Decentralized Collaborative Learning with Probabilistic Data Protection

Tsuyoshi Idé
IBM Research, T. J. Watson Research Center
Yorktown Heights, NY, USA
tide@us.ibm.com

Rudy Raymond
IBM Research–Tokyo
Tokyo, Japan
rudyhar@jp.ibm.com

***Abstract*—We discuss future directions of Blockchain as a collaborative value co-creation platform, in which network participants can gain extra insights that cannot be accessed when disconnected from the others. As such, we propose a decentralized machine learning framework that is carefully designed to respect the values of democracy, diversity, and privacy. Specifically, we propose a federated multi-task learning framework that integrates a privacy-preserving dynamic consensus algorithm. We show that a specific network topology called the expander graph dramatically improves the scalability of global consensus building. We conclude the paper by making some remarks on open problems.**

***Index Terms*—Blockchain, decentralized learning, multi-task learning, federated learning, expander graphs, data privacy, secret sharing**

I. Introduction

Spurred by the remarkable commercial success of cryptocurrencies, there have been many attempts to date to extend Blockchain as a decentralized management platform for general business transactions. In most application scenarios, such as product traceability [1]–[3] and those involving smart contracts [4], Blockchain has been used essentially as an immutable data storage whose goal is simply to manage identical replicas of data among distributed nodes. Although this is a meaningful first step, we argue that the true value of Blockchain lies in its potential for value co-creation by network participants ("agents") through knowledge sharing [5]–[7]. The platform should help the agents obtain extra insights that cannot be accessed when looking at their own data alone, disconnected from the others. We envision that the next generation of Blockchain will integrate *collaborative learning* capabilities at the core.

Inspired by the Nakamoto's original agenda [8], our collaborative learning platform features three main characteristics: 1) The entire learning procedure is done in a decentralized manner, i.e., without relying on the central authority. 2) The outcomes of the learning reflect specific situations of the individual agents, resulting in generally different models for each of the agents. 3) The data and the learned model of the agents are protected as a private property.

The *first* characteristic is to ensure *democracy* in the platform. The agents voluntarily communicate with the others through given network infrastructure, but there is no such thing as the central server that collects a piece of data from the agents and perform, e.g., stochastic gradient descent to train a deep learning model. The *second* characteristic is to ensure *diversity* in the platform. Here, a 'model' refers to a probability distribution in general that captures patterns hidden in the data, as implied in Fig. 1. In the machine learning literature, this is commonly called *multi-task learning* because the goal is to learn S different models ('tasks') simultaneously, where S is the number of agents participated in the network. The *third* characteristic is to ensure the *privacy* of the agents in the platform. Although they learn from the other agents and share their learnings in return in a certain way, their original data and the resulting model must be protected.

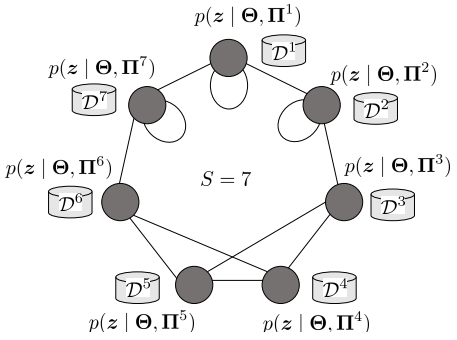

Fig. 1. Illustration of decentralized multi-task learning illustrated on a 3-regular expander graph of $S = 7$ nodes, where $p(\cdot \mid \boldsymbol{\Theta}, \boldsymbol{\Pi}^s)$ denotes a statistical machine learning model for the agent s with $\boldsymbol{\Theta}$ being common model parameters and $\boldsymbol{\Pi}^s$ being agent-specific model parameters.

This paper proposes a new framework of decentralized collaborative learning with certain privacy guarantees. Given a specific parametric model agreed upon, the goal of the agents is to maximize the likelihood function of the entire system in a collaborative manner. Higher likelihood implies higher model fidelity, which will lead to better business outcomes through more accurate predictions. In our previous work [9], we presented a decentralized multi-task learning protocol for a mixture of exponential family distributions, where the dynamic consensus algorithm [10]–[12] eliminates the need for the central server. We found that the spectral structure of network topology plays a critical role in the convergence of the algorithm, but the analysis was mainly on the cycle graph, in which an analytic form of the eigenspectrum is available. In

this paper, we show that a family of graphs called the *expander graph* [13], [14] can dramatically speedup global consensus. We also provide a systematic analysis on the probability of privacy breach in a couple of different scenarios. Decentralized collaborative learning is a new research field. We conclude the paper with some remarks on open problems.

II. Related work

One recent major trend in distributed learning is to distributedly train deep neural networks using SGD (stochastic gradient descent) [15]–[19], in which managing huge computational overhead has been an issue. Although most of the existing work falls into the category of distributed *single*-task learning, some recent works point out that model biases and thus a lack of diversity can be a serious issue [20], [21]. A multi-task extension is discussed in [22] without a data privacy context. A unique role of the exponential family in data privacy is pointed out in [23].

For privacy preservation in distributed learning, a common approach is to use differential privacy [24], [25], but for real-valued data, performance degradation in learning due to introduced noise is an open question, as discussed in [26]. Privacy preservation in a decentralized environment is a challenging task in general. Almost the only solution known so far is the use of homomorphic encryption and its variants [15], [27], [28], but its computational cost is known to be often prohibitive.

III. Multi-Task learning framework

This section summarizes the problem setting from our previous work [9].

A. Problem setup

As shown in Fig. 1, there are $S \geq 3$ agents in the network. Each agent indexed by a privately keeps its own data set

$$\mathcal{D}^a \triangleq \{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(N^a)}\} \tag{1}$$

about a random variable $\boldsymbol{z}$. Here, N^a is the number of samples of the a-th agent. As a general rule, we use the superscript such as $^{(n)}$ to represent the n-th instance of a random variable. The random variable $\boldsymbol{z}$ can be a pair of a feature vector and its label like $\boldsymbol{z} = (y, \boldsymbol{x})$ or simply a feature vector alone $\boldsymbol{z} = \boldsymbol{x}$. We assume $\boldsymbol{x}$ is *real-valued* and noisy in general. The goal of the agents is to learn a predictive distribution for $\boldsymbol{z}$. As illustrated in Fig. 1, the distribution will have model parameters $\boldsymbol{\Theta}$ and $\boldsymbol{\Pi}^a$, where $\boldsymbol{\Theta}$ is parameters shared by all the agents and $\boldsymbol{\Pi}^a$ is agent-specific parameters. Both $\boldsymbol{\Theta}$ and $\boldsymbol{\Pi}^a$ are to be learned from the total data $\mathcal{D} \triangleq \{\mathcal{D}^1, \ldots, \mathcal{D}^S\}$. The question is how the agents leverage information from the other ones while keeping data privacy.

