbf{x}}^{(k+1)} = a^{(k)}\bar{\mathbf{x}}^{(k)} + (1 - a^{(k)})\bar{\mathbf{z}}^{(k)} - \gamma \bar{\mathbf{e}}^{(k)} = \bar{\mathbf{x}}^{(k)} - \gamma \bar{\mathbf{e}}^{(k)}.$$

□

Lemma B.8 (DESCENT LEMMA)**.** *Under Assumptions 2.1–2.2 and step-size $\gamma < \frac{1}{4L}$, it holds for $k = 0, 1, \cdots$ that*

$$\begin{aligned}\mathbb{E} f(\bar{\mathbf{x}}^{(k+1)}) \leq \mathbb{E} f(\bar{\mathbf{x}}^{(k)}) - \frac{\gamma}{4}\mathbb{E}\|\nabla f(\bar{\mathbf{z}}^{(k)})\|^2 + \frac{\gamma^2 L \sigma^2}{2n} \\ + \frac{3\gamma L^2}{4n}\Big(\mathbb{E}\|\mathbf{x}^{(k)} - \mathbb{1}_n(\bar{\mathbf{x}}^{(k)})^\top\|_F^2 + \mathbb{E}\|\mathbf{y}^{(k)} - \mathbb{1}_n(\bar{\mathbf{y}}^{(k)})^\top\|_F^2\Big).\end{aligned} \tag{24}$$

Proof. We first introduce the following filtration to simplify the analysis

$$\mathcal{F}^{(k)} = \sigma\{\mathbf{x}^{(0)}, \boldsymbol{\xi}^{(0)}, \boldsymbol{\xi}^{(1)}, \boldsymbol{\xi}^{(2)}, \ldots, \boldsymbol{\xi}^{(k-1)}\}. \tag{25}$$

It is the σ algebra of all random variables before the kth iteration. With $\bar{\mathbf{x}}^{(k+1)} = \bar{\mathbf{x}}^{(k)} - \gamma \bar{\mathbf{e}}^{(k)}$ and Assumption 2.1, it holds that

$$\begin{aligned}\mathbb{E}[f(\bar{\mathbf{x}}^{(k+1)})|\mathcal{F}^{(k-1)}] &\leq f(\bar{\mathbf{x}}^{(k)}) - \gamma\mathbb{E}[\langle \nabla f(\bar{\mathbf{x}}^{(k)}), \bar{\mathbf{e}}^{(k)}\rangle|\mathcal{F}^{(k-1)}] + \frac{L\gamma^2}{2}\mathbb{E}[\|\bar{\mathbf{e}}^{(k)}\|^2|\mathcal{F}^{(k-1)}] \\ &= f(\bar{\mathbf{x}}^{(k)}) - \gamma\langle \nabla f(\bar{\mathbf{x}}^{(k)}), \frac{1}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{z}_i^{(k)})\rangle + \frac{L\gamma^2}{2}\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{z}_i^{(k)})\|^2 + \frac{\gamma^2 L\sigma^2}{2n}\end{aligned} \tag{26}$$

Note that

$$\begin{aligned}-\langle \nabla f(\bar{\mathbf{x}}^{(k)}), \frac{\gamma}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{z}_i^{(k)})\rangle &= -\langle \nabla f(\bar{\mathbf{z}}^{(k)}), \frac{\gamma}{n}\sum_{i=1}^n [\nabla f_i(\boldsymbol{z}_i^{(k)}) - \nabla f_i(\bar{\mathbf{z}}^{(k)}) + \nabla f_i(\bar{\mathbf{z}}^{(k)})]\rangle \\ &\leq -\gamma\|\nabla f(\bar{\mathbf{z}}^{(k)})\|^2 + \frac{\gamma}{2}\|\nabla f(\bar{\mathbf{z}}^{(k)})\|^2 + \frac{\gamma}{2n}\sum_{i=1}^n \|\nabla f_i(\boldsymbol{z}_i^{(k)}) - \nabla f_i(\bar{\mathbf{z}}^{(k)})\|^2 \\ &\leq -\frac{\gamma}{2}\|\nabla f(\bar{\mathbf{z}}^{(k)})\|^2 + \frac{\gamma L^2}{2n}\|\mathbf{z}^{(k)} - \mathbb{1}_n(\bar{\mathbf{z}}^{(k)})^\top\|_F^2,\end{aligned} \tag{27}$$

