

Received November 21, 2019, accepted December 13, 2019, date of publication December 18, 2019, date of current version December 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2960494

Fast Decentralized Data Analytics in IoT Wireless Networks

LIANG ZHAO 

Division of Mathematics and Computer Science, University of South Carolina Upstate, Spartanburg, SC 29303, USA

e-mail: lzhaoz@uscupstate.edu

This work was supported in part by the Scholarly Start-Up Package program from the Office of Sponsored Awards and Research Support at University of South Carolina Upstate.

ABSTRACT This paper presents an innovative mechanism for decentralized data analytics in the Internet of Things (IoT) networks. In contrast to traditional centralized (eg. Cloud) based solution, our proposed decentralized framework can avoid the problems raised in data collection such as the constraint in high latency, and the associated data breach and privacy issues. The invented mechanism enables each IoT node in the network to independently obtain the global optimal analytics with local computation and communication. The proposed framework has been tested in seismic imaging and machine learning applications. The evaluation results validate its fast convergence to the optimal model, and demonstrates that it allows (near) real-time data analytics in IoT networks under bandwidth and energy limitation, particularly in uncertain and dynamic environments.

INDEX TERMS Asynchronous algorithms, decentralized data analytics, Internet of Things.

I. INTRODUCTION


[1] The Internet of Things (IoT) is considered to be the next evolution of the Internet with advances in collecting and analyzing data that can be ultimately abstracted into insights and knowledge [2]. In order to extract deep insights, Big Data and Big Model are usually required and adopted. Thus, most IoT applications today are built on a centralized architecture: huge volume of data from IoT devices are firstly collected and then transferred into a central place (eg. Cloud) for processing and analytics. However, this cloud-based “collecting then processing” architecture raises serious constraints and issues as follows. a). High data transfer cost. Most of the Big Data comes in high volume and it is very costly and sometimes even infeasible to move all the raw data into Cloud due to bandwidth limitation [3]. b). Relatively high latency. Cloud-based centralized framework is not suitable for many time-sensitive IoT applications where near real-time analytics is critical [3]. c). Data breach issues. If a cloud-based solution is adopted where data are stored and processed on remote servers, the risk of data breach is present and inevitable [4]. d). Privacy-concerned applications. For applications in some fields where privacy and security is of prime concern (such as health, military, bank etc.), their raw

data is not allowed to be shared and has to remain within their controllable. These industries cannot directly benefit from traditional cloud-based solution.

To address the issues above, several alternatives have been proposed such as Fog computing [5] and Edge computing [6]. Both of them aim to push data analytics close to where the data is generated. In Fog computing, the data is processed within a fog node or IoT gateway while with Edge computing, data is usually processed on the devices at the edge of the network. There are several works in Fog/Edge computing aiming to reduce the latency by offloading the computation [7]. Although problem a and b mentioned above can be addressed in Fog and Edge computing, problem c and d remain unsolved. Data collection from users to perform analytics are required in these two alternatives and thus they are still “centralized” type solutions, which may not solve the associated data breach or privacy concerns. Considering the issues above, we propose a novel fast decentralized data analytics framework for IoT networks in this paper.

II. RELATED WORK

Bringing computing into IoT networks is nontrivial. A mismatch exists between the “centralized” nature (in traditional computing) and the spatially “distributed” feature (in sensor networks), making computing and data analytics in IoT networks very challenging. Designing suitable numerical

The associate editor coordinating the review of this manuscript and approving it for publication was Yuedong Xu .

optimization solution is at the core of solving many of these similar problems. In signal processing community, much attention has been paid to solving similar distributed estimation problems. For instance, Sayed and Lopes [8] developed a distributed least-squares estimation strategy by appealing to collaboration techniques that exploit the space-time structure of the data. This strategy achieves an exact recursive solution in a distributed manner. In addition, a cyclic path in the network is required in order to perform the computation node by node. However, a master node usually exists in the network in order to fuse the estimates from local nodes and spread mixed information over the network nodes [9]–[14].

Decentralized computing solutions are considered then in literature. In this decentralized paradigm, each node holds an objective function privately known and can only communicate with its immediate neighbors (no multi-hop required). In optimization community, a great effort has been devoted to solving decentralized (fully distributed) consensus optimization problem, especially in applications like distributed machine learning, multi-agent optimization, etc. Several algorithms have been proposed for solving general convex and (sub)differentiable functions.

Decentralized algorithms can be synchronous or asynchronous. A series of algorithms based on synchronous models have been proposed in literature. In synchronous model, each node might need to wait all its neighbors' information in order to perform the next computation round. decentralized (sub)gradient-based methods have been proposed in [15], [16], [17]–[20]. However, the aforementioned methods can only converge to a neighborhood of an optimal solution in the case of fixed step size [21]. Modified algorithms have been developed in [19] and [20], which use diminishing step sizes in order to guarantee to converge to a true solution. Other related algorithms were discussed in [22]–[28], which share similar ideas. The D-NC algorithm proposed in [20] was demonstrated to have an outer-loop convergence rate of $O(1/k^2)$ in terms of objective value error. The rate is same as the centralized Nesterov's accelerated gradient method. However, the number of consensus iterations within outer-loop is growing significantly along the iteration. Shi *et al.* [29] developed a method based on correction on mixing matrix for Decentralized Gradient Descent (DGD) method [21] without diminishing step sizes.

Asynchronous distributed solutions have been proposed in [30], [31] and [24]. For computation, the work in [30] and [31] leverage the alternating direction method of multipliers (admm). Regarding communication, unicast (random gossip) has been used such that in each iteration, one node randomly wake up one of its neighbors to exchange information. Tsitsiklis *et al.* [24] designed an asynchronous model for distributed optimization, while in its model each node estimates a partial vector of the global variable, which is different from our goal of decentralized consensus such that every node maintains an estimate for the global model.

The first work for asynchronous decentralized consensus using random broadcast protocol was developed in [32].

However, the objective is solving the average consensus problem. The setting of the average consensus problem is that every node holds some value and all the nodes in the network want to obtain the average of their values. Nedic considers general decentralized convex optimization problem (more general and more difficult) adopting the random broadcast setting in [33]. For the computation, a (sub)gradient-based update rule is adopted. By replacing (sub)gradient computation with full local optimization, an improved algorithm has been designed in terms of the number of communication rounds [34]. Our proposed algorithm aims to solve the same general convex optimization problem and adopts the random broadcast communication setting since we think this communication protocol is more practical for IoT networks. The key difference in our algorithm is that we have a different design on what local nodes compute and what they communicate with their neighbors. The idea of our method is combing neighbors' (sub)gradient information in order to further speed up the algorithms in [33], [34]. The main contribution of this paper is three-fold: 1) We develop a new decentralized optimization algorithm for data analytics in IoT networks. 2) We provide convergence guarantee on the new algorithm such that each node in the network would eventually converge to the same optimal solution. 3) We conduct experiments on various applications demonstrating that the proposed algorithm outperforms the benchmark, and the resulting framework is a promising solution for efficient data analytics in IoT networks.

III. PROBLEM FORMULATION

The formulation of the decentralized data analytics problem can be described as follows. Consider an undirected connected network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where \mathcal{V} denotes the node set and \mathcal{E} is the edge set. The size of network is $m = |\mathcal{V}|$ (cardinality of the set \mathcal{V}) and two nodes i and j are neighbors if $(i, j) \in \mathcal{E}$. Each node i has a local private objective function $F_i : \mathbb{R}^n \rightarrow \mathbb{R}$ and it captures the data acquisition process at node i . The goal is that each node can find the global consensus solution $x \in X$ minimizing the summation of all the local objective functions. The resulting optimization problem is expressed as follows.

$$\min_{x \in X} \left\{ F(x) := \sum_{i=1}^m F_i(x) \right\}. \quad (1)$$

Solving (1) in IoT wireless networks is very challenging. First, data is generated in a distributed way in nature. Simply transmit all the raw data from IoT nodes into a Cloud for post-processing is undesirable due to many issues. First, data movement would be very costly or even infeasible due to bandwidth and energy constraints. Second, traditional Cloud-computing solution is not suitable for time-sensitive IoT applications due to its high latency. Third, there are privacy issues involved since the Cloud would be able to access all the raw data. To address the issues above, a decentralized solution is highly demanded. We are thus motivated to propose a decentralized framework for data analytics in IoT networks

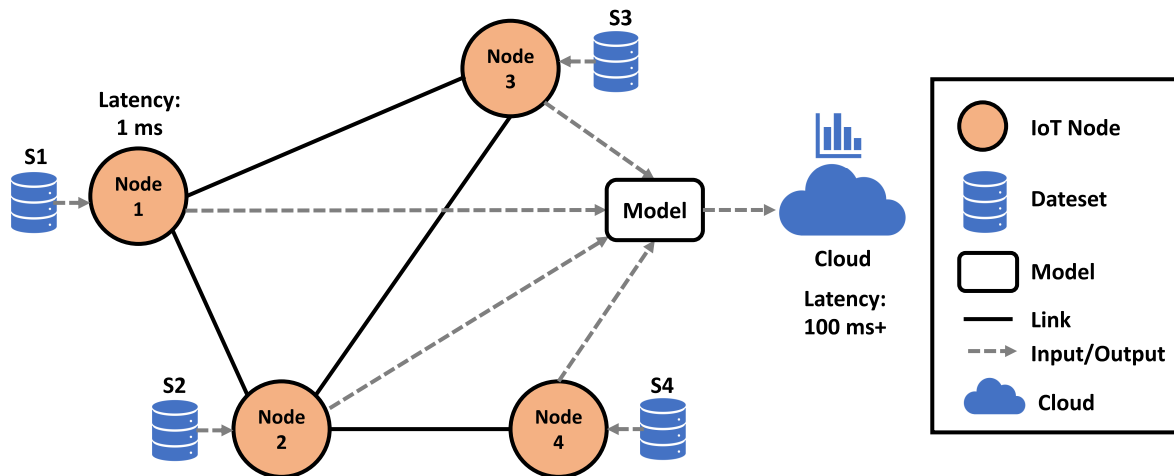


FIGURE 1. Architecture for Decentralized Data Analytics in IoT Networks. Features: Data is distributed and privately owned by the IoT nodes in the network. All the nodes must converge to a single model. Only the final model rather than the raw data would be transmitted to Cloud if needed. Only local exchanges between immediate neighbors. No central or coordinate (fusion) node in the network. Leveraging local IoT node processing power, which has latency around 1 ms while latency for Cloud computing is 100 ms+.

as follows. First, each node maintains an estimate for the global model. Second, the raw data generated does not leave the nodes and are processed locally. Each node refines its estimate by leveraging the local data and exchanging information with its immediate neighbors only. Third, all the nodes should reach consensus on the estimate for the global model eventually. The architecture of the proposed framework is illustrated in Figure 1.

The key challenge in implementing the framework is the potential high communication cost because each node has only partial knowledge of the whole network. Hence, a fast and communication-efficient algorithm needs to be designed in order to make decentralized data analytics in IoT networks feasible. In the next section, we will describe the design of our proposed algorithm.

IV. ALGORITHM DESIGN

Solving decentralized data analytics problem in IoT networks requires that each node performs local computation based on its own data and communicates its estimate with neighbors. We can categorize the possible solution methodologies for the problem in (1) as follows.

Updating Fashion: Synchronous vs Asynchronous. In synchronous models, each node needs to wait for all its neighbors' information to perform the next update on its estimate. Thus, node i has to wait for its slowest neighbor's estimate in order to proceed. In contrast, asynchronous models allow each node to perform its action independent of others. We adopt asynchronous updating rule in this work since it is more practical for decentralized computing in IoT networks.