As part of the network infrastructure, a set of bi-directional communication paths are given as an undirected graph, whose nodes are the agents and the edges are pairwise communication paths, as shown in the figure. As is the case in the IP (internet protocol) network of the Internet, the infrastructure is assumed to do basic bookkeeping jobs such as network routing and clock synchronization without any interest in the contents communicated. Network failures are unavoidable in real networks but we do not consider them for simplicity. We also assume a consortium-based network, where the agents have verified identities. The agents are *honest but curious*, meaning that they do not lie about computed statistics but they always try to selfishly get as much information as possible from the other agents.

B. Mixture of exponential family

To capture the diversity among the agent, we employ a *mixture* of the exponential family. In what follows, we focus on the case where $\boldsymbol{z} = \boldsymbol{x} \in \mathbb{R}^M$ with M being the dimensionality of the variable and the learning task is unsupervised (a.k.a. density estimation). Extension to the supervised setting can be done easily. Practical applications of this setting include failure detection of industrial robots, where all the S agents are assumed to have the same set of physical sensors. See [29] for more detail.

Now the observation model of the a-th agent is given by:

$$p(\boldsymbol{x} \mid \boldsymbol{\Theta}, \boldsymbol{u}^a) = \prod_{k=1}^{K} f(\boldsymbol{x} \mid \boldsymbol{\theta}_k)^{u_k^a} \tag{2}$$

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}_k) = G(\boldsymbol{\theta}_k) H(\boldsymbol{x}) \exp\{\boldsymbol{\eta}(\boldsymbol{\theta}_k)^\top \boldsymbol{T}(\boldsymbol{x})\}, \tag{3}$$

where K is the number of mixture components and $\boldsymbol{u}^a \triangleq (u_1^a, \ldots, u_K^a)^\top$ is the one-hot indicator variable representing cluster assignment. Notice that the dependency on a in $\boldsymbol{u}^a$ represents diversity of the model. In $f(\boldsymbol{z} \mid \boldsymbol{\theta}_k)$, functional forms of G, H (scalar function) and $\boldsymbol{\eta}, \boldsymbol{T}$ (vector-valued function) are given by a specific choice in the exponential family [30]. We will give a Gaussian-based model as an example below.

In the unsupervised scenario, the observation model in the form Eq. (2) is almost always combined with a prior distribution of $\boldsymbol{u}^a$:

$$p(\boldsymbol{u}^a \mid \boldsymbol{\pi}^a) = \mathrm{Cat}(\boldsymbol{u}^a \mid \boldsymbol{\pi}^a) \triangleq \prod_{k=1}^{K} (\pi_k^a)^{u_k^a} \tag{4}$$

where Cat denotes the categorical distribution. The parameter $\boldsymbol{\pi}^a \triangleq (\pi_1^a, \ldots, \pi_K^a)$ can be viewed as the probability distribution over the K clusters and satisfies $\sum_{k=1}^{K} \pi_k^a = 1$.

For stable numerical estimation, prior distributions are imposed also on $\boldsymbol{\Theta} \triangleq \{\boldsymbol{\theta}_k\}$ and $\boldsymbol{\Pi} \triangleq \{\boldsymbol{\pi}^a\}$ as

$$p(\boldsymbol{\Theta}) = \prod_{k=1}^{K} p(\boldsymbol{\theta}_k), \quad p(\boldsymbol{\Pi}) = \prod_{a=1}^{S} p(\boldsymbol{\pi}^a), \tag{5}$$

where we have used $p(\cdot)$ to generically represent potentially different probability distributions.

Here we give an example when $f(\boldsymbol{z} \mid \boldsymbol{\theta}_k)$ is the Gaussian $\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, (\Lambda_k)^{-1})$, where $\boldsymbol{\mu}_k$ is the mean and Λ_k the precision matrix. For $\boldsymbol{\theta}_k = \{\boldsymbol{\mu}_k, \Lambda_k\}$, one practically recommended choice for the prior distribution is the Gauss-Laplace distribution:

$$p(\boldsymbol{\mu}_k, \Lambda_k) \propto \mathcal{N}(\boldsymbol{\mu}_k | \boldsymbol{m}_0, (\lambda_0 \Lambda_k)^{-1}) \exp\left(-\frac{\rho}{2} \|\Lambda_k\|_1\right) \tag{6}$$

where $\rho, \lambda_0, \boldsymbol{m}_0$ are predefined constants and $\|\cdot\|_1$ is the ℓ_1 norm. For $p(\boldsymbol{\pi}^a)$, a common choice is the Dirichlet distribution $p(\boldsymbol{\pi}^a) \propto (\pi_1^a \cdots \pi_K^a)^\gamma$ with $\gamma \sim 1$ is a given constant.

C. Model estimation algorithm

The probabilistic model presented above contains unknown model parameters $\boldsymbol{\Theta}, \boldsymbol{\Pi}$ in addition to the latent variable $\boldsymbol{U} \triangleq \{\boldsymbol{u}^1, \ldots, \boldsymbol{u}^S\}$. The standard strategy to learn the model in such a case is to maximize the log marginalized likelihood L_0 with respect to $\boldsymbol{\Theta}, \boldsymbol{\Pi}$:

$$L_0 \triangleq \ln p(\boldsymbol{\Pi}) p(\boldsymbol{\Theta}) + \ln \left\{ \sum_{\boldsymbol{U}} \prod_{a=1}^{s} \prod_{n=1}^{N^s} p(\boldsymbol{z}^{a(n)} | \boldsymbol{\Theta}, \boldsymbol{u}^{a(n)}) p(\boldsymbol{u}^{a(n)} \mid \boldsymbol{\pi}^a) \right\}.$$

Unfortunately, this maximization problem is intractable even in the simple Gaussian case. We instead maximize the lower bound of L_0 derived by applying Jensen's inequality. We can derive a simple iterative algorithm to estimate the unknown model parameters $\boldsymbol{\Theta}, \boldsymbol{\Pi}$ as well as the posterior distribution of the latent variable $\boldsymbol{U}$:

$$Q(\boldsymbol{U}) = \prod_{a=1}^{S} \prod_{n=1}^{N^s} \prod_{k=1}^{K} (r_k^{a(n)})^{u_k^{a(n)}}. \tag{7}$$

Here, we summarize the result presented in our previous work [9]. With an initialized $\{r^{a(n)}\}$ so $\sum_{k=1}^K r_k^{a(n)} = 1$, we repeat the following steps until convergence:

