Walk for Learning: A Random Walk Approach for Federated Learning From Heterogeneous Data

Ghadir Ayache, *Member, IEEE*, Venkat Dassari, and Salim El Rouayheb

***Abstract*— We consider the problem of a Parameter Server (PS) that wishes to learn a model that fits data distributed on the nodes of a graph. We focus on Federated Learning (FL) as a canonical application. One of the main challenges of FL is the communication bottleneck between the nodes and the parameter server. A popular solution in the literature is to allow each node to do several local updates on the model in each iteration before sending it back to the PS. While this mitigates the communication bottleneck, the statistical heterogeneity of the data owned by the different nodes has proven to delay convergence and bias the model. In this work, we study random walk (RW) learning algorithms for tackling the communication and data heterogeneity problems. The main idea is to leverage available direct connections among the nodes themselves, which are typically "cheaper" than the communication to the PS. In a random walk, the model is thought of as a "baton" that is passed from a node to one of its neighbors after being updated in each iteration. The challenge in designing the RW is the data hetErogeneity and the uncertainty about the data distributions. Ideally, we would want to visit more often nodes that hold more informative data. We cast this problem as a sleeping multi-armed bandit (MAB) to design near-optimal node sampling strategy that achieves a variance reduced gradient estimates and approaches sub-linearly the optimal sampling strategy. Based on this framework, we present an adaptive random walk learning algorithm. We provide theoretical guarantees on its convergence. Our numerical results validate our theoretical findings and show that our algorithm outperforms existing random walk algorithms.**

***Index Terms*— Decentralized learning, distributed learning, random walk, incremental algorithms, multi-armed bandit.**

I. Introduction

A. Overview and Motivation

DISTRIBUTED Machine Learning has proven to be an important framework for training machine learning models without moving the available data from its local devices, which ensures privacy and scalability. Federated Learning (FL) has risen to be one of the main applications [1], [2], [3] that has been attracting significant research attention and has been deployed in real-world systems with millions of uses [2]. Other applications include learning in IoT networks [4], smart cities and healthcare [5], [6]. To see how a typical learning algorithm works in this setting, consider the FL setting in Fig 1. There is as Parameter Server (PS) (typically sitting in the cloud) and a number of nodes (phones, IoT-devices, smart sensors, etc.) each having its own local data. The PS wishes to learn a global model on all the data without moving the data away from its original owner. The algorithm would work in a batch SGD fashion. In each iteration, the PS samples a batch of nodes and sends the current model to it. Each node in this batch will update the model based on its local data and sends back its updated model to the PS. The PS then aggregates all the received models and starts over again.

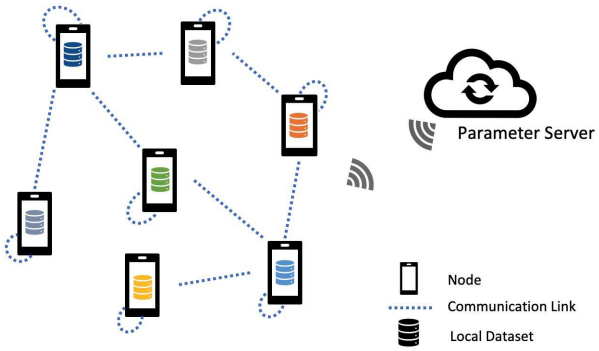

Fig. 1. Distributed Learning on Graph through local computations at the graph's nodes and through local communication between the connected nodes.

1) Locality vs. Heterogenity: One of the main bottlenecks here is the communication with the parameter server (PS) needed in each iteration to aggregate the updates sent by the nodes and to coordinate the learning process. A popular solution is to reduce the communication cost with the PS is to let each node perform several local model updates on its data before reporting back to the PS [7]. However, the local computations may induce local biases to the model and slow the convergence of the learning algorithm. This is due to the data that is heterogeneous across the different nodes, which imposes an inconsistency between the local and the global objectives [8], [9].

2) Random Walks: We propose Random Walks (RW) as a way to simultaneously achieve two seemingly opposing goals: extending the benefits of locality and mitigating the drawbacks of data heterogeneity. The idea being that instead of restricting local computations to the node itself, they can be extended

Manuscript received 17 April 2022; revised 5 September 2022; accepted 30 November 2022. Date of publication 14 February 2023; date of current version 17 March 2023. This work was supported in part by the Army Research Laboratory (ARL) under Grant W911NF-21-2-0272 and in part by the National Science Foundation (NSF) under Grant CNS-2148182. *(Corresponding author: Ghadir Ayache.)*

Ghadir Ayache and Salim El Rouayheb are with the Electrical and Computer Engineering Department, Rutgers University, Piscataway, NJ 08854 USA (e-mail: ghadir.ayache@rutgers.edu).

Venkat Dassari is with the U.S. Army Research Laboratory, White Oak, MD 20783 USA.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSAC.2023.3244250.

Digital Object Identifier 10.1109/JSAC.2023.3244250

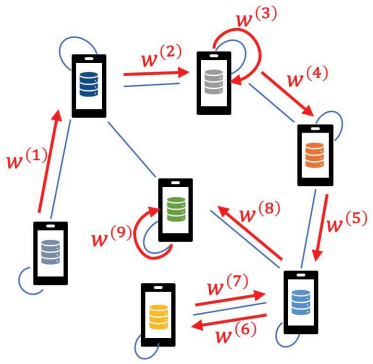

Fig. 2. Random Walk (RW) on the graph: the updated model $w^{(k)}$ is transmitted at time k. The self-loops at $k = 3$ and $k = 9$, indicate that the RW algorithm decided to make a second update at the same node. The model is reported back to the PS on a regular basis.

to its neighboring nodes. This is achieved by leveraging existing local connections among the nodes themselves, which are typically cheaper than communication to the PS [10], [11], [12]. We represent the local connections by a graph structure where each node is connected to a subset of other neighboring nodes. Thus, each node can exchange information with its neighbors and, through these local communications, information can propagate through the whole network. This setting can arise in mobile and edge networks, IoT applications and ad-hoc networks, to name a few.

In our proposed framework, the learning algorithm will run as a random walk where in each iteration the model gets updated at one of the nodes and then passed to one of its neighbors to take over the next update, as shown in Figure 2. The model is then passed to the PS on a regular basis and/or depending on the network resources to mitigate the communication cost to the PS.

Random walk learning algorithms have been well studied in the literature on optimization [13], [14], wireless networks [15] and signal processing [16]. What distinguishes this work is that it tackles the problem of designing random walk learning algorithms in the presence of arbitrary data heterogeneity across the nodes. Our main contribution is a random walk algorithm that, along with updating the model, it learns and adapts to the different nodes' distributions by carefully combining exploration and exploitation. Our main tool is the theory of Multi-Armed Bandit (MAB).

3) Random Walk via Multi-Armed Bandit: Typically, the distributions of the local data at each node are not known a priori. Therefore, we want to devise a random walk strategy that overcomes data heterogeneity by learning about the local data distributions along the way. The goal is to minimize the variance of the global objective gradient estimates computed locally by adjusting the nodes' sampling strategy. More specifically, at each iteration k, one has to design the probabilities with which the next node in the RW is chosen among the neighboring nodes. Note that these probabilities will depend on k to adapt to the information learned so far, leading to a time-varying RW. Therefore, the random walk will start with an exploration phase before gradually transitioning into an exploitation phase, once more robust estimates about the nodes' data are obtained.

We design the RW by casting our problem as Mulit-Armed bandit. The multi-armed bandit (MAB) is a learning framework to decide optimally under uncertainty [17], [18], [19]. It features N arms with unknown random costs (negative rewards). At each iteration, one pulls. The problem is to decide on which arm to pull each time so that the accumulated cost, called *regret*, after playing T times is minimized. Our solution is an algorithm that explores the different arms and, in parallel, it exploits the collected information so far, where it observes the outcome of playing an arm and uses it to tune its expected reward estimate to adjust future selections [17], [18]. In our distributed learning setting, we model the node selection in the RW as arm pulling in the MAB framework. At each iteration, the random walk picks a node in graph to activate for the next update, observes the update, and receives the local gradient as a cost.

Under this analogy, the performance of the learning algorithm is measured by the regret, which is the difference between the cost of a random walk with optimal transition matrix when all the distributions are known, and the cost of the nodes visited by the algorithm.

B. Contribution

In this paragraph, we summarize our contribution as follows:

- In this work, we propose a distributed learning algorithm to learn a model on the distributed data over the nodes in a graph. Our algorithm selects the nodes to update the model by an adaptive random walk on the network to address the statistical heterogeneity of distributed data.
- We model the random walk transition design as a sleeping multi-armed bandit problem to compete with the optimal transition probabilities that mitigate the high variance in the local gradients estimates.
- We provide the theoretical guarantee on the rate of convergence of our proposed algorithm approaching a rate $\mathcal{O}(1/\sqrt{T})$. The rate depends on the graph spectral property and the minimal gradient variance.
- Finally, we simulate our algorithm on real and synthetic data, for different graph settings and heterogeneity levels and show that it outperforms existing baseline random walk designs.

