Decentralized Asynchronous Non-convex Stochastic Optimization on Directed Graphs

Vyacheslav Kungurtsev, Mahdi Morafah, Tara Javidi *Member, IEEE,* Gesualdo Scutari *Member, IEEE,*

Abstract—**Distributed Optimization is an increasingly important subject area with the rise of multi-agent control and optimization. We consider a decentralized stochastic optimization problem where the agents on a graph aim to asynchronously optimize a collective (additive) objective function consisting of agents' individual (possibly non-convex) local objective functions. Each agent only has access to a noisy estimate of the gradient of its own function (one component of the sum of objective functions). We proposed an asynchronous distributed algorithm for such a class of problems. The algorithm combines stochastic gradients with tracking in an asynchronous push-sum framework and obtain the standard sublinear convergence rate for general non-convex functions, matching the rate of centralized stochastic gradient descent SGD. Our experiments on a non-convex image classification task using convolutional neural network validate the convergence of our proposed algorithm across different number of nodes and graph connectivity percentages.**

I. INTRODUCTION

In this paper we consider the global optimization problem,

$$\min_{\mathbf{x} \in \mathbb{R}^n} \mathbb{E}_\xi \left[F(\mathbf{x}, \xi) \right] \triangleq \sum_{i=1}^m \mathbb{E}_\xi \left[f_i(\mathbf{x}, \xi) \right] \tag{1}$$

where the agents' local objective functions $\mathbb{E}[f_i(\cdot, \xi)], i = 1, \ldots, m$ are smooth and generally nonconvex and known only locally to the m networked agents. In addition, each agent is assumed to only have access to noisy estimates of f_i and its gradients. Communication across the network is performed asynchronously in a gossip fashion, i.e., there is a (possibly) directed graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, and for each edge $e \in \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the vertices elements $\{i, j\} \in e$ implies that node i can send information to node j. Define $\mathcal{N}_i^{in} := \{j \in \mathcal{V} : (j, i) \in \mathcal{E}\}$ and $\mathcal{N}_i^{out} := \{j \in \mathcal{V} : (i, j) \in \mathcal{E}\}$.

We make the following assumption on the problem,

Assumption I.1. 1) *For a.e. ξ, $f_i(\cdot, \xi)$ is proper, closed, and L_i-Lipschitz continuously differentiable. Furthermore (for a.e. ξ) $F(\cdot, \xi)$ is bounded from below.*
2) *The (di)graph $\mathcal{G}$ is strongly connected.*

In this paper we uniquely address several concomitant challenges 1) the objective function is stochastic nonconvex 2) the sum-components of the objective function are known only locally to each agent 3) communication is fully asynchronous, modeled as each iteration consisting of a random activation

V. Kungurtsev are with Department of Computer Science, Czech Technical University in Prague

M. Morafah and T. Javidi are with the Department of Electrical Engineering, University of California-San Diego

G. Scutari is with the School of Industrial Engineering and the School of Electrical and Computer Engineering (ECE) at Purdue University

among the agents as well as delays in the communicated information and 4) the topology of the network is arbitrary (i.e., no hierarchical structure) and communication is *directed*, i.e., that agent i being able to send data to j does not necessarily imply that j can also send information to i.

Previous Works There are a number of works in the literature that consider *distributed* or *decentralized* stochastic optimization addressing a partial subset of these challenges.

A bulk of works consider distributing computation in a shared memory setting while allowing for asynchronous updating (and thus read and/or write lock-free) including the classic [1] and the seminal work [2], the general framework for block/coordinate parallel updates given in [3], and many thereafter. These, however, assume that every computing node has access to the entire function, or a noisy estimate thereof, rather than a component of it, and does not consider communication across an arbitrary network.

A standard structure for distributed optimization is the "hub-spoke", "parameter-server" or "master-slave" architecture. This is considered, for instance, in [4] and [5]. In this case, it is assumed that the nodes have a hierarchical structure of communication, with one node aggregating information and coordinating the computation to be performed across the other nodes, which communicate solely with the central node and not at all with each other. In this paper we consider a more general arbitrary graph topology modeling the communication links across agents.

Schemes that consider an arbitrary graph topology include [6] which considers *convex* problems and uses variance reduction (i.e., accessing the entire local gradient vector periodically), and [7] which only considers *undirected* graphs and with synchronization barriers.

Moving closer to our setting, contemporary works considering a decentralized graph communication structure and asynchronous communication include [8], which analyzes nonconvex problems and [9] considers convex ones. However, they consider only *undirected* graphs, and furthermore the ultimate function being minimized is not the desired objective, but a scaled one, based on the frequency of updates of each agent. This can frequently not be known a priori, and thus is a poor target for the objective function. Without this knowledge, any solution to the problem would be biased.

The push-sum framework was introduced in [10] to avoid systematic bias in the solution of multi-agent optimizations problems on directed graphs. The analysis of distributed consensus with delays was first given in [11], who introduced *virtual* nodes which model information as passing from one to the next as one less delay until it arrives at the real-time node.

Note that these are purely theoretical instruments, and need not be stored.

In [12] a stochastic gradient algorithm is presented based on the push sum approach to handling optimization over a network with asynchronous directed communication. However, convergence is only proven for strongly convex problems.

The paper [13] describes an algorithm for using the push-sum framework in a nonconvex setting with asynchronous parallel communication, for deterministic objectives.

The paper [14] considers asynchronous communication and directed graphs, and presents an algorithm with provably linear convergence towards the optimum under the Polyak-Łojasiewicz condition. Finally [15] considers nonconvex federated learning using gradient tracking, and prove convergence, consensus, and asymptotic agreement of each agent's average gradient estimate. They consider the synchronous setting and undirected graphs.

Contributions In this paper we study the theoretical and numerical convergence properties of decentralized stochastic nonconvex optimization on directed graphs with asynchronous communication. Thus, this paper extends the work of [13] to consider noisy function data and [12] to the case of nonconvex objectives, closing an important gap in the literature for decentralized stochastic optimization.

II. ALGORITHM

The algorithm is presented as Algorithm 1, and described below.