where the first equality holds because $\bar{\mathbf{z}}^{(k)} = \bar{\mathbf{x}}^{(k)}$ for any $k = 0, 1, 2, \cdots$. Furthermore, it also holds that

$$\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{z}_i^{(k)})\|^2 \leq \frac{2L^2}{n}\|\mathbf{z}^{(k)} - \mathbb{1}_n(\bar{\mathbf{z}}^{(k)})^\top\|_F^2 + 2\|\nabla f(\bar{\mathbf{z}}^{(k)})\|^2 \tag{28}$$

and

$$\|\mathbf{z}^{(k)} - \mathbb{1}_n(\bar{\mathbf{z}}^{(k)})^\top\|_F^2 \leq \|\mathbf{x}^{(k)} - \mathbb{1}_n(\bar{\mathbf{x}}^{(k)})^\top\|_F^2 + \|\mathbf{y}^{(k)} - \mathbb{1}_n(\bar{\mathbf{y}}^{(k)})^\top\|_F^2 \tag{29}$$

Substituting (27), (28) and (29) into (26), taking expectations over $\mathcal{F}^{(k)}$, and using the fact that $\gamma < \frac{1}{4L}$, we reach (24). □

To bound the term $\mathbb{E}\|\mathbf{x}^{(k)} - \mathbb{1}_n(\bar{\mathbf{x}}^{(k)})^\top\|_F^2 + \mathbb{E}\|\mathbf{y}^{(k)} - \mathbb{1}_n(\bar{\mathbf{y}}^{(k)})^\top\|_F^2$ in (24), we introduce the auxiliary variables

$$\boldsymbol{\varphi}^{(k)} = \begin{bmatrix} \mathbf{x}^{(k)} - \mathbb{1}_n(\bar{\mathbf{x}}^{(k)})^\top \\ \mathbf{y}^{(k)} - \mathbb{1}_n(\bar{\mathbf{y}}^{(k)})^\top \end{bmatrix} \in \mathbb{R}^{2n\times d}, \quad \boldsymbol{\psi}^{(k)} = \begin{bmatrix} \mathbf{g}^{(k)} - \mathbb{1}_n(\bar{\mathbf{g}}^{(k)})^\top \\ \mathbf{h}^{(k)} - \mathbb{1}_n(\bar{\mathbf{h}}^{(k)})^\top \end{bmatrix} \in \mathbb{R}^{2n\times d}.$$

It holds that $\|\boldsymbol{\varphi}^{(k)}\|_F^2 = \|\mathbf{x}^{(k)} - \mathbb{1}_n(\bar{\mathbf{x}}^{(k)})^\top\|_F^2 + \|\mathbf{y}^{(k)} - \mathbb{1}_n(\bar{\mathbf{y}}^{(k)})^\top\|_F^2$.

Lemma B.9 (CONSENSUS LEMMA)**.** *Under Assumptions 2.1-2.3, if the learning rate $\gamma \le \frac{1}{8\tau L}$, it holds that*

$$\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2 \le \frac{2}{T+1}\sum_{k=0}^{\tau}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2 + 32n\tau^2\gamma^2(\sigma^2+2b^2) \tag{30}$$

Proof. We prove (30) in three steps.