C. Prior Work

1) Random Walk Learning: Several works have studied random walk learning algorithms focusing on the convergence under different sets of assumptions. The works of [14], [20], and [21] established theoretical convergence guarantees for uniform random walks for different convex problem settings and using first-order methods. Later work [22] employs more advanced stochastic updates based on gradient tracking technique that uses Hessian information to accelerate convergence. The work of [23] proposed to speed-up the convergence by using non-reversible random walks. In [16], the authors studied the convergence of random walks learning

for the alternating direction method of multipliers (ADMM). In [24], the paper proposes to improve the convergence guarantees by designing a weighted random walk that accounts for the importance of the local data to speed up convergence. An asymptotic fundamental bound on the convergence rate of these algorithms was proven by [25] and it approaches $O(1/\sqrt{k})$ under convexity and bounded-gradient assumptions. From our MAB perspective, a common aspect of these algorithms is that they are purely exploitative. They require a priori information about the local data (e.g., gradient-Lipschitz constants, bounds on the gradients) to design a non-adaptive (time-invariant) random walk.

Random Walk algorithms belong to a more general class of decentralized learning algorithms where no central entity, such as a PS, is involved to handle the learning process. Gossip algorithms are another class of decentralized algorithms that are not based on random walks (see for e.g. [13], [26], [27], [28]). In (synchronous) gossip algorithms, at each round, each node updates and exchanges its *local model* with its neighboring nodes. Hence, in each iteration, all the nodes and all the links in the graph are activated. The goal of a Gossip algorithm is to ensure that all nodes, and not just the PS, learn the global model and assume convergence once a consensus is reached. Hence, it is less efficient in terms of computations and communication costs [24].

2) Data Heterogeneity: Recently, there has been lots of work addressing the problem of data heterogeneity especially in the FL literature. The data across the nodes is typically not generated in an iid fashion. A local dataset tends to be more personalized and biased towards its specific owner's profile. Therefore, multiple local updates on the global model can drift the global objective optimization towards its local one. This may slow down the convergence and can lead to converging to a suboptimal model [9], [29]. Several measures have been proposed in the literature to quantify statistical heterogeneity, which refers to this local vs. global objectives' inconsistency in the distributed data. The focus has been on quantifying the gap between the local update direction and the global one [8], [29], [30], [31], [32]. The proposed solutions vary between controlling the update direction [31] or the learning objective [29].

3) Multi-Armed Bandit Sampler: The multi-armed bandit (MAB) problem aims to devise optimal sampling strategies by balancing together exploration and exploitation [17], [18], [19], [33]. Results from MAB have been used in problems related to standard SGD training in a non-distributed settings [34], [35], [36], [37], [38]. The idea there is to use an MAB sampler to select more often the data points that can better guide the learning algorithm.

In the classical MAB setting there is no constraint on which node to sample (visit) at a given time (which arm to pull in the MAB language). However, in our case, we are restricted by the graph topology, so only neighboring nodes can be visited. To account for this constraint, we cast our problem as Sleeping Multi-Armed Bandit in which nodes that are not neighbors of the current nodes are assumed to be sleeping (not available) at the time of the sampling. The Sleeping MAB literature has studied various assumptions on sleeping reliance: independent availabilities, general availabilities, and adversarial availabilities. The lower bound on the regret is known to be $\Omega(\sqrt{NT})$ if we consider stochastic independent availabilities. For a harder sleeping-MAB setting with adversarial availabilities, the lower bound is $\Omega(N\sqrt{T})$ for N being the total number of nodes and T being the total number of rounds [33], [39], [40]. In our work, we model the RW design problem as dependent availabilities Sleeping MAB learning algorithm [40]. For the proof technicality, we use a harder upper bound on the performance that assume oblivious adversarial availibilities.

A related line of work is the work on importance sampling which can be thought of as a pure exploitation scheme with no exploration. The literature has studied different aspects of importance sampling using prior information on the local datasets (e.g., [41], [42], [43], [44]). For instance, in [45], the paper proposes to sample proportionally to the smoothness bounds of the local objectives. While the scheme in [41] suggests to select the data points based on the bounds of the gradients of the local objectives.

D. Organization

The rest of the paper is organized as follows. We present the problem setup in Section II. In Section III, we present the detailed random walk learning algorithm. In Section IV, we provide the optimal sampling scheme and its theoretical motivation. In Section V, we outline the analogy between the RW design problem and the sleeping MAB problem. In Section VI, we present the MAB RW learning algorithm and the main theorem on its convergence. Moreover, we provide the technical definitions and assumptions used into the main theorem proof in Section VII. Finally, we provide numerical results on the convergence of our proposed algorithm in Section VIII. The full proofs of the technical results are deferred to the appendices.

II. Setup

A. Network Model

We represent a network of N nodes by an undirected graph $\mathcal{G}(V, E)$ with $V = \{1, \ldots, N\}$ being the set of nodes and E being the set of edges such that $E = \{(i, j) \in V \times V,$ if i is connected to $j\}$. Since the graph is undirected, then $\forall (i, j) \in E,$ we have $(j, i) \in E$. Any two connected nodes i and j are called neighbor nodes and we denote it by $i \sim j$. Moreover, we assume that all the nodes have self-loops, thus, $\forall i,\ (i, i) \in E$.

B. Data Model

We assume that every node i owns a dataset $\mathcal{D}_i$ of size n such that $\mathcal{D}_i = \left\{\xi_{i,j} := (x_{i,j}, y_{i,j}) \in \mathbb{R}^d \times \mathbb{R} \text{ for } j \in [n]\right\}$, which is sampled from an unknown local distribution Π_i.

C. Learning Objective

Our goal is to minimize a global objective function $F(w)$ where $w \in \mathcal{W} \subset \mathbb{R}^d$, $\mathcal{W}$ being the feasible set assumed to be

closed and bounded. The objective function $F(.)$ represents the empirical mean of local losses on the distributed data over the graph of N nodes. Therefore, we are looking to solve the following problem:

$$\min_{w\in\mathcal{W}}\left\{F(w):=\frac{1}{N}\sum_{i=1}^{N}F_i\left(w\right)\right\}, \tag{1}$$

where the function F_i is the local objective at the node i and it is defined as

$$F_i\left(w\right)=\mathbb{E}_{\xi_i}\left[F_i\left(w;\xi_i\right)\right] \text{ for } \xi_i\sim\Pi_i. \tag{2}$$

The optimal model is denoted by w^* and defined as follows:

$$w^*=\arg\min_{w\in\mathcal{W}}F(w).$$

D. Data Heterogeneity

The data distributions across the nodes of the network are assumed to be arbitrary. Therefore, when a node performs a local update on the global model it may bias it by its local dataset that may not be a good representative of the global learning objective. Multiple definitions have been recently proposed to quantify the degree of local heterogeneity in distributed systems [9], [29], [30], [32]. These definitions focus on the variance of the local gradients with respect to the global gradient at a given model w. In our work, we adopt the definition used by [31] as stated below.

Definition 1 (Data Heterogeneity): The local objectives F_is are (α,σ)-locally dissimilar at w if

$$\mathbb{E}_i\left[\left\|\nabla F_i\left(w\right)\right\|_2^2\right]\leq\alpha^2+\sigma^2\left\|\nabla F\left(w\right)\right\|_2^2,$$

for $\alpha\geq 0$, $\sigma\geq 1$, $\mathbb{E}_i$ is the expectation over the nodes and $\nabla F_i\left(w\right)$ is the gradient at node i. For $\alpha=0$ and $\sigma=1$, we restore the homogeneous case.

This definition is a generalization of other definitions [29], [32].

E. Model Update

In distributed learning applications, the nodes are assumed to operate on a limited computational budget as they tend to be personal devices or hard-to-reach sensors.. Thus, we focus in our analysis on first-order methods using stochastic gradient descent.[1]

Thus, the model update at round k will be as follows

$$w^{(k+1)}=\mathbf{\Pi}_{\mathcal{W}}\left(w^{(k)}-\gamma^{(k)}\frac{1}{p^{(k)}(i^{(k)})}\hat{\nabla}F_{i^{(k)}}\left(w^{(k)}\right)\right), \tag{3}$$

where $\gamma^{(k)}$ is the step size, $\hat{\nabla}F_{i^{(k)}}\left(w^{(k)}\right)$ is an unbiased estimate of the local gradient at node $i^{(k)}$ computed on a uniformly sampled data point from $\mathcal{D}_{i^{(k)}}$, $p^{(k)}(i^{(k)})$ is the probability of picking node $i^{(k)}$ at round k, and $\mathbf{\Pi}_{\mathcal{W}}$ is the projection operator onto the feasible set $\mathcal{W}$. For our convergence analysis, we will need the following technical assumptions.

[1]Our work is applicable to any iterative algorithm that uses an unbiased descent update [46], [47], [48].

Assumption 1: For every node $i\in[N]$, the local loss function $F_i(.):\mathcal{W}\rightarrow\mathbb{R}^d$ is differentiable and convex function on the closed bounded domain $\mathcal{W}$.

Assumption 2: The step size $\gamma^{(k)}$ is decreasing and satisfies the following

$$\sum_{k=1}^{\infty}\gamma^{(k)}=+\infty \quad \textit{and} \quad \sum_{k=1}^{\infty}\ln k.(\gamma^{(k)})^2<+\infty. \tag{4}$$

Assumption 3 (Bounded Gradient): There exits a constant D such that, $\forall i\in[V]$ and $\forall w\in\mathcal{W}$, we have $\|\nabla F_i(w)\|_2^2\leq D$.

The last assumption is actually a result that follows from the functions F_i's being convex on a closed bounded subset $\mathcal{W}\subset\mathbb{R}$. A complementary proof can be found in [24].