All agents update asynchronously and continuously without coordination, using noisy gradient estimates and possibly delayed information from their neighbors. Each agent i maintains and updates the following local variables: i) a local estimate $\mathbf{x}_i$ of the common optimization vector $\mathbf{x}$; ii) the auxiliary variable $\mathbf{z}_i$, aiming at tracking the sample gradient $\tilde{\nabla} F$ of the sum-loss (we use $\tilde{\nabla} F$ a sample instance of ∇F), not available locally; and iii) some mass counters ρ_{ij} and buffer variables $\tilde{\rho}_{ij}$, $j \in \mathcal{N}_i^{in}$, which are instrumental to track properly the sum-gradient $\nabla F(\cdot, \xi)$ in the presence of asynchrony (their update is commented below). The k-th iterate of the above variables is denoted by $\mathbf{x}_i^k, \mathbf{z}_i^k, \rho_{ij}^k$, and $\tilde{\rho}_{ij}^k$, respectively. In Algorithm 1, the iteration index k is understood as a *global iteration counter* k, unknown to the agents, which increases by 1 whenever a variable of the agents changes. Let i^k be the agent triggering iteration $k \to k+1$; it executes Steps (S1)-(S3) (no necessarily within the same activation), as described below.

(S1) Stochastic gradient step: The active agent i^k updates its local variable $\mathbf{x}_{i^k}^k$ by moving along the direction of the sample gradient estimate $\mathbf{z}_{i^k}^k$, with a step-size $\gamma \in (0,1]$, generating $\mathbf{v}_{i^k}^{k+1}$.

(S2) Consensus step with delays: Agent i^k may receive delayed variables from its in-neighbors $j \in \mathcal{N}_{i^k}^{in}$, whose iteration index is $k - d_j^k$, where $d_j^k \geq 0$ is the delay. To perform its update, it first sorts the "age" of all the received variables from agent j since $k = 0$, and then picks the most recently generated one. This is implemented maintaining a local counter $\tau_{i^k j}^k$, updated recursively as $\tau_{i^k j}^k = \max(\tau_{i^k j}^{k-1}, k - d_j^k)$. Thus, the variable agent i^k uses from j has iteration index

$\tau_{i^k j}^k$. Given this (outdated) information, agent i^k performs a consensus update with mixing matrix $\mathbf{W} = (w_{ij})_{i,j=1}^I$ (to be properly chosen, see Assumption II.1 below), generating $\mathbf{x}_{i^k}^{k+1}$.

(S3) Robust gradient tracking: This step aims at tracking the sample sum-gradient $\tilde{\nabla} F$ in the presence of asynchrony; it builds on the the asynchronous sum-push scheme introduced in [13] (note that [13] does not deal with stochastic gradients), and works as follows. Each agent i maintains mass counters ρ_{ji} associated to $\mathbf{z}_i$ that record the cumulative mass generated by i for $j \in \mathcal{N}_i^{out}$ since $k = 0$; and transmits ρ_{ji}. In addition, agent i also maintains buffer variables $\tilde{\rho}_{ij}$ to track the latest mass counter ρ_{ij} from $j \in \mathcal{N}_i^{in}$ that has been used in its update. The update of the $\mathbf{z}$- and ρ-variables employed by agent i^k is as follows. Agent i^k first performs the sum step (S3.1) using a possibly delayed mass counter $\rho_{ij}^{\tau_{ij}^k}$ received from j. By computing the difference $\rho_{i^k j}^{\tau_{i^k j}^k} - \tilde{\rho}_{i^k j}^k$, it collects the sum of the $a_{i^k j} z_j$'s generated by j that it has not yet added. Then, agent i^k sums them together with the gradient correction term $\tilde{\nabla} f_{i^k}(\mathbf{x}_{i^k}^{k+1}, \zeta^k) - \tilde{\nabla} f_i(\mathbf{x}_{i^k}^k, \zeta^{j(i^k, k)})$ to its current state variable $\mathbf{z}_{i^k}^k$ to form the intermediate mass $\mathbf{z}_{i^k}^{k+\frac{1}{2}}$, where $j(i^k, k)$ is the last iteration j before k for which i^k is the chosen agent. Next, in the push step (S3.2), agent i^k splits $\mathbf{z}_{i^k}^{k+\frac{1}{2}}$, maintaining $a_{i^k i^k} \mathbf{z}_{i^k}^{k+\frac{1}{2}}$ for itself and accumulating $a_{ji^k} \mathbf{z}_{i^k}^{k+\frac{1}{2}}$ to its local mass counter $\rho_{ji^k}^k$, to be transmit to $j \in \mathcal{N}_{i^k}^{out}$. Since the last mass counter agent i^k processed is $\rho_{i^k j}^{\tau_{i^k j}^k}$, it sets $\tilde{\rho}_{i^k j} = \rho_{i^k j}^{\tau_{i^k j}^k}$.

We make the following Assumption regarding the communication network, activation, delays, and stochastic gradient estimates.

Assumption II.1. 1) *It holds that there exists an $\bar{m}$ such that for all $i \in \mathcal{V}$, $w_{ij} \geq \bar{m}$ and $a_{ij} \geq \bar{m}$ for all $(i,j) \in \mathcal{E}$. Furthermore the matrix $\mathbf{W}$ composed of w_{ij} is row-stochastic ($\mathbf{W}\mathbf{1} = \mathbf{1}$) and $\mathbf{A}$ composed of a_{ij} is column-stochastic ($\mathbf{A}^T\mathbf{1} = \mathbf{1}$).*
2) *There is a $T \in \mathbb{R}^+$ such that the activations satisfy $\cup_{t=k}^{k+T-1} i^t = \mathcal{V}$.*
3) *There is a $D \in \mathbb{R}^+$ such that the delays satisfy $0 \leq d_j^k \leq D$ for all $j \in \mathcal{N}_{i^k}^{in}$ for all $k \in \mathbb{N}$*
4) *The assumptions on the stochastic estimate are the standard unbiased estimate with bounded variance conditions,*

$$\mathbb{E}\left[\tilde{\nabla} f_{i^k}(\mathbf{x}_{(i^k)}^{k+1}, \zeta^k)\right] = \nabla f_{i^k}(\mathbf{x}_{(i^k)}^{k+1}),$$
$$\mathbb{E}\left[\left\|\tilde{\nabla} f_{i^k}(\mathbf{x}_{(i^k)}^{k+1}, \zeta^k) - \nabla f_{i^k}(\mathbf{x}_{(i^k)}^{k+1})\right\|^2\right] = \sigma^2 \quad (2)$$

III. CONVERGENCE

In this section we prove the convergence properties of Algorithm 1 for stochastic nonconvex objectives. We begin introducing some intermediate results, instrumental for our proofs.

