Decentralized Asynchronous Nonconvex Stochastic Optimization on Directed Graphs

Vyacheslav Kungurtsev, Mahdi Morafah, Tara Javidi, Fellow, IEEE, and Gesualdo Scutari, Fellow, IEEE

Abstract—In this article, we consider a decentralized stochastic optimization problem over a network of agents, modeled as a directed graph: Agents aim to asynchronously minimize the average of their individual losses (possibly nonconvex), each one having access only to a noisy estimate of the gradient of its own function. We propose an asynchronous distributed algorithm for such a class of problems. The algorithm combines stochastic gradients with tracking in an asynchronous push-sum framework and obtains a sublinear convergence rate, matching the rate of the centralized stochastic gradient descent applied to the nonconvex minimization. Our experiments on a nonconvex image classification task using a convolutional neural network validate the convergence of our proposed algorithm across a different number of nodes and graph connectivity percentages.

Index Terms—Decentralized applications, distributed computing, federated learning, machine learning, optimization, optimization methods.

I. INTRODUCTION

We consider the following multiagent optimization problem:

\[
\min_{x \in \mathbb{R}^n} \mathbb{E}_\xi [F(x, \zeta)] = \sum_{i=1}^{m} \mathbb{E}_\xi [f_i(x, \zeta)]
\]

where the agents’ local objective functions \( \mathbb{E}[f_i(\cdot, \zeta)] \), \( i = 1, \ldots, m \) are smooth and generally nonconvex. Each agent is assumed to have access only to noisy estimates of its own \( f_i \) and its gradients. Agents are embedded in a communication network, modeled as a (possibly) directed graph \( \mathcal{G} = (\mathcal{V}, \mathcal{E}) \), where \( \mathcal{V} \) is the set of vertices (agents) and \( e \in \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \) denotes an edge; \( (i, j) \in e \) implies that node \( i \) can send information to node \( j \). We define by \( \mathcal{N}_{\text{in}}^i := \{ j \in \mathcal{V} : (j, i) \in \mathcal{E} \} \) the set of in-neighbors of agent \( i \) (the set of agents that can send information to \( i \)) while \( \mathcal{N}_{\text{out}}^i := \{ j \in \mathcal{V} : (i, j) \in \mathcal{E} \} \) is the set of out-neighbors of agent \( i \) (agents that can receive information from \( i \)). Communications among the agents and individual local optimizations can happen in a fully asynchronous fashion, with each iteration consisting of an arbitrary activation among the agents as well as delays in the communicated information, subject only to conditions on the delay bound and a relative bounded uniformity in agent activation frequency (as per the work in [1]). We make the following assumptions on the optimization problem (1) and networking setting.

Assumption I.1:

1) For a.e. \( \zeta \), \( f_i(\cdot, \zeta) \) is \( L_i \)-Lipschitz continuously differentiable. Furthermore (for a.e. \( \zeta \)), \( F(\cdot, \zeta) \) is bounded from below.
2) The (di)graph \( \mathcal{G} \) is (strongly) connected.

In this article, we uniquely address several concomitant challenges, which are as follows.

1) The objective function is stochastic nonconvex.
2) The sum-components of the objective function are known only to the intended agent.
3) Communications and local optimization from the agents are fully asynchronous [1].
4) The topology of the network is arbitrary (i.e., no hierarchical structure) and communications are directed (i.e., that agent \( i \) being able to send data to \( j \) does not necessarily imply that \( j \) can also send information to \( i \)).

A. 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, as discussed next.

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 work [2] and the seminal work [3], the general framework for block/coordinate parallel updates given in [4], 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.

2325-5870 © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.
A standard structure for distributed optimization is the “hub–spoke,” “parameter–server,” or “master–slave” architecture. This is considered, for instance, in [5] and [6]. In that setting, 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. A similar, in spirit, approach is the “local SGD” framework introduced in [7] wherein there is no central node, but all of the worker nodes can communicate directly to each other and periodically average their local parameter estimates. Also, see the work in [8] for an effective extension incorporating momentum at the periodic averaging iterations.

In this article, we consider a more general arbitrary graph topology modeling the communication links across agents. This setting can appear in “Internet of Things”-type settings wherein there are independently computing nodes that cooperatively solve learning problems, or particular distributed memory high-performance computing environments wherein the communication bus has, e.g., a ring or lattice structure (see, e.g., [9]).

Schemes that consider an arbitrary graph topology include the scheme in [10], which considers convex problems and uses variance reduction (i.e., accessing the entire local gradient vector periodically), and the scheme in [11], which only considers undirected graphs and with synchronization barriers. We also mention the work in [12] as presenting an interesting and nuanced picture of the relative performance potential of a stochastic gradient for (nonconvex) optimization on (undirected) graphs. An important point, which we shall see again in the presentation of the numerical results, is that the presence of parallel speedup in this scenario depends on problem and algorithm constants. Finally, we point to the work in [13], which considers the stochastic undirected graph setting, however, with looser assumptions on the nature of the gossip communication, incorporating a wide variety of practical settings, including random selection, as well as the recent work [14], which presents a tighter analysis of convergence rates for gradient tracking algorithms.