III. Random Walk Algorithm

Our objective is to design a random walk on the graph $\mathcal{G}$ algorithm to learn the optimal model w^*. The algorithm starts uniformly at random at an initial node in the graph, say $i^{(0)}$, with an initial model $w^{(0)}$ also sampled uniformly at random from the feasible set $\mathcal{W}$. Let $i^{(k)}$ be the node visited (active) at the k^{th} round of the algorithm, $k=0,1,\ldots,T$. At each round k, the active node $i^{(k)}$ will receive the latest model update $w^{(k)}$ from a neighbor node $i^{(k-1)}$, that was active at the previous round. Then, the model $w^{(k)}$ is updated via a gradient descent update using data sampled from the local dataset of node $i^{(k)}$.

The main question we are after is how to design the transition probabilities defining the random walk, which govern how the RW is sampling the nodes in the graph. In addition to the explicit objective of learning the model, the random walk will simultaneously learn information about the heterogeneity of each node's data. Therefore, as the random walk progresses, it can adapt with the information gained on the importance of a given node's data to speed up the convergence. For this reason, we allow the transition probabilities of the random walk to adapt over time (algorithm rounds). We denote by $p^{(k)}(i)$ the probability distribution to select the node i to be active at round k. We denote by $P^{(k)}$ the transition matrix at time k. Therefore, we have $P^{(k)}(i,j)>0$ if $j\sim i$ and $P^{(k)}(i,j)=0$ otherwise. Moreover, we have $p^{(k)}=p^{(0)}P^{(1)}\ldots P^{(k)}$ and $p^{(m,k)}=p^{(0)}P^{(m+1)}\ldots P^{(k)}$.

IV. Node Sampling Strategy

We aim to design a sampling strategy that mitigates the effect of heterogeneity on the performance of the learning algorithm. Such strategy is constrained by an environment with two essential properties to consider: 1) The node sampling is restricted by the topology of the graph where the node to pick next has to be connected to the current active node; 2) No full information about the distributed heterogeneous data is available except what has been learned in the rounds so far and what can be shared among neighbor nodes.

In our algorithm, each node i is sampled (visited) with probability $p^{(k)}(i)$ at round k. And the gradient is computed on one data point sampled uniformly among the n local data points at the visited node. A crucial quantity for our analysis is

the second moment of the unbiased gradient estimate at round k which is

$$\mathbb{E}\left[\left\|\hat{\nabla}F_{i^{(k)}}\left(w^{(k)}\right)\right\|_2^2\right] = \sum_{i\in[N]}\frac{1}{p^{(k)}(i)}\sum_{j=1}^{n}\frac{1}{n^2}\left\|\nabla F_i\left(w^{(k)};\xi_{i,j}\right)\right\|_2^2. \quad (5)$$

This quantity affects the convergence rate of the Random Walk SGD algorithm as we show in equation (6) (see the Appendix for the details)

$$\mathcal{O}\left(\left(\sum_{k=1}^{T}\gamma^{(k)}\right)^{-1}\sum_{k=1}^{T}\mathbb{E}\left[\left\|\hat{\nabla}F_{i^{(k)}}\left(w^{(k)}\right)\right\|_2^2\right]\right). \quad (6)$$

Thus, the second moment of the gradients' updates imposes a burden on the convergence, especially in heterogeneous data settings where the diversity of the gradients is high. This dependence on the second moment is a common property of SGD based algorithms and has been well studied in the literature. Variance reduction techniques via importance sampling have been proposed to improve the convergence guarantees [41], [42], [44], [45], [49].

Our goal is to design the node sampling strategy to approach the optimal probability that minimizes the convergence bound [41], [42], [45] given by

$p^{(k)}(i) \propto \sqrt{g_i^{(k)}\left(w^{(k)}\right)}$, such that

$$g_i^{(k)}\left(w^{(k)}\right) := \sum_{\xi_{i,j}\in\mathcal{D}_i}\frac{1}{n^2}\left\|\nabla F_i\left(w^{(k)};\xi_{i,j}\right)\right\|_2^2.$$

Note that computing the $g_i^{(k)}$'s is very costly since it requires computing the gradients related to all the data points owned by the node and its neighbors. Moreover, the $g_i^{(k)}$'s need to be re-computed at the new model $w^{(k)}$ at each iteration. Instead, we propose to estimate in each iteration the $g_i^{(k)}$'s using the already computed gradients for the update step in (3). Therefore, at each iteration, the random walk has a double-fold objective: (i) update the model in each iteration and (ii) refine the estimates of the $g_i^{(k)}$'s by adjusting the RW level of exploitation vs. exploration using tools from the theory of sleeping multi-armed bandit.

V. Sleeping Bandit for RW Node Sampling

The multi-armed bandit (MAB) problem [19], [50] is a decision framework that features a set of N arms, where each arm $i \in [N]$ has an unknown cost $c^{(k)}(i)$ at round k. A player selects a sequence of arms $i^{(0)}, i^{(1)}, \ldots$ up to the final round T (also called called horizon) of the algorithm. The goal is to design an arm selection strategy to minimize the accumulated regret $R(T)$ over the total number of rounds T:

$$\mathrm{R}(T) = \sum_{k=1}^{T}\left(\mathbb{E}\left[c^{(k)}\left(i^{(k)}\right)\right] - \min_{i\in[N]}\mathbb{E}\left[c^{(k)}(i)\right]\right). \quad (7)$$

The first term in the regret, $\mathbb{E}\left[c^{(k)}\left(i^{(k)}\right)\right]$, is the average cost of the arm selected by the player. The second term, $\min_{i\in[N]}\mathbb{E}\left[c^{(k)}(i)\right]$, is the "best" arm with the minimum cost that could have been selected were the costs known. The expectation is taken over the selection strategy and the cost randomness.

TABLE I
Our Proposed Analogy Between the Sleeping MAB and the Random Walk Design Problems

MAB	RW Learning on Graph
Arm	Node
Action	Select the next node in the RW
Cost	Variance of the local gradient at the selected node as shown in (8)
Regret	Gap between the accumulated variance and the minimal variance under the optimal transition probabilities in full information setting as shown in (9)

Our work is based on establishing an analogy between MAB and RW. This allows us to use results from the vast literature on MAB to design the RW in order to speed up the learning process. To that end, we think of each node as an arm, and visiting a node in the RW as selecting an arm in the MAB problem. What is not clear in this analogy is what the cost of visiting a node and updating the model is. Based on the discussion in section IV and the the upper bound in (6), minimizing the accumulated variance of the gradient serves to tighten the convergence guarantees. Thus, we propose the cost of visiting a node i, at a given round k in our Random Walk algorithm, to be:

$$c^{(k)}(i) = \sum_{\xi_{i,j}\in\mathcal{D}_i}\frac{1}{n^2}\left\|\nabla F_i\left(w^{(k)};\xi_{i,j}\right)\right\|_2^2. \quad (8)$$

However, this analogy between "standard" MAB and RW cannot be fully established here. That is because in MAB any arm can be selected at any time. Whereas in RW only neighboring nodes can be visited in each iteration. To take into account the graph topology, we consider a variant of the standard MAB called sleeping MAB, where in each iteration only a subset or arms is available (the rest are sleeping) [33], [39], [40]. Moreover, the available nodes to select from are the ones that are connected to the currently visited node. In Table I, we summarise this analogy between MAB and RW.

Within this sleeping multi-armed bandit framework, our goal is to minimize the regret given the available arms and approach the best node sampling strategy denoted by $\pi : 2^{[N]} \to [N]$, which is a mapping from a set of available arms $\mathcal{N}^{(k)}$ to a selected arm. The goal is to minimize the following regret

$$\mathrm{R}(T) = \sum_{k=1}^{T}\left(\mathbb{E}\left[c^{(k)}\left(i^{(k)}\right)\right] - \min_{\pi}\sum_{k=1}^{T}\mathbb{E}\left[c^{(k)}\left(\pi\left(\mathcal{N}^{(k)}\right)\right)\right]\right), \quad (9)$$

where the cost $c^{(k)}$ is defined in (8), the expectation is taken w.r.t. the availabilities and the randomness of the player's strategy, and $\mathcal{N}^{(k)}$ is the set of available nodes at time k which consists of the neighbors of the currently visited node. Therefore, the regret function is defined as the gap between

Algorithm 1 Sleeping MAB Random Walk SGD

Input: Exploration parameter $\lambda^{(k)}$. Learning parameter $\eta = \sqrt{\frac{\log N}{NT}}$. Horizon T. Graph $\mathcal{G}(E, V)$.
Initialization: Initial control weight $q^{(0)}(i) = 1\ \forall i \in [N]$, Initial model $w^{(0)}$ chosen uniformly at random from $\mathcal{W}$. Starter node $i^{(0)}$ chosen uniformly at random from $[N]$.
for $k = 1$ **to** T **do**
  Compute $P^{(k)}(i^{(k-1)}, i) \propto q^{(k-1)}(i)\ \ \forall i \in \mathcal{N}\left(i^{(k-1)}\right)$ and $P^{(k)}(i^{(k-1)}, i) = 0$ otherwise.
  Choose a neighbor node $i^{(k)} \sim P^{(k)}(i^{(k-1)}, .)$.
  Choose $\xi_{i,j}$ uniformly at random from $\mathcal{D}_{i^{(k)}}$ and compute $\hat{\nabla} F_{i^{(k)}}\left(w^{(k)}\right)$.
  Compute the cost estimate $c_i^{(k)}$.
  Update the model using the SGD update in (3).
  Update the control weight $q^{(k)}(i^{(k)})$ using (10).
end for

the local variance of the local gradient estimate implied by the our selection strategy and the minimal variance that requires full information about the local datasets.