A. Preliminaries

Following [13], we define augmented variables $\mathbf{h}^k \triangleq \left[(\mathbf{x}^k)^T (\mathbf{v}^k)^T (\mathbf{v}^{k-1})^T ...(\mathbf{v}^{k-D})^T\right]$, where D is the maximum

Algorithm 1 - Asynchronous Stochastic Gradient Descent with Tracking

Initialization: Set $k = 0$, Set $\mathbf{x}_i^0 = \mathbf{0}$ and $\mathbf{z}_i^0 = \tilde{f}_i(\mathbf{0}, \xi^0)$ for all i.

while Not converged **do**

Choose (i^k, d^k);

Set $\tau_{i^k j}^k = \max\{\tau_{i^k j}^{k-1}, k - d_j^k\}, \quad \forall j \in \mathcal{N}_{i^k}^{in}$;

(S1) (Stochastic gradient update): Set $\mathbf{v}_{i^k}^{k+1} = \mathbf{x}_{i^k}^k - \gamma^k \mathbf{z}_{i^k}^k$

(S2) Consensus (with delayed info):
$\mathbf{x}_{i^k}^{k+1} = w_{i^k i^k} \mathbf{v}_{i^k}^{k+1} + \sum_{j \in \mathcal{N}_{i^k}^{in}} w_{i^k j} \mathbf{v}_j^{\tau_{i^k j}^k}$

(S3) Robust gradient tracking:

(S3.1) Sum step:

$$\mathbf{z}_{i^k}^{k+\frac{1}{2}} = \mathbf{z}_{i^k}^k + \sum_{j \in \mathcal{N}_{i^k}^{in}} (\rho_{i^k j}^{\tau_{i^k j}^k} - \tilde{\rho}_{i^k j}^k)$$
$$+ \tilde{\nabla} f_{i^k}(\mathbf{x}_{i^k}^{k+1}, \zeta^k) - \tilde{\nabla} f_i(\mathbf{x}_{i^k}^k, \zeta^{j(i^k, k)});$$

(S3.2) Push step:

$$\mathbf{z}_{(i^k)}^{k+1} = a_{i^k i^k} \mathbf{z}_{(i^k)}^{k+\frac{1}{2}};$$
$$\rho_{j i^k}^{k+1} = \rho_{j i^k}^k + a_{j i^k} \mathbf{z}_{i^k}^{k+\frac{1}{2}}, \forall j \in \mathcal{N}_{i^k}^{out};$$

(S3.3) Mass-Buffer update:

$$\tilde{\rho}_{i^k j}^{k+1} = \rho_{i^k j}^{\tau_{i^k j}^k}, \forall j \in \mathcal{N}_{i^k}^{in};$$

(S4): Untouched state variables shift to state $k+1$ while keeping the same value; $k \leftarrow k + 1$.

end while

---

possible delay time. We denote the augmented gradient estimate stacked vector as $\hat{\mathbf{z}}^k$. Ultimately, there is a matrix $\hat{\mathbf{A}}^k$ representing the mixing of $\hat{z}$, i.e., $\hat{\mathbf{z}}^{k+1} = \hat{\mathbf{A}}^k \hat{\mathbf{z}}^k + \mathbf{p}^k$ [13], where $\mathbf{p}^k$ is simply the stacked change in the vector from the new stochastic gradient updates. For the consensus of the expanded model vector $\mathbf{h}^k$ we denote by $\hat{\mathbf{W}}^k$ the corresponding mixing matrix, i.e., $\mathbf{h}^{k+1} = \hat{\mathbf{W}}^k (\mathbf{h}^k + \delta^k)$, with δ^k defining the stacked update vector. The explicit expressions of the matrices $\hat{\mathbf{A}}^k$ and $\hat{\mathbf{W}}^k$ are immaterial for our subsequent convergence analysis; all it is needed are their mixing rate properties, as recalled next.

Lemma III.1. *[13, Lemma 14] In the setting of Algorithm 1, there exists a sequence of stochastic vectors $\{\xi^k\}$ such that, for any $k \geq t \in \mathbb{N}$ and $i, j \in \mathcal{V}$, there holds*

$$|\hat{\mathbf{A}}_{ij}^{k:t} - \xi_i^k| \leq C \rho^{k-t},$$

for some $C > 0$ and $\rho < 1$.

Lemma III.2. *[13, Lemma 16] In the setting of Algorithm 1, there exists a sequence of stochastic vectors $\{\psi^k\}$ such that, for any $k \geq t \in \mathbb{N}$ and $i, j \in \mathcal{V}$, there holds*

$$|\hat{\mathbf{W}}^{k:t} - \mathbf{1}(\psi^t)^T| \leq C \rho^{k-t},$$

for some $C > 0$ and $\rho < 1$.

Finally, we define a new vector $\bar{\mathbf{z}}_{i^k}^k$. This represents the update that would be made if actual rather that stochastic gradients were computed, i.e. $\bar{\mathbf{z}}_{(i)}^k = \nabla f_i(\mathbf{x}_i^0) + \sum_{t=k: i^k = i} (\nabla f_i(\mathbf{x}_i^{t+1}) - f_i(\mathbf{x}_i^t))$.