Step I. In this step, we will provide a rough upper bound to $\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2$. By left-multiplying both sides of (12) by

$$\mathbf{I}_{2n} - \begin{bmatrix} \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top & \mathbf{0}_n \\ \mathbf{0}_n & \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top \end{bmatrix}$$

and utilizing the commutativity property proved in Lemma B.3, we have

$$\boldsymbol{\varphi}^{(k+1)} = \mathbf{W}^{(k)}\boldsymbol{\varphi}^{(k)} - \gamma\mathbf{W}_g^{(k)}\boldsymbol{\psi}^{(k)}. \tag{31}$$

For any $k \ge \tau$, we let $m = \lfloor k/\tau \rfloor - 1$ and hence $k - m\tau \ge \tau$. Keep iterating (31), we have for any $k \ge \tau$ that

$$\begin{aligned}
\boldsymbol{\varphi}^{(k)} &= \mathbf{W}^{(k-1)}\boldsymbol{\varphi}^{(k-1)} - \gamma\mathbf{W}_g^{(k-1)}\boldsymbol{\psi}^{(k-1)} \\
&\overset{(a)}{=} \left(\prod_{j=m\tau}^{k-1}\mathbf{W}^{(j)}\right)\boldsymbol{\varphi}^{(m\tau)} - \gamma\sum_{l=1}^{k-m\tau}\left(\prod_{j=k-l+1}^{k-1}\mathbf{W}^{(j)}\right)\mathbf{W}_g^{(k-l)}\boldsymbol{\psi}^{(k-l)} \\
&\overset{(b)}{=} -\gamma\sum_{l=1}^{k-m\tau}\left(\prod_{j=k-l+1}^{k-1}\mathbf{W}^{(j)}\right)\mathbf{W}_g^{(k-l)}\boldsymbol{\psi}^{(k-l)}.
\end{aligned}$$

where in equality (a) we define $\prod_{j=p}^{q}\mathbf{W}^{(j)} = I$ if $p > q$, and (b) holds by applying (17). The above equality leads to

$$\begin{aligned}
\|\boldsymbol{\varphi}^{(k)}\|_F^2 &\overset{(a)}{\le} \gamma^2(k-m\tau)\sum_{l=1}^{k-m\tau}\left\|\left(\prod_{j=k-l+1}^{k-1}\mathbf{W}^{(j)}\right)\mathbf{W}_g^{(k-l)}\boldsymbol{\psi}^{(k-l)}\right\|_F^2 \\
&\overset{(b)}{\le} 2\tau\gamma^2\sum_{l=1}^{k-m\tau}\left\|\left(\prod_{j=k-l+1}^{k-1}\mathbf{W}^{(j)}\right)\mathbf{W}_g^{(k-l)}\boldsymbol{\psi}^{(k-l)}\right\|_F^2 \\
&\overset{(c)}{\le} 4\tau\gamma^2\sum_{l=1}^{k-m\tau}\left\|\boldsymbol{\psi}^{(k-l)}\right\|_F^2.
\end{aligned}$$

Here, inequality (a) holds due to Jensen's inequality, (b) holds since $k - m\tau \le 2\tau$, and (c) holds due to the fact that $\left(\prod_{j=k-l+1}^{k-1}\mathbf{W}^{(j)}\right)\mathbf{W}_g^{(k-l)} \in \mathscr{F}$ and Lemma B.2. Summing up the above inequality over $k = \tau, \tau+1, \ldots, T$ and dividing it by $T+1$, we get

$$\frac{1}{T+1}\sum_{k=\tau}^{T}\|\boldsymbol{\varphi}^{(k)}\|_F^2 \le 4\tau\gamma^2\frac{1}{T+1}\sum_{k=\tau}^{T}\sum_{l=1}^{k-m\tau}\left\|\boldsymbol{\psi}^{(k-l)}\right\|_F^2 \le 8\tau^2\gamma^2\frac{1}{T+1}\sum_{k=0}^{T}\left\|\boldsymbol{\psi}^{(k)}\right\|_F^2. \tag{32}$$

Adding $\frac{1}{T+1}\sum_{k=0}^{\tau-1}\|\boldsymbol{\varphi}^{(k)}\|_F^2$ to the both sides of (32), then taking expectations to both sides of the above inequality, we have

$$\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2 \le \frac{1}{T+1}\sum_{k=0}^{\tau}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2 + 8\tau^2\gamma^2\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\left\|\boldsymbol{\psi}^{(k)}\right\|_F^2 \tag{33}$$