In [33], [39], and [19], it was shown that one can achieve a sublinear regret $\mathcal{O}(\sqrt{T})$ for sleeping MAB and it is asymptotically optimal. This is achieved by applying the EXP3 algorithm. Initially, the algorithm assigns equal importance to all arms. Then, at every round, the player receives the subset of non-sleeping arms, selects one among them, and observes the outcome of the chosen arm. The player then updates its cost estimation and keeps track of the empirical probability of the appearance of a given arm in the non-sleeping set. The goal of the player is to balance between exploration and exploitation and gradually shifts to exploitation as the costs estimates become more robust after playing enough rounds.

The multi-armed bandit modeling implies an algorithm design on the random walk that guarantees a sublinear decaying of the regret in (9) such that $\lim_{T\to\infty} \frac{\text{Regret}(T)}{T} = 0$, thus, it approaches asymptotically the optimal transition scheme of the random walk.

VI. Main Results

In this section, we summarize our main technical results. First, we present the details of our Sleeping Multi-Armed Bandit Random Walk SGD algorithm in Algorithm 1. Second, we prove in Theorem 1 that the proposed algorithm has an asymptotically optimal convergence rate.

A. Algorithm

Algorithm 1 leverages the analogy between Sleeping MAB and RW that we established in the previous section to design the RW learning algorithm. In the literature of Sleeping MAB [40], there are two versions of the EXP3 algorithm based on the availabilities of the arms: dependent versus independent availabilities. The case with dependent availabilities fits our RW model since the graph structure dictates the joint availability of any set of nodes.

In Algorithm 1, each node i keeps an accumulated control (importance) value $q^{(k)}(i)$ of the observed cost up to round k. The nodes with higher values will be favored in the selection.

In each round of the algorithm, the active node has to make a decision to select a neighboring node to carry the next update. In order to do that, the active node receives the control value from each neighboring node, and does the selection proportionally to the control values $q^{(k-1)}(i)$.

The selection starts with pure exploration using uniform control values and keeps refining it with time given the observed average cost. The selected node will turn active, samples one of its local data point uniformly at random, performs the update in (3) and computes the cost estimate based on the local gradient.

Each node i keeps tracks of the empirical estimate $\bar{P}^{(k)}(i) = \frac{1}{k}\sum_{t=1}^{k}\left(P^{(t)}(i^{(t-1)}, i)\right)$. The exploration is implicitly adjusted by a decreasing exploration parameter $\lambda^{(k)} = \sqrt{\frac{2^{N+2}}{k}\ln(T) + \frac{2^{N+2}}{3k}\ln(T)}$. At early stage of the training $\lambda^{(k)}$ gives less importance to the observed cost contribution. Lastly the control value is updated as follows

$$q^{(k)}\left(i^{(k)}\right) = q^{(k-1)}\left(i^{(k)}\right)\exp\left(-\eta\frac{c_i^{(k)}\left(i^{(k)}\right)}{\bar{P}^{(k)}\left(i^{(k)}\right) + \lambda^{(k)}}\right). \tag{10}$$

Each round of the Random Walk algorithm consists of one local update followed by the updated model passage to a neighbor node that uses the communicated importance weights. Thus, for a total number of round T, the communication cost follows an order of $\mathcal{O}(T.d)$ where d, is the dimension of the model.

B. Convergence Guarantees

Theorem 1: Under assumptions 1, 2 and 3, for a connected graph $\mathcal{G}$, particularly for $\gamma^{(k)} = \frac{1}{k^q}$ for $\frac{1}{2} < q < 1$, the convergence rate of Algorithm 1 is as follows:

$$\mathbb{E}\left[F\left(\bar{w}^{(T)}\right) - F\left(w^*\right)\right] \leq \frac{\frac{c.D^2}{\ln(1/\mu_{\mathcal{G}})} + N.E + \frac{1}{2}R + 3C^*}{T^{1-q}},$$

where,

$$\bar{w}^{(T)} = \frac{\sum_{k=1}^{T}\gamma^{(k)}w^{(k)}}{\sum_{t=1}^{T}\gamma^{(t)}}, \quad C^* = \min_p \sum_{k=1}^{T}\mathbb{E}_p\left[c^{(k)}(i)\right],$$

$$R = \frac{1}{2}\left\|w^{(0)} - w^*\right\|_2^2, \quad \text{and } E = \sum_{k=1}^{T}\gamma^{(k)}\left(\frac{1}{2k} + \frac{1}{\sqrt{k}}\right).$$

Moreover, c is a constant function of the convexity constants and the step size and $\mu_{\mathcal{G}}$ is the spectral norm of the transition matrix defined in Section VII.

To better understand its significance, one must compare it with other first-order random walk algorithms, such as uniform node sampling. For all these algorithms, one can get a similar bound as in Theorem 1 with the same constants (depending on the data and graph topology, etc.). The only difference would be in the term C^* which is the cumulated variance of the gradients corresponding to the optimal sampling strategy p^*. Any other node sampling strategy p will lead to a higher cost

C and a looser bound. Of course, here we are optimizing the upper bound which we are taking as a proxy for the actual performance. Our numerical results in Section VIII substantiate our theoretical conclusions and show that our proposed algorithm outperform other baselines.

VII. Proof Outline

We outline here the different steps needed to establish the result in Theorem 1. The details can be found in the Appendix. First, we state the results on the regret rate of the sleeping multi-armed bandit selection scheme used in Algorithm 1. Furthermore, we show that the multi-armed bandit random walk is strongly ergodic which is an essential assumption for the convergence of our algorithm.

Lemma 2 (Regret Rate of Sleeping Multi-Armed Bandit Sampler): Let $C^* = \min_p \sum_{k=1}^{T} \mathbb{E}\left[c^{(k)}(i)\right]$ *is the minimal cost if the optimal transition scheme is known. Under algorithm 1, The multi-armed bandit sleeping algorithm approximates the optimal cost asymptotically as follows*

$$\lim_{T\to\infty} \frac{1}{T}\left(\sum_{k=1}^{T} \mathbb{E}\left[c^{(k)}(i^{(k)})\right] - 3C^*\right) \leq 0,$$

The proof follows the multiplicative weight approach for EXP3 algorithms introduced in [18].

Next, we state the definition on Strongly Ergodic Non-Homogeneous Random Walk [51]. The sleeping multi-armed bandit algorithm guarantees that this property applies on the sequence of employed transition, which in the end guarantees the convergence stated in Theorem 1.

Definition 2 (Strongly Ergodic Non-Homogeneous Random Walk [51]): We denote by $p^{(k)}(i)$ *the probability distribution to select the node* i *to be active at round* k*. We denote by* $P^{(k)}$ *the transition matrix at time* k*. Therefore, we have* $P^{(k)}(i,j) > 0$ *if* $j \sim i$ *and* $P^{(k)}(i,j) = 0$ *otherwise. Moreover, we have* $p^{(k)} = p^{(0)}P^{(1)}\dots P^{(k)}$ *and* $p^{(m,k)} = p^{(0)}P^{(m+1)}\dots P^{(k)}$*. A non-homogeneous random walk, with uniform starting distribution* $p^{(0)}$*, is called strongly ergodic if there exists a vector* p^* *such that for all* $m \geq 0$*,*

$$\lim_{k\to\infty} \left\| p^{(m,k)} - p^* \right\| = 0.$$

Lastly, we present the result on the rate of convergence of the transition probability distribution.

Proposition 3 (Convergence Non-Homogeneous Strongly Ergodic Random Walk):

The non-homogeneous random walk in Algorithm 1 is strongly ergodic. Thus, it exists a stochastic matrix P *such that* $\lim_{k\to\infty} \left\| P^{(k)} - P \right\| = 0$*. Moreover, it exists a stochastic matrix* Q *such that* $\left\| P^k - Q \right\| \leq c\beta_2^k$*, where* $\beta_1 = 1 > \beta_2 \geq \dots \geq \beta_N$ *are the eigenvalues of the matrix* P*. Moreover, it exists a function* $g(k) = \mathcal{O}(\sqrt{k})$ *such that*

$\lim_{k\to\infty} \min\left\{1/\mu_{\mathcal{G}}^k,\, g\,(k)\right\} \left\| P^{(0,\,k)} - Q \right\| = 0$, *where* $1 < 1/\mu_{\mathcal{G}} < \sqrt{1/\beta_2}$.

VIII. Simulations

In this section, we present the numerical performance of our proposed Multi-Armed Bandit Random Walk (RW) SGD algorithm described in Algorithm 1.

A. Baseline Algorithms

We compare the performance of our algorithm to three baselines, namely: (1) Uniform Random Walk, (2) Static Weighted Random Walk, and (3) Adaptive Weighted Random Walk.

1) The Uniform Random Walk: This algorithm assigns equal importance to all nodes in the network [20] imitating uniform sampling in centralized SGD. We implement the Metropolis Hasting (MH) decision rule to design the transition probabilities, so the random walk converges to a uniform stationary. The MH rule can be described as follows:

1) At the k^{th} step of the random walk, the active node $i^{(k)}$ selects uniformly at random one of its neighbors, say j, as a candidate to be the next active node. This selection gets accepted with probability

$$a_u\left(i^{(k)},\, j\right) = \min\left(1,\, \frac{\deg\left(i^{(k)}\right)}{\deg\left(j\right)}\right).$$

Upon the acceptance, we have $i^{(k+1)} = j$.

2) Otherwise, if the candidate node gets rejected, the random walk stays at the same node, i.e., $i^{(k+1)} = i^{(k)}$.