Communication Scheme: Unicast vs Broadcast. For decentralized computing applications in IoT wireless networks, broadcast communication is preferable than unicast since estimates from the nodes can be propagated through the

whole network more quickly, which can leads to obtaining faster convergence and solution.

Local Computation Complexity: It is natural that a simple and light-weight local computation rule is demanded since IoT nodes are nonstandard computing devices and have limited memory and computing power.

Considering the design choices above, we are motivated to propose an asynchronous, broadcast-based algorithm involving only gradient calculation in local computation.

A. LOCAL + NEIGHBOR'S (SUB)GRADIENT

The asynchronous algorithm developed by Nedic in [33] can be expressed as follows.

$$\begin{aligned} y_k^i &= \theta x_{k-1}^{ik} + (1 - \theta) x_{k-1}^i, \\ x_k^i &= P_X \left[y_k^i - \alpha_{i,k} \tilde{\nabla} F_i(y_k^i) \right] \end{aligned} \quad (2)$$

We propose a new method by adding neighbor's (sub)gradient information into (2) in its local update rule (second equation). The main computation step is:

$$\begin{aligned} y_k^i &= \theta x_{k-1}^{ik} + (1 - \theta) x_{k-1}^i, \\ x_k^i &= P_X \left[y_k^i - \alpha_{i,k} \tilde{\nabla} F_i(y_k^i) - \rho_{i,k} \left(\sum_{u \in \mathcal{N}_i} \tilde{\nabla} F_u(x_{\tau_{u,k}}^u) \right) \right] \end{aligned} \quad (3)$$

where k is the virtual global iteration number. \mathcal{N}_i is the set of neighbors of node i . x_k^i represents node i 's solution at k -th iteration. $\tilde{\nabla} F_u(x_{\tau_{u,k}}^u)$ means some "delayed" (sub)gradient of node u at point $x_{\tau_{u,k}}^u$. P_X represents projection to the set X . θ , $\alpha_{i,k}$ and $\rho_{i,k}$ are three parameters for node i at k -th iteration. The choices of these parameters will be discussed in the next section.

The proposed algorithm can be summarized as follows.

Algorithm 1 Fast Decentralized Data Analytics (FDDA)

Input: Starting point $x_0^1, x_0^2, \dots, x_0^m$.
Setting: Each node is equipped with a local clock.
Main Algorithm:
 1: **while** each node $i, i \in \{1, 2, \dots, m\}$ asynchronously **do**
 2: **if** (node i_k 's local clock ticks now) **then**
 3: Node i_k broadcasts its estimate $x_{k-1}^{i_k}$ and (sub)gradient $\tilde{\nabla}F_i(x_{k-1}^{i_k})$ to its neighbors;
 4: Node i who receives node i_k 's broadcast updates its solution x_k^i based on (3).
 5: **end if**
 6: **end while**

V. INTERPRETATION OF THE PROPOSED ALGORITHM
A. ALGORITHM INTERPRETATION

In this section, we will interpret and show the rationale of proposing Algorithm 1. Now assume every node i in the network can access all the local objective functions $F_i, i \in \{1, 2, \dots, m\}$. The optimal strategy for every node i to obtain the solution then becomes as follows.

$$x^i = \arg \min_x \sum_{j=1}^m f_j(x), \quad \forall i \in \{1, 2, \dots, m\}. \quad (4)$$

That is, each node can directly try to minimize the summation of all the local objective functions as a ‘‘centralized’’ machine does (assuming all the data is available in this centralized node). To solve (4), we can evaluate a proximal operator as follows [35].

$$x^i = \mathbf{prox}_{\alpha F}(v) = \arg \min_x \left\{ \frac{1}{2\alpha} \|x - v\|^2 + F(x) \right\}, \quad (5)$$

for $\forall i \in \{1, 2, \dots, m\}$, with certain constant v (independent of decision variable x) and parameter $\alpha > 0$. Note that each node i can obtain the optimal solution by evaluating (5), and more importantly there is no communication needed between nodes since $F(x)$ contains all the information in the network. However, this is under an ideal scenario (every node i has the knowledge of all the local functions f_j) which will not be valid in our setting of decentralized sensor networks.

Considering that F_i is only available to node i locally (according to our assumption in this paper), if we replace the term $F(x)$ in (5) with F_i and let $v = y_k^i$ and $\alpha = \alpha_{i,k}^i$, the update rule for node i in [34] is then derived as follows.

$$\begin{aligned} y_k^i &= \theta x_{k-1}^{i_k} + (1 - \theta) x_{k-1}^i, \\ x_k^i &= \mathbf{prox}_{\alpha_{i,k} F_i}(y_k^i) \\ &= \arg \min_x \left\{ \frac{1}{2\alpha_{i,k}} \|x - y_k^i\|^2 + F_i(x) \right\}. \end{aligned} \quad (6)$$

Further linearizing $F_i(x)$ in (6) yields Nedic’s algorithm as follows [33].

$$\begin{aligned} y_k^i &= \theta x_{k-1}^{i_k} + (1 - \theta) x_{k-1}^i, \\ x_k^i &= \arg \min_x \left\{ \frac{1}{2\alpha_{i,k}} \|x - y_k^i\|^2 + \left\langle \tilde{\nabla}F_i(y_k^i), x \right\rangle \right\} \\ &= y_k^i - \alpha_{i,k} \tilde{\nabla}F_i(y_k^i). \end{aligned} \quad (7)$$

The deduction of the last step in (7) is based on the optimality condition below.

$$\frac{1}{\alpha_{i,k}} (x_k^i - y_k^i) + \tilde{\nabla}F_i(y_k^i) = 0.$$

The first step in (6) and (7) takes the weighted average of node i 's solution and neighbor i_k 's solution which is the most recent broadcast received. This averaging step aims to mix the neighbor’s information and enforce consensus of solutions among all the nodes in the network. Next, the proximal step in (6) forces the new solution x to be close to y_k^i (the weighted average) and optimizes the local objective function F_i simultaneously. Parameter $\alpha_{i,k}$ controls the trade-off between the aforementioned two objectives.

To speed up the process of decentralized consensus optimization, we are motivated to propose a new algorithm by adding the following item into the proximal steps in (7).

$$\sum_{u \in \mathcal{N}_i} \tilde{\nabla}F_u(x_{\tau_{u,k}}^u)^T (x - y_k^i) \quad (8)$$

The main rational behind is that when designing distributed/decentralized algorithms, a common strategy is to mimic the operations in its centralized counterpart in the distributed environment setting. Decentralized algorithms that have better approximate operations would achieve better convergence performance. In this scenario, equation (8) contains node i 's neighbors’ (sub)gradient information and it can be seen that (8) is a linear approximation to $\sum_{u \in \mathcal{N}_i} F_u$. Comparing (3) with (6) we can see that in (3) node i is (approximately) optimizing $F_i(x) + \sum_{u \in \mathcal{N}_i} F_u(x)$ while Nedic’s method in (6) is optimizing local objective function $F_i(x)$ only. In (5), the second item is $F(x) := \sum_{i=1}^m F_i(x)$ (according to (1)). Hence, (3) (contains $F_i(x) + \sum_{u \in \mathcal{N}_i} F_u(x)$ as the second item) is a better approximation than Nedic’s method (contains $F_i(x)$ only) to $F(x)$ in the ideal centralized case in (5). In order to execute the computations in (3), node i_k needs to broadcast its estimate $x_{k-1}^{i_k}$ and (sub)gradient $\tilde{\nabla}F_i(x_{k-1}^{i_k})$ to its neighbors (described in Algorithm 1).

B. PARAMETER SETTINGS

In the main computation step (3), there are three parameters each node needs to configure. In this section, we describe the settings for them and also the assumptions required for the algorithm.

1) SETTING FOR θ

The first equation in (3) shows that node i mixes its neighbor’s estimate at $x_{k-1}^{i_k}$ with its own estimate x_{k-1}^i in a weighted

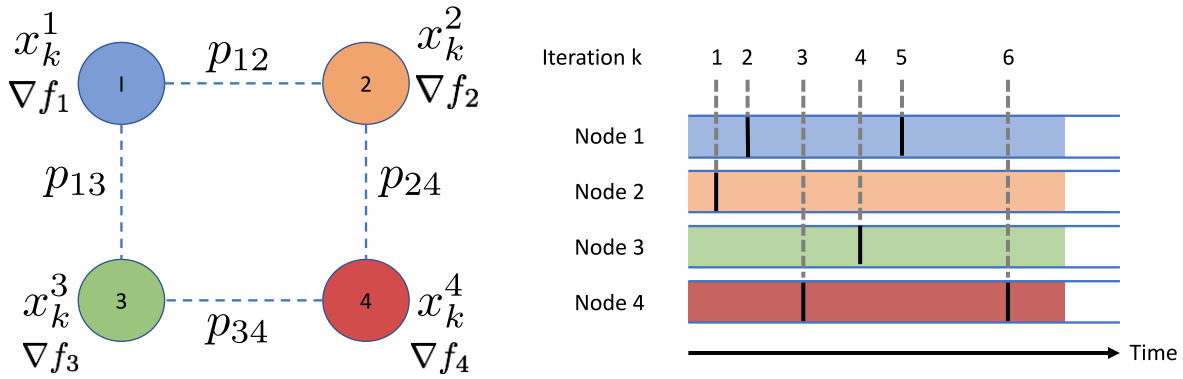


FIGURE 2. Network model and asynchronous computing. Left: An example of decentralized IoT wireless networks. Right: Asynchronous computing model. An iteration begins when any node's local clock ticks.

average manner. Thus, the parameter θ determines the relative weights between the aforementioned two estimates. In theory, the proposed algorithm can converge with any positive constant satisfying $0 < \theta < 1$. For simplicity, we set $\theta = 0.5$ for all the tests in this paper.

2) SETTING FOR $\alpha_{i,k}$

In the second equation of (3), we can see the term $y_k^i - \alpha_{i,k} \tilde{\nabla} F_i(y_k^i)$. This can be considered as the gradient descent step at point y_k^i , and $\alpha_{i,k}$ is the step size for node i at iteration k [36]. This sequence needs to be decreasing in order to ensure convergence of Algorithm 1. In the meanwhile, small step sizes will cause poor convergence performance. Thus, a sequence that “gradually” decreases is needed. In this work, we choose $\alpha_{i,k}$ to be the reciprocal of the number of updates node i has performed till k -th iteration. Also, this setting allows node i to choose the parameter independently and asynchronously at each iteration [33].

3) SETTING FOR $\rho_{i,k}$

As discussed in previous, one key component of the proposed algorithm is integrating neighbors gradient information in order to speed up the process of finding the optimal solution. Similar to the role of $\alpha_{i,k}$, $\rho_{i,k}$ can be considered as the step size for neighbors gradient. It is used to control how much neighbors information (including error) is combined at each iteration for convergence. In our convergence analysis, we realized that parameter $\rho_{i,k}$ needs to satisfy Assumption 3 (described in the next section). Users have some flexibility to choose combination of $\alpha_{i,k}$ and $\rho_{i,k}$ but a simple and reasonable choice is setting $\rho_{i,k} = \alpha_{i,k}$. Note that different parameter setting may affect the convergence performance and to have a fair comparison with the benchmark, we adopt the same setting for all the methods in the experiment section.

4) ASSUMPTIONS

Assumption 1: The (sub)gradient of function $F_i, \forall i \in \{1, 2, \dots, m\}$ is bounded. That is, $\|\tilde{\nabla} F_i\| \leq G$ where G is some positive constant.

Assumption 2: The constraint set X is bounded.

Assumption 3: $\sum_{k=1}^{\infty} \frac{\rho_{i,k}}{k\alpha_{i,k}} < \infty$ almost surely.