It can be seen that,

$$\mathbb{E}\left[\bar{\mathbf{z}}_{i^k}^k - \mathbf{z}_{i^k}^k\right] = 0 \quad \text{and} \quad \mathbb{E}\left[\left\|\bar{\mathbf{z}}_{i^k}^k - \mathbf{z}_{i^k}^k\right\|^2\right] \leq m\sigma^2. \quad (3)$$

To study convergence of Algorithm 1, we introduce the following error terms, defining the gradient tracking, consensus, and gradient norm errors for the evolving iterations:

$$E_t^k = \left\|\bar{\mathbf{z}}_{(i)}^k - \xi_{i^k}^{k-1}\left(\mathbf{1}\mathbf{1}^T \otimes \mathbf{I}_n\right)\bar{\mathbf{z}}^k\right\|^2,$$
$$E_c^k = \left\|\mathbf{h}^k - \mathbf{1}_m \otimes \mathbf{x}_\psi^k\right\|^2, E_z^k = \left\|\bar{\mathbf{z}}_{(i^k)}^k\right\|^2. \quad (4)$$

Note that,

$$\mathbb{E}|\mathbf{z}_{i^k}^k| \leq \sqrt{E_z^k} + \mathbb{E}|\mathbf{z}_{i^k}^k - \bar{\mathbf{z}}_{i^k}^k| \leq \sqrt{E_z^k} + \sqrt{m}\sigma.$$

B. Convergence Theory

The proof of the main convergence theory begins similarly as in [13], however, subsequently changes significantly in order to account for the noise and then also set up the possibility of deriving specific convergence rates for the optimization, consensus, and tracking errors.

Theorem III.1. *Let Assumptions I.1 and II.1 hold.*

Assume that the stepsize sequence $\{\gamma^k\}$ satisfies,

$$\sum_{k=1}^\infty \gamma^k = \infty, \ \sum_{k=1}^\infty (\gamma^k)^2 < \infty,$$
$$\gamma^0 \leq \min\left\{\frac{1}{1+\eta}, \frac{1}{4(L+C_2^c+C_2^t)}\right\}$$

where C_2^c and C_2^t are constants to be defined in the proof.

The merit function $M(\bar{\mathbf{z}}^k, \mathbf{h}^k, \mathbf{x}_\psi^k) = \mathbb{E}\left[E_t^k + E_c^k + E_z^k\right]$ is sublinearly convergent with the standard ergodic rate,

$$\sum_{l=0}^k \gamma^l M(\bar{\mathbf{z}}^l, \mathbf{h}^l, \mathbf{x}_\psi^l) \leq \frac{C}{\sum_{l=0}^k \gamma^l}$$

for some constant $C > 0$.

Proof. Consider the application of the Descent Lemma to F applied at x_ψ^k and x_ψ^{k+1}.

$$
\begin{aligned}
\mathbb{E}\left[F(\mathbf{x}_{(\psi)}^{k+1})\right] &\leq \mathbb{E}\left[F(\mathbf{x}_\psi^k)\right] + \gamma^k\psi_{i^k}^k \mathbb{E}\left[\left\langle \nabla F(\mathbf{x}_\psi^k), -\mathbf{z}_{(i^k)}^k\right\rangle\right] \\
&\quad + \frac{L(\gamma^k\psi_{ik}^k)^2}{2}\mathbb{E}\left[\left\|\mathbf{z}_{(i^k)}^k\right\|^2\right] \\
&\leq \mathbb{E}\left[F(\mathbf{x}_\psi^k)\right] + \gamma^k\psi_{i^k}^k\mathbb{E}\left[\left\langle\nabla F(\mathbf{x}_\psi^k), -\bar{\mathbf{z}}_{(i^k)}^k\right\rangle\right] \\
&\quad + \gamma^k\psi_{i^k}^k\mathbb{E}\left[\left\langle\nabla F(\mathbf{x}_\psi^k), \bar{\mathbf{z}}_{(i^k)}^k - \mathbf{z}_{(i^k)}^k\right\rangle\right] \\
&\quad + L(\gamma^k\psi_{i^k}^k)^2\mathbb{E}\left[\left\|\bar{\mathbf{z}}_{(i^k)}^k\right\|^2 + \left\|\mathbf{z}_{(i^k)}^k - \bar{\mathbf{z}}_{(i^k)}^k\right\|^2\right] \\
&\leq \mathbb{E}\left[F(\mathbf{x}_\psi^k)\right] + L(\gamma^k)^2\mathbb{E}\left[\left\|\bar{\mathbf{z}}_{(i^k)}^k\right\|^2\right] \\
&\quad + \gamma^k\psi_{i^k}^k\mathbb{E}\left[\left\langle(\xi_{i^k}^{k-1})^{-1}\bar{\mathbf{z}}_{(i^k)}^k, -\bar{\mathbf{z}}_{(i^k)}^k\right\rangle\right] \\
&\quad + \gamma^k\psi_{i^k}^k\mathbb{E}\left[\left\langle\left(\mathbf{1}\mathbf{1}^T\otimes\mathbf{I}_n\right)\bar{\mathbf{z}}^k - (\xi_{i^k}^{k-1})^{-1}\bar{\mathbf{z}}_{(i^k)}^k, -\bar{\mathbf{z}}_{(i^k)}^k\right\rangle\right] \\
&\quad + \gamma^k\psi_{i^k}^k\mathbb{E}\left[\left\langle\nabla F(\mathbf{x}_\psi^k) - \left(\mathbf{1}\mathbf{1}^T\otimes\mathbf{I}_n\bar{\mathbf{z}}^k\right), -\bar{\mathbf{z}}_{(i^k)}^k\right\rangle\right] \\
&\quad + \gamma^k\psi_{i^k}^k\mathbb{E}\left[\left\langle\nabla F(\mathbf{x}_\psi^k), \bar{\mathbf{z}}_{(i^k)}^k - \mathbf{z}_{(i^k)}^k\right\rangle\right] \\
&\quad + L(\gamma^k\psi_{i^k}^k)^2\mathbb{E}\left[\left\|\mathbf{z}_{(i^k)}^k - \bar{\mathbf{z}}_{(i^k)}^k\right\|^2\right] \\
&\leq \mathbb{E}\left[F(\mathbf{x}_\psi^k)\right] + L(\gamma^k)^2\mathbb{E}\left[\left\|\bar{\mathbf{z}}_{(i^k)}^k\right\|^2\right] \\
&\quad + \gamma^k\psi_{i^k}^k\mathbb{E}\left[\left\langle(\xi_{i^k}^{k-1})^{-1}\bar{\mathbf{z}}_{(i^k)}^k, -\bar{\mathbf{z}}_{(i^k)}^k\right\rangle\right] \\
&\quad + \gamma^k\psi_{i^k}^k\left(\frac{\beta_1}{2}E_t^k + \frac{1}{2\beta_1}E_z^k\right) \\
&\quad + \gamma^k\psi_{i^k}^k\left(\frac{\beta_1 Lm}{2}E_c^k + \frac{1}{2\beta_1}E_z^k\right) \\
&\quad + \gamma^k\psi_{i^k}^k\mathbb{E}\left[\left\langle\nabla F(\mathbf{x}_\psi^k), \bar{\mathbf{z}}_{(i^k)}^k - \mathbf{z}_{(i^k)}^k\right\rangle\right] \\
&\quad + L(\gamma^k\psi_{i^k}^k)^2\mathbb{E}\left[\left\|\mathbf{z}_{(i^k)}^k - \bar{\mathbf{z}}_{(i^k)}^k\right\|^2\right]
\end{aligned}
$$