- In each $a \in \{1, \ldots, S\}$, with the latest $\{r_k^{a(n)}\}$, locally compute

$$N_k^a \triangleq \sum_{n=1}^{N^a} r_k^{a(n)} \quad \text{and} \quad \boldsymbol{T}_k^a \triangleq \sum_{n=1}^{N^a} r_k^{a(n)} \boldsymbol{T}(\boldsymbol{z}^{a(n)}). \tag{8}$$

- Among all $a = 1, \ldots, S$, build a global consensus on

$$N_k \triangleq \sum_{a=1}^{S} N_k^a, \quad \text{and} \quad \boldsymbol{T}_k \triangleq \sum_{a=1}^{S} \boldsymbol{T}_k^a, \tag{9}$$

- In each $a \in \{1, \ldots, S\}$, locally solve

$$\max_{\boldsymbol{\theta}_k} \left\{ \ln p(\boldsymbol{\theta}_k) + N_k \ln G(\boldsymbol{\theta}_k) + \boldsymbol{T}_k^\top \boldsymbol{\eta}(\boldsymbol{\theta}_k) \right\} \tag{10}$$

- In each $a \in \{1, \ldots, S\}$, with the latest $\{\boldsymbol{\theta}_k\}$ and $\{\boldsymbol{\pi}^a\}$ with $\pi_k^a = \frac{N_k^a + \gamma}{N^a + K\gamma}$, locally update

$$r_k^{a(n)} = \frac{\pi_k^a f(\boldsymbol{z}^{a(n)} \mid \boldsymbol{\theta}_k)}{\sum_{m=1}^K \pi_m^a f(\boldsymbol{z}^{a(n)} \mid \boldsymbol{\theta}_m)}. \tag{11}$$

Here, we assumed that we have used the Dirichlet distribution $p(\boldsymbol{\pi}^a) \propto (\pi_1^a \cdots \pi_K^a)^\gamma$ for $p(\boldsymbol{\Pi})$. Notice that agent-agent communication is involved only in the second step; All the other steps need only local computation that can be complete within each agent. The complexity per agent per iteration is $\mathcal{O}(N^a + M^3 + \ln S)$, assuming Eq. (10) takes M^3 and the consensus step Eq. (9) takes $\ln S$ (See Theorem 5).

D. Gaussian example

To be concrete, we provide parameter updating equations for the Gaussian observation model with the Gauss-Laplace prior Eq. (6). In this case, instead of $\boldsymbol{T}_k^a$ in Eq. (8), we compute

$$\boldsymbol{m}_k^a = \sum_{n=1}^{N^s} r_k^{a(n)} \boldsymbol{x}^{a(n)}, \quad \mathsf{C}_k^a = \sum_{n=1}^{N^s} r_k^{a(n)} \boldsymbol{x}^{a(n)} \boldsymbol{x}^{a(n)\top}. \tag{12}$$

for each $a \in \{1, \ldots, S\}$ and each mixture component $k \in \{1, \ldots, K\}$. Then, in the step of global consensus in Eq. (9), we compute aggregated values as

$$\bar{\boldsymbol{m}}_k = \sum_{a=1}^{S} \boldsymbol{m}_k^a, \quad \bar{\mathsf{C}}_k = \sum_{a=1}^{S} \mathsf{C}_k^a. \tag{13}$$

Finally, in the step of optimization in Eq. (10), we compute

$$\boldsymbol{\mu}_k = \frac{1}{\lambda_0 + N_k} \bar{\boldsymbol{m}}_k, \quad \Sigma_k = \frac{1}{N_k} \bar{\mathsf{C}}_k + \boldsymbol{\mu}_k \boldsymbol{\mu}_k^\top, \tag{14}$$

$$\Lambda_k = \arg\max_{\Lambda_k} \left\{ \ln \det \Lambda_k - \mathrm{Tr}(\Lambda_k \Sigma_k) - \frac{\rho}{N_k} \|\Lambda_k\|_1 \right\}, \tag{15}$$

where Tr is the matrix trace and det is the matrix determinant. This optimization problem is well-known in covariance selection and can be solved very efficiently with the graphical lasso algorithm [31], [32]. To ensure that all the agents have the same $\{\boldsymbol{\mu}_k, \Lambda_k\}$, they can run another global consensus step to register the average as the final outcome in each iteration round.

IV. Aggregation via secret sharing

This section focuses on Eq. (9), i.e., how to securely aggregate the local statistics. Since aggregation can be done element-wise, without loss of generality, we consider the problem of computing the sum of scalars $\{\xi_a\}$:

$$\bar{\xi} = \sum_{a=1}^{S} \xi_a = \mathbf{1}_S^\top \boldsymbol{\xi}(0), \tag{16}$$

where ξ_a's are constants to be summed, $\mathbf{1}_S$ is the S-dimensional vector of ones, and we defined $\xi_a(0) = \xi_a$. Aggregation would be trivial if a trusted coordinator existed in the network. The question is how to compute the summation only through local communications and how to make it secure.

A. Aggregation through Markov transitions on graph

Let $\mathsf{A} \in \{0,1\}^{S \times S}$ be the incidence matrix of the communication graph, where only connected nodes (or neighboring nodes) can communicate with each other. As illustrated in Fig. 1, the graph is undirected but may have self-loops and multiple edges. Given $\mathsf{A} = [A_{a,j}]$, consider the following updates:

$$\xi_a(t+1) = \xi_a(t) + \epsilon \sum_{j=1}^{S} A_{a,j} [\xi_j(t) - \xi_a(t)], \tag{17}$$

where t is the number of update rounds, and ϵ is a given parameter controlling convergence. All the S nodes perform

this update by communicating with their neighbors. In the matrix form, Eq. (17) is written as

$$\boldsymbol{\xi}(t+1) = \mathsf{W}_\epsilon \boldsymbol{\xi}(t) \quad \text{with} \quad \mathsf{W}_\epsilon \triangleq \mathsf{I}_S - \epsilon(\mathsf{D} - \mathsf{A}), \tag{18}$$

where $\mathsf{D} \triangleq \mathrm{diag}(d_1, \ldots, d_S)$ is the degree matrix with d_s being the degree of the s-th node, I_S is the S-dimensional identity matrix, and $\boldsymbol{\xi}(t) \triangleq (\xi_1(t), \ldots, \xi_S(t))^\top$.