Step II. In this step, we derive the bound on the term $\frac{1}{T+1}\sum_{k=0}^{T}\|\boldsymbol{\psi}^{(k)}\|_F^2$. Recall that $\|\boldsymbol{\psi}^{(k)}\|_F^2 = \|\mathbf{g}^{(k)} - \mathbb{1}_n(\bar{\mathbf{g}}^{(k)})^\top\|_F^2 + \|\mathbf{h}^{(k)} - \mathbb{1}_n(\bar{\mathbf{h}}^{(k)})^\top\|_F^2$. With $\mathcal{F}^{(k-1)}$ defined in (25), we have

$$
\begin{aligned}
&\mathbb{E}[\|\mathbf{g}^{(k)} - \mathbb{1}_n(\bar{\mathbf{g}}^{(k)})^\top\|_F^2|\mathcal{F}^{(k-1)}] \\
&= \mathbb{E}\left[\left\|(\mathbf{I}_n - \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top)\mathbf{g}^{(k)}\right\|_F^2 \middle| \mathcal{F}^{(k-1)}\right] \\
&= \mathbb{E}\left[\left\|(\mathbf{I}_n - \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top)(\mathbf{g}^{(k)} - \nabla F(\mathbf{x}^{(k)}) + \nabla F(\mathbf{x}^{(k)}))\right\|_F^2 \middle| \mathcal{F}^{(k-1)}\right] \\
&\overset{(a)}{=} \mathbb{E}\left[\|(\mathbf{I}_n - \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top)(\mathbf{g}^{(k)} - \nabla F(\mathbf{x}^{(k)}))\|_F^2 \middle| \mathcal{F}^{(k-1)}\right] + \|(\mathbf{I}_n - \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top)\nabla F(\mathbf{x}^{(k)})\|_F^2 \\
&\leq \mathbb{E}\left[\|\mathbf{g}^{(k)} - \nabla F(\mathbf{x}^{(k)})\|_F^2 \middle| \mathcal{F}^{(k-1)}\right] + \|(\mathbf{I}_n - \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top)(\nabla F(\mathbf{x}^{(k)}) - \nabla F(\bar{\mathbf{x}}^{(k)}) + \nabla F(\bar{\mathbf{x}}^{(k)}))\|_F^2 \\
&\overset{(b)}{\leq} n\sigma^2 + 2\|(\mathbf{I}_n - \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top)(\nabla F(\mathbf{x}^{(k)}) - \nabla F(\bar{\mathbf{x}}^{(k)}))\|_F^2 + 2\|(\mathbf{I}_n - \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top)(\nabla F(\bar{\mathbf{x}}^{(k)}))\|_F^2 \\
&\leq n\sigma^2 + 2\|\nabla F(\mathbf{x}^{(k)}) - \nabla F(\bar{\mathbf{x}}^{(k)})\|_F^2 + 2\|(\mathbf{I}_n - \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top)(\nabla F(\bar{\mathbf{x}}^{(k)}))\|_F^2 \\
&\overset{(c)}{\leq} n\sigma^2 + 2L^2\|\mathbf{x}^{(k)} - \mathbb{1}_n(\bar{\mathbf{x}}^{(k)})^\top\|_F^2 + 2nb^2.
\end{aligned}
$$

In the above inequality, (a) holds because of Assumption 2.2, (b) holds because of Assumption 2.2 and Jensen's inequality, and (c) holds due to Assumption 2.1 and Assumption 2.3. By taking expectation over $\mathcal{F}^{(k-1)}$, we have

$$
\mathbb{E}\|\mathbf{g}^{(k)} - \mathbb{1}_n(\bar{\mathbf{g}}^{(k)})^\top\|_F^2 \leq 2L^2\mathbb{E}\|\mathbf{x}^{(k)} - \mathbb{1}_n(\bar{\mathbf{x}}^{(k)})^\top\|_F^2 + n\sigma^2 + 2nb^2.
$$