where in the last inequality we used the Cauchy-Schwartz and Young's inequality, $ab \leq \frac{\beta}{2}a^2 + \frac{1}{2\beta}b^2$, twice.

Set $\beta_1 = \beta_2 = 2/\eta$, then,

$$
\begin{aligned}
\mathbb{E}\left[F(\mathbf{x}_{(\psi)}^{k+1})\right] &\leq \mathbb{E}\left[F(\mathbf{x}_\psi^k)\right] - \left(\frac{\eta\gamma^k}{2} + L(\gamma^k)^2\right)\mathbb{E}\left[\left\|\bar{\mathbf{z}}_{(i^k)}^k\right\|^2\right] \\
&\quad + \frac{\gamma^k}{\eta}E_t^k + \frac{\gamma^k}{\eta}E_c^k \\
&\quad + \gamma^k\psi_{i^k}^k\mathbb{E}\left[\left\langle\nabla F(\mathbf{x}_\psi^k), \bar{\mathbf{z}}_{(i^k)}^k - \mathbf{z}_{(i^k)}^k\right\rangle\right] \\
&\quad + L(\gamma^k)^2 m\sigma^2
\end{aligned}
\tag{5}
$$

Now it holds that by [13, Proposition 17],

$$
\begin{aligned}
\mathbb{E}\left[\sqrt{E_c^k}\right] &\leq C_2\rho^k\mathbb{E}\left[\sqrt{E_c^0}\right] + C_2\sum_{l=0}^{k-1}\rho^{k-l}\gamma^l\mathbb{E}|\mathbf{z}_{i^l}^l| \\
&\leq C_2\rho^k\mathbb{E}\left[\sqrt{E_c^0}\right] + C_2\sum_{l=0}^{k-1}\rho^{k-l}\gamma^l\left(\mathbb{E}\sqrt{E_z^l} + \sqrt{m}\sigma\right)
\end{aligned}
\tag{6}
$$

which implies, by [13, Lemma 26] that, after taking full expectations, that there exist C_1^c and C_2^c such that,

$$
\sum_{l=0}^k\mathbb{E}\left[E_c^l\right] \leq C_1^c + C_2^c\sum_{l=0}^k(\gamma^l)^2\left(\mathbb{E}\left[E_z^l\right] + m\sigma^2\right)
\tag{7}
$$

Similarly, from the proof of [13, Proposition 18], it can be seen that,

$$
\begin{aligned}
\mathbb{E}\left[\sqrt{E_t^k}\right] &\leq 3C_0C_L\sum_{l=0}^{k-1}\rho^{k-l}\mathbb{E}\left[\sqrt{E_c^l} + \gamma^l(\sqrt{E_z^l} + \sqrt{m}\sigma)\right] \\
&\quad + C_0\rho^k\|\mathbf{z}^0\|
\end{aligned}
\tag{8}
$$

and so, again as in [13, Lemma 26]

$$
\sum_{l=0}^k\mathbb{E}\left[E_t^l\right] \leq C_1^t + C_2^t\sum_{l=0}^k(\gamma^l)^2\left(\mathbb{E}\left[E_z^l\right] + m\sigma^2\right)
\tag{9}
$$

Now, summing up (5), taking full expectations we get, using (3)

$$
\begin{aligned}
\sum_{l=0}^k\left(\frac{\gamma^l\eta}{2} + L(\gamma^l)^2\right)\mathbb{E}\left\|\bar{\mathbf{z}}_{(i^l)}^l\right\|^2 &\leq F(\mathbf{x}^0) - F_m \\
&\quad + \sum_{l=0}^k\left[\frac{\gamma^l}{\eta}\mathbb{E}[E_t^l] + \frac{\gamma^l}{\eta}\mathbb{E}[E_c^l]\right] + \sum_{l=0}^k(\gamma^l)^2 Lm\sigma^2
\end{aligned}
\tag{10}
$$

Now add $\sum_{l=0}^k\left[\gamma^l\mathbb{E}[E_t^l] + \gamma^l\mathbb{E}[E_c^l]\right]$ to both sides, use $\frac{\gamma^l}{\eta} + \gamma^l \leq 1$ and plug in (7) and (9) to get,

$$
\begin{aligned}
\sum_{l=0}^k\left(\frac{\gamma^l\eta}{2} - (L + C_2^c + C_2^t)(\gamma^l)^2\right)\mathbb{E}\left\|\bar{\mathbf{z}}_{(i^l)}^l\right\|^2 \\
+ \sum_{l=0}^k\left[\gamma^l E_t^l + \gamma^l E_c^l\right] \\
\leq F(\mathbf{x}^0) - F_m + C_1^c + C_1^t + C_\sigma\sum_{l=0}^k(\gamma^l)^2
\end{aligned}
$$

and so for sufficiently small γ^0, we have,

$$
\sum_{l=0}^k\gamma^l\left[E_z^l + E_t^l + E_c^l\right] \leq F(\mathbf{x}^0) - F_m + C_\sigma m\sigma^2\sum_{l=0}^k(\gamma^l)^2
$$

and the proof claim follows from the general stepsize conditions. $\square$

Corollary III.1. *With the specific step-size choice of,*

$$
\gamma^k = \frac{1}{k^\alpha}, \ \alpha \in (1/2, 1]
$$

we have the following convergence rates,

$$
\begin{aligned}
M(\bar{\mathbf{z}}^k, \mathbf{h}^k, \mathbf{x}_\psi^k) &= o\left(\tfrac{1}{k^{1-\alpha}}\right), \ \mathbb{E}\left[E_t^k\right] = o\left(\tfrac{1}{k}\right), \\
\mathbb{E}\left[E_c^k\right] &= o\left(\tfrac{1}{k}\right), \ \mathbb{E}\left[E_z^k\right] = o\left(\tfrac{1}{k^{1-\alpha}}\right)
\end{aligned}
$$

Proof. The first follows directly from Theorem III.1, i.e., the right hand side is bounded and thus the sum on the left must be bounded, and so $\gamma^k M^k = o\left(\frac{1}{k}\right)$, thus the form of γ^k implies the rate for M^k.