Remark 1: Assumptions 1-3 are mainly for the convergence analysis purpose. They are conditions for our analysis to prove the convergence of the proposed algorithm. Assumption 1 is about bounded gradient, assumption 2 is about bounded constraint set (basically means the optimization problem has finite instead of infinite number of solutions). Both of them are common assumptions in analyzing distributed algorithms in literature [18]–[21], [29]. Assumption 3 is specific to our proposed algorithm. There are two user-defined parameters alpha and rho involved and assumption 3 indeed gives condition on selecting the two parameters. Once parameters are chosen appropriately, assumption 3 can be satisfied.

C. A TOY EXAMPLE

We assume Algorithm 1 is performed in a decentralized IoT network illustrated in Figure 2. There are four nodes in this cyclic network and it contains that $\mathcal{N}_1 = \{2, 3\}$, $\mathcal{N}_2 = \{1, 4\}$, $\mathcal{N}_3 = \{1, 4\}$, $\mathcal{N}_4 = \{2, 3\}$. The algorithm runs as follows.

Iteration 1: Node 2's clock ticks and it broadcasts x_0^2 and $\tilde{\nabla} f_2(x_0^2)$. Node 1 and 4 receive the broadcast and use x_0^2 and $\tilde{\nabla} f_2(x_0^2)$ to update x_1^1 and x_1^4 based on (3). Set $x_1^2 \leftarrow x_0^2$, $x_1^3 \leftarrow x_0^3$.

Iteration 2: Node 1's clock ticks and it broadcasts x_1^1 and $\tilde{\nabla} f_1(x_1^1)$. Node 2 and 3 receive the broadcast and use x_1^1 and $\tilde{\nabla} f_1(x_1^1)$ to update x_2^2 and x_2^3 based on (3). Set $x_2^1 \leftarrow x_1^1$, $x_2^4 \leftarrow x_1^4$.

Iteration 3: Node 4's clock ticks and it broadcasts x_2^4 and $\tilde{\nabla} f_4(x_2^4)$. Node 2 and 3 receive the broadcast and use x_2^4 and $\tilde{\nabla} f_1(x_1^1) + \tilde{\nabla} f_4(x_2^4)$ to update x_3^2 and x_3^3 based on (3). Set $x_3^1 \leftarrow x_2^1$, $x_3^4 \leftarrow x_2^4$.

Iteration 4: Node 3's clock ticks and it broadcasts x_3^3 and $\tilde{\nabla} f_3(x_3^3)$. Node 1 and 4 receive the broadcast and use x_3^3 and $\tilde{\nabla} f_2(x_0^2) + \tilde{\nabla} f_3(x_3^3)$ to update x_4^1 and x_4^4 based on (3). Set $x_4^2 \leftarrow x_3^2$, $x_4^3 \leftarrow x_3^3$.

It can be seen that after four iterations, each node has gathered all its neighbors' (sub)gradient information. As the

algorithm goes on, the neighbors' (sub)gradient information will be updated for each node.

VI. CONVERGENCE ANALYSIS

In this section, we conduct convergence analysis on the proposed Algorithm 1. The main results are summarized in Theorem 7 and Theorem 8. Required lemmas for deriving the main results are shown as follows.

Lemma 1 (The supermartingale convergence theorem [37], Proposition 8.2.10): Assume $\sigma_k, \varphi_k, \omega_k$, and ε_k are nonnegative random variables and assume the following hold

$$\mathbb{E}(\sigma_{k+1}|\Omega_k) \leq \sigma_k - \varphi_k + \varepsilon_k \text{ almost surely,}$$

$$\sum_{k=1}^{\infty} \varepsilon_k < \infty \text{ almost surely}$$

where $\mathbb{E}(\sigma_{k+1}|\Omega_k)$ represents the conditional expectation given all the past history of σ_k, φ_k , and ε_k up to iteration k . Then it concludes that

$$\sigma_k \rightarrow \sigma \text{ almost surely, } \sum_{k=1}^{\infty} \varphi_k < \infty \text{ almost surely}$$

where $\sigma \geq 0$ is some random variable.

Lemma 2 ([33] lemma 2): The random matrix $W_k - \frac{1}{m}\mathbf{1}\mathbf{1}^T W_k$ is independent and has identical distribution with each other that $\mu < 1$.

$$\mu := \mu \left(\mathbb{E} \left[\left(W_k - \frac{1}{m}\mathbf{1}\mathbf{1}^T W_k \right)^T \left(W_k - \frac{1}{m}\mathbf{1}\mathbf{1}^T W_k \right) \right] \right) \quad (9)$$

where $\mu(\mathbf{A})$ denotes the largest eigenvalue of a symmetric matrix \mathbf{A} .

Lemma 3 [33]: The upperbounds of step size $\alpha_{i,k}$ are obtained as follows when k is large enough ($k > \tilde{k}(m, q)$)

$$\alpha_{i,k} \leq \frac{2}{k\delta_i}, \quad \alpha_{i,k}^2 \leq \frac{4m^2}{k^2 p_*^2}, \quad \left| \alpha_{i,k} - \frac{1}{k\delta_i} \right| \leq \frac{2}{k^{\frac{3}{2}-q} p_*^2},$$

where δ_i is the total probability that node i updates. p_* denotes the minimum among all p_{ij} 's. $q \in (0, \frac{1}{2})$ is some constant. $\tilde{k}(m, q)$ is an integer determined by the number of nodes m and q .

Lemma 4 ([38] Proposition 1.1.9): The projection operation P_X in (3) is nonexpansive.

Lemma 5: The following inequality holds for arbitrary $a, b, c \in X, \forall i \in \mathcal{V}$.

$$\tilde{\nabla} F_i(a)^T (a - b) \geq F_i(c) - F_i(b) - G \|a - c\|.$$

Proof: By the property of (sub)gradient and the assumption of bounded (sub)gradient, we can have:

$$\begin{aligned} \tilde{\nabla} F_i(a)^T (a - b) &\geq F_i(a) - F_i(b) \\ &\geq F_i(c) - \tilde{\nabla} F_i(c)^T (c - a) - F_i(b) \\ &\geq F_i(c) - \left\| \tilde{\nabla} F_i(c) \right\| \|a - c\| - F_i(b) \\ &\geq F_i(c) - G \|a - c\| - F_i(b). \end{aligned}$$

□

Lemma 6: The following two relations hold

$$\sum_{i=1}^m \mathbb{E} \left[\left\| y_k^i - x \right\| \middle| \Omega_{k-1} \right] \leq \sum_{i=1}^m \left\| x_{k-1}^i - x \right\|,$$

$$\sum_{i=1}^m \mathbb{E} \left[\left\| y_k^i - x \right\|^2 \middle| \Omega_{k-1} \right] \leq \sum_{i=1}^m \left\| x_{k-1}^i - x \right\|^2.$$

Proof: By the convexity of norm function, the definition of y_k^i and the property that W_k is a stochastic matrix, $\mathbb{E}[W_k]$ is doubly stochastic, it follows

$$\begin{aligned} &\sum_{i=1}^m \mathbb{E} \left[\left\| y_k^i - x \right\| \middle| \Omega_{k-1} \right] \\ &= \sum_{i=1}^m \mathbb{E} \left[\left\| \sum_{j=1}^m [W_k]_{i,j} (x_{k-1}^j - x) \right\| \middle| \Omega_{k-1} \right] \\ &\leq \sum_{i=1}^m \mathbb{E} \left[\sum_{j=1}^m [W_k]_{i,j} \left\| x_{k-1}^j - x \right\| \middle| \Omega_{k-1} \right] \\ &= \sum_{i=1}^m \sum_{j=1}^m \mathbb{E} [[W_k]_{i,j} \middle| \Omega_{k-1}] \left\| x_{k-1}^j - x \right\| \\ &= \sum_{j=1}^m \sum_{i=1}^m \mathbb{E} [[W_k]_{i,j} \middle| \Omega_{k-1}] \left\| x_{k-1}^j - x \right\| \\ &\leq \sum_{j=1}^m \left\| x_{k-1}^j - x \right\|. \end{aligned}$$

It completes the proof for the first relation and the second relation can be proved in a similar manner with the convexity of square norm function. □

Remark 2: For our convergence analysis, the key lemma is the first one "The supermartingale convergence theorem". It provides a framework for proving convergence of iterative algorithms. The challenge is that how to bound the sequences in our algorithm and fit them into this framework in order to obtain the convergence proof.

A. ALMOST SURE CONVERGENCE

Theorem 7: Let $\{x_k^i\}, \forall i \in \mathcal{V}, k \geq 0$ be the sequence generated by Algorithm 1 and given that all the assumptions are satisfied. Then we can have the following almost surely:

$$\sum_{k=1}^{\infty} \frac{1}{k} \|x_{k-1}^i - \bar{x}_{k-1}\| < \infty, \text{ and } \lim_{k \rightarrow \infty} \|x_k^i - \bar{x}_k\| = 0,$$

where $\bar{x}_{k-1} = \frac{1}{m} \sum_{i=1}^m x_{k-1}^i$.

Proof: See APPENDIX A. □

Theorem 8: Let $\{x_k^i\}, \forall i \in \mathcal{V}, k \geq 0$ be the sequences generated by Algorithm 1 and given that all the assumptions are satisfied. Then the sequences converges to a same optimal point almost surely for any node i .

Proof: See APPENDIX B. □

Remark 3: Theorem 7 indicates that every node in the network will reach to a consensus on the estimate of the

global model eventually. Theorem 8 implies that each node would reach to the same optimal solution.

Both results in Theorem 7 and 8 are interesting in theory since we would be able to randomly select one node's estimate as our final solution. Also, this solution through decentralized collaborative computing would be the same as the centralized solution leveraging all the data in the nodes. In some time-sensitive real-world applications, we would like to have a solution that can be quickly generated and also it optimizes the global function (summation of all the local objective functions). But if we just pick up one node's estimate at early stage (the algorithm might not converge yet), it might not be desirable to use it as the final solution since the estimates among the nodes might have vast difference at that point and we will not know if the solution is good or bad. Thus a common strategy is to average the estimates from some nodes available. In this situation, the speed of the algorithm in decreasing the error (optimize the objective value) would be a more important metric than how fast the nodes can reach consensus on their estimates. Note that if we are allowed to choose only a few nodes and access their estimates for calculating the final solution, a list of "critical" nodes needs to be determined. Fortunately, the weight of each node is easy to determine. The objective function in the decentralized consensus problem is the summation of all the local objective functions thus the node with more data is considered to be more important. The node with more data will dominate the overall objective function naturally.

VII. CASE STUDY

In this section, we perform experiments on two applications: decentralized seismic tomography and decentralized machine learning in this section. The proposed algorithm 1 and the benchmark method (Nedic's method in [33]) are tested on both applications for comparison. The experiments are conducted in a distributed network emulator named Common Open Research Emulator (CORE) and EMANE network emulator [40], [41]. Tests are performed on a MacBook Pro with an Intel dual core i5-2.3G HZ CPU and 8 GB memory.

A. DECENTRALIZED SEISMIC TOMOGRAPHY

In this section, we illustrate a motivating application - Decentralized Seismic Tomography and demonstrate how the proposed Algorithm 1 fits into it.