A similar bound can also be derived for $\mathbb{E}\|\mathbf{h}^{(k)} - \mathbb{1}_n(\bar{\mathbf{h}}^{(k)})^\top\|_F^2$. As a result, we achieve

$$
\mathbb{E}\|\boldsymbol{\psi}^{(k)}\|_F^2 \leq 2n\sigma^2 + 4nb^2 + 2L^2\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2.
$$

Summing up the inequality for $k = 0, 1, 2, \ldots, T$ and then dividing the result by $T+1$, we get

$$
\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\boldsymbol{\psi}^{(k)}\|_F^2 \leq 2n\sigma^2 + 4nb^2 + 2L^2\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2. \tag{34}
$$

Step III. In this step, we will derive (30) based on (33) and (34). Substituting (34) to (33), we get

$$
\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2 \leq \frac{1}{T+1}\sum_{k=0}^{\tau}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2 + \frac{16\tau^2\gamma^2L^2}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2 + 16n\tau^2\gamma^2(\sigma^2 + 2b^2) \tag{35}
$$

When $\gamma \leq \frac{1}{8\tau L}$, it holds that $16\tau^2L^2\gamma^2 \leq \frac{1}{2}$. Regrouping terms associated with $\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2$, we get (30). □

With Lemmas B.8 and B.9, we can establish the following convergence property of DSGD-CECA.

Theorem B.10 (CONVERGENCE PROPERTY)**.** *Suppose Assumptions 2.1-2.3 hold and $\boldsymbol{x}_i^{(k)} = \bar{\mathbf{x}}^{(k)}$, $\boldsymbol{y}_i^{(k)} = \bar{\mathbf{y}}^{(k)}$ for $k < \tau$ by conducting global averaging in the first τ iterations. If learning rate γ satisfies*

$$
\gamma = \frac{1}{\left(\frac{2n\Delta}{L\sigma^2(T+1)}\right)^{-\frac{1}{2}} + \left(\frac{\Delta}{24L^2\tau^2(\sigma^2+2b^2)(T+1)}\right)^{-\frac{1}{3}} + 8\tau L} \tag{36}
$$

where $\Delta = \mathbb{E}f(\bar{\mathbf{x}}^0) - f^\star$, then DSGD-CECA will converge as follows

$$
\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\nabla f(\bar{\mathbf{x}}^{(k)})\|^2 \leq 16\left(\frac{\Delta L\sigma^2}{n(T+1)}\right)^{\frac{1}{2}} + 24\left(\frac{\Delta^2L^2\tau^2(\sigma^2+2b^2)}{(T+1)^2}\right)^{\frac{1}{3}} + \frac{32\tau\Delta L}{T+1} \tag{37}
$$

Proof. Recall that $\|\boldsymbol{\varphi}^{(k)}\|_F^2 = \|\mathbf{x}^{(k)} - \mathbb{1}_n(\bar{\mathbf{x}}^{(k)})^\top\|_F^2 + \|\mathbf{y}^{(k)} - \mathbb{1}_n(\bar{\mathbf{y}}^{(k)})^\top\|_F^2$. By averaging (24) over $k = 0, 1, \cdots, T$, we have

$$\begin{aligned}\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\nabla f(\bar{\mathbf{z}}^{(k)})\|^2 &\leq \frac{4(\mathbb{E}f(\bar{\mathbf{x}}^0) - f^\star)}{\gamma(T+1)} + \frac{3L^2}{n(T+1)}\sum_{k=0}^{T}\mathbb{E}\|\boldsymbol{\varphi}^{(k)}\|_F^2 + \frac{2\gamma L\sigma^2}{n} \\ &\leq \frac{4(\mathbb{E}f(\bar{\mathbf{x}}^0) - f^\star)}{\gamma(T+1)} + 96L^2\tau^2\gamma^2(\sigma^2 + 2b^2) + \frac{2\gamma L\sigma^2}{n},\end{aligned} \tag{38}$$