The rates for $\mathbb{E}\left[E_c^k\right]$ and $\mathbb{E}\left[E_t^k\right]$ follow from (7) and (9), respectively, as the right hand side is bounded and thus the sum on the left must be.

Finally the bound for $\mathbb{E}\left[E_t^k\right]$ follows from the finiteness of the right hand side of (10) and the form of γ^k. $\square$

We note how, similarly as in [16], the consensus errors converge quicker than the optimization and asymptotically the optimization dominates the overall convergence rate, in this case arbitrarily close to the standard SGD nonconvex rate of $O\left(\frac{1}{\sqrt{k}}\right)$ [17].
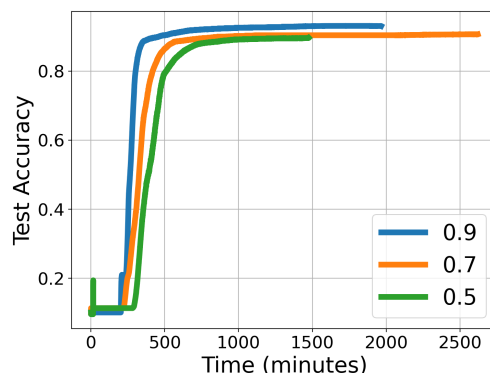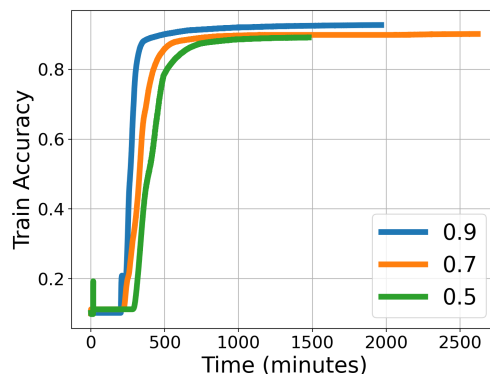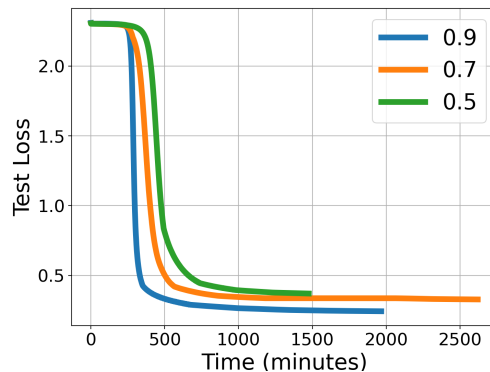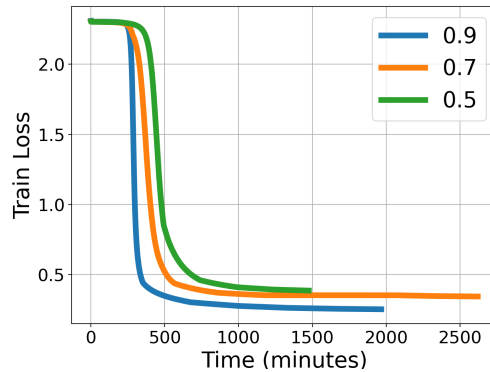
IV. Experiments

In this section we aim to numerically study different aspects of our proposed Algorithm 1 on a non-convex optimization task.

The non-convex optimization task is image classification using neural network on MNIST [18] dataset where we aim to solving it using Algorithm 1 on a randomly generated digraph. We randomly shuffle the dataset and uniformly partition it across the nodes where the data partitions are disjoint. In our randomly generated digraph structure each agent i sends its updates to 3 out-neighbors; one of them is the next agent $i+1$ in the cycle order and the other two are selected randomly from a uniform distribution, following [13]. We use the convolutional neural network (CNN) architecture used in Tensorflow tutorial [19]. For each experiment we used a step-wise learning rate reduction schedule to achieve the best results. We selected the initial learning rate (step-size), step reduction interval, and size of reduction from the grid $[1, 0.8, 0.6, 0.4, 0.3, 0.2, 0.1]$, $[5000, 7500, 10000]$, and $[1.5, 1.6, 1.8, 2]$ respectively. Unless stated, otherwise we fixed the number of iterations that each node performs to 45000 and stored results at every 30 seconds. The experiments are done in Python environment using Tensorflow V2 and MPI (mpi4py) on RCI[1] clusters over the cpu nodes.

Convergence and Scalability The experiments in this section analyze the convergence property of our method for the described task. We perform experiments for $I = 2, 4, 8, 16$ nodes with a fixed graph connectivity of 0.7. We report the results on the node-wise average parameters, i.e. $x^{avg} = \sum_i x_i$.

Figure 1 shows the time-wise convergence results for different number of nodes. We can see that the accuracy drops monotonically as the number of nodes increased. This suggests for this scale we do not witness speedup with decentralized parallelism, although accurate training is still achievable.