Moving closer to our setting, contemporary works considering a decentralized graph communication structure and asynchronous communication include the work in [9], which analyzes nonconvex problems, and the work in [15], which 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, thus resulting in 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 [16] to avoid systematic bias in the solution of multiagent optimization problems on directed graphs. Incorporating stochastic gradient updates, studying the performance on nonconvex objectives, and demonstrating performance on a series of standard learning tasks was given in [17].

The analysis of distributed consensus with delays was first given in [18], 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. Another important analytical breakthrough is the introduction of buffers to handle dropped communication across a (possibly directed) graph of computing links [19]. For a recent survey of asynchronous distributed and decentralized optimization, see [20].

The first work that appears in studying the convergence rate of a gradient push-sum algorithm for asynchronous directed gossip communication is by Assran and Rabbat [21], which considers strongly convex objectives. In [22], the stochastic setting is studied, with stochastic gradients computed in lieu of deterministic gradients and a convergence rate proven for strongly convex problems.

The work in [1] describes an algorithm for the push-sum framework in a nonconvex setting with asynchronous parallel communication for deterministic objectives.

The work in [23] considers asynchronous communication and directed graphs and presents an algorithm with provably linear convergence toward the optimum under the Polyak–Łojasiewicz condition. Finally, Zhang and You [24] consider 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.

B. Contributions

In this article, we study the theoretical and numerical convergence properties of decentralized stochastic nonconvex optimization on directed graphs with asynchronous communication. Thus, this article extends the work in [1] to consider noisy function data and complements the work in [22] considering an arbitrary bounded delay model and proving convergence in the case of nonconvex objectives, closing an important gap in the literature of decentralized stochastic optimization. The consideration of directed graphs introduces a number of challenges due to a loss of a lot of mathematical algorithmic symmetries.

II. ALGORITHM

The algorithm is presented as Algorithm 1 and described ahead. 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:

1) a local estimate $x_i \in \mathbb{R}^n$ of the common optimization vector $x$;
2) the auxiliary variable $z_i \in \mathbb{R}^n$, aiming at tracking the sample gradient $\nabla F$ of the sum-loss (we use $\nabla F$ a sample instance of $\nabla^c F$), not available locally;
3) some mass counters $\rho_{ij} \in \mathbb{R}^n$ and buffer variables $\tilde{\rho}_{ij} \in \mathbb{R}^n$, $j \in \mathcal{N}_i^n$, which are instrumental to track properly the sum-gradient $\nabla F(\cdot, \zeta)$ in the presence of asynchrony (their update is commented ahead).

The $k$th iterate of the above variables is denoted by $x^{k}_i$, $z^{k}_i$, $\rho^{k}_{ij}$, and $\tilde{\rho}^{k}_{ij}$, respectively. In addition, we denote $x^k$, $z^k \in \mathbb{R}^{nm}$ the vertical stack of the agents’ vectors, i.e., $x^k = \left( (x^k_1)^T \ (x^k_2)^T \ \ldots \ (x^k_n)^T \right)$.
In Algorithm 1, the iteration index $k$ is understood as a global iteration counter, 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 ahead.

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

(S2) Consensus step with delays: Agent $i^k$ may receive delayed variables from its in-neighbors $j \in \mathcal{N}^{in}_{ik}$, whose iteration index is $k - d^{ij}_j$, where $d^{ij}_j \geq 0$ 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 $\tau_{ikj}$, updated recursively as $\tau_{ikj}^k = \max(\tau_{ikj}^{k-1}, k - d^{ij}_j)$. Thus, the variable agent $i^k$ uses from $j$ has iteration index $\tau_{ikj}^k$. Given this (outdated) information, agent $i^k$ performs a consensus update with mixing matrix $W = (w_{ij})_{i,j=1}^n$ (to be properly chosen, see Assumption II.1), generating $x_i^{k+1}$.

(S3) Robust gradient tracking: This step aims at tracking the sample sum-gradient $\nabla F$ in the presence of asynchrony; it builds on the asynchronous sum-push scheme introduced in [1] (note that the work in [1] does not deal with stochastic gradients) and works as follows. Each agent $i$ maintains mass counters $\rho_{ij}$ associated to $z_i$ that record the cumulative mass generated by $i$ for $j \in \mathcal{N}^{out}_{ik}$ since $k = 0$ and transmits $\rho_{ij}$. In addition, agent $i$ also maintains buffer variables $\tilde{\rho}_{ij}$ to track the latest mass counter $\rho_{ij}$ from $j \in \mathcal{N}^{out}_{ik}$ that has been used in its update. The update of the $\mathbf{z}$- and $\rho$-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 $\tau_{ikj}^k$, received from $j$. By computing the difference $\rho_{ik} - \rho_{ikj}$, it collects the sum of the $a_{ij}z_i^j$’s generated by $j$ that has not yet added. Then, agent $i^k$ sums them together with the gradient correction term $\nabla f_i(x_i^{k+1}, c^k) - \nabla f_i(x_i^k, c^{(i^k),k})$ to its current state variable $z_i^k$ to form the intermediate mass $z_{ik}^{k+\frac{1}{2}}$, 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 $z_{ik}^{k+\frac{1}{2}}$, maintaining $a_{ik}z_{ik}^{k+\frac{1}{2}}$ for itself and accumulating $a_{ik}z_{ik}^{k+\frac{1}{2}}$ to its local mass counter $\rho_{ik}^{k+1}$, to be transmitted to $j \in \mathcal{N}^{out}_{ik}$ since the last mass counter agent $i^k$ processed is $\rho_{ikj}$. It sets $\tilde{\rho}_{ikj} = \rho_{ikj}$.