where the last inequality holds because of inequality (30) and the fact that $\boldsymbol{x}_i^{(k)} = \bar{\mathbf{x}}^{(k)}$, $\boldsymbol{y}_i^{(k)} = \bar{\mathbf{y}}^{(k)}$ for $k < \tau$ due to the global averaging in the first τ iterations. We next let $\Delta := \mathbb{E}f(\bar{\mathbf{x}}^0) - f^\star$ and define

$$\gamma_1 = \left(\frac{2n\Delta}{L\sigma^2(T+1)}\right)^{\frac{1}{2}}, \quad \gamma_2 = \left(\frac{\Delta}{24L^2\tau^2(\sigma^2+2b^2)(T+1)}\right)^{\frac{1}{3}}. \tag{39}$$

If we set

$$\gamma = \frac{1}{\gamma_1^{-1} + \gamma_2^{-1} + 8\tau L},$$

it holds that $\gamma \leq \min\{\gamma_1, \gamma_2, \frac{1}{8\tau L}, \frac{1}{4L}\}$. Substituting the above γ to (38), we achieve

$$\begin{aligned}\frac{1}{T+1}\sum_{k=0}^{T}\mathbb{E}\|\nabla f(\bar{\mathbf{z}}^{(k)})\|^2 &\leq \frac{4\Delta}{T+1}(\gamma_1^{-1} + \gamma_2^{-1} + 8\tau L) + 96L^2\tau^2\gamma_2^2(\sigma^2 + 2b^2) + \frac{2\gamma_1 L\sigma^2}{n} \\ &= 16\left(\frac{\Delta L\sigma^2}{n(T+1)}\right)^{\frac{1}{2}} + 24\left(\frac{\Delta^2L^2\tau^2(\sigma^2+2b^2)}{(T+1)^2}\right)^{\frac{1}{3}} + \frac{32\tau\Delta L}{T+1}.\end{aligned} \tag{40}$$

Substituting $\bar{\mathbf{x}}^{(k)} = \bar{\mathbf{z}}^{(k)}$ to the above inequality (see Lemma B.7), we achieve (37). □

C. Additional experiments

CIFAR-10: We utilize the ResNet-18 model (He et al., 2016) implemented by (Liu, 2021). Similar to the MNIST experiments, we employ BlueFog for decentralized training using 5 NVIDIA GeForce RTX 2080 GPUs. The training process consists of 130 epochs without momentum, with a weight decay of 10^{-4}. A local batch size of 64 is used, and the base learning rate is set to 0.01. The learning rate is reduced by a factor of 10 at the 50th, 100th, and 120th epochs. Data augmentation is performed similarly to the method described in the work (Liu, 2021). Please refer to Fig. 10 for a comparison of the test accuracy between O.-P. Exp and DSGD-CECA-2P. It is noteworthy that DSGD-CECA-2P outperforms O.-P. Exp in terms of test accuracy.

Table 4. Comparison of test accuracy(%) with O.-P. Exp and DSGD-CECA-2P over MNIST and CIFAR-10 datasets.

Topology	MNIST Test Acc.	CIFAR-10 Test Acc.
O.-P. Exp.	98.33	90.99
DSGD-CECA-2P	**98.50**	**92.07**

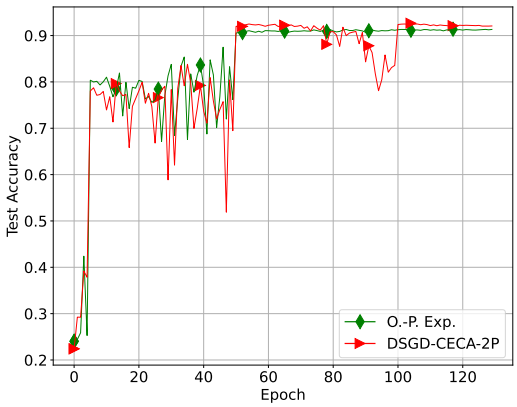

Figure 10. Test accuracy of O.-P. Exp and DSGD-CECA-2P algorithms for ResNet-18 on CIFAR-10.