Graph Connectivity This part experimentally studies the behaviour of our algorithm for different percentages of graph connectivity. We fixed the number of nodes to 16 and did experiments with $\{0.5, 0.7, 0.9\}$ graph connectivity percentages. Figure 2 shows the time-wise convergence results for different graph connectivity percentages. Our observation is that as the graph topology gets more connected, the convergence results get improve and our algorithm finds a better optimal parameters.
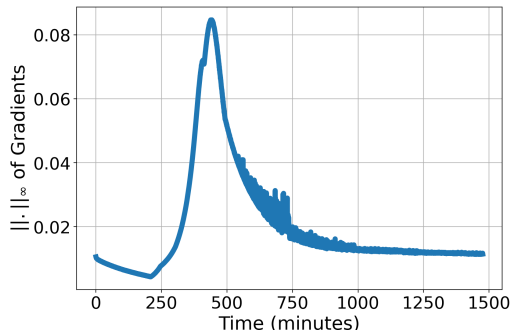
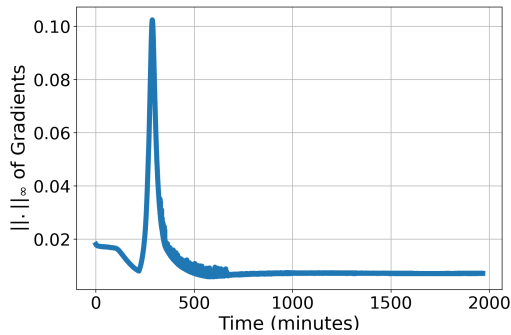Fig. 2: Time-wise convergence results for different connectivity

Parameter deviations and Norm of gradients

Figure 3 shows the average L_∞ distance of each node's parameters and the node-wise average, i.e. $\frac{1}{K}\sum_{i=1}^{K}||x_i - x^{avg}||_\infty$ at the end of each snapshot, for two different graph connectivity percentages, i.e. $(0.5, 0.9)$. We observe that each node's parameters are approximately equidistant from the average. Moreover, there is a gradual increase at the beginning time around $500, 300$ minutes for $0.5, 0.9$ graph connectivity percentages respectively which is due to the initial learning-rate warm-up. After that point, we can see multiple reductions at the learning-rate reduction intervals. In fact, it is noticeable that the parameter deviations are smaller for higher connectivity percentages.

Figure 4 presents the norm infinity of gradients on the whole dataset using the node-wise averaged parameters. As the graph connectivity percentage increases, norm infinity of the gradients gets smaller and reduces faster. We can see a gradual increase at the beginning time around $500, 300$ minutes for $0.5, 0.9$ graph connectivity percentages respectively which is consistent with our observation in parameter deviation plots in Figure 3.
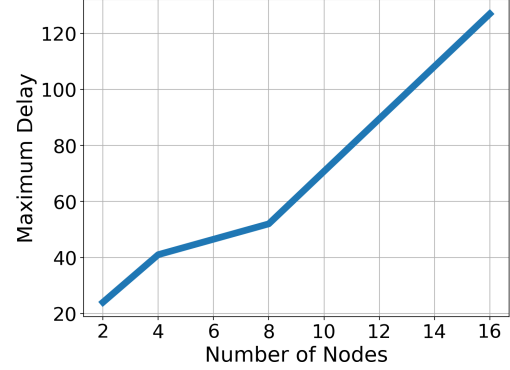

(a) 0.5 graph connectivity

(b) 0.9 graph connectivity

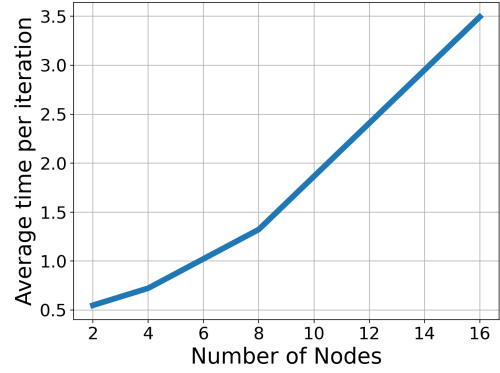Fig. 4: Norm of gradients for different graph connectivity percentages

Maximum delay and Time per iteration

Figures 5a,5b show maximum delay, and average time per iteration as a function of number of nodes for a fixed 0.7 graph connectivity percentage respectively. Maximum delay is the delay between the fastest and slowest nodes. We observe that maximum delay increased as the number of nodes increased and it was always bounded. Indeed, from figure 5b we can see that average time per iteration also increased as the number of

nodes increased. Figure 6 represents the behaviour of maximum delay w.r.t graph connectivity percentage. We can observe that maximum delay has been increased by increasing connectivity percentage and it was always bounded.

(a) Maximum Delay

(b) Average Time Per Iteration

Fig. 5: Maximum delay and average time per iteration across different number of nodes for 0.7 graph connectivity percentage
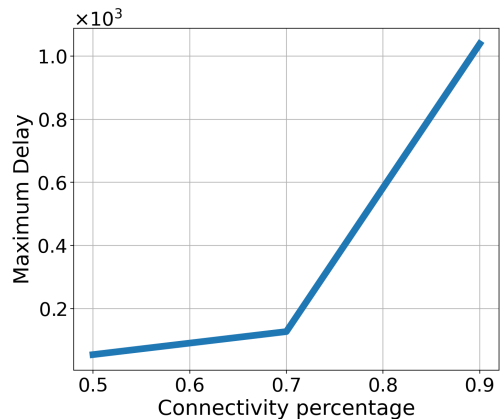