The investigated problem is called travel-time tomography inversion, which can be formulated as solving a linear system of equations as follows [42]:

$$Ax = b, \tag{10}$$

where matrix A stores the ray information from the sources to the receivers. Vector b is constructed using travel-time information. x is the decision variable we want to solve, which contains the slowness value (reciprocal of velocity) in each block. The inversion problem in (10) is equivalent to

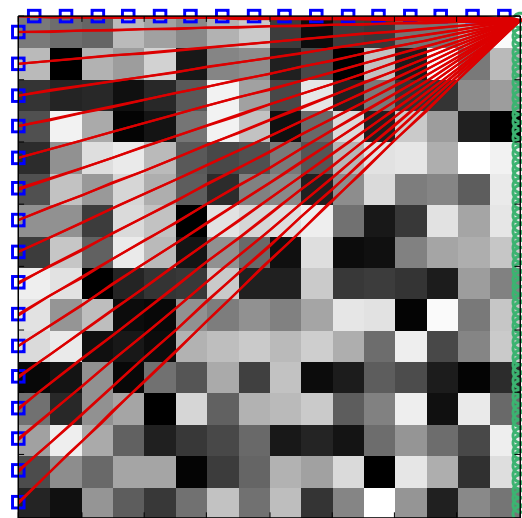


FIGURE 3. 2-D Seismic Imaging Model. The green dots are the seismic event sources. The blue squares are the sensors. The red lines are the rays traveling from the sources to the sensors (receivers). The goal is reconstruction of the seismic velocity model (structure) of the square area in which the computed travel times agree with the observed travel times.

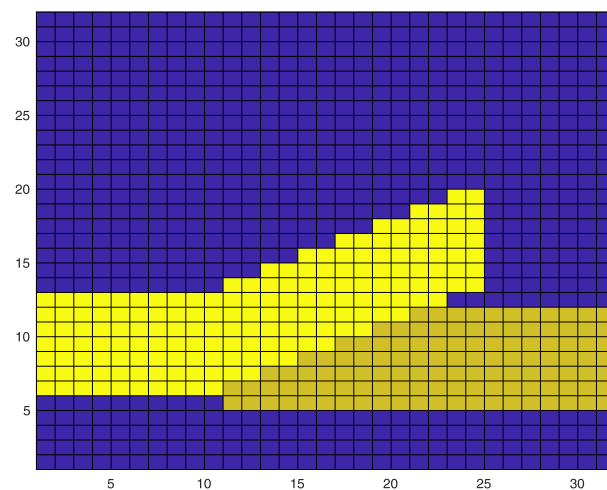


FIGURE 4. Ground Truth for the structure of the square region in Figure 3. The color in each small block represents its velocity. Blocks with high velocity will take less time for waves to go through them.

finding the least-squares solution x as follows.

$$x = \arg \min_x \|Ax - b\|_2^2 \tag{11}$$

The linear system in (10) is usually inconsistent due to noise-corrupted vector b . Regularization technique is thus utilized to reconstruct a better tomography. We adopt Tikhonov regularization as follows since it is the most popular one used for the seismic inversion problem in literature.

$$\min_x \frac{1}{2} \|Ax - b\|_2^2 + \lambda^2 \|x\|_2^2 \tag{12}$$

In the next, we introduce the formulation for the decentralized seismic imaging problem. The ray information and travel-time information in A and b are originally generated in a distributed manner. Thus we can equivalently

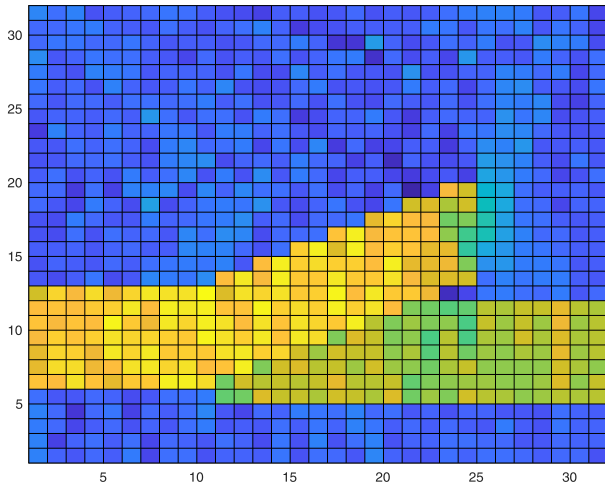


FIGURE 5. 2-D Model-Centralized Solution. This is the solution for the original centralized problem in (12) assuming that all the raw data are collected in a central place for processing. This centralized solution is computed using LSQR method [39], and is seen as the best benchmark for all the decentralized algorithms in terms of the tomography results.

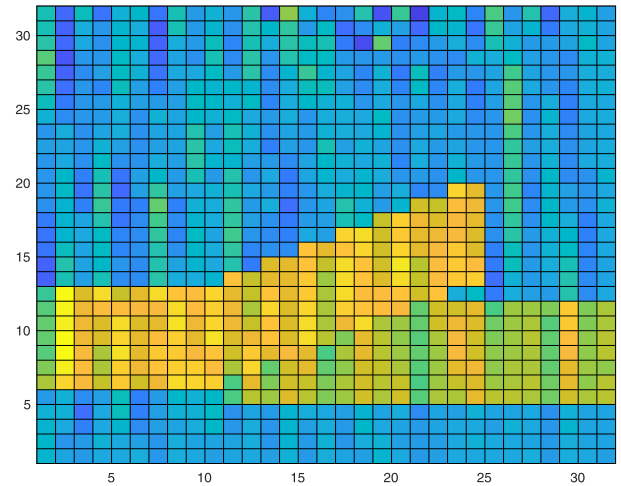


FIGURE 7. 2-D Model-Decentralized Solution using the benchmark in [33]. We pick the same node as in Figure 6 for Algorithm 1 and generate the image.

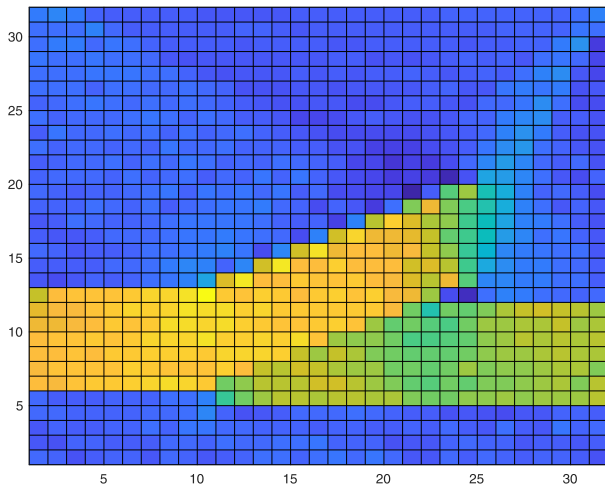


FIGURE 6. 2-D Model-Decentralized Solution using Algorithm 1. We let each node in the network run Algorithm 1 and randomly pick a node's solution after 80 global iterations and generate its tomography.

decompose (12) such that, the local objective function F_i for node i is expressed as follows.

$$F_i = \frac{1}{2} \|A_i x - b_i\|_2^2 + \lambda_i^2 \|x\|_2^2, \quad (13)$$

where the data in A_i and b_i is privately owned by i -th IoT node only. For the seismic tomography problem, two different data sets have been used to test the performance of the proposed algorithm: a 2-D synthetic data set, and a 3-D synthetic seismic tomography data set. The data sets are constructed and used to simulate performing seismic tomography in IoT wireless networks. The regularization parameter λ is set to 1 in all scenarios and λ_i can be set accordingly (all the nodes have the same regularization parameter). The resolution in the seismic model represents the number of blocks along the x , y and z -axis over the interested region.

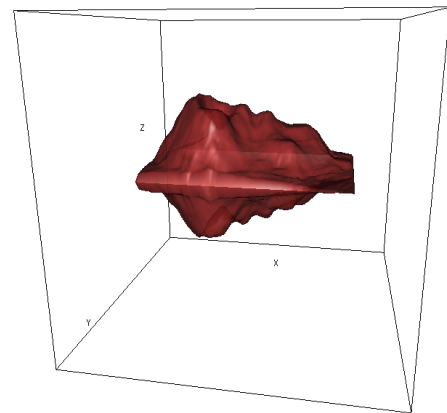


FIGURE 8. Ground Truth of the 3-D Seismic Imaging Model. The model contains a magma chamber (low velocity area) in a 10 km^3 cube.

1) TESTS ON SYNTHETIC 2-D SEISMIC DATA SET

We first test the performance of Algorithm 1 on a synthetic 2-D seismic data set. The data set is generated by using the code in [43]. A seismic tomography test problem is created with a 2-D square region. The receivers (seismographs) (blue dots) are scattered along the left and top boundary. The seismic event sources are located on the right boundary (green dots). The seismic rays travel from each source to each receiver (red lines) (See Fig. 3 for details).

The test region is partitioned into 32×32 small blocks (the resolution is 32×32) and our goal is to calculate the velocity information in these blocks (i.e. value of x). The number of nodes in the networks is 64 and the number of seismic events is 128. Thus, the size of matrix A is 8192×1024 and x is 1024×1 , b is 8192×1 , accordingly. In addition, the size of each sub-matrix A_i , $\forall i$ is 128×1024 . Vector b has been corrupted with a 5% Gaussian noise. The communication network is constructed such that each node has an average degree of 3 (3 immediate neighbors).

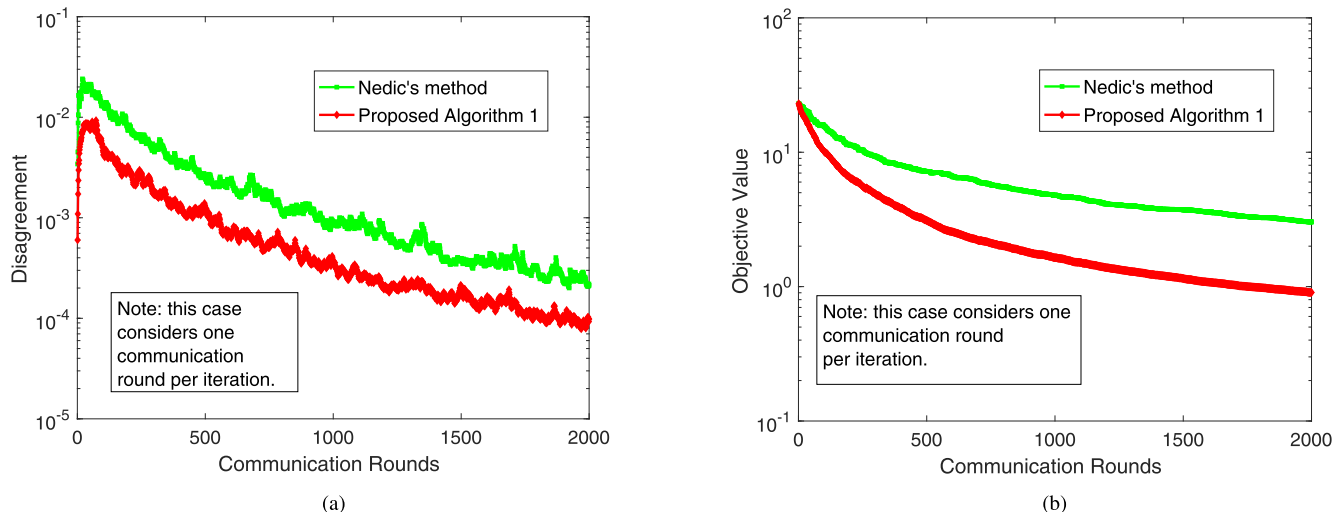


FIGURE 9. Evaluation of convergence speed using the 3-D seismic imaging data set. (a) is the plot for the disagreement term $\|x_k^i - \bar{x}_k\|$ described in Theorem 7. It tracks how close every node in the network reaches the consensus along the iterations. (b) is the illustration of average objective value defined in (13) considering all the nodes in the network. In this setting, each node i performs one broadcast to send its estimate x_{k-1}^i and (sub)gradient $\tilde{\nabla}F_i(x_{k-1}^i)$ within one iteration.