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

**Assumption II.1:**

1) There exists some $\bar{k}$ such that for all $i \in \mathcal{V}$, $w_{ij} \geq \bar{k}$ and $a_{ij} \geq \bar{k}$ for all $(i,j) \in \mathcal{E}$. Furthermore, the matrix $W \in \mathbb{R}^{m \times m}$ composed of $w_{ij}$ is row-stochastic ($W1 = 1$) and $A \in \mathbb{R}^{m \times m}$ composed of $a_{ij}$ is column-stochastic ($A^{-1} = 1$).

2) There is a positive integer $T$ such that the activations satisfy $u_{i=k}^{T-1} \in \mathcal{V}$.

---

**Algorithm 1:** Asynchronous Stochastic Gradient Descent With Tracking.

**Initialization:** Set $k = 0$, Set $x_i^0 = 0$ and $z_i^0 = f_i(0, \zeta^0)$ for all $i$.

**while** Not converged do

**Choose** ($i^k, d^k$):

Set $\tau_{ikj}^k = \max(\tau_{ikj}^{k-1}, k - d^{ij}_j) \forall j \in \mathcal{N}^{in}_{ik}$;

(S1) (Stochastic gradient update): Set $x_i^{k+1} = x_i^k - \gamma \tilde{z}_{ik}$;

(S2) Consensus (with delayed info):

$x_i^{k+1} = w_{ikj}x_j^{k+1} + \sum_{j \in \mathcal{N}^{in}_{ik}} w_{ikj} \tilde{z}_{ikj}^k$;

(S3) Robust gradient tracking:

(S3.1) Sum step:

$z_{ik}^{k+\frac{1}{2}} = z_i^k + \sum_{j \in \mathcal{N}^{in}_{ik}} (\rho_{ikj}^k - \rho_{ikj})$

$+ \nabla f_i(x_i^{k+1}, c^k) - \nabla f_i(x_i^k, c^{(i^k),k})$;

(S3.2) Push step:

$z_{ik}^{k+1} = a_{ik}z_{ik}^{k+\frac{1}{2}}$

$\rho_{ikj}^{k+1} = \rho_{ikj}^k + a_{ik}z_{ik}^{k+\frac{1}{2}} \forall j \in \mathcal{N}^{out}_{ik}$;

(S3.3) Mass-Buffer update:

$\tilde{\rho}_{ikj}^{k+1} = \rho_{ikj}^{k+1} \forall j \in \mathcal{N}^{in}_{ik}$.

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

end while

3) There is a $D \in \mathbb{R}^{+}$, such that the delays satisfy $0 \leq d^k_j \leq D$ for all $j \in \mathcal{N}^{in}_{ik}$, for all $k \in \mathbb{N}$.

4) The stochastic estimates are unbiased and with bounded variance:

$E_{\zeta^k} \left[ \nabla f_i(x_i^{k+1}, \zeta^k) - \nabla f_i(x_i^{k+1}) \right] = \sigma^2$

$E_{\zeta^k} \left[ \left\| \nabla f_i(x_i^{k+1}, \zeta^k) - \nabla f_i(x_i^{k+1}) \right\|^2 \right] = \sigma^2$

(2)

where the expectation is taken over the sample, conditional on the filtration up to iteration $k$.

---

**III. CONVERGENCE**

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

**A. Preliminaries**

Following the work in [1], we define augmented variables

$h^k \triangleq (x_i^k)^T (v^k)^T (v^{k-1})^T \cdots (v^{k-D})^T \in \mathbb{R}^{mn(D+2)}$

where $D$ is the maximum possible delay time. By augmenting the graph with virtual “noncomputing agents,” we obtain a new...
Finally, we define the following merit function measuring the progresses of the algorithm toward stationarity:

$$M_k \triangleq \max \left[ \mathbb{E} \left[ \| \nabla F(x^k, \zeta) \|^2 \right], \| x^k - 1_m \otimes x^k \| \right].$$  \hspace{1cm} (7)

Note that

$$\mathbb{E} | z_{i,k} | \leq \sqrt{E^2_x + \mathbb{E} | z_{i,k} - z_{i,k}^\ast |} \leq \sqrt{E_x^2 + \sqrt{m} \sigma}.$$

### B. Convergence Theory

The proof of the main convergence theory begins similarly as in [1], however, with significantly differences in order to account for the noise and set up the possibility of deriving specific convergence rates for the optimization, consensus, and tracking errors.