Fig. 6: Maximum delay w.r.t graph connectivity percentage

V. CONCLUSION

In this paper we have studied stochastic nonconvex decentralized optimization on directed graphs with asynchronous
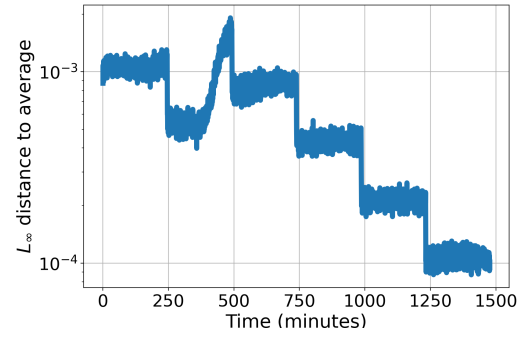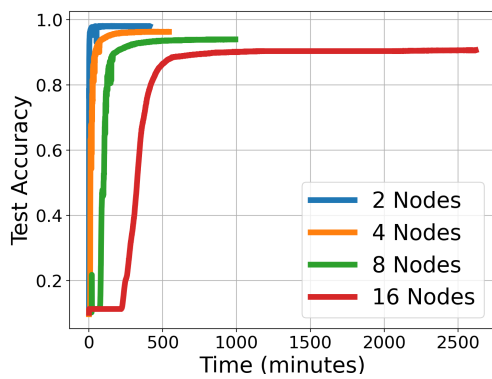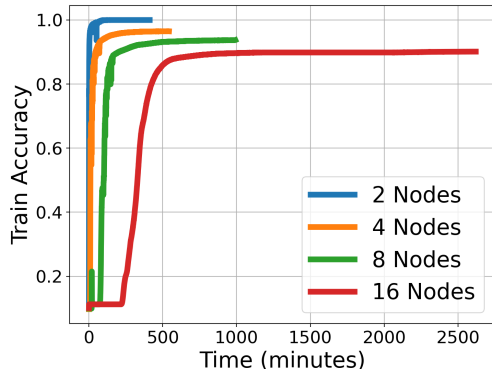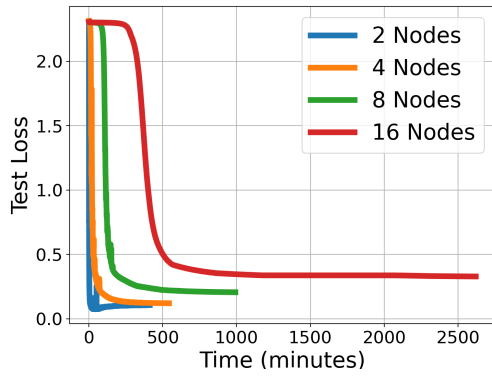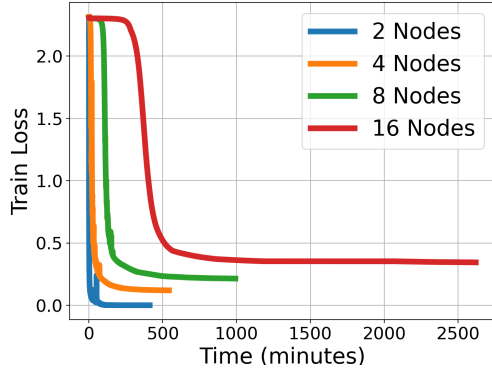
communication, closing an important gap in the literature on distributed optimization. The theoretical results confirm the expected sublinear convergence rate, and corroborate a similar pattern of faster consensus and tracking convergence, leaving the optimization to dominate the error asymptotically. Our numerical results confirm the convergence of the algorithm and show the scalability with the number of nodes and different graph connectivity percentages on a non-convex image classification task.
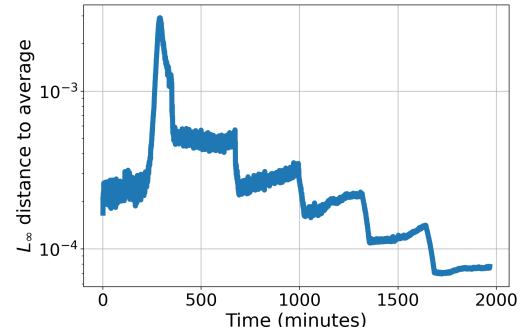
ACKNOWLEDGMENT

REFERENCES

[1] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE transactions on automatic control*, vol. 31, no. 9, pp. 803–812, 1986.

[2] X. Lian, Y. Huang, Y. Li, and J. Liu, "Asynchronous parallel stochastic gradient for nonconvex optimization," in *Advances in Neural Information Processing Systems*, 2015, pp. 2737–2745.

[3] Z. Peng, Y. Xu, M. Yan, and W. Yin, "Arock: an algorithmic framework for asynchronous parallel coordinate updates," *SIAM Journal on Scientific Computing*, vol. 38, no. 5, pp. A2851–A2879, 2016.

[4] A. Agarwal and J. C. Duchi, "Distributed delayed stochastic optimization," in *Advances in Neural Information Processing Systems*, 2011, pp. 873–881.

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[6] R. Xin, U. A. Khan, and S. Kar, "Variance-reduced decentralized stochastic optimization with gradient tracking," *arXiv preprint arXiv:1909.11774*, 2019.

[7] H. Tang, X. Lian, M. Yan, C. Zhang, and J. Liu, "D2: Decentralized training over decentralized data," in *International Conference on Machine Learning*, 2018, pp. 4848–4856.

[8] X. Lian, W. Zhang, C. Zhang, and J. Liu, "Asynchronous decentralized parallel stochastic gradient descent," *arXiv preprint arXiv:1710.06952*, 2017.

[9] T. Wu, K. Yuan, Q. Ling, W. Yin, and A. H. Sayed, "Decentralized consensus optimization with asynchrony and delays," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 2, pp. 293–307, 2018.

[10] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Push-sum distributed dual averaging for convex optimization," in *2012 ieee 51st ieee conference on decision and control (cdc)*. IEEE, 2012, pp. 5453–5458.

[11] A. Nedić and A. Ozdaglar, "Convergence rate for consensus with delays," *Journal of Global Optimization*, vol. 47, no. 3, pp. 437–456, 2010.

[12] A. Olshevsky, I. C. Paschalidis, and A. Spiridonoff, "Robust asynchronous stochastic gradient-push: Asymptotically optimal and network-independent performance for strongly convex functions," *arXiv preprint arXiv:1811.03982*, 2018.

[13] Y. Tian, Y. Sun, and G. Scutari, "Achieving linear convergence in distributed asynchronous multi-agent optimization," *IEEE Trans. on Automatic Control*, 2020.

[14] J. Zhang and K. You, "Fully asynchronous distributed optimization with linear convergence in directed networks," *arXiv preprint arXiv:1901.08215*, 2019.

[15] ——, "Decentralized stochastic gradient tracking for empirical risk minimization," *arXiv preprint arXiv:1909.02712*, 2019.

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

[17] S. Ghadimi and G. Lan, "Stochastic first-and zeroth-order methods for nonconvex stochastic programming," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2341–2368, 2013.

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

[19] T. Tutorials. https://www.tensorflow.org/tutorials/quickstart/advanced. [Online]. Available: "https://www.tensorflow.org/tutorials/quickstart/advanced"

Fig. 1: Convergence results for different number of nodes

(a) 0.5 graph connectivity

(b) 0.9 graph connectivity

Fig. 3: L_∞ distance to average for different graph connectivity percentages