2) Static Weighted Random Walk: This algorithm assigns a static importance metric to each node that is proportional to the gradient-Lipschitz constant of the local loss function [45], [52]. The random walk is designed by the MH again with a stationary distribution that is proportional to the local gradient-Lipschitz constants. In order to achieve that stationary, the probability of acceptance looks as follows:

$$a_w\left(i^{(k)},\, j\right) = \min\left(1,\, \frac{L_j}{L_{i^{(k)}}} \frac{\deg\left(i^{(k)}\right)}{\deg\left(j\right)}\right). \tag{11}$$

3) Adaptive Weighted Random Walk: In this algorithm, we adapted the importance sampling scheme that is used in [41] and [43] for centralized settings. The importance is computed for each node i as the average of of gradients computed so far at that node which is at time k, $\frac{\sum_{t=1:i^{(t)}=i}^{k} \left\|\hat{\nabla} F_i\left(w^{(t)}\right)\right\|_2^2}{n_{i,k}}$, for $n_{i,k}$ is the total number of rounds when node i has been active up to k. We call it the pure Exploitation scheme.

B. Datasets and Comparison

Our simulations are run on both synthetic dataset and on real benchmarks to confirm our theoretical results. They show that a bandit based random walk in decentralized learning consistently outperforms existing random walk baselines that uses static or exploitation based importance estimation.

1) Synthetic Data: For each node i, we sample the dataset $\mathcal{D}_i$ from a a normal distribution with $\mathcal{N}\left([\mu_{i,1}, \mu_{i,2}], \sigma\mathbb{I}_2\right)$. We assigned manually a label to each node dataset such that half the nodes has the label $y_i = 1$ and the other half has the $y_i = -1$. We run our simulations on an Expander graph which known to be sparse (The Margulis-Gabber-Galil graph).[2]

[2]We call the generator function of the Python library NetworkX v2.8 [53].

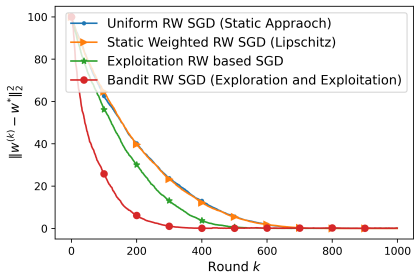

Fig. 3. Classification model trained on a Synthetic dataset distributed over an Expander graph of 100 nodes.

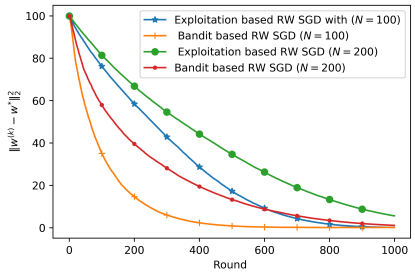

Fig. 4. Classification model trained on the Synthetic dataset. The same dataset is been distributed over a Expander graph of size 100 nodes and 200 nodes.

TABLE II

NUMBER OF ROUNDS TO REACH 0.45 TEST ACCURACY FORLOGISTIC REGRESSION ON MNIST AS WE VARY THE LEVEL OF SIMILARITY. BANDIT RW SGD IS CONSISTENTLY FASTER THAN UNIFORM RW SGD

Similarity	0%	10%	100%
Uniform RW SGD	202	87	61
Bandit RW SGD	138	44	34

2) MNIST Dataset and Fashion-MNIST: We run experiments on the MNIST dataset and the Fashion-MNIST to train a multi-class logistic regression model. We divide the data among the nodes as follows: for a level $s\%$ of similarity, each client has $s\%$ of its local dataset drawn i.i.d. from a shared pool of data. Another non-public pool of the data is sorted with respect to label and partitioned into non-overlapping chunks. Each node is assigned a different chunk that consists of its remaining $(100 - s)\,\%$ data [31].

Table II and Table IV report the results for different level of similarity from fully homogeneous to fully heterogeneous to highlight how different similarity levels affect the convergence of our algorithm vs. the oblivious uniform algorithm. Table II is on the MNIST dataset. The gap between both algorithms becomes wider once the system turns more heterogeneous. Table IV is on the Fashion-MNIST dataset. The gap between both algorithms becomes more significant as the system turns into more heterogeneous state.[3]

To elaborate on how our algorithm performs given the graph structure, we consider multiple scenarios of simulations where we use an Erdos-Renyi with different probabilities of connectivity that go from sparser to denser. Table V measures the performance given the probability of connectivity in the graph.

[3]The fact that the ratio between the number of iterations of the two algorithms is constant (roughly 2) is an artifact of the data and is not reproducible for other datasets. See for example Table IV which is on the Fashion-MNIST dataset that shows an increasing ratio with decreasing similarity.

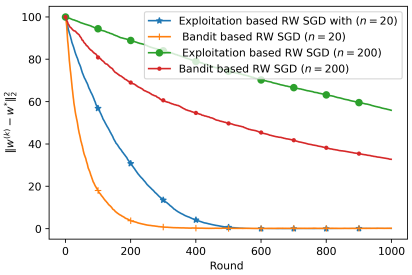

Fig. 5. Classification model trained on the Synthetic dataset. The local dataset size has been augmented from 20 to 200 in a Expander graph of size 100.

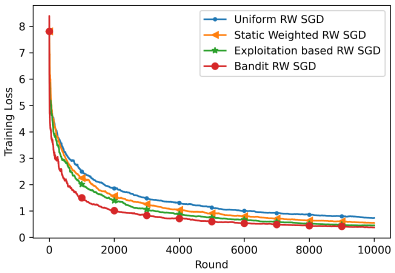

Fig. 6. Multi-class MNIST dataset of 10 classes distributed over 100 nodes with 0% similarity on Expander graph.

TABLE III

NUMBER OF ROUNDS TO REACH 0.45 TEST ACCURACY FORLOGISTIC REGRESSION ON FMNIST AS WE VARY THE LEVEL OF SIMILARITY. BANDIT RW SGD IS CONSISTENTLY FASTER THAN UNIFORM RW SGD

Similarity	0%	10%	100%
Uniform RW SGD	124	90	66
Bandit RW SGD	82	69	54

TABLE IV

NUMBER OF ROUNDS TO REACH 0.75 TEST ACCURACY FORLOGISTIC REGRESSION ON FMNIST FOR COMPLETE AND INCOMPLETE GRAPHS FOR 0% SIMILARITY. BANDIT RW SGD IS CONSISTENTLY FASTER THAN UNIFORM RW SGD

Similarity	Expander graph	Complete graph
Uniform RW SGD	190	143
Bandit RW SGD	169	139

TABLE V

NUMBER OF ROUNDS TO REACH 0.45 TEST ACCURACY FORLOGISTIC REGRESSION ON MNIST AS WE VARY THE GRAPH CONNECTIVITY PARAMETER. BANDIT RW SGD IS CONSISTENTLY FASTER THAN UNIFORM RW SGD. AS WE DECREASE THE PROBABILITY OFCONNECTIVITY, THE INCREASE IN THE NUMBER OF ROUNDS IS LESS SIGNIFICANT FOR THE BANDIT ALGORITHM

Probability of Connectivity	0.1	0.5	0.8
Uniform RW SGD	202	128	94
Bandit RW SGD	138	103	90

APPENDIX

C. Optimal Sampling

Here is a sketched proof on the optimal sampling to minimize the gradient variance in the full information setting [41]. Consider the optimization problem in hand which can be formulated as follows:

$$\min \sum_{i\in[N]} \frac{g_i^{(k)}\left(w^{(k)}\right)}{p^{(k)}(i)},$$
$$\text{such that} \quad p^{(k)}(i) \in (0,1)\,\forall\, i, \text{ and } \sum_i p^{(k)}(i) = 1.$$

The optimality conditions of the Lagrangian expression of the problem above gives the following:

$$-\frac{g_i^{(k)}\left(w^{(k)}\right)}{\left(p^{(k)}(i)\right)^2} + \eta = 0 \text{ and } \sum_i p^{(k)}(i) = 1,$$

where η is the Lagrange multiplier. Thus, by a simple algebraic manipulation, we get $p^{(k)}(i) = \frac{\sqrt{g_i^{(k)}\left(w^{(k)}\right)}}{\sum_j \sqrt{g_j^{(k)}\left(w^{(k)}\right)}}$.

D. Connection to the Heterogeneity Definition

The cost function in Equation (8) that we chose for our multi-armed bandit is an upper bound on the heterogeneity $\mathbb{E}_i\left[\left\|\nabla F_i(w)\right\|_2^2\right]$ defined in Definition 1. Below we clarify this connection:

Assume F_i for any $i \in [N]$ are convex and L-smooth, then we have

$$\begin{aligned}
&\mathbb{E}_i\left[\left\|\nabla F_i(w)\right\|_2^2\right] \\
&= \frac{1}{N}\sum_{i=1}^N \left\|\nabla F_i(w)\right\|_2^2 \\
&\le \frac{2}{N}\sum_{i=1}^N \left\|\nabla F_i(w^*)\right\|_2^2 + \frac{2}{N}\sum_{i=1}^N \left\|\nabla F_i(w) - \nabla F_i(w^*)\right\|_2^2 \\
&\overset{(a)}{\le} \frac{2}{N}\sum_{i=1}^N \left\|\nabla F_i(w^*)\right\|_2^2 + 4L\left(F(w) - F(w^*)\right).
\end{aligned}$$

(a) follows from Theorem 2.1.5 in [54].

Now taking $w = w^{(T)}$ in the upper bound above, the term $\left(F\left(w^{(T)}\right) - F\left(w^{(*)}\right)\right)$ depends on the cost which is the accumulated variance of the gradients at the sequence of selected nodes up to time T, as shown in Equation (6).