The key idea of multi-agent coordination is to associate a stationary solution of the Markov transition defined by Eq. (18) with the process of consensus building. As can be easily verified, $\boldsymbol{u}_1 = \frac{1}{\sqrt{S}}\mathbf{1}_S$ is an ℓ_2-normalized eigenvector of W_ϵ whose eigenvalue is $\lambda_1 = 1$. If this is non-degenerated and the other absolute eigenvalues are less than one, Eq. (18) will converge to the stationary solution $\boldsymbol{\xi}^*$

$$\boldsymbol{\xi}^* = {\mathsf{W}_\epsilon}^\infty \boldsymbol{\xi}(0) \approx {\lambda_1}^\infty \boldsymbol{u}_1 \boldsymbol{u}_1^\top \boldsymbol{\xi}(0) = \frac{1}{S}\bar{\xi}\mathbf{1}_S \tag{19}$$

because only the largest eigenvalue survives in the spectral expansion after an infinite number of transitions [33]. This means that all of the agents end up having the same value of $\frac{1}{S}\bar{\xi}$, which is the aggregation we wanted. We call this approach the *dynamical consensus algorithm*.

One obvious limitation of this approach is that the connected peers can see the original value in the first iteration, which is a privacy breach. To address this issue, we propose two secret sharing approaches: One is Shamir's secret sharing [34] and the other is random chunking we proposed originally in [9].

B. Shamir's secret sharing

In Shamir's secret sharing scheme, each of the S agents first generates random numbers $R_1^a, \ldots, R_{S-1}^a$ in a large enough integer domain to define an $(S-1)$-th order polynomial function. For the a-th agent, the polynomial is defined by

$$g_a(n) = \xi_a + R_1^a n + \cdots + R_{S-1}^a n^{S-1}. \tag{20}$$

Using this polynomial, each agent locally computes S values $\{g_s(1), \ldots, g_s(S)\}$. Then, by repeating the dynamical consensus algorithm S times (i.e., by setting $\xi_s(0) = g_s(n)$ for $n = 1, \ldots, S$ in Eq. (17)), the agents build a consensus on S aggregated values $\{\bar{g}(1), \ldots, \bar{g}(S)\}$, where $\bar{g}(l) \triangleq \sum_{s=1}^S g_s(l)$. Since R_1^a etc. are random numbers, raw data $\{\xi_a\}$ will not be revealed to the peers in the communication process.

At this point, each agent has the S input-output pairs $\{(1, \bar{g}(1)), \ldots, (S, \bar{g}(S))\}$ of the polynomial

$$\bar{g}(n) = \bar{\xi} + \bar{R}_1 n + \cdots + \bar{R}_{S-1} n^{S-1} \tag{21}$$

at hand, where $\bar{R}_i \triangleq \sum_{a=1}^S R_i^a$. Notice that we have $\bar{\xi}$ on the r.h.s. as the intercept. Since an $(S-1)$-th order polynomial is uniquely determined by distinctive S points, each agent can uniquely identify the functional form of $\bar{g}(n)$. With Lagrange's interpolation formula, the agents obtain the intercept by

$$\bar{\xi} = \bar{g}(0) = \sum_{l=1}^S \bar{g}(l) \prod_{m \neq l} \frac{m}{(m-l)}. \tag{22}$$

Shamir's algorithm is secure in the sense that it satisfies rigorous cryptographic conditions [35]. However, when S is large, the product form of Eq. (22) needs a special attention to avoid numerical issues. Also, the need for S repeated consensus makes it computationally less efficient, as will be empirically shown in Section VII.

Algorithm 1 Dynamical consensus with random chunking

```
Input: ϵ, N_C. Initialize ξ̄ = 0
Split ξ_a into N_C chunks for ∀a.
for i_C ← 1, N_C do
    Randomly generate A. Confirm all the links are available for communication (re-generate A otherwise).
    Initialize ξ_a(0) as ξ_a^[i_C] for ∀a.
    repeat
        Each agent synchronously perform (17).
    until convergence
    ξ̄ ← ξ̄ + ξ̄^[i_C]
end for
```

C. Random chunking

Shamir's algorithm is based on the idea of recovering the data from seemingly non-informative multiple signatures. Here it is interesting to see what happens if we use a datum itself as the signature. Specifically, each of the agents secretly splits a local datum ξ_s into a few chunks $\{\xi_s^{[h]}\}$ such that $\sum_h \xi_s^{[h]} = \xi_s$. Then, the agents run the dynamical consensus algorithm on each of the chunks to get $\{\bar{\xi}^{[1]}, \ldots, \bar{\xi}^{[N_\mathrm{C}]}\}$, where $N_\mathrm{C} \ll S$ is the predefined number of chunks. Since aggregation is an linear operation, we have

$$\bar{\xi}^{[1]} + \cdots + \bar{\xi}^{[N_\mathrm{C}]} = \sum_{s=1}^S (\xi_s^{[1]} + \cdots + \xi_s^{[N^c]}) = \bar{\xi}. \tag{23}$$

This means that the total aggregation can be computed by performing chunk-wise aggregations without sharing the raw data $\{\xi_s\}$ explicitly. The procedure is summarized in Algorithm 1.

Here, to prevent the agents from recovering the raw data by receiving all the N_C chunks, the chunk-wise aggregations must be made on different incidence matrices. There are two major solutions having different levels of intervention of the network infrastructure (see Section III-A). For the higher intervention side, upon starting aggregation, the router may randomly pick A from a set of prepopulated incidence matrices, in which any pair of the graphs do not share the same edge. If the router always picks an A that is different from the previous aggregation round, $N_\mathrm{C} = 2$ suffices.

If we cannot expect too much from the infrastructure, the security guarantee becomes probabilistic. This is reminiscent of Bitcoin's probabilistic security guarantee, which makes it the *probabilistic finality* protocol [36]. In the next section, we discuss the detail of probabilistic guarantee of the random chunking algorithm.

V. Privacy Breach Analysis

This section provides probabilistic guarantees of the random chunking algorithm under three major attack scenarios.

A. Independent agent scenario

As suggested in Algorithm 1, one practical scenario is for the router to randomly assign the node name whenever starting aggregation, keeping the graph structure itself fixed. To study the risk of privacy breach, consider the event that the j-th node is the breach node to s, meaning that the node j is able to fully recover ξ_s by summing over $N_{\rm C}$ chunks it received. This happens when the j-th node stays as the neighbor over the $N_{\rm C}$ rounds, regardless of the other nodes. Since the j-th node gets chosen in each of the $N_{\rm C}$ chunking round with a probability

$$\binom{S-2}{d_s-1}\binom{S-1}{d_s}^{-1} = \frac{d_s}{S-1}, \tag{24}$$

the probability that the j-th node is the breach node to s regardless of the other nodes is given by $\left(\frac{d_s}{S-1}\right)^{N_{\rm C}}$. Since some of the other $S-2$ nodes may be the breach node as well, we have

$$p^s_{\rm breach} \le \sum_{j\neq s}\left(\frac{d_s}{S-1}\right)^{N_{\rm C}} = (S-1)\left(\frac{d_s}{S-1}\right)^{N_{\rm C}} \tag{25}$$

by Boole's inequality (or the union bound). Therefore, we conclude that the probability of not having any breach node in the network is lower-bounded as

$$\begin{aligned} p_{\rm secure} &\ge \prod_{s=1}^{S}\left\{1-(S-1)\left(\frac{d_s}{S-1}\right)^{N_{\rm C}}\right\} \\ &\ge 1 - S(S-1)\left(\frac{d_{\max}}{S-1}\right)^{N_{\rm C}}, \end{aligned} \tag{26}$$

where the second inequality is due to Bernoulli's inequality. The second term of the bound can be made arbitrarily small for *sparse graphs* $d_{\max} \ll S$ by appropriately choosing the value of $N_{\rm C}$. To summarize, we have proved the following theorem:

Theorem 1. *The dynamical consensus algorithm with random chunking has a privacy guarantee of Eq.* (26)*. The probability of privacy breach can be made arbitrarily small.*

B. Colluded agent scenario

Another interesting question is whether the algorithm is secure under collusion. Although the risk of collusion is minimal in consortium-based networks, where the identity and the motivation of the agents are known to each other, we have the following guarantee:

Theorem 2. *Let $N_{\rm L}$ be the number of colluded agents. The probability that the privacy of a non-colluded node (indexed by s) is compromised due to collusion is given by*

$$p^s_{\rm c.breach} = \left\{1-\prod_{l=1}^{N_{\rm L}}\left(1-\frac{d_s}{S-l}\right)\right\}^{N_{\rm C}} \tag{27}$$

$$\le \exp\left\{-N_{\rm C}\left(1-\frac{d_s}{S-N_{\rm L}}\right)^{N_{\rm L}}\right\} \tag{28}$$

for $N_{\rm L} < S-d_s$, and 1 otherwise.

(*Proof*) For an s-th non-colluded node, privacy breach occurs when at least one colluded node receives the chunk in all the $N_{\rm C}$ chunking rounds. Therefore, we have

$$p^s_{\rm c.breach} = (1-p_{\rm L})^{N_{\rm C}}, \tag{29}$$

where $p_{\rm L}$ is the probability that no colluded nodes are chosen in the neighbor set. The probability $p_{\rm L}$ can be evaluated as

$$\begin{aligned} p_{\rm L} &= \binom{S-1-N_{\rm L}}{d_s}\binom{S-1}{d_s}^{-1} \\ &= \frac{(S-d_s-1)(S-d_s-2)\cdots(S-d_s-N_{\rm L})}{(S-1)(S-2)\cdots(S-N_{\rm L})}, \end{aligned} \tag{30}$$

which holds only for $N_{\rm L} \le S-d_s-1$. If there are so many colluded nodes that $N_{\rm L} \ge S-d_s$ holds, it is impossible not to choose a colluded nodes $N_{\rm C}$ times in a row and thus $p_{\rm L}$ must be zero. From Eqs. (29) and (30), we have the equality of Theorem 2:

$$p^s_{\rm c.breach} = \left\{1-\prod_{l=1}^{N_{\rm L}}\left(1-\frac{d_s}{S-l}\right)\right\}^{N_{\rm C}}. \tag{31}$$

Next, let us derive the upper bound. By replacing the product in Eq. (31) with the power of the minimum term and apply the Bernoulli inequality, we have

$$\begin{aligned} p^s_{\rm c.breach} &\le \left\{1-\left(1-\frac{d_s}{S-N_{\rm L}}\right)^{N_{\rm L}}\right\}^{N_{\rm C}} \\ &\le \exp\left\{-N_{\rm C}\left(1-\frac{d_s}{S-N_{\rm L}}\right)^{N_{\rm L}}\right\}., \end{aligned} \tag{32}$$

which completes the proof. □

Figure 2 (a) visualizes the privacy breach probability due to collusion (Eq. (28)) as a function of $N_{\rm C}$ and $N_{\rm L}$. As expected, the probability is almost one if $N_{\rm L} \sim \frac{S}{2}$ and $N_{\rm C} \sim 1$. However, we also see that the risk can be made negligible by properly choosing $N_{\rm C}$, as claimed in Theorem 2. Also, Fig. 2 (b) shows comparison among the exact probability and its upper bound computed by Eq. (32).

The inequality enables us to reasonably set the number of chunks, $N_{\rm C}$, so that the privacy of all non-colluded nodes is guaranteed with a sufficiently high probability, say, at least $1-\eta$ for a small η. This corresponds to $p^s_{\rm c.breach} \le \eta$ and is translated into

$$N_{\rm C} \ge |\ln\eta|\left(1-\frac{d_s}{S-N_{\rm L}}\right)^{-N_{\rm L}}. \tag{33}$$

As long as the graph is sparse ($d_s \ll S$) and $N_{\rm L}$ is expected to be much smaller than S, a useful rule-of-thumb will be $2|\ln\eta|$. In consortium-based networks, $N_{\rm L}$ should be much smaller than S. In that case, the use of graphs having $d_s \ll S$ is critical to guarantee data privacy under the risk of collusion. Again, this motivates us to consider a family of *sparse graphs*, as discussed in the next section.

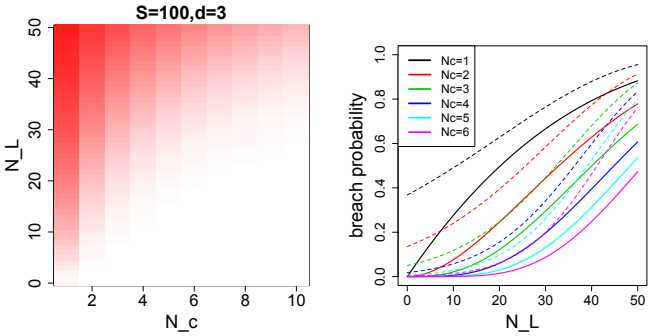

Fig. 2. Privacy breach probability due to collusion computed for 3-regular graph of $S = 100$. (a) Left panel: $p^s_{\text{c.breach}}$ computed by Eq. (27) with a gradation from red being 1 to white being 0. (b) Right panel: $p^s_{\text{c.breach}}$ (solid lines) and its upper bound (Eq. (32); dashed lines) at a few N_{C} values. Best viewed in color.