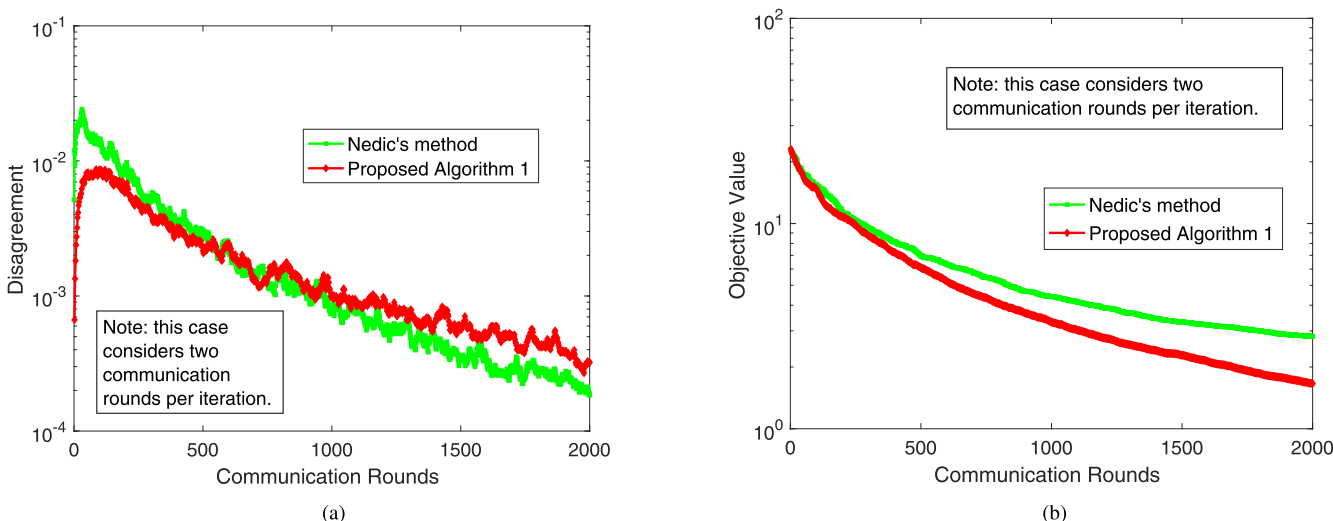


FIGURE 10. Evaluation of convergence speed using the 3-D seismic imaging data set. (a) is the plot for the disagreement term $\|x_k^i - \bar{x}_k\|$ described in Theorem 7. It measures how close each node in the network reaches the agreement along the iterations. (b) illustrates how fast the average objective value defined in (13) decreases along the iterations. In this setting, each node i performs two broadcasts to send its estimate x_{k-1}^i and (sub)gradient $\tilde{\nabla}F_i(x_{k-1}^i)$ within one iteration.

The goal of seismic tomography is to generate the “tomography” of the physical structure around some region. In Figure 4, it shows the Ground Truth for the structure of the square region in the 2-D seismic imaging model. We aim to recover the same tomography as shown in Figure 4 but in reality we are not able to do that due to noise in the measurement data. Figure 5 depicts the tomography result calculated using a centralized solver assuming all the measurement data is collected in a central place. Since the data set generated has added noise, Figure 5 is not exactly the same as the ground truth. Nevertheless, this is the best we can do. Our proposed algorithm tries to obtain the same “ideal” solution but in a decentralized manner. The centralized solution can

be used to determine if any decentralized solution is close to the optimal. The tomography results in Figure 6 and 7 are obtained using the proposed Algorithm 1 and Nedic’s method (the benchmark), respectively. Note that Figure 6 is closer to the centralized solution shown in Figure 5 than Figure 7 visually. This demonstrates the superiority of our algorithm over the benchmark.

2) TESTS ON SYNTHETIC 3-D SEISMIC DATA SET

In this section, we evaluate the proposed algorithm using a simulated seismic data set from a 3-D synthetic model of resolution $32 \times 32 \times 32$. The ground truth of the 3-D seismic model is shown in Figure 8. There are 100 nodes that are

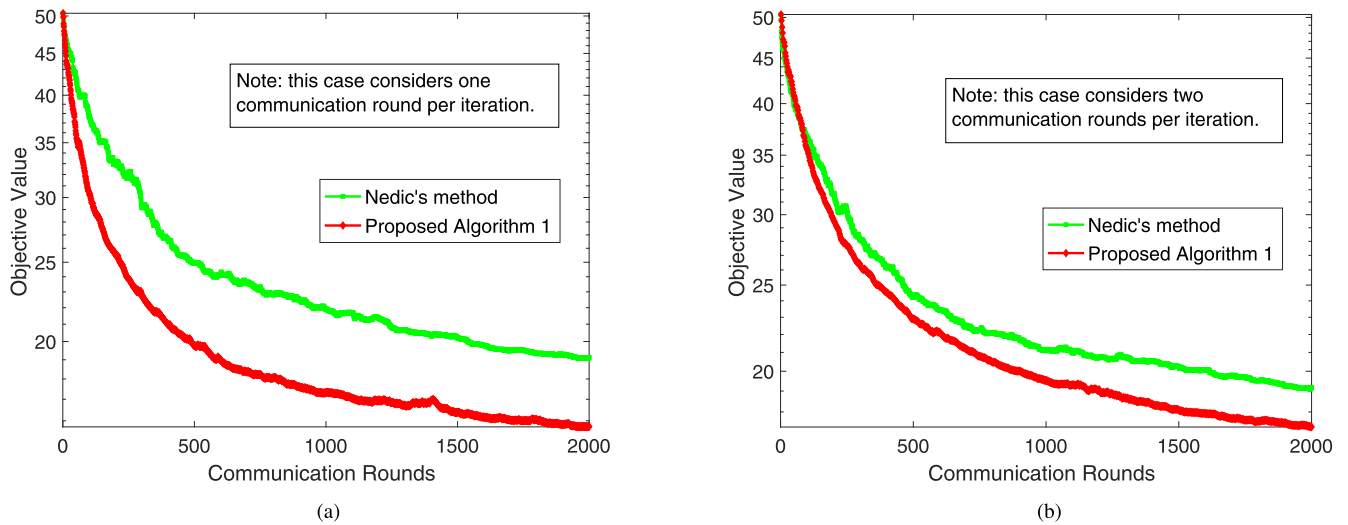


FIGURE 11. Evaluation of convergence speed using the logistic regression data set. (a) and (b) demonstrate the performance of algorithm in optimizing the average objective value along the iterations. Setting in (a) uses one broadcast in each iteration to send both model estimate and the gradient information while two communication rounds are taken in (b) to send the aforementioned two quantities separately.

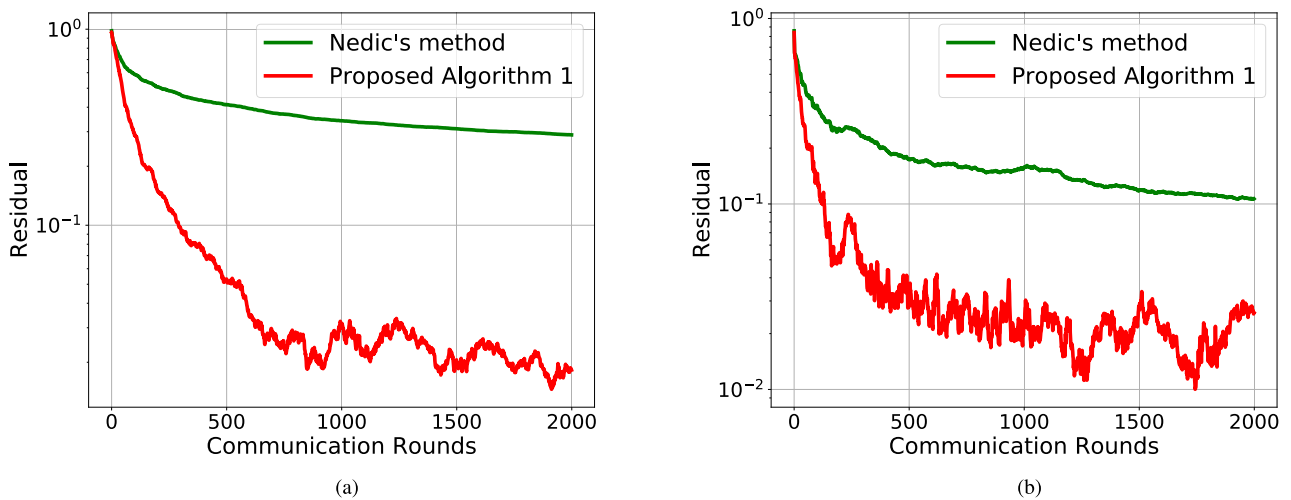


FIGURE 12. Evaluation of convergence speed for least-square problem. (a) and (b) demonstrate the performance of algorithm in terms of the residual value along the iterations. The residual at iteration k is defined as $\frac{\|x_k - x^*\|}{\|x_0 - x^*\|}$, where $x_k = [x_k^1, x_k^2, \dots, x_k^m]$ contains each node's estimate at iteration k , and x^* denotes the matrix assuming all the nodes obtain the optimal solution x^* (ideal case). Intuitively, the residual measures how close the nodes' estimates to the optimal solution. Setting in (a) uses a wheel network and (b) uses a circle network.

randomly distributed on top of the region. We simulate 400 seismic events and then compute the travel times from each source to each receiver based on the ground truth. Similar as the 2-D data set, the dimension of matrix A is $40,000 \times 32,768$. For the communication network, the average degree for each node is set to 10.

The convergence behavior of the proposed algorithm is depicted in Figure 9 and 10. Note that we test our algorithm using the same data set. The difference is that we used the normal setting in Figure 9 that one communication round is required in each iteration while in Figure 10 we assumed that each node in our proposed algorithm needs to perform two communication rounds to send the info to its neighbors (the

benchmark still uses the same setting in Figure 9 that only one communication round is needed). We would like to show how much improvement our proposed algorithm has over the benchmark. For example, the accuracy level the red curve reaches at 1000-th communication round in Figure 9(a) would be achieved at 2000-th communication round in Figure 10(a). Two metrics are used to test the convergence speed: how fast the algorithm optimizes the objective value and how fast all the nodes reach consensus. In short, our proposed algorithm converges faster than the benchmark in this 3-D seismic imaging data set illustrated in Figure 9(a) and the similar performance (even the red curve is above the green curve at the large communication rounds) shown in Figure 10(a) is due

to “unfair” setting for our proposed algorithm, which we did in purpose.

B. DECENTRALIZED MACHINE LEARNING

In this section, we evaluate the performance of our proposed methods in a machine learning task: logistic regression. The task can be formulated as an optimization problem below:

$$\min_x \sum_{i=1}^N \log(1 + \exp(-b_i \langle a_i, x \rangle)) + \lambda \|x\|_2^2. \quad (14)$$

We use the pre-processed data set in [44], which has around 20,000 newsgroup documents. To apply the decentralized algorithms into this problem, a random communication network is constructed with 10 nodes and each node has 2 immediate neighbors on average. The local objective function for node i is expressed as:

$$F_i(x^{(i)}) = \sum_{j \in \mathcal{N}_i} \log(1 + \exp(-b_j \langle a_j, x^{(i)} \rangle)) + \lambda_i \|x^{(i)}\|_2^2. \quad (15)$$

Notice that our goal is to test the speed of the proposed decentralized algorithm in obtaining the “ideal” solution for the problem in (14). We are not focus on testing whether the logistic regression model used is good for the data set or not.

Figure 11 demonstrates the fast convergence of the proposed algorithm in decreasing the objective value (training the data) and its effectiveness in decentralized machine learning application.

C. DECENTRALIZED LEAST-SQUARE PROBLEM

In this section, we evaluate our algorithm for the least-square [45], which has wide applications in statistics and data analytics such as linear regression, data fitting [46]. The formulation for the conventional centralized least-square can be cast as a optimization problem as follows.

$$\min_x \frac{1}{2} \|Ax - b\|_2^2. \quad (16)$$

To apply the decentralized algorithms, we first convert the above into its decentralized formulation as follows.

$$\min_x \sum_{i=1}^m \frac{1}{2} \|A_i x - b_i\|_2^2. \quad (17)$$

The local objective function for node i is thus $F_i(x) = \frac{1}{2} \|A_i x - b_i\|_2^2$. Matrix A is randomly generated with size 50×10 . Each element in the matrix A and vector b is sampled from a uniform distribution over $[0, 1)$. We conduct the experiments with 10 (i.e. $m = 10$) nodes in the network. A and b are evenly decomposed for all the nodes. The dimension of A_i and b_i are 5×10 and 5×1 , respectively. Regarding the network topology, a circle network and a wheel network are both tested. The experiment results are illustrated in Figure 12. Note that in both cases, our proposed FDDA algorithm significantly outperforms the benchmark.

VIII. CONCLUSION

We proposed a fast decentralized data analytics framework in IoT wireless networks. The framework is based on a new light-weight, asynchronous, and broadcast-based algorithm, which exhibits the following merits: 1) No data collection to central place required; 2) Fast convergence in terms of optimizing the objective; 3) No coordinator required and each IoT node can run the algorithm asynchronously; 4) Data Privacy of the nodes in the network is preserved. Furthermore, our developed decentralized mechanism is scalable in the sense that the mechanism leverages the computational resource of all the nodes and let them process their data locally and in parallel. Finally, various case studies demonstrate that the proposed framework is transferable once the application problem can be formulated in the convex optimization context, and capable of solving a large class of Big Data computing problems.

APPENDICES

APPENDIX A

PROOF OF THEOREM VI.7

Proof: We follow the framework adopted in [33] to prove this theorem. Let s_k^t be the vector with components $[x_k^i]_t, \forall i \in \mathcal{V}$, where $[x_k^i]_t$ is the t -th element of node i 's estimate at iteration k . Then it follows that:

$$s_k^t = W_k s_{k-1}^t + d_k^t, \quad (18)$$

where d_k^t is a vector defined as follows.

$$[d_k^t]_i = \left[-\alpha_{i,k} \left(\tilde{\nabla} F_i(y_k^i) + e_k^i \right) \right]_t, i \in J_k \quad (19)$$

And $[d_k^t]_i = 0$ otherwise. Using the definition for s_k^t , it follows that:

$$[\bar{x}_k]_t = \frac{1}{m} \mathbf{1}^T s_k^t \quad (20)$$

where $\mathbf{1} = [1, 1, \dots, 1]^T$ denotes a vector containing all 1's. Also, plugging (20) into (18) yields:

$$[\bar{x}_k]_t = \frac{1}{m} \left(\mathbf{1}^T W_k s_{k-1}^t + \mathbf{1}^T d_k^t \right). \quad (21)$$

Using (18) and (21) leads to the following.

$$s_k^t - [\bar{x}_k]_t \mathbf{1} = \left(W_k - \frac{1}{m} \mathbf{1} \mathbf{1}^T W_k \right) s_{k-1}^t + \left(\mathbf{I} - \frac{1}{m} \mathbf{1} \mathbf{1}^T \right) d_k^t \quad (22)$$

where \mathbf{I} represents the identity matrix. Using the definition and the stochasticity property of matrix W_k , it follows that $W_k \mathbf{1} = \mathbf{1}$. Further, it derives the following:

$$\left(W_k - \frac{1}{m} \mathbf{1} \mathbf{1}^T W_k \right) [\bar{x}_{k-1}]_t \mathbf{1} = 0. \quad (23)$$

Adding the right part of (23) into both sides of (21) yields:

$$s_k^t - [\bar{x}_k]_t \mathbf{1} = \left(W_k - \frac{1}{m} \mathbf{1} \mathbf{1}^T W_k \right) (s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1})$$

$$+ \left(\mathbf{I} - \frac{1}{m} \mathbf{1}\mathbf{1}^T \right) d_k^t \quad (24)$$

To simplify the notation, we define

$$Q_k = W_k - \frac{1}{m} \mathbf{1}\mathbf{1}^T W_k, \quad U = \mathbf{I} - \frac{1}{m} \mathbf{1}\mathbf{1}^T$$

Taking the norm and conditional expectation on both sides of equation (24) yields:

$$\begin{aligned} & \mathbb{E} \left[\left\| s_k^t - [\bar{x}_k]_t \mathbf{1} \right\| \mid \Omega_{k-1} \right] \\ & \leq \mathbb{E} \left[\left\| Q_k (s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1}) \right\| \mid \Omega_{k-1} \right] \\ & \quad + \mathbb{E} \left[\left\| U d_k^t \right\| \mid \Omega_{k-1} \right] \end{aligned} \quad (25)$$

where Ω_k is the σ -algebra containing the past history up to iteration k , i.e.

$$\Omega_k = \left\{ x_0^i, i_t, j_t, \forall i \in \mathcal{V}, t = 0, 1, \dots, k \right\}. \quad (26)$$

Lemma 2 implies that the matrix is independent of the past history and $\mu < 1$ (μ is the largest eigenvalue mentioned there). Based on that, the first term on the right-hand side of (25) can be bounded as follows.

$$\begin{aligned} & \mathbb{E} \left[\left\| Q_k (s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1}) \right\|^2 \mid \Omega_{k-1} \right] \\ & \leq \mu \left\| s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1} \right\|^2. \end{aligned} \quad (27)$$

Applying the property $\mathbb{E} [\|z\|] \leq \sqrt{\mathbb{E} [\|z\|^2]}$ into (27) yields:

$$\begin{aligned} & \mathbb{E} \left[\left\| Q_k (s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1}) \right\| \mid \Omega_{k-1} \right] \\ & \leq \sqrt{\mu} \left\| s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1} \right\|. \end{aligned} \quad (28)$$

Now the second item in the right hand side of (25) needs to be bounded. It can be shown that U is a projection matrix:

$$U \mathbf{1} = \left(\mathbf{I} - \frac{1}{m} \mathbf{1}\mathbf{1}^T \right) \mathbf{1} = \mathbf{0}. \quad (29)$$

The norm of matrix U is thus 1. Furthermore, the following can be obtained (also from the definition of d_k^t in (19)):

$$\left\| U d_k^t \right\|^2 \leq \left\| d_k^t \right\|^2 \leq \sum_{i \in J_k} \left\| \alpha_{i,k} \left(\tilde{\nabla} F_i(y_k^i) + e_k^i \right) \right\|^2. \quad (30)$$

Lemma 3 indicates the bound for $\alpha_{i,k}$. Based on that, the right hand side of (30) can be further bounded as follows.

$$\begin{aligned} \left\| U d_k^t \right\|^2 & \leq \sum_{i \in J_k} \alpha_{i,k}^2 \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2 \\ & \leq \frac{4m^2}{k^2 p_*^2} \sum_{i \in J_k} \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2. \end{aligned} \quad (31)$$

Taking conditional expectation on both sides of (31) together with the property that $|J_k| \leq m$ yields:

$$\mathbb{E} \left[\left\| U d_k^t \right\|^2 \mid \Omega_{k-1} \right] \leq \frac{4m^3}{k^2 p_*^2} \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2. \quad (32)$$

We can obtain the following by using the inequality $\mathbb{E} [\|z\|] \leq \sqrt{\mathbb{E} [\|z\|^2]}$ again.

$$\mathbb{E} \left[\left\| U d_k^t \right\| \mid \Omega_{k-1} \right] \leq \frac{2m\sqrt{m}}{k p_*} \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|. \quad (33)$$

Now the upper bounds for all the terms on the right hand side of (25) have been obtained. Plugging (28) and (33) into (25) yields

$$\begin{aligned} & \mathbb{E} \left[\left\| s_k^t - [\bar{x}_k]_t \mathbf{1} \right\| \mid \Omega_{k-1} \right] \\ & \leq \sqrt{\mu} \left\| s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1} \right\| + \frac{2m\sqrt{m}}{k p_*} \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|. \end{aligned} \quad (34)$$

Then the following can be obtained (since $\frac{1}{k-1} > \frac{1}{k}$):

$$\begin{aligned} & \frac{1}{k} \mathbb{E} \left[\left\| s_k^t - [\bar{x}_k]_t \mathbf{1} \right\| \mid \Omega_{k-1} \right] \leq \frac{1}{k-1} \left\| s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1} \right\| \\ & \quad - \frac{1-\sqrt{\mu}}{k} \left\| s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1} \right\| \\ & \quad + \frac{2m\sqrt{m}}{k^2 p_*} \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|. \end{aligned} \quad (35)$$

Leveraging the second statement in Lemma 1, it can be shown that the following holds almost surely for any t .

$$\sum_{k=1}^{\infty} \frac{1}{k} \left\| s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1} \right\| < \infty. \quad (36)$$

Using the definition of s_k^t again can show that the following holds almost surely.

$$\sum_{k=1}^{\infty} \frac{1}{k} \left\| x_{k-1}^i - \bar{x}_{k-1} \mathbf{1} \right\| < \infty. \quad (37)$$

At this point, we have verified the first claim in Theorem 7. For the remaining part, it needs to show the following holds almost surely first.

$$\lim_{k \rightarrow \infty} \left\| s_k^t - [\bar{x}_k]_t \mathbf{1} \right\| = 0. \quad (38)$$

From equation (36), we can derive that:

$$\liminf_{k \rightarrow \infty} \left\| s_k^t - [\bar{x}_k]_t \mathbf{1} \right\| = 0. \quad (39)$$

To prove (38), we need to show that $\left\| s_k^t - [\bar{x}_k]_t \mathbf{1} \right\|$ converges when $k \rightarrow \infty$. To achieve that, first take the square norm and conditional expectation on both sides of (24). The derivations are shown as follows.

$$\begin{aligned} & \mathbb{E} \left[\left\| s_k^t - [\bar{x}_k]_t \mathbf{1} \right\|^2 \mid \Omega_{k-1} \right] \\ & \leq \mathbb{E} \left[\left\| Q_k (s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1}) \right\|^2 \mid \Omega_{k-1} \right] \\ & \quad + 2 \sqrt{\mathbb{E} \left[\left\| Q_k (s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1}) \right\|^2 \mid \Omega_{k-1} \right]} \\ & \quad \times \sqrt{\mathbb{E} \left[\left\| U d_k^t \right\|^2 \mid \Omega_{k-1} \right]} + \mathbb{E} \left[\left\| U d_k^t \right\|^2 \mid \Omega_{k-1} \right]. \end{aligned} \quad (40)$$

Plugging (27) - (28) and (32) - (33) into (40) yields

$$\begin{aligned} & \mathbb{E} \left[\left\| s_k^t - [\bar{x}_k]_t \mathbf{1} \right\|^2 \mid \Omega_{k-1} \right] \\ & \leq \mu \left\| s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1} \right\|^2 + \frac{4m^3}{k^2 p_*^2} \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2 \\ & \quad + \sqrt{\mu} \left\| s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1} \right\| \frac{2m\sqrt{m}}{k p_*} \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\| \end{aligned}$$

$$\begin{aligned} &\leq \mu \|s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1}\|^2 + \frac{8m^3}{k^2 p_*^2} \left(\frac{4m^4}{k^2 p_*^2} + 1 \right) G^2 \\ &\quad + \sqrt{\mu} \|s_{k-1}^t - [\bar{x}_{k-1}]_t \mathbf{1}\| \frac{2m\sqrt{m}}{kp_*} \left(\frac{2m^2}{kp_*} + 1 \right) G \quad (41) \end{aligned}$$

Using (36) and the first statement in Lemma 1, it can be seen that $\|s_k^t - [\bar{x}_k]_t \mathbf{1}\|$ converges almost surely for any t . Now (38) is proved. The almost sure convergence of the disagreement can be immediately obtained (second conclusion) according to the definition of s_k^t . This completes the entire proof of Theorem 7. \square

APPENDIX B PROOF OF THEOREM VI.8

Proof: The proof plan is to use the supermartingale convergence theorem in Lemma 1, which is commonly adopted to prove the almost sure convergence. We follow the framework adopted in [33], [34] to prove this theorem. The challenge lies in bounding and fitting the terms in Lemma 1. Now consider node i with $i \in J_k$. First, subtract x (some point in the feasible set) and then take square norm on both sides of the second equation in (3). We can obtain the following using the nonexpansive property of the projection operation described in Lemma 4.

$$\begin{aligned} \|x_k^i - x\|^2 &\leq \|y_k^i - x\|^2 + \alpha_{i,k}^2 \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2 \\ &\quad - 2\alpha_{i,k} \left(\tilde{\nabla} F_i(y_k^i) + e_k^i \right)^T (y_k^i - x), \quad (42) \end{aligned}$$

where $e_k^i = \frac{\rho_{i,k}}{\alpha_{i,k}} \left(\sum_{u \in \mathcal{N}_i} \tilde{\nabla} F_u(x_{\tau_{u,k}}^u) \right)$. Combing $\alpha_{i,k} = \left(\alpha_{i,k} - \frac{1}{k\delta_i} \right) + \frac{1}{k\delta_i}$ and the third inequality in Lemma 3, the following inequality holds.

$$\begin{aligned} \|x_k^i - x\|^2 &\leq \|y_k^i - x\|^2 + \alpha_{i,k}^2 \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2 \\ &\quad - \frac{2}{k\delta_i} \left(\tilde{\nabla} F_i(y_k^i) + e_k^i \right)^T (y_k^i - x) \\ &\quad + \left| \alpha_{i,k} - \frac{1}{k\delta_i} \right| \left(\tilde{\nabla} F_i(y_k^i) + e_k^i \right)^T (y_k^i - x) \\ &\leq \|y_k^i - x\|^2 + \alpha_{i,k}^2 \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2 \\ &\quad - \frac{2}{k\delta_i} \left(\tilde{\nabla} F_i(y_k^i) + e_k^i \right)^T (y_k^i - x) \\ &\quad + \frac{4}{k^{3/2-q} p_*^2} \left(\tilde{\nabla} F_i(y_k^i) + e_k^i \right)^T (y_k^i - x) \quad (43) \end{aligned}$$

Using the fact that $2a^T b \leq \|a\|^2 + \|b\|^2$, the inner product on the right hand side of (43) can be further bounded and the the following inequality holds.

$$\begin{aligned} \|x_k^i - x\|^2 &\leq \|y_k^i - x\|^2 + \alpha_{i,k}^2 \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2 \\ &\quad - \frac{2}{k\delta_i} \left(\tilde{\nabla} F_i(y_k^i) + e_k^i \right)^T (y_k^i - x) \\ &\quad + \frac{2}{k^{3/2-q} p_*^2} \left(\left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2 + \|y_k^i - x\|^2 \right) \end{aligned}$$

$$\begin{aligned} &\leq (1 + b_k) \|y_k^i - x\|^2 - \frac{2}{k\delta_i} \left(\tilde{\nabla} F_i(y_k^i) + e_k^i \right)^T (y_k^i - x) \\ &\quad + \left(\alpha_{i,k}^2 + b_k \right) \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2, \quad (44) \end{aligned}$$

where $b_k = \frac{2}{k^{3/2-q} p_*^2}$. Since function F_i is convex and the (sub)gradient is bounded, the following inequality holds for arbitrary a, b , and c (Lemma 5).

$$\tilde{\nabla} F_i(a)^T (a - b) \geq F_i(c) - F_i(b) - G \|a - c\|. \quad (45)$$

Plugging (45) into (44) with $a = y_k^i$, $b = x$, $c = \bar{x}_{k-1}$ yields

$$\begin{aligned} \|x_k^i - x\|^2 &\leq (1 + b_k) \|y_k^i - x\|^2 \\ &\quad - \frac{2}{k\delta_i} (F_i(\bar{x}_{k-1}) - F_i(x)) \\ &\quad + \frac{2G}{k\delta_i} \|y_k^i - \bar{x}_{k-1}\| - \frac{2}{k\delta_i} (e_k^i)^T (y_k^i - x) \\ &\quad + \left(\alpha_{i,k}^2 + b_k \right) \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2. \quad (46) \end{aligned}$$

Take conditional expectation on both sides of (46) and then apply Lemma 6, it follows that

$$\begin{aligned} \mathbb{E} \left[\|x_k^i - x\|^2 \mid \Omega_{k-1}, i_k, J_k \right] &\leq (1 + b_k) \|y_k^i - x\|^2 \\ &\quad - \frac{2}{k\delta_i} (F_i(\bar{x}_{k-1}) - F_i(x)) + \frac{2G}{k\delta_i} \|y_k^i - \bar{x}_{k-1}\| \\ &\quad + \frac{2}{k\delta_i} \mathbb{E} \left[- (e_k^i)^T (y_k^i - x) \mid \Omega_{k-1}, i_k, J_k \right] \\ &\quad + \left(\alpha_{i,k}^2 + b_k \right) \mathbb{E} \left[\left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2 \mid \Omega_{k-1}, i_k, J_k \right]. \quad (47) \end{aligned}$$

Leveraging the facts that $|a^T b| \leq \|a\| \|b\|$, $\|a + b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$ and the bounded (sub)gradient assumption again, the two terms associated with e_k^i in (47) can be bounded as follows.

$$\begin{aligned} - (e_k^i)^T (y_k^i - x) &\leq \|e_k^i\| \|y_k^i - x\| \\ &\leq \frac{\rho_{i,k}}{\alpha_{i,k}} \left\| \sum_{u \in \mathcal{N}_i} \tilde{\nabla} F_u(x_{\tau_{u,k}}^u) \right\| \|y_k^i - x\| \\ &\leq \frac{\rho_{i,k}}{\alpha_{i,k}} \left(\sum_{u \in \mathcal{N}_i} \left\| \tilde{\nabla} F_u(x_{\tau_{u,k}}^u) \right\| \right) \|y_k^i - x\| \\ &\leq \frac{\rho_{i,k} |\mathcal{N}_i| G}{\alpha_{i,k}} \|y_k^i - x\|, \quad (48) \end{aligned}$$

$$\begin{aligned} \left\| \tilde{\nabla} F_i(y_k^i) + e_k^i \right\|^2 &\leq 2 \left\| \tilde{\nabla} F_i(y_k^i) \right\|^2 + 2 \left\| e_k^i \right\|^2 \\ &\leq 2 \left(\frac{\rho_{i,k} |\mathcal{N}_i| G}{\alpha_{i,k}} \right)^2 + 2G^2, \quad (49) \end{aligned}$$

where $|\mathcal{N}_i|$ is the cardinality of set \mathcal{N}_i . Recall that δ_i is the probability of node i updates (the event that it receives broadcast from any of its neighbors and each node i will be selected uniformly at certain iteration). Hence, inequality

$\delta_i \geq \frac{\rho_{i,k}}{m}$ holds. Combining this fact with (48)-(49), (47) can be further deduced as follows.

$$\begin{aligned} & \mathbb{E} \left[\left\| x_k^i - x \right\|^2 \mid \Omega_{k-1}, i_k, J_k \right] \\ & \leq (1 + b_k) \left\| y_k^i - x \right\|^2 - \frac{2}{k\delta_i} (F_i(\bar{x}_{k-1}) - F_i(x)) \\ & \quad + \frac{2mG}{kp_*} \left\| y_k^i - \bar{x}_{k-1} \right\| + \frac{2m|\mathcal{N}_i|G\rho_{i,k}}{kp_*\alpha_{i,k}} \left\| y_k^i - x \right\| \\ & \quad + (\alpha_{i,k}^2 + b_k) \left[2 \left(\frac{\rho_{i,k}|\mathcal{N}_i|G}{\alpha_{i,k}} \right)^2 + 2G^2 \right]. \end{aligned} \quad (50)$$

Now let $x = x^*$ where x^* is an optimal point of the objective function. Substituting this into (50) yields

$$\begin{aligned} & \mathbb{E} \left[\left\| x_k^i - x^* \right\|^2 \mid \Omega_{k-1}, i_k, J_k \right] \\ & \leq (1 + b_k) \left\| y_k^i - x^* \right\|^2 - \frac{2}{k\delta_i} (F_i(\bar{x}_{k-1}) - F_i(x^*)) \\ & \quad + \frac{2mG}{kp_*} \left\| y_k^i - \bar{x}_{k-1} \right\| + \frac{2m|\mathcal{N}_i|G\rho_{i,k}}{kp_*\alpha_{i,k}} \left\| y_k^i - x^* \right\| \\ & \quad + (\alpha_{i,k}^2 + b_k) \left[2 \left(\frac{\rho_{i,k}|\mathcal{N}_i|G}{\alpha_{i,k}} \right)^2 + 2G^2 \right]. \end{aligned} \quad (51)$$

It can be shown that $y_k^i \in X$ using the definition of y_k^i and the assumption that constraint set X is bounded. Thus, $\|y_k^i - x^*\|$ can be upper-bounded as follows.

$$\left\| y_k^i - x^* \right\| \leq d_X, \text{ where } d_X = \max_{a,b \in X} \|a - b\|. \quad (52)$$

Incorporating the case when $i \notin J_k$ ($x_k^i = y_k^i$) with the current formula (which assumes $i \in J_k$), and also with the definition that δ_i denotes the total probability that node i updates, we can obtain (also with fact in (52))

$$\begin{aligned} & \mathbb{E} \left[\left\| x_k^i - x^* \right\|^2 \mid \Omega_{k-1} \right] \\ & \leq (1 + b_k) \mathbb{E} \left[\left\| y_k^i - x^* \right\|^2 \mid \Omega_{k-1} \right] \\ & \quad + \frac{2mG}{kp_*} \mathbb{E} \left[\left\| y_k^i - \bar{x}_{k-1} \right\| \mid \Omega_{k-1} \right] + \frac{2m|\mathcal{N}_i|Gd_X\rho_{i,k}}{kp_*\alpha_{i,k}} \\ & \quad + \delta_i (\alpha_{i,k}^2 + b_k) \left[2 \left(\frac{\rho_{i,k}|\mathcal{N}_i|G}{\alpha_{i,k}} \right)^2 + 2G^2 \right] \\ & \quad - \frac{2}{k} (F_i(\bar{x}_{k-1}) - F_i(x^*)). \end{aligned} \quad (53)$$

In the next, sum up for all the nodes $i \in \mathcal{V}$ in the network on both sides of (53). We can then obtain the following by using Lemma 6 and (1).

$$\begin{aligned} & \sum_{i=1}^m \mathbb{E} \left[\left\| x_k^i - x^* \right\|^2 \mid \Omega_{k-1} \right] \\ & \leq (1 + b_k) \sum_{i=1}^m \left\| x_{k-1}^i - x^* \right\|^2 - \frac{2}{k} (F(\bar{x}_{k-1}) - F(x^*)) \end{aligned}$$

$$\begin{aligned} & + \frac{2mG}{kp_*} \sum_{i=1}^m \left\| x_{k-1}^i - \bar{x}_{k-1} \right\| + \frac{2m^2NGd_X\rho_{i,k}}{kp_*\alpha_{i,k}} \\ & + \sum_{i=1}^m \delta_i (\alpha_{i,k}^2 + b_k) \left[2 \left(\frac{\rho_{i,k}NG}{\alpha_{i,k}} \right)^2 + 2G^2 \right], \end{aligned} \quad (54)$$

where $N = \max_s |\mathcal{N}_s|$. Pick parameter $\rho_{i,k}$ such that $\sum_{k=1}^{\infty} \frac{\rho_{i,k}}{k\alpha_{i,k}} < \infty$ (Assumption 3). Then it can be seen that (considering the definitions of $\alpha_{i,k}$ and b_k).

$$\sum_{k=1}^{\infty} b_k < \infty, \quad \sum_{k=1}^{\infty} (\alpha_{i,k}^2 + b_k) < \infty. \quad (55)$$

In addition, based on the first conclusion of Theorem 7, it is known that the following holds almost surely.

$$\sum_{k=1}^{\infty} \frac{2mG}{kp_*} \sum_{i=1}^m \left\| x_{k-1}^i - \bar{x}_{k-1} \right\| < \infty. \quad (56)$$

Considering the last three terms in (54) as one item along with the fact that $F(\bar{x}_{k-1}) - F(x^*) \geq 0$, we can see that all the conditions of Lemma 1 have been satisfied. Hence, it concludes that the sequence $\left\{ \sum_{i=1}^m \left\| x_k^i - x^* \right\|^2 \right\}$ converges and

$$\sum_{k=\bar{k}}^{\infty} \frac{1}{k} (F(\bar{x}_{k-1}) - F(x^*)) < \infty. \quad (57)$$

Similar to the proof for Theorem 7, it can be deduced from (57) that

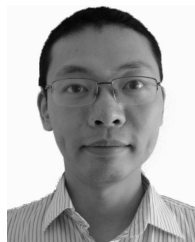
$$\liminf_{k \rightarrow \infty} F(\bar{x}_{k-1}) = F(x^*) \quad (58)$$

Now the sequence $\left\{ \sum_{i=1}^m \left\| x_k^i - x^* \right\|^2 \right\}$ converges and (58) holds for any point x^* in the set of optimal solutions X^* , it is known that there exists a subsequence $\{\bar{x}_{k_j}\}$ (of sequence $\{\bar{x}_k\}$) such that $\bar{x}_{k_j} \rightarrow \hat{x}$ for some \hat{x} in the feasible set X and $\lim_{j \rightarrow \infty} F(\bar{x}_{k_j}) = F(x^*)$. Since function F is continuous and \bar{x}_{k_j} converges to \hat{x} , we can obtain that $\lim_{j \rightarrow \infty} F(\bar{x}_{k_j}) = F(\hat{x})$. Thus, it follows that $F(\hat{x}) = F(x^*)$, which implies that \hat{x} belongs to the optimal solution set X^* . At this point, we know that the sequence $\left\{ \sum_{i=1}^m \left\| x_k^i - \hat{x} \right\|^2 \right\}$ converges. Using the second claim in Theorem 7) such that $\left\{ \sum_{i=1}^m \left\| x_k^i - \bar{x}_k \right\|^2 \right\} \rightarrow 0$ as $k \rightarrow \infty$, it is known that $\left\| \bar{x}_k - \hat{x} \right\|^2$ converges. Since the subsequence $\left\| \bar{x}_{k_j} - \hat{x} \right\|^2 \rightarrow 0$, there is $\left\| \bar{x}_k - \hat{x} \right\| \rightarrow 0$, which indicates that $\{\bar{x}_k\}$ converges to an optimal point (\hat{x}) of the problem. Lastly, applying the second statement in Theorem 7 one more time, it can be obtained that the sequence $\{x_k^i\}$ generated by any node $i \in \mathcal{V}$ converges to the same optimal solution point almost surely. The proof of the theorem for Algorithm 1 is thus complete. \square

REFERENCES

[1] S. S. L. Rosencrance and I. Wigmore, (2016). *Internet Things (IoT)*. Accessed: Nov. 1, 2018. [Online]. Available: <https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT>

- [2] D. Evans, "The Internet of Things: How the next evolution of the Internet is changing everything," CISCO, San Jose, CA, USA, White Paper, 2011, vol. 1, pp. 1–11. [Online]. Available: https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf
- [3] "Fog computing and the Internet of Things: Extend the cloud to where the things are," CISCO, San Jose, CA, USA, White Paper, 2016, vol. 1, pp. 1–6. [Online]. Available: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf
- [4] R. Barona and E. A. M. Anita, "A survey on data breach challenges in cloud computing security: Issues and threats," in *Proc. Int. Conf. Circuit, Power Comput. Technol. (ICCPCT)*, Apr. 2017, pp. 1–8.
- [5] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog computing: Platform and applications," in *Proc. 3rd IEEE Workshop Hot Topics Web Syst. Technol. (HotWeb)*, Nov. 2015, pp. 73–78.
- [6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.
- [7] F. Wang, J. Xu, and Z. Ding, "Multi-antenna NOMA for computation offloading in multiuser mobile Edge computing systems," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2450–2643, Mar. 2019.
- [8] A. H. Sayed and C. G. Lopes, "Distributed recursive least-squares strategies over adaptive networks," in *Proc. 14th Asilomar Conf. Signals, Syst. Comput. (ACSSC)*, Oct. 2006, pp. 233–237.
- [9] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion recursive least-squares for distributed estimation over adaptive networks," *IEEE Trans. Signal Process.*, vol. 56, no. 5, pp. 1865–1877, May 2008.
- [10] G. Mateos and G. B. Giannakis, "Distributed recursive least-squares: Stability and performance analysis," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3740–3754, Jul. 2012.
- [11] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1035–1048, Mar. 2010.
- [12] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3122–3136, Jul. 2008.
- [13] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS algorithms with information exchange," in *Proc. 42nd Asilomar Conf. Signals, Syst. Comput.*, Oct. 2008, pp. 251–255.
- [14] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov. 2010.
- [15] I. Matic and J. S. Baras, "Performance evaluation of the consensus-based distributed subgradient method under random communication topologies," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 4, pp. 754–771, Aug. 2011.
- [16] 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.
- [17] A. Nedić and A. Olshevsky, "Distributed optimization over time-varying directed graphs," in *Proc. 52nd IEEE Conf. Decis. Control*, Dec. 2013, pp. 6855–6860.
- [18] A. Nedic and A. Olshevsky, "Stochastic gradient-push for strongly convex functions on time-varying directed graphs," 2014, *arXiv:1406.2075*. [Online]. Available: <https://arxiv.org/abs/1406.2075>
- [19] I.-A. Chen, "Fast distributed first-order methods," M.S. thesis, Dept. Elect. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, USA, 2012.
- [20] J. M. F. M. Dusan Jakovetic and J. Xavier, "Fast distributed gradient methods," 2014, *arXiv:1112.2972v4*. [Online]. Available: <https://arxiv.org/abs/1112.2972>
- [21] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," 2013, *arXiv:1310.7063*. [Online]. Available: <https://arxiv.org/abs/1310.7063>
- [22] M. Zargham, A. Ribeiro, and A. Jadbabaie, "A distributed line search for network optimization," in *Proc. Amer. Control Conf. (ACC)*, Jun. 2012, pp. 472–477.
- [23] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. 4th Int. Symp. Inf. Process. Sensor Netw. (IPSN)*, 2005.
- [24] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Autom. Control*, vol. 31, no. 9, pp. 803–812, Sep. 1986.
- [25] J. N. Tsitsiklis, "Problems in decentralized decision making and computation," DTIC Document, Fort Belvoir, VA, USA, Tech. Rep. DTIC ADA150025, 1984.
- [26] H. Terelius, U. Topcu, and R. M. Murray, "Decentralized multi-agent optimization via dual decomposition," *IFAC Proc. Volumes*, vol. 44, no. 1, pp. 11245–11251, 2011.
- [27] G. Shi and K. H. Johansson, "Finite-time and asymptotic convergence of distributed averaging and maximizing algorithms," *arXiv:1205.1733*. [Online]. Available: <https://arxiv.org/abs/1205.1733>
- [28] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proc. 3rd Int. Symp. Inform. Process. Sensor Netw.*, Apr. 2004, pp. 20–27.
- [29] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," 2014, *arXiv:1404.6264*. [Online]. Available: <https://arxiv.org/abs/1404.6264>
- [30] E. Wei and A. Ozdaglar, "On the $o(1/k)$ convergence of asynchronous distributed alternating direction method of multipliers," 2013, *arXiv:1307.8254*. [Online]. Available: <https://arxiv.org/abs/1307.8254>
- [31] P. C. F. Iutzeler, P. Bianchi, and W. Hachem, "Asynchronous distributed optimization using a randomized alternating direction method of multipliers," 2013, *arXiv:1303.2837*. [Online]. Available: <https://arxiv.org/abs/1303.2837>
- [32] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione, "Broadcast gossip algorithms for consensus," *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2748–2761, Jul. 2009.
- [33] A. Nedic, "Asynchronous broadcast-based convex optimization over a network," *IEEE Trans. Autom. Control*, vol. 56, no. 6, pp. 1337–1351, Jun. 2011.
- [34] L. Zhao, W.-Z. Song, X. Ye, and Y. Gu, "Asynchronous broadcast-based decentralized learning in sensor networks," *Int. J. Parallel, Emergent Distrib. Syst.*, to be published. [Online]. Available: <http://www.tandfonline.com/doi/abs/10.1080/17445760.2017.1294690>
- [35] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 127–239, Jan. 2014. [Online]. Available: <http://dx.doi.org/10.1561/24000000003>
- [36] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Nashua, NH, USA: Athena Scientific, 1999.
- [37] D. Bertsekas, A. Nedic, and A. Ozdaglar, *Convex Analysis and Optimization*. Nashua, NH, USA: Athena Scientific, 2003.
- [38] D. P. Bertsekas, *Convex Optimization Theory*. Nashua, NH, USA: Athena Scientific, 2009.
- [39] C. C. Paige and M. A. Saunders, "LSQR: An algorithm for sparse linear equations and sparse least squares," *ACM Trans. Math. Softw.*, vol. 8, no. 1, pp. 43–71, Mar. 1982.
- [40] J. Ahrenholz, "Comparison of core network emulation platforms," in *Proc. Milcom Mil. Commun. Conf.*, Oct. 2010, pp. 166–171.
- [41] J. Ahrenholz, T. Goff, and B. Adamson, "Integration of the CORE and EMANE Network Emulators," in *Proc. Military Commun. Conf. (Milcom)*, Nov. 2011, pp. 1870–1875.
- [42] A. Michelini, "An adaptive-grid formalism for traveltime tomography," *Geophys. J. Int.*, vol. 121, no. 2, pp. 489–510, May 1995. [Online]. Available: <http://dx.doi.org/10.1111/j.1365-246x.1995.tb05728.x>
- [43] P. C. Hansen and M. Saxild-Hansen, "Air tools—A MATLAB package of algebraic iterative reconstruction methods," *J. Comput. Appl. Math.*, vol. 236, no. 8, pp. 2167–2178, 2012. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0377042711005188>
- [44] R. F. Corra, *Text Categorization Datasets*. [Online]. Available: <https://sites.google.com/site/renatocorra02/textcategorizationdatasets>
- [45] G. del Pino, "The unifying role of iterative generalized least squares in statistical algorithms," *Statist. Sci.*, vol. 4, no. 4, pp. 394–403, Nov. 1989, doi: 10.1214/ss/1177012408.
- [46] X. Yan and X. G. Su, *Linear Regression Analysis: Theory and Computing*. River Edge, NJ, USA: World Scientific, 2009.



LIANG ZHAO received the Ph.D. degree in computer science from Georgia State University, GA, USA, and the M.S. degree in electrical engineering from Lehigh University, PA, USA, in 2016 and 2012, respectively. He was a Research Data Scientist at IBM, USA. He is currently an Assistant Professor in computer science with the University of South Carolina Upstate, SC, USA. His current research interest includes optimization and data analytics for distributed systems.