Definition 3: The local loss function F_i for each node $i \in V$ has an L_i-Lipschitz continuous gradient; that is, any $w, w' \in \mathcal{W}$, there exists a constant $L_i > 0$ such that

$$\left\|\nabla f_i(w) - \nabla f_i(w')\right\|_2 \le L_i \left\|w - w'\right\|_2.$$

Lemma 4: Under assumptions 1, 2 and 3, the Random Walk SGD algorithm, that uses the update in equation 3 for transition matrix P, has the following rate of convergence.

$$\begin{aligned}
&\mathbb{E}\left[F\left(w^{(T)}\right) - F(w^*)\right] \\
&\le \mathcal{O}\left(\left(\sum_{k=1}^T \gamma^{(k)}\right)^{-1} \sum_{k=1}^T \mathbb{E}\left[\left\|\hat{\nabla} F_{i^{(k)}}\left(w^{(k)}\right)\right\|_2^2\right]\right).
\end{aligned}$$

In order to prove Lemma 4 and Theorem 1, we present some technical results that we use in the proof. The proof techniques are essentially inspired by the work of [24] and they are adapted to the assumptions and setting of this work.

Lemma 5 (Convexity and Lipschitzness): If F_i is a convex function on an open subset $\Omega \subseteq \mathbb{R}$, then for a closed bounded subset $\mathcal{W} \subset \Omega$, there exists a constant $D_i \ge 0$, such that, for any $w_1, w_2 \in \mathcal{W}$,

$$\left|F_i(w_1) - F_i(w_2)\right| \le D_i \left\|w_1 - w_2\right\|_2.$$

We define $D = \sup_{i\in V} D_i$. Therefore,

$$\left|F_i(w_1) - F_i(w_2)\right| \le D \left\|w_1 - w_2\right\|_2.$$

A proof for Lemma 5 can be found in [55].

Corollary 1 (Boundedness of the Gradient): If F_i is a convex function on $\mathbb{R}$, then for a closed bounded subset $\mathcal{W} \subset \mathbb{R}$, $\left\|\nabla F_i(w)\right\|_2 \le D, \quad \forall w \in \mathcal{W}$.

Proof: Taking $v = w + \nabla F_i(w)$,

$$\begin{aligned}
D\left\|\nabla F_i(w)\right\|_2 &= D\left\|v - w\right\|_2 \\
&\overset{(a)}{\ge} \left|F_i(v) - F_i(w)\right| \\
&\overset{(b)}{\ge} \left\langle \nabla F_i(w), \nabla F_i(w)\right\rangle \\
&= \left\|\nabla F_i(w)\right\|_2^2.
\end{aligned}$$

(a) follows Lemma 5 and (b) follows from F_i being convex. ■

Now, we present the steps of the the proof:

$$\begin{aligned}
&\left\|w^{(k+1)} - w^*\right\|_2^2 \\
&= \left\|\mathbf{\Pi}_{\mathcal{W}}\left(w^{(k)} - \gamma^{(k)}\hat{\nabla} F_{i^{(k)}}\left(w^{(k)}\right)\right) - \mathbf{\Pi}_{\mathcal{W}} F(w^*)\right\|_2^2 \\
&\overset{(a)}{\le} \left\|w^{(k)} - \gamma^{(k)}\hat{\nabla} F_{i^{(k)}}\left(w^{(k)}\right) - w^*\right\|_2^2 \\
&= \left\|w^{(k)} - w^*\right\|_2^2 - 2\gamma^{(k)}\left\langle w^{(k)} - w^*, \hat{\nabla} F_{i^{(k)}}\left(w^{(k)}\right)\right\rangle \\
&\quad + \left(\gamma^{(k)}\right)^2 \left\|\hat{\nabla} F_{i^{(k)}}\left(w^{(k)}\right)\right\|_2^2
\end{aligned}$$

(a) follows from $\mathcal{W}$ being a convex closed set, so one can apply nonexpansivity theorem in [55].

For the next we use the convexity of F_i,

$$\begin{aligned}
&\left\|w^{(k+1)} - w^*\right\|_2^2 \\
&\le \left\|w^{(k)} - w^*\right\|_2^2 - 2\gamma^{(k)}\left(F_{i^{(k)}}\left(w^{(k)}\right) - F_{i^{(k)}}(w^*)\right) \\
&\quad + (\gamma^{(k)})\left\|\hat{\nabla} F_{i^{(k)}}\left(w^{(k)}\right)\right\|_2^2. \qquad (12)
\end{aligned}$$

Re-arranging the above equation gives

$$\begin{aligned}
&\gamma^{(k)}\left(F_{i^{(k)}}\left(w^{(k)}\right) - F_{i^{(k)}}(w^*)\right) \\
&\le \frac{1}{2}\left(\left\|w^{(k)} - w^*\right\|_2^2 - \left\|w^{(k+1)} - w^*\right\|_2^2\right) \\
&\quad + \frac{\left(\gamma^{(k)}\right)^2}{2}\left\|\hat{\nabla} F_{i^{(k)}}\left(w^{(k)}\right)\right\|_2^2. \qquad (13)
\end{aligned}$$

Summing (13) over k and using Assumption and the boundness of $\mathcal{W}$,

$$\sum\nolimits_k \gamma^{(k)} \left(F_{i^{(k)}}\left(w^{(k)}\right) - F_{i^{(k)}}\left(w^*\right)\right) \leq \frac{1}{2}\left\|w^{(0)} - w^*\right\|_2^2 + \sum\nolimits_k \left(\gamma^{(k)}\right)^2 \left\|\nabla F_{i^{(k)}}\left(w^{(k)}\right)\right\|_2^2. \quad (14)$$

Next we give some results we need on the Markov chain. We denote by μ the stationary distribution, P is the transition matrix and P^k is the k^{th} power of matrix P. We refer to i^{th} row of a matrix P by $P(i,:)$.

Lemma 6 (Convergence of Markov Chain [56]): Assume the graph $\mathcal{G}$ is connected with self-loop, therefore a random walk is aperiodic and irreducible, we have

$$\max_i \left\|\mu - P^k(i,:)\right\| \leq C\lambda_P^{(k)}$$

for $k > K_p$, where K_P is a constant that depends and λ_P and $\lambda_2(P)$ and C is a constant that depends on the Jordan canonical form of P.

Corollary 2: Using the previous lemma, we get

$$\max_i \left\|\mu - P^{T_k}(i,:)\right\| \leq C\lambda_P^{T_k} \leq \frac{1}{2k}$$

for $T^{(k)} = \min\{k, \max\{\frac{\ln(2Ck)}{\ln(1/\lambda_P)}, K_P\}\}$.

Here, we state the next corollary on the convergence of the random walk.

$$\begin{aligned}
&\gamma^{(k)}\mathbb{E}\left[F_{j^{(k)}}\left(w^{\left(k-T^{(k)}\right)}\right) - F_{j^{(k)}}\left(w^{(k)}\right)\right] \\
&\overset{(a)}{\leq} D\gamma^{(k)}\mathbb{E}\left\|w^{\left(k-T^{(k)}\right)} - w^{(k)}\right\| \\
&\overset{(b)}{\leq} D\gamma^{(k)}\mathbb{E}\left(\sum_{n=k-T^{(k)}}^{k-1}\left\|w^{(n+1)} - w^{(n)}\right\|\right) \\
&\overset{(c)}{\leq} D\gamma^{(k)}\sum_{n=k-T^{(k)}}^{k-1}\mathbb{E}\left(\left\|w^{(n+1)} - w^{(n)}\right\|\right) \\
&\overset{(d)}{\leq} D^2\gamma^{(k)}\sum_{n=k-T^{(k)}}^{k-1}\gamma^{(n)} \\
&\overset{(e)}{\leq} \frac{D^2}{2}\sum_{n=k-T^{(k)}}^{k-1}\left(\left(\gamma^{(n)}\right)^2 + \left(\gamma^{(k)}\right)^2\right) \\
&\leq \frac{D^2}{2}T^{(k)}\left(\gamma^{(k)}\right)^2 + \frac{D^2}{2}\sum_{n=k-T_k}^{k-1}\left(\gamma^{(n)}\right)^2.
\end{aligned}$$

(a) follows from Lemma 5, (b) using triangle inequality, (c) using linearity of expectation and (d) follows from the Cauchy–Schwarz inequality.
Now taking the summation over k:

$$\begin{aligned}
&\sum_k \gamma^{(k)}\mathbb{E}\left[F_{j^{(k)}}\left(w^{\left(k-T^{(k)}\right)}\right) - F_{j^{(k)}}\left(w^{(k)}\right)\right] \\
&\leq \sum_k \frac{D^2}{2}T_k\left(\gamma^{(k)}\right)^2 + \frac{D^2}{2}\sum_k\sum_{n=k-T_k}^{k-1}\left(\gamma^{(n)}\right)^2.
\end{aligned}$$

By simply using the assumption on the step size summability, the result is as follows:

$$\begin{aligned}
\sum_{k=K}^{\infty}\sum_{n=k-T^{(k)}}^{k-1}\left(\gamma^{(n)}\right)^2 &\leq \sum_{k=K}^{\infty} T^{(k)}\left(\gamma^{(k)}\right)^2 \\
&\leq \frac{2}{\ln(1/\lambda_P)}\sum_{k=K}^{\infty}\ln k.\left(\gamma^{(k)}\right)^2 \\
&< \infty. \quad (15)
\end{aligned}$$