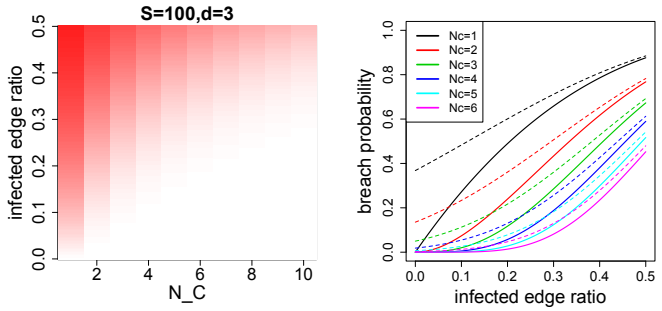

Fig. 3. Privacy breach probability due to eavesdropping computed for 3-regular graph of $S = 100$. (a) Left panel: $p^s_{\text{e.breach}}$ computed by Eq. (34) as a function of N_{C} and the infected edge ratio N_{E}/E with a gradation from red being 1 to white being 0. (b) Right panel: $p^s_{\text{e.breach}}$ (solid lines) and its upper bound (Eq. (35); dashed line) at a few values of N_{C}. Best viewed in color.

C. Eavesdropping scenario

Eavesdropping is one of the major attacks in communication network [37]. Although the risk of eavesdropping is limited in consortium-based networks, we can compute the probability of privacy breach in a similar way to the incidental privacy breach. The Shamir-based algorithm is safe even under eavesdropping and collusion because each agent sends only obfuscated values as Eq. (20) to the connected agents.

In the random chunking algorithm, however, we again have a probabilistic guarantee. Consider the scenario that an eavesdropper picks a set of the edges and captures all the contents of communications. Let N_{E} be the maximum number of edges the eavesdropper can tap. Let $p^s_{\text{e.breach}}$ be the probability that the s-th agent incurs a privacy breach under eavesdropping. A breach occurs when the eavesdropper receives all of the N_{C} chunks. This happens when the eavesdropper successfully tapped any of the d_s edges from the s-th agent in every chunking round. Thus

$$\begin{aligned} p^s_{\text{e.breach}} =& [1 - \Pr(\text{No edges from } s \text{ are eavesdropped})]^{N_{\text{C}}}, \\ =& [1 - \Pr(\text{All the } N_{\text{E}} \text{ edges are chosen} \\ & \qquad \text{from the other } E - d_s \text{ edges})]^{N_{\text{C}}}, \\ =& \left\{1 - \binom{E - d_s}{N_{\text{E}}}\binom{E}{N_{\text{E}}}^{-1}\right\}^{N_{\text{C}}} \\ =& \left\{1 - \prod_{l=0}^{d_s-1}\left(1 - \frac{N_{\text{E}}}{E-l}\right)\right\}^{N_{\text{C}}} \end{aligned} \tag{34}$$

where $\Pr(\cdot)$ denotes the probability of the event specified by the argument, and $E \triangleq \sum_{i=1}^{S} d^i$ is the total number of edges of the graph. Using the Bernoulli's inequality as in Eq. (32), we have

$$\begin{aligned} p^s_{\text{e.breach}} &\le \left\{1 - \left(1 - \frac{N_{\text{E}}}{E - d_s + 1}\right)^{d_s}\right\}^{N_{\text{C}}} \\ &\le \exp\left\{-N_{\text{C}}\left(1 - \frac{N_{\text{E}}}{E - d_s + 1}\right)^{d_s}\right\}. \end{aligned} \tag{35}$$

We summarize this result as bellow:

Theorem 3. *The probability of privacy breach due to eavesdropping in the dynamic consensus algorithm with random chunking is upper-bounded as Eq.* (35)*.*

Note that for d-regular graphs, $E = dS$. Since $S \gg d_s$ for sparse graphs, as long as the infected edge ratio $\frac{N_{\text{E}}}{E}$ is expected to be much smaller than one, we can always make $p^s_{\text{e.breach}}$ negligible by choosing a sufficiently large N_{C}. To keep $p^s_{\text{e.breach}}$ less than $\eta > 0$, we need

$$N_{\text{C}} \ge |\ln \eta| \left(1 - \frac{N_{\text{E}}}{E - d_s + 1}\right)^{-d_s}. \tag{36}$$

For sparse graphs, unless a majority of edges are tapped, $N_{\text{C}} \sim 2|\ln \eta|$ again is a useful rule-of-thumb.

Figure 3 illustrates $p^s_{\text{e.breach}}$ as a function of N_{E}/E and N_{C} for a 3-regular graph with $S = 100$. As shown, even under massive eavesdropping where 20% of the edges are tapped, $N_{\text{C}} = 6$ gives only about 1% of breach risk.

In addition to the above case, we may consider the case where an eavesdropper hijacks an agent and capture all the outgoing communications. In this case, the random chunking algorithm is no longer secure although the Shamir-based method is still safe. This issue may be handled by combining with a more sophisticated secret sharing scheme such as in [38]. Exploring this topic would be an interesting future topic.

VI. Network Topology Analysis

In the previous section, we pointed out that the graph sparsity plays an important role in privacy preservation. This section discusses another important aspect of the network topology: *Spectral* structure. Specifically, we discuss properties of a specific type of graph called the *expander graph*.

A. Estimating the number of iterations to converge

In the dynamic consensus algorithm discussed in Section IV-A, the convergence speed is governed by the ratio between the first and second absolute largest eigenvalues. Let

$\lambda_1 = 1, \lambda_2, \ldots, \lambda_S$ be the eigenvalues of W_ϵ arranged in the decreasing order. For ease of exposition, let us assume that ϵ has been chosen so λ_2 is the absolute second largest as well, which is always possible.

We are interested in the scalability of the algorithm in terms of the network size S. Let us first think about how the convergence rate is evaluated. Equation (19) shows that

$$\|\sqrt{S}{\mathsf{W}_\epsilon}^t \boldsymbol{\xi}(0)\|_2 \to \left\| \frac{1}{\sqrt{S}} \mathbf{1}_S \bar{\xi} \right\|_2 = |\bar{\xi}|$$

as $t \to \infty$, where $\|\cdot\|_2$ denotes the ℓ_2 norm. Thus it makes sense to define the relative error at iteration round t as

$$e_\epsilon(t) \triangleq \frac{1}{|\bar{\xi}|} \sqrt{\|\sqrt{S}{\mathsf{W}_\epsilon}^t \boldsymbol{\xi}(0)\|_2^2 - \bar{\xi}^2} \tag{37}$$

Let $\boldsymbol{v}_2$ be the ℓ_2-normalized eigenvector corresponding to λ_2. As t grows to infinity, we asymptotically have:

$$e_\epsilon(t) \to \sqrt{S}(\lambda_2)^t \times \frac{\boldsymbol{v}_2^\top \boldsymbol{\xi}(0)}{\bar{\xi}} \sim \sqrt{S}(\lambda_2)^t \times \mathrm{O}(1). \tag{38}$$

Thus, we conclude that $\sqrt{S}(\lambda_2)^t$ is a reasonable nondimensional metric for convergence. To achieve a relative error δ, we need a number of iterations as $t \sim \mathrm{O}\left(\frac{\ln(\sqrt{S}/\delta)}{|\ln \lambda_2|}\right)$. Let us summarize this result in Theorem 4:

Theorem 4. *In the dynamical consensus algorithm, the number of iterations t to achieve a relative error δ is given by*

$$t \sim \mathrm{O}\left(\frac{\ln(\sqrt{S}/\delta)}{|\ln(1-\Delta_\lambda)|}\right), \tag{39}$$

where $\Delta_\lambda \triangleq \lambda_1 - \lambda_2$.