**Theorem III.1**: Let Assumptions I.1 and I.1 hold. Define the following set of constants:

$$C_0^* \triangleq \frac{2 \sqrt{(D+2)m(1+\bar{K}_A)}}{1+\bar{K}_A}, \quad C_0^\prime \triangleq C_A \sqrt{28m/\rho},$$

$$C_L \triangleq \max_{0 \leq i \leq |m|} L_i, \quad L \triangleq \sum_{i \in [m]} L_i,$$

$$C_{M_i} \triangleq \frac{\eta}{2(L+c_1^* C_2^* + c_2^*)}.$$

Assume that the stepsize sequence \( \{ \gamma_k \} \) satisfies

$$\sum_{k=1}^{\infty} \gamma_k = \infty, \quad \sum_{k=1}^{\infty} (\gamma_k)^2 < \infty, \quad \text{and} \quad \gamma_0 \leq \min \left\{ \frac{1}{1+T}, \frac{1}{L(L+c_1^* C_2^* + c_2^*)} \right\}$$

where \( C_2^\ast \) and \( C_2^\prime \) are constants to be defined in the proof.

The merit function \( M_k \) is sublinearly convergent with the standard ergodic rate

$$\sum_{l=0}^{k} \gamma_l M_l \leq \max \{ C_{M_i}^2, 3\eta^{-2} \} \mathbb{E} \left[ F(x_\psi^{k+1}) \right] + L m \sigma^2 \sum_{l=0}^{k} (\gamma_l)^2$$

with \( C_{M}^2 \neq F(x^0) - F_m + C_1^2 + C_1^\prime \), and \( C_1^2, C_1^\prime > 0 \).

**Proof**: By the Descent Lemma applied to \( F \) at \( x_\psi^k \) and \( x_\psi^{k+1} \), yields

$$\mathbb{E} \left[ F \left( x_{(i)}^{k+1} \right) \right] \leq \mathbb{E} \left[ F \left( x_{(i)}^k \right) \right] + \gamma^k \psi_{ik} \mathbb{E} \left[ \nabla F \left( x_{(i)}^k \right), -z_{(i,k)}^k \right]$$

$$+ \frac{L(\gamma^k \psi_{ik})^2}{2} \mathbb{E} \left[ \| z_{(i,k)}^k \|^2 \right]$$

$$\leq \mathbb{E} \left[ F \left( x_{(i)}^k \right) \right] + \gamma^k \psi_{ik} \mathbb{E} \left[ \nabla F \left( x_{(i)}^k \right), -z_{(i,k)}^k \right]$$

$$+ \gamma^k \psi_{ik} \mathbb{E} \left[ \nabla F \left( x_{(i)}^k \right), z_{(i,k)}^k - z_{(i,k)}^k \right]$$

$$+ \frac{L(\gamma^k \psi_{ik})^2}{2} \mathbb{E} \left[ \| z_{(i,k)}^k \|^2 \right] + \mathbb{E} \left[ \| z_{(i,k)}^k \|^2 \right]$$

$$\leq \mathbb{E} \left[ F \left( x_{(i)}^k \right) \right] + L(\gamma^k)^2 \mathbb{E} \left[ \| z_{(i,k)}^k \|^2 \right]$$

$$+ \gamma^k \psi_{ik} \mathbb{E} \left[ \left( (\xi_{ik})^{-1} - z_{(i,k)}^k \right) - z_{(i,k)}^k \right]$$

$$+ \gamma^k \psi_{ik} \mathbb{E} \left[ \left( 11T \otimes I_m \right) z_{(i,k)}^k - \left( (\xi_{ik})^{-1} \right) z_{(i,k)}^k, -z_{(i,k)}^k \right]$$

$$+ \gamma^k \psi_{ik} \mathbb{E} \left[ \nabla F \left( x_{(i)}^k \right) - \left( 11T \otimes I_m \right) z_{(i,k)}^k \right].$$

Finally, we define the following merit function measuring the progresses of the algorithm toward stationarity:

$$M_k \triangleq \max \left[ \mathbb{E} \left[ \| \nabla F(x^k, \zeta) \|^2 \right], \| x^k - 1_m \otimes x^k \| \right].$$  \hspace{1cm} (7)
+ \gamma \psi E \left[ \frac{\gamma}{\eta} z_{k}^{i} - z_{k}^{i} \right]
\leq \mathbb{E} \left[ F(x_{k}^{i+1}) \right] - \frac{\gamma}{\eta} E_{t} + \gamma \psi E \left[ \frac{\gamma}{\eta} z_{k}^{i} - z_{k}^{i} \right]
+ \gamma \psi E \left[ \frac{\gamma}{\eta} z_{k}^{i} - z_{k}^{i} \right]
+ L(\gamma \psi)^{2} \mathbb{E} \left[ \left\| z_{k}^{i} - z_{k}^{i} \right\|^{2} \right]

where in the last inequality we used the Cauchy–Schwarz and Young’s inequality, \( ab \leq \frac{b^{2}}{2}\alpha + \frac{a^{2}}{2}\beta \), twice.

Set \( \beta_{1} = \beta_{2} = 2/\eta \), then

\[ \mathbb{E} \left[ F(x_{k}^{i+1}) \right] \leq \mathbb{E} \left[ F(x_{k}^{i}) \right] - \frac{\gamma}{2} E_{t} + L(\gamma \psi)^{2} \mathbb{E} \left[ \left\| z_{k}^{i} \right\|^{2} \right] \]

Invoking [1, Proposition 18] (note that the precise form of the update \( z_{k}^{i} \) does not affect the result), we can write

\[ \mathbb{E} \left[ \frac{1}{\sqrt{E_{k}^{i}}} \right] \leq C_{0} \rho/k \mathbb{E} \left[ \frac{1}{\sqrt{E_{k}^{i}}} \right] + C_{0} \sum_{l=0}^{k-1} \rho^{k-l-1} \mathbb{E} \left[ \left\| z_{l}^{i} \right\| \right] \]

\[ \leq C_{0} \rho/k \mathbb{E} \left[ \frac{1}{\sqrt{E_{k}^{i}}} \right] + C_{0} \sum_{l=0}^{k-1} \rho^{k-l-1} \mathbb{E} \left[ \sqrt{E_{k}^{i}} + \sqrt{m} \right] \]

which implies, by [1, Lemma 27], after taking full expectations, that

\[ \sum_{l=0}^{k} \mathbb{E} \left[ E_{l}^{i} \right] \leq C_{1} + C_{2} \sum_{l=0}^{k} (\gamma \psi)^{2} \left( \mathbb{E} \left[ E_{l}^{i} \right] + m\sigma^{2} \right) \]

with

\[ C_{2} = \frac{2C_{0}}{1 - \rho^{2}} \quad \text{and} \quad C_{1} > 0. \]

Similarly, from the proof of [1, Proposition 19], it can be seen that

\[ \mathbb{E} \left[ \sqrt{E_{k}^{i}} \right] \leq 3C_{0}C_{L} \sum_{l=0}^{k-1} \rho^{k-l-1} \left( \sqrt{E_{l}^{i}} + \sqrt{m} \sigma \right) \]

and so, again as in [1, Lemma 27]

\[ \sum_{l=0}^{k} \mathbb{E} \left[ E_{l}^{i} \right] \leq C_{1} + C_{2} \sum_{l=0}^{k} (\gamma \psi)^{2} \left( \mathbb{E} \left[ E_{l}^{i} \right] + m\sigma^{2} \right) \]

with

\[ C_{2} = \frac{36(C_{0}C_{L})^{2}(2(C_{0}C_{L})^{2} + (1 - \rho)^{2})}{(1 - \rho)^{4}} \]

and some \( C_{1} > 0 \).

Now, summing up (8), taking full expectations we get, using (5)

\[ \sum_{l=0}^{k} \left( \frac{\gamma}{2} - (L + C_{1} + C_{2} \gamma \psi)^{2} \mathbb{E} \left[ \left\| z_{l}^{i} \right\|^{2} \right] \right) + \sum_{l=0}^{k} \left[ \frac{\gamma}{\eta} \mathbb{E} \left[ E_{l}^{i} \right] + \frac{\gamma}{\eta} \mathbb{E} \left[ E_{l}^{i} \right] \right] \]

\[ \leq F(x_{0}) - F_{m} + \sum_{l=0}^{k} \frac{(\gamma \psi)^{2}}{2(L + C_{1} + C_{2} \gamma \psi)^{2}} \mathbb{E} \left[ \left\| z_{l}^{i} \right\|^{2} \right] \]

\[ + \sum_{l=0}^{k} \left[ \frac{\gamma}{\eta} \mathbb{E} \left[ E_{l}^{i} \right] + \frac{\gamma}{\eta} \mathbb{E} \left[ E_{l}^{i} \right] \right] \]

\[ \leq F(x_{0}) - F_{m} + C_{1} + C_{2} \gamma \psi \mathbb{E} \left[ \left\| z_{l}^{i} \right\|^{2} \right] \]

Therefore, for sufficiently small \( \gamma \), specifically

\[ \gamma \leq \frac{\eta}{2(L + C_{1} + C_{2} \gamma \psi)} \]

we have

\[ \sum_{l=0}^{k} \gamma \left[ E_{l}^{i} + E_{l}^{i} + E_{l}^{i} \right] \leq F(x_{0}) - F_{m} + C_{1} + C_{2} \gamma \psi \mathbb{E} \left[ \left\| z_{l}^{i} \right\|^{2} \right] \]

Following the same reasoning as [1, Lemma 25], we deduce

\[ M_{k} \leq \mathbb{E} \left[ \left\| \nabla F(x_{k}, \zeta) \right\|^{2} + 2\left\| x_{k} - 1 \otimes x_{k} \right\|^{2} \right] \]

\[ + 2 \mathbb{E} \left[ \left\| J(1 \otimes x_{k}) \right\|^{2} \right] \leq \mathbb{E} \left[ \left\| \nabla F(x_{k}, \zeta) \right\|^{2} + 4\left\| x_{k} - 1 \otimes x_{k} \right\|^{2} \right] \]

Furthermore

\[ \mathbb{E} \left[ \left\| \nabla F(x_{k}, \zeta) \right\| \leq \mathbb{E} \left[ \left\| \nabla F(x_{k}, \zeta) \right\| + L \left\| x_{k} - x_{k} \right\| \right] \right. \]

\[ \leq \mathbb{E} \left[ \left\| \nabla F(x_{k}, \zeta) - \bar{z}_{k} \right\| + ||\bar{z}_{k} - \left( \xi_{k}^{k-1} \right)^{-1} z_{k}^{k} || \right] \]

\[ + \left( \xi_{k}^{k-1} \right)^{-1} \left\| z_{k}^{k} \right\| + \frac{L}{\sqrt{m}} \mathbb{E} \left[ \left\| J(x_{k} - 1 \otimes x_{k}) \right\| \right] \]

\[ \leq C_{M} \sqrt{E_{k}^{i}} + \eta^{-1} \sqrt{E_{k}^{i}} + \eta^{-1} \sqrt{E_{k}^{i}} \]
and so $M_k \leq C^2 M, E^k + 3\eta^{-2}(E^k + E^k_y)$, which finally implies the statement of the theorem.

**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:

$$M_k = o\left(\frac{1}{k^{1-\alpha}}\right)$$

$$\mathbb{E}[E^k_i] = o\left(\frac{1}{k}\right)$$

$$\mathbb{E}[E^k_c] = o\left(\frac{1}{k}\right)$$

$$\mathbb{E}[E^k_z] = o\left(\frac{1}{k^{1-\alpha}}\right).$$

**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 $\gamma^k M^k = o(1/k^\alpha)$; thus, the form of $\gamma^k$ implies the rate for $M^k$.

The rates for $\mathbb{E}[E^k_i]$ and $\mathbb{E}[E^k_c]$ follow from (10) and (13), respectively, as the right-hand side is bounded, and thus, the sum on the left must be.

Finally, the bound for $\mathbb{E}[E^k_i]$ follows from the finiteness of the right-hand side of (15) and the form of $\gamma^k$.

We note how, similarly as in [25], 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 $O(1/\sqrt{k})$ [26]. Compared to the work in [9], which studies stochastic gradient descent with asychronous updates on an undirected graph, we see that, quite similarly, the topology does not influence the rate of convergence but appears in the required stepsize to achieve convergence, effectively requiring a smaller stepsize, and thus slower convergence, with looser connectivity. Their results are stated in terms of a fixed budget of minimal iterations on which the stepsize depends, whereas our work provides an overall rate and, hence, not directly comparable. However, similarly, it can still be seen that with the stepsize bounds, convergent behavior is expected to be seen only after an initial set of developing iterations and the presence of speedup is subtle and ambiguous, depending on multiple constants.

### IV. Experiments

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

The nonconvex optimization task is image classification using a neural network (CNN) architecture used in a Tensorflow tutorial [28], a commonly used model to test SGD-based algorithms requiring a Lipschitz smooth gradient. For each experiment, we used a stepwise 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 $[1, 0.8, 0.6, 0.4, 0.3, 0.2, 0.1, 0.05, 0.01]$, respectively. Unless stated, otherwise, we fixed the number of iterations that each node performs to 45 000 and stored results at every 30 s. The weight connection matrices $W$ and $A$ were randomly generated by the standard Erdős–Rényi random graph technique with the desired the connectivity probability. Note that by taking the tail of the stepsize to be a stepwise reduction by $\gamma^k = ((k - k_0)/s)$ for some $k_0, s \in \mathbb{N}$, the required stepsize conditions hold. The experiments are done in a Python environment using Tensorflow V2 and MPI (mpi4py) on RCI clusters over the cpu nodes.

#### A. 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., $w^{new} = \sum_i x_i$.

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

#### B. Graph Connectivity

This part experimentally studies the behavior of our algorithm for different percentages of graph connectivity. We fixed the

---

1 The access to the computational infrastructure of the OPVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 “Research Center for Informatics” is also gratefully acknowledged.
number of nodes to 16 and did experiments with \{0.5, 0.7, 0.9\} graph connectivity percentages. Fig. 2 shows the timewise convergence results for different graph connectivity percentages. Our observation is that as the graph topology gets more connected, the convergence results improve. We note that, as is standard with decentralized tracking based, and especially asynchronous, methods, we restricted our study to connectivity at least 0.5, as, otherwise, the convergence is known to be less stable and reliable.

**C. Parameter Deviations and Norm of Gradients**

Fig. 3 shows the average \(L_{\infty}\) distance of each node’s parameters and the nodewise average, i.e., \(1/K \sum_{i=1}^{K} ||x_i - x_{\text{avg}}||_{\infty}\) 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 min 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.

Fig. 4 presents the infinity norm of gradients on the whole dataset using the nodewise averaged parameters. As the graph connectivity percentage increases, the norm of the gradients gets smaller and reduces faster. We can see a gradual increase at the beginning time around 500, 300 min for 0.5, 0.9 graph connectivity percentages, respectively, which is consistent with our observation in parameter deviation plots in Fig. 3. The visible bound away from zero for the gradient norms in the low connectivity case, however, with a slow asymptotic trend toward zero, highlights that while theoretically long convergence to stationarity should occur, the time scale and tuning required can be practically challenging.

**D. Maximum Delay and Time per Iteration**

Fig. 5(a) and (b) shows maximum delay, and average time per iteration as a function of number of nodes for a fixed 0.7 graph connectivity percentage. We can observe that the maximum delay increased as the number of nodes increased and it was always bounded. Indeed, from Fig. 5(b), we can see that the average time per iteration also increased as the number of nodes increased. Fig. 6 represents the behavior of the maximum delay w.r.t graph connectivity percentage. We can observe that
the maximum delay has been increased by increasing the connectivity percentage and it was always bounded. The one-to-one increase in a delay relative to the total number of connections (nodes and connectivity) indicates the necessity of asynchronous communication—communication barriers would result in ever significant end-to-end training time increase.

V. CONCLUSION

In this article, 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 nonconvex image classification task.

REFERENCES

[1] Y. Tian, Y. Sun, and G. Scutari, “Achieving linear convergence in distributed asynchronous multi-agent optimization,” IEEE Trans. Autom. Control, vol. 65, no. 12, pp. 5264–5279, Dec. 2020.
[2] J. Tsitsiklis, D. Bertsekas, and M. Athans, “Distributed asynchronous deterministic and stochastic gradient optimization algorithms,” IEEE Trans. Autom. Control, vol. AC-31, no. 9, pp. 803–812, Sep. 1986.
[3] X. Lian, Y. Huang, Y. Li, and J. Liu, “Asynchronous parallel stochastic gradient for nonconvex optimization,” in Proc. Adv. Neural Inf. Process. Syst., 2015, pp. 2737–2745.
[4] Z. Peng, Y. Xu, M. Yan, and W. Yin, “Arock: An algorithmic framework for asynchronous parallel coordinate updates,” SIAM J. Comput., vol. 38, no. 5, pp. A2851–A2879, 2016.
[5] A. Agarwal and J. C. Duchi, “Distributed delayed stochastic optimization,” in Proc. Adv. Neural Inf. Process. Syst., 2011, pp. 873–881.
[6] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in Proc. Artif. Intell. Statist., 2017, pp. 1273–1282.
[7] S. U. Stich, “Local SGD converges fast and communicates little,” in Proc. Int. Conf. Learn. Representations, 2018.
[8] J. Wang, V. Tantia, N. Ballas, and M. Rabbat, “Slowmo: Improving communication-effcient distributed SGD with slow momentum,” in Proc. Int. Conf. Learn. Representations, 2019.
[9] X. Lian, W. Zhang, C. Zhang, and J. Liu, “Asynchronous decentralized parallel stochastic gradient descent,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 3043–3052.
[10] R. Xin, U. A. Khan, and S. Kar, “Variance-reduced decentralized stochastic optimization with accelerated convergence,” IEEE Trans. Signal Process., vol. 68, pp. 6255–6271, 2020.
[11] H. Tang, X. Lian, M. Yan, C. Zhang, and J. Liu, “D2: Decentralized training over decentralized data,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 4848–4856.
[12] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, “Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent,” Adv. Neural Inf. Process. Syst., vol. 30, pp. 5330–5340, 2017.
[13] A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. Stich, “A unified theory of decentralized SGD with changing topology and local updates,” in Proc. Int. Conf. Mach. Learn., 2020, pp. 5381–5393.
[14] A. Koloskova, T. Lin, and S. U. Stich, “An improved analysis of gradient tracking for decentralized machine learning,” Adv. Neural Inf. Process. Syst., vol. 34, pp. 11422–11435, 2021.
[15] T. Wu, K. Yuan, Q. Ling, W. Yin, and A. H. Sayed, “Decentralized consensus optimization with asynchrony and delays,” IEEE Trans. Signal Inf. Process. Netw., vol. 4, no. 2, pp. 293–307, Jun. 2018.
[16] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, “Push-sum distributed dual averaging for convex optimization,” in Proc. IEEE 51st Conf. Decis. Control, 2012, pp. 5453–5458.
[17] M. Assran, N. Loizou, N. Ballas, and M. Rabbat, “Stochastic gradient push for distributed deep learning,” in Proc. Int. Conf. Mach. Learn., 2019, pp. 344–353.
[18] A. Nedić and A. Ozdaglar, “Convergence rate for consensus with delays,” J. Glob. Optim., vol. 47, no. 3, pp. 437–456, 2010.
[19] C. N. Hadjicostis, N. H. Vaidya, and A. D. Domínguez-García, “Robust distributed average consensus via exchange of running sums,” IEEE Trans. Autom. Control, vol. 61, no. 6, pp. 1492–1507, Jun. 2016.
[20] M. Assran, A. Aytekin, H. R. Feyzmahdavian, M. Johansson, and M. G. Rabbat, “Advances in asynchronous parallel and distributed optimization,” Proc. IEEE, vol. 108, no. 11, pp. 2013–2031, Nov. 2020.
[21] M. S. Assran and M. G. Rabbat, “Asynchronous gradient push,” IEEE Trans. Autom. Control, vol. 66, no. 1, pp. 168–183, Jan. 2021.
[22] A. Oshevoy, I. C. Paschalidis, and A. Spiridonoff, “Robust asynchronous stochastic-gradient-push: Asymptotically optimal and network-independent performance for strongly convex functions,” J. Mach. Learn. Res., vol. 21, no. 58, 2020.
[23] J. Zhang and K. You, “Fully asynchronous distributed optimization with linear convergence in directed networks,” 2019, arXiv:1901.08215.
[24] J. Zhang and K. You, “Decentralized stochastic gradient tracking for empirical risk minimization,” 2019, arXiv:1909.02712.
[25] S. Pu, A. Oshevoy, and I. Paschalidis, “A sharp estimate on the transient time of distributed stochastic gradient descent,” IEEE Trans. Automat. Control, vol. 67, no. 11, pp. 5900–5915, 2021.
[26] S. Ghadimi and G. Lan, “Stochastic first-and zeroth-order methods for nonconvex stochastic programming,” SIAM J. Optim., vol. 23, no. 4, pp. 2341–2368, 2013.
[27] Y. LeCun, C. Cortes, and C. Burges, “MNIST handwritten digit database,” ATT Labs, vol. 2, 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist
[28] T. Tutorials. 2019. [Online]. Available: https://www.tensorflow.org/tutorials/quickstart/advanced

Vyacheslav Kungurtsev received the B.S. degree in mathematics from Duke University, Durham, NC, USA, in 2007, and the Ph.D. degree in mathematics with a specialization in computational science from the University of California, San Diego, La Jolla, CA, USA, in 2013. Subsequently, he was a Postdoctoral Fellow with KU Leuven as part of the “optimization for engineering” project. Since 2014, he has been a Postdoctoral Researcher and later a key researcher with the Czech Technical University, Prague, the Czech Republic. His research interests include mathematical programming and numerical optimization, with applications in machine learning and deep learning, process control engineering, distributed computing, and uncertainty quantification for mathematical physics.

Mahdi Morafah was born in Tehran, Iran. He received the B.S. degree in electrical engineering from Tehran Polytechnic University, Tehran, in 2019, and the M.S. degree in electrical and computer engineering in 2021 from the University of California, San Diego, La Jolla, CA, USA, where he is currently working toward the Ph.D. degree in electrical and computer engineering. He was a Machine Learning Researcher with Tesla autopilot team in 2021. His research interests include machine learning and deep learning, federated learning, continual learning, and optimization.
Tara Javidi (Fellow, IEEE) received the B.S. degree in electrical engineering from the Sharif University of Technology, Tehran, Iran, in 1996, and the M.S. degrees in electrical engineering (systems) and applied mathematics (stochastic analysis) and the Ph.D. degree in electrical engineering and computer science from the University of Michigan, Ann Arbor, MI, USA, in 1998, and 2002, respectively. In 2005, she joined the University of California, San Diego, CA, USA, where she is currently a Jacobs Family Scholar and Professor of Electrical and Computer Engineering and a Founding Faculty and Fellow of Halıcıoğlu Data Science Institute. She is also a founding Co-Director of UCSD Center for Machine-Intelligence, Computing and Security. Her research interests include theory of active learning, information acquisition and statistical inference, information theory with feedback, stochastic control theory, and wireless communications and communication networks.

Dr. Javidi was a Distinguished Lecturer of IEEE Information Theory Society during 2017–2018 as well as Communications Society during 2019–2020. She is also the Editor-in-Chief of the IEEE JOURNAL OF SELECTED AREAS IN INFORMATION THEORY. She and her Ph.D. students are recipients of the 2021 IEEE Communications Society and Information Theory Society Joint Paper Award. In addition to her many research awards, she has also received awards for her exceptional university service/leadership and for her sustained contributions to diversity.

Gesualdo Scutari (Fellow, IEEE) received the degree in electrical engineering and the Ph.D. (Hons.) degree in electrical engineering from the University of Rome “La Sapienza,” Rome, Italy, in 2001 and 2005, respectively. He is currently a Professor with the School of Industrial Engineering, Purdue University, West Lafayette, IN, USA. His research interests include continuous and distributed optimization, equilibrium programming, and their applications to signal processing and machine learning.

Dr. Scutari was a recipient of the 2013 NSF CAREER Award, the 2015 IEEE Signal Processing Society Young Author Best Paper Award, and the 2020 IEEE Signal Processing Society Best Paper Award. He was a Senior Area Editor of IEEE TRANSACTIONS ON SIGNAL PROCESSING and is currently an Associate Editor for the SIAM Journal on Optimization. He is one of the 2023 IEEE Signal Processing Society Distinguished Lecturers.