Now, we compute the following lower bound:

$$\begin{aligned}
&\mathbb{E}_{j^{(k)}}\left[F_{j^{(k)}}\left(w^{\left(k-T^{(k)}\right)}\right) - F_{j^{(k)}}\left(w^*\right) \mid X_0, X_1, \ldots, X_{k-T^{(k)}}\right] \\
&= \sum_{i=1}^{N}\left(F_i\left(w^{\left(k-T^{(k)}\right)}\right) - F_i\left(w^*\right)\right) \\
&\quad \times P\left(j^{(k)} = i \mid X_0, X_1, \ldots, X_{k-T^{(k)}}\right) \\
&\overset{(a)}{=} \sum_{i=1}^{N}\left(F_i\left(w^{\left(k-T^{(k)}\right)}\right) - F_i\left(w^*\right)\right) \\
&\quad \times P\left(j^{(k)} = i \mid X_{k-T^{(k)}}\right) \\
&= \sum_{i=1}^{N}\left(F_i\left(w^{\left(k-T^{(k)}\right)}\right) - F_i\left(w^*\right)\right) \\
&\quad \times P^{T^{(k)}}\left[X_{k-T^{(k)}} \mid j^{(k)} = i\right] \\
&\overset{(b)}{\geq} \left(F\left(w^{\left(k-T^{(k)}\right)}\right) - F\left(w^*\right)\right) - \frac{N}{2k}. \quad (16)
\end{aligned}$$

(a) using Markov property and (b) using Lemma 2 in [23]. Therefore,

$$\begin{aligned}
&F\left(w^{\left(k-T^{(k)}\right)}\right) - F\left(w^*\right) \\
&\leq \frac{N}{2k} + \mathbb{E}_{j^{(k)}}\left[F_{j^{(k)}}\left(w^{\left(k-T^{(k)}\right)}\right) - F_{j^{(k)}}\left(w^*\right) \mid \right. \\
&\qquad \left. \times X_0, \ldots, X_{k-T^{(k)}}\right]
\end{aligned}$$

Taking the total expectation, we have

$$\begin{aligned}
&\gamma^{(k)}\mathbb{E}\left[F\left(w^{\left(k-T^{(k)}\right)}\right) - F\left(w^*\right)\right] \\
&\leq \frac{N\gamma^{(k)}}{2k} + \gamma^{(k)}\mathbb{E}\left[F_{j^{(k)}}\left(w^{\left(k-T^{(k)}\right)}\right) - F_{j^{(k)}}\left(w^*\right)\right]
\end{aligned}$$

Rearranging the equation above, we get:

$$\begin{aligned}
&\sum_k \gamma^{(k)}\mathbb{E}\left[F\left(w^{\left(k-T^{(k)}\right)}\right) - F\left(w^*\right)\right] \\
&\leq \sum_k \frac{N\gamma^{(k)}}{2k} + \sum_k \gamma^{(k)}\mathbb{E}\left[F_{j^{(k)}}\left(w^{\left(k-T^{(k)}\right)}\right) - F_{j^{(k)}}\left(w^*\right)\right] \\
&\leq \sum_k \frac{N\gamma^{(k)}}{2k} + \frac{1}{2}\left\|w^{(0)} - w^*\right\|_2^2 \\
&\quad + \sum\nolimits_k \left(\gamma^{(k)}\right)^2 \mathbb{E}\left[\left\|\nabla F_{i^{(k)}}\left(w^*\right)\right\|_2^2\right]
\end{aligned}$$

Next, we get a bound on

$$\sum_k \gamma^{(k)} \mathbb{E}\left[F\left(w^{(k)}\right) - F\left(w^{(k-T^{(k)})}\right)\right].$$

$$\begin{aligned}
&\gamma^{(k)} \mathbb{E}\left[F\left(w^{(k)}\right) - F\left(w^{(k-T_k)}\right)\right] \\
&\overset{(a)}{\leq} ND\gamma^{(k)} \mathbb{E}\left\|w^{(k)} - w^{(k-T_k)}\right\| \\
&\overset{(b)}{\leq} ND\gamma^{(k)} \mathbb{E}\left(\sum_{n=k-T_k}^{k-1} \left\|w^{(n+1)} - w^{(n)}\right\|\right) \\
&\overset{(c)}{\leq} ND\gamma^{(k)} \sum_{n=k-T_k}^{k-1} \mathbb{E}\left(\left\|w^{(n+1)} - w^{(n)}\right\|\right) \\
&\overset{(d)}{\leq} ND^2\gamma^{(k)} \sum_{n=k-T_k}^{k-1} \gamma^{(n)} \\
&\overset{(e)}{\leq} \frac{ND^2}{2} \sum_{n=k-T_k}^{k-1} \left(\left(\gamma^{(n)}\right)^2 + \left(\gamma^{(k)}\right)^2\right) \\
&\leq \frac{ND^2}{2} T_k \left(\gamma^{(k)}\right)^2 + \frac{ND^2}{2} \sum_{n=k-T_k}^{k-1} \left(\gamma^{(n)}\right)^2. \qquad (17)
\end{aligned}$$

(a) follows from Lemma 4, (b) using triangle inequality, (c) using linearity of expectation, (d) follows Corollary 1 and (e) follows from the Cauchy–Schwarz inequality. The upper bound summability over k follows from previous discussion in equation (15).

Combining with the results in (14) and (15), we get

$$\begin{aligned}
&\mathbb{E}\left[F\left(w^{(k)}\right) - F\left(w^*\right)\right] \\
&\leq \frac{\sum_{k=1}^T \frac{N\gamma^{(k)}}{2k} + \frac{C.D^2}{\ln(1/\lambda_P)} + \frac{1}{2}\left\|w^{(0)} - w^*\right\|_2^2}{\sum_{k=1}^T \gamma^{(k)}} \\
&\quad + \frac{\left(\gamma^{(0)}\right)^2 \sum_{k=1}^T \mathbb{E}\left[\left\|\hat{\nabla} F_{i^{(k)}}\left(w^{(k)}\right)\right\|_2^2\right]}{\sum_{k=1}^T \gamma^{(k)}}.
\end{aligned}$$

By this step we proved Lemma 4. Next we present the essential technical results to use in the proof of Theorem 1.

Proposition 7: [Sleeping multi-armed bandit convergence [33]] The sleeping multi-armed bandit sampling scheme under adversarial availability guarantees the following: $\left\|P^{(k)} - P\right\| \leq \mathcal{O}(\frac{1}{\sqrt{k}})$. *Therefore, using Definition 1, the random walk with transition matrices* $P^{(k)}$ *is strongly ergodic.*

We state next Lemma of [51] about the convergence of strongly ergodic random walk.

Lemma 8 (Theorem II.7 in [51]): Given strongly ergodic non-homogenous transition matrices $P^{(k)}$ *with a stochastic matrix* P *such that* $\lim_{k\to\infty} g(2k)\left\|P^{(k)} - P\right\| = 0$, *given* Q *such* $\left\|P^k - Q\right\| \leq c\beta_2^k$, *then* $\lim_{k\to\infty} \min\left\{(1/\mu)^k, g(k)\right\}\left\|P^{(0,k)} - Q\right\| = 0$, *where* $1 < 1/\mu < \sqrt{1/\beta_2}$.

Using the previous lemma, we get

$$\max_i \left\|\mu - P^{(0,T_k)}(i,:)\right\| \leq \mathcal{O}\left(\frac{1}{2k} + \frac{1}{\sqrt{k}}\right) \qquad (18)$$

for $T_k = \min\left\{k, \max\left\{\frac{\ln(2Ck)}{\ln(1/\lambda)}, K_P\right\}\right\}$. Therefore,

$$\begin{aligned}
&\mathbb{E}_{j^{(k)}}\left[F_{j^{(k)}}\left(w^{(k-T^{(k)})}\right) - F_{j^{(k)}}(w^*) \,|\, X_0, X_1, \ldots, X_{k-T^{(k)}}\right] \\
&= \sum_{i=1}^N \left(F_i\left(w^{(k-T^{(k)})}\right) - F_i(w^*)\right) \\
&\quad \times P\left(j^{(k)} = i \,|\, X_0, X_1, \ldots, X_{k-T^{(k)}}\right) \\
&= \sum_{i=1}^N \left(F_i\left(w^{(k-T^{(k)})}\right) - F_i(w^*)\right) P\left(j^{(k)} = i \,|\, X_{k-T^{(k)}}\right) \\
&= \sum_{i=1}^N \left(F_i\left(w^{(k-T^{(k)})}\right) - F_i(w^*)\right) \\
&\quad \times P^{(0,T^{(k)})}\left[X_{k-T^{(k)}} \,|\, j^{(k)} = i\right] \\
&\geq \left(F\left(w^{(k-T^{(k)})}\right) - F(w^*)\right) - \frac{N}{2k} - \frac{cte.N}{\sqrt{k}}. \qquad (19)
\end{aligned}$$

Finally,

$$\begin{aligned}
&\sum_{k=1}^T \gamma^{(k)} \mathbb{E}\left[F\left(w^{(k)}\right) - F(w^*)\right] \\
&\leq \sum_{k=1}^T N\gamma^{(k)}\left(\frac{1}{2k} + \frac{cte}{\sqrt{k}}\right) + \frac{C.D^2}{\ln(1/\lambda)} \\
&\quad + \frac{1}{2}\left\|w^{(0)} - w^*\right\|_2^2 + \sum_{k=1}^T \left(\gamma^{(k)}\right)^2 \mathbb{E}\left[\left\|\hat{\nabla} F_{i^{(k)}}\left(w^{(k)}\right)\right\|_2^2\right].
\end{aligned}$$

Using previous results and the the convexity assumption, we get

$$\begin{aligned}
&\mathbb{E}\left[F\left(w^{(k)}\right) - F(w^*)\right] \\
&\leq \frac{\sum_{k=1}^T N\gamma^{(k)}\left(\frac{1}{2k} + \frac{cte}{\sqrt{k}}\right) + \frac{C.D^2}{\ln(1/\lambda)} + \frac{1}{2}\left\|w^{(0)} - w^*\right\|_2^2}{\sum_{k=1}^T \gamma^{(k)}} \\
&\quad + \frac{\left(\gamma^{(0)}\right)^2 C^*}{\sum_{k=1}^T \gamma^{(k)}}.
\end{aligned}$$

Employing the assumptions on the step size and the gradient boundness in Corollary 1, we get an order of convergence $O(T^{1-q})$ for a step size choice $\gamma^{(k)} = \frac{1}{k^q}$ where $\frac{1}{2} < q < 1$.