In Eq. (39), we have used the fact that $\lambda_1 = 1$ and thus $\lambda_2 = 1 - \Delta_\lambda$. In our previous work [9], we showed that the cycle graph scales quadratically as $t \sim S^2 \ln S$. The question is whether we can improve this (relatively poor) scalability by using a different communication graph. The next subsection answers this question.

B. Expander graph

Intuitively, we expect that the more information the agents circulate, the faster convergence we will get. The complete graph is clearly an extreme case, where the agents share information most generously with the peers and thus data privacy is least protected. Let $\mu_1, \ldots, \mu_S$ be the eigenvalues of A in the decreasing order. In the complete graph, it is easy to verify that $\mu_1 = S - 1$ and $\mu_2 = \cdots = \mu_S = -1$. Hence, we have $\Delta_\lambda = \epsilon(\mu_1 - \mu_2) = \epsilon S$. The complete graph is a d-regular graph with $d = S - 1$. By Theorem 4, with a choice of $\epsilon \sim \mathrm{O}(1/d)$, the convergence speed measured by t scales as $\ln S$, which is much better than the quadratic scalability of the cycle graph.

Is there any *sparse* graph that behaves like the complete graph when communicating with the other agents? This may sound like a ridiculous question, but surprisingly, there exists a class of sparse graph called the *expander graph* that has the same logarithmic convergence. Formally, the expander graph is defined as a graph whose *expansion constant* is lower-bounded, where

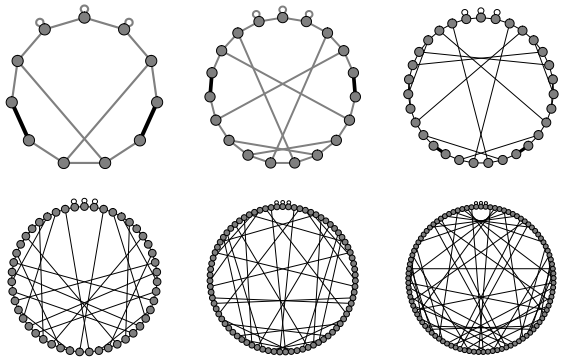

Fig. 4. Example of 3-regular expander graphs with $S = 11, 19, 31$ (top row) and $47, 79, 97$ (bottom row) from left to right. Note that self-loops and double edges (thick lines) exist.

$$\text{(expansion constant)} \triangleq \inf_{\mathcal{V}_1} \left\{ \frac{|\partial \mathcal{V}_1|}{\min\{|\mathcal{V}_1|, S - |\mathcal{V}_1|\}} \right\} \tag{40}$$

with $\mathcal{V}_1$ being an arbitrary subset of the graph nodes and $|\partial \mathcal{V}_1|$ is the number of outgoing edges from the subgraph [39]. Intuitively, the more the expansion constant is, the more "talkative" the nodes tend to be. Very interestingly, this purely geometric definition has a hidden connection to the graph spectrum, and, in our case, we have

$$\Delta_\lambda \geq \epsilon \frac{\alpha^2}{2d} \quad (d\text{-regular expander graphs}) \tag{41}$$

by Cheeger's inequality [40], where α is the lower bound of the expansion constant. Thus we again have the logarithmic scalability $t \sim \ln S$ with a choice of $\epsilon \sim \mathrm{O}(1/d)$. We now summarize the result as follows.

Theorem 5. *On d-regular expander graphs, $t \sim \mathrm{O}(\ln(\sqrt{S}/\delta))$ with a choice of $\epsilon \sim \mathrm{O}(1/d)$.*

Construction of expander graphs is nontrivial. Fortunately, there are two ways available to obtain expander graphs for our purpose. The first one is a 3-regular expander graph called the cycle with inverse chords [13]. As the name suggests, it starts with the cycle graph (2-regular graph), and connects each node s to the nodes $j = s \pm 1, (s-1)(j-1) = 1 \mod S$ for a prime S. See Figs. 1 and 4 for a few examples. As mentioned before, the node indices need to be randomly shuffled prior to each round of the random chunking algorithm. Since the eigenspectrum is invariant to re-labeling the nodes, it does not affect the convergence of dynamic consensus.

The other possible approach is to use a random graph. It is known that uniformly sampled d-regular random graphs approximate Δ_λ of the expander graph Eq. (41) almost always perfectly [41]. To construct such a graph, we can leverage the configuration model in random graph theory [42]. Specifically, each agent simply picks d neighbors randomly and uniformly to form a d-regular graph. This approach is preferable in

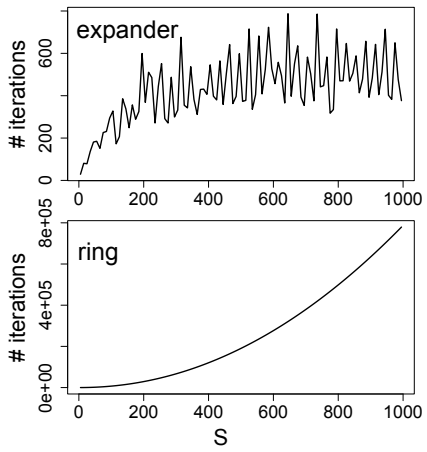

Fig. 5. Comparison of the number of iterations t for $\delta = 10^{-3}$.

the sense that it does not require any intervention of the network router and does not assume network connection that is stable throughout the entire consensus process. On the other hand, however, the randomness in the network structure may lead to some unpredictability in the eigenspectrum, which is in contrast to the first option. To avoid extra randomness, we use the cycle with inverse chords in the empirical study in the next section. Evaluating and designing random graph construction algorithms in dynamic consensus is an interesting future research topic.

VII. Experiments

This section reports on experimental results of the proposed model. All the experiments were conducted locally on a laptop PC with a Core i7 Processor and 32 GB memory.

A. Number of iterations to converge

Figure 5 compares between the expander graph (cycle with a random chord) and the 2-regular cycle graph (or the ring) on the number of iterations t to achieve $\delta = 10^{-3}$ as defined in Theorem 4 and 5. It is interesting to observe that just adding an extra edge to the ring by the rule $(s-1)(j-1) = 1 \mod S$ drastically changes the convergence behavior. As mentioned before, the ring scales quadratically as $t \sim S^2 \ln S$. In contrast, t grows very slowly in the expander graph, being consistent to the logarithmic scalability predicted by Theorem 5.

The original construction of the 3-regular expander graph is for S that is a prime. One interesting question in practice is that the excellent convergence behavior is maintained for non-prime S's. We extended the original construction by giving a self-loop whenever the equation $(s-1)(j-1) = 1 \mod S$ does not have a solution, so that the graph is always 3-regular. Fortunately, apart from the visible fluctuations, the figure suggests that the overall convergence behaviors are robust and we should be able to safely use the model even for non-primes.

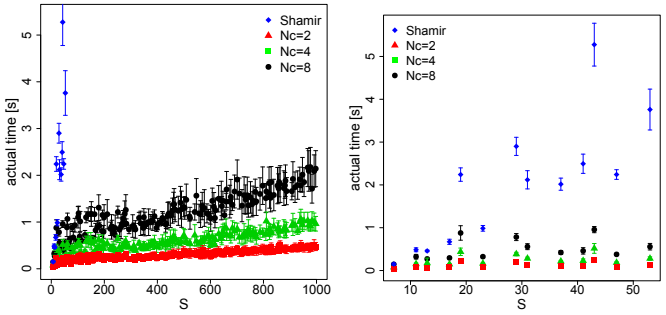

Fig. 6. Actual computation time for aggregation on the 3-regular expander graph. The right panel covers the range of $7 \leq S \leq 53$.

B. Shamir vs. random chunking

Since one iteration needs one matrix-vector multiplication (see Eq. (18)), actual computational time may have a different scalability from that of the number of iterations to converge. Specifically, one may expect an extra $S\bar{d}$ factor, where $\bar{d}$ is the mean degree.

Figure 6 compares actual computation times in different N_{C}s. We randomly initialized $\xi_s(0)$ with the uniform distribution in $[-1, 2]$. Convergence was declared when the relative error gets smaller than 10^{-5}. The mean and standard deviation (s.d.) were computed by repeating computation by 10 times with a different random seed. The figure confirms the expected *linear* dependency in the random chunking algorithm.

Figure 6 also compares the Shamir's secret sharing algorithm with the random chunking algorithm. Since the Shamir-based method needs to repeat the process of global consensus S times, the actual computation time is expected to depend quadratically on S, which is consistent with the figure. Apart from $S \lesssim 10$, where the both methods are comparable, the Shamir-based method tended to have a larger variability than the random chunking algorithm, which might suggest numerical issues in the implementation.

Finally, we comment on the origin of non-smoothness of the computational time in Fig. 6. In the figure, close inspection shows that the computational times are correlated: whenever the Shamir-based method gets much time, so does the random chunking method. This suggests that the source of the non-smoothness is in the spectral structure of the underlying network because the same dynamical consensus algorithm was shared by both. Although we have obtained a very solid result on the convergence behavior, how the spectral gap Δ_λ scales as S increases is not straightforwardly predictable. Further investigation on this point is left to future work.

C. Shamir vs. homomorphic encryption

Table I compares computation time for a few small S's between the proposed Shamir-based method and homomorphic encryption (HE)-based method of [28]. Both approaches have cryptographic security and share the same dynamical consensus algorithm. For a fair comparison, we ran both on the same expander graph and used the same experimental design as above. For HE, we used an implementation of the

TABLE I
COMPARISON OF ACTUAL COMPUTATION TIMES IN AGGREGATION [SEC].

S	Shamir		HE	
	mean	s.d.	mean	s.d.
7	0.159	0.024	155	18.8
11	0.485	0.045	288	44.1
13	0.458	0.020	375	24.8
17	0.671	0.060	574	55.1
19	2.24	0.156	699	47.0

Paillier cryptosystem [43], which encrypts and decrypts every communication between the agents with a new key. From the table, and in the light of Fig. 6, we conclude that the proposed random chunking-based method is several orders of magnitude faster than the HE-based alternative.

D. Remarks on synchronization issues

In Section VII-B, we mentioned that actual computational time of the random chunking algorithm on the expander graph should have a linear dependency on S. This statement implicitly assumes that computation is made sequentially. Although in theory it is true that the agents can perform updates in parallel, sequential execution should be a reasonable assumption for evaluating computational time. In real distributed environments, we always need to handle the issue of synchronization across the network. If the cost of local computation is negligible, network delays will almost always dominate the time required to move on to the next iteration. In that case, most of the agents would spend most of the time waiting for the last message to be delivered to one of the agents.

On the other hand, if the computational cost is on average much higher than network delays, we will need to consider parallelization as a realistic option. This is actually the case in homomorphic encryption (HE)-based secure computation, as recently pointed out in [27]. There may also be some room for improvement in the HE implementation itself, where we employed `homomorpheR` [43] with the key size of $1\,024$ to follow the protocol proposed in [28], [44], although handling signed floating-point numbers can be a subtle issue (see, e.g. [45]). The key exchange protocol can be simplified, too. Regarding Table I, it would be an interesting future research topic to compare the random chunking method with a highly optimized HE implementation in a realistic setting.

VIII. CONCLUDING REMARKS

We have presented new research directions of Blockchain as a collaborative value co-creation platform rather than a mere immutable data storage. Our platform is designed to respect the values of democracy, diversity, and privacy in its collaborative learning process. As such an instance, we have proposed a multi-task federated learning framework combined with a global consensus-building algorithm.

We discussed two topics in the global consensus-building process. The first topic was data privacy. We proposed two secure consensus-building approaches built upon the idea of secret sharing: The Shamir-based dynamic consensus, which has a cryptographic security guarantee, and the random chunking algorithm, which falls into the category of the probabilistic finality protocol. For the latter, we have provided the upper bound of privacy breach under three different attack scenarios. The second topic discussed was the issue of network design in global consensus. Based on the profound results in spectral graph theory, we showed that expander graphs dramatically improve the scalability of the algorithm.

We conclude this paper by summarizing a few future research topics:

a) Learning under network errors: We have assumed perfectly synchronized and stable communication among the agents. As suggested in Sections VI-B and VII-D, extensions to include network errors is of primary importance to make our platform truly useful.

b) Meta-agreement issues: Another potential issue in practice is how to agree on the learning task itself (choice of the algorithm, data dimensionality M, the definition of each dimension, etc.) and how to initiate peer-to-peer communication (who to communicate with).

c) External data privacy: One interesting business scenario is to sell the learned model to external parties. This calls for a guarantee that is different from the *internal* data privacy among