REFERENCES

[1] H. B. McMahan, E. Moore, D. Ramage, and B. A. Y. Arcas, "Federated learning of deep networks using model averaging," 2016, *arXiv:1602.05629*.

[2] P. Kairouz et al., "Advances and open problems in federated learning," 2019, *arXiv:1912.04977*.

[3] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[4] Y. Yang, R. G. L. D'Oliveira, S. El Rouayheb, X. Yang, H. Seferoglu, and Y. Chen, "Secure coded computation for efficient distributed learning in mobile IoT," in *Proc. 18th Annu. IEEE Int. Conf. Sens., Commun., Netw. (SECON)*, Jul. 2021, pp. 1–9.

[5] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *Int. J. Med. Inform.*, vol. 112, pp. 59–67, Apr. 2018.

[6] J. C. Jiang, B. Kantarci, S. Oktug, and T. Soyata, "Federated learning in smart city sensing: Challenges and opportunities," *Sensors*, vol. 20, no. 21, p. 6230, Oct. 2020.

[7] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, A. Singh and J. Zhu, Eds. vol. 54, Apr. 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[8] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*.

[9] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. Vincent Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," 2020, *arXiv:2007.07481*.

[10] K. Dolui and S. K. Datta, "Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing," in *Proc. Global Internet Things Summit (GIoTS)*, Jun. 2017, pp. 1–6.

[11] K. Hsieh et al., "Gaia: Geo-distributed machine learning approaching LAN speeds," in *Proc. NSDI*, 2017, pp. 1–21.

[12] S. Wang et al., "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 1205–1221, Jun. 2019.

[13] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[14] B. Johansson, M. Rabi, and M. Johansson, "A randomized incremental subgradient method for distributed optimization in networked systems," *SIAM J. Optim.*, vol. 20, no. 3, pp. 1157–1170, 2009.

[15] P. Zhang, J. Wang, and K. Guo, "Compressive sensing and random walk based data collection in wireless sensor networks," *Comput. Commun.*, vol. 129, pp. 43–53, Sep. 2018.

[16] X. Mao, K. Yuan, Y. Hu, Y. Gu, A. H. Sayed, and W. Yin, "Walkman: A communication-efficient random-walk algorithm for decentralized optimization," *IEEE Trans. Signal Process.*, vol. 68, pp. 2513–2528, 2020.

[17] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2, pp. 235–256, 2002.

[18] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The nonstochastic multiarmed bandit problem," *SIAM J. Comput.*, vol. 32, no. 1, pp. 48–77, 2011.

[19] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," 2012, *arXiv:1204.5721*.

[20] B. Johansson, M. Rabi, and M. Johansson, "A simple peer-to-peer algorithm for distributed optimization in sensor networks," in *Proc. 46th IEEE Conf. Decis. Control*, Dec. 2007, pp. 4705–4710.

[21] S. Sundhar Ram, A. Nedic, and V. V. Veeravalli, "Asynchronous gossip algorithms for stochastic optimization," in *Proc. Int. Conf. Game Theory Netw.*, May 2009, pp. 80–81.

[22] H.-T. Wai, W. Shi, A. Nedic, and A. Scaglione, "Curvature-aided incremental aggregated gradient method," in *Proc. 55th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Oct. 2017, pp. 526–532.

[23] T. Sun, Y. Sun, and W. Yin, "On Markov chain gradient descent," in *Proc. NeurIPS*, 2018, pp. 1–10.

[24] G. Ayache and S. E. Rouayheb, "Private weighted random walk stochastic gradient descent," *IEEE J. Sel. Areas Inf. Theory*, vol. 2, no. 1, pp. 452–463, Mar. 2021.

[25] J. C. Duchi, A. Agarwal, M. Johansson, and M. I. Jordan, "Ergodic mirror descent," in *Proc. 49th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2011, pp. 701–706.

[26] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *J. Optim.*, vol. 25, no. 2, pp. 944–966, May 2015.

[27] A. I. Chen, "Fast distributed first-order methods," Tech. Rep., 2012.

[28] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Fast distributed gradient methods," *IEEE Trans. Autom. Control*, vol. 59, no. 5, pp. 1131–1146, May 2014.

[29] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. S. Talwalkar, and V. Smith, "On the convergence of federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*.

[30] F. Haddadpour and M. Mahdavi, "On the convergence of local descent methods in federated learning," 2019, *arXiv:1910.14425*.

[31] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 5132–5143.

[32] A. Khaled, K. Mishchenko, and P. Richtárik, "Tighter theory for local SGD on identical and heterogeneous data," in *Proc. AISTATS*, 2020, pp. 1–10.

[33] R. Kleinberg, A. Niculescu-Mizil, and Y. Sharma, "Regret bounds for sleeping experts and bandits," *Mach. Learn.*, vol. 80, nos. 2–3, pp. 245–272, 2010.

[34] H. Namkoong, A. Sinha, S. Yadlowsky, and J. C. Duchi, "Adaptive sampling probabilities for non-smooth optimization," in *Proc. ICML*, 2017, pp. 1–10.

[35] F. Salehi, L. Elisa Celis, and P. Thiran, "Stochastic optimization with bandit sampling," 2017, *arXiv:1708.02544*.

[36] Z. Borsos, A. Krause, and K. Y. Levy, "Online variance reduction for stochastic optimization," 2018, *arXiv:1802.04715*.

[37] L. Zhang, S. Lu, and Z. Zhou, "Adaptive online learning in dynamic environments," in *Proc. NeurIPS*, 2018, pp. 1–11.

[38] A. El Hanchi and D. A. Stephens, "Adaptive importance sampling for finite-sum optimization and sampling with decreasing step-sizes," 2021, *arXiv:2103.12243*.

[39] V. Kanade, H. B. McMahan, and B. Bryan, "Sleeping experts and bandits with stochastic action availability and adversarial rewards," in *Proc. AISTATS*, 2009, pp. 272–279.

[40] A. Saha, P. Gaillard, and M. Valko, "Improved sleeping bandits with stochastic actions sets and adversarial rewards," in *Proc. ICML*, 2020, pp. 1–9.

[41] P. Zhao and T. Zhang, "Stochastic optimization with importance sampling," 2014, *arXiv:1401.2753*.

[42] G. Alain, A. Lamb, C. Sankar, A. Courville, and Y. Bengio, "Variance reduction in SGD by distributed importance sampling," 2015, *arXiv:1511.06481*.

[43] S. U. Stich, A. Raj, and M. Jaggi, "Safe adaptive importance sampling," in *Proc. NIPS*, 2017, pp. 1–11.

[44] G. Papa, P. Bianchi, and S. Clémençon, "Adaptive sampling for incremental optimization using stochastic gradient descent," in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2015, pp. 317–331.

[45] D. Needell, R. Ward, and N. Srebro, "Stochastic gradient descent, weighted sampling, and the randomized Kaczmarz algorithm," in *Advances in Neural Information Processing Systems*, vol. 27, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2014, pp. 1017–1025.

[46] L. Bottou and O. Bousquet, "The tradeoffs of large scale learning," in *Proc. NIPS*, 2007, pp. 1–8.

[47] A. Defazio, F. Bach, and S. Lacoste-Julien, "SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives," in *Proc. NIPS*, 2014, pp. 1–9.

[48] L. Xiao and T. Zhang, "A proximal stochastic gradient method with progressive variance reduction," *SIAM J. Optim.*, vol. 24, no. 4, pp. 2057–2075, 2014.

[49] G. Bouchard, T. Trouillon, J. Perez, and A. Gaidon, "Online learning to sample," 2015, *arXiv:1506.09016*.

[50] T. Lattimore and C. Szepesvari, *Bandit Algorithms*, 2020.

[51] C.-C. Huang, "Non-homogeneous Markov chains and their applications," Ph.D. dissertations, Dept. Math., Lowa State Univ., Ames, IA, USA, 1977.

[52] G. Ayache and S. El Rouayheb, "Random walk gradient descent for decentralized learning on graphs," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshops (IPDPSW)*, May 2019, pp. 926–931.

[53] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using networkx," in *Proc. 7th Python Sci. Conf.*, G. Varoquaux, T. Vaught, and J. Millman, Eds. Pasadena, CA, USA, 2008, pp. 11–15.

[54] Y. Nesterov et al., *Lectures on Convex Optimization*, vol. 137. Cham, Switzerland: Springer, 2018.

[55] J. C. Dattorro, *Convex Optimization and Euclidean Distance Geometry*, 2004.

[56] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov Chains and Mixing Times*. Providence, RI, USA: American Mathematical Society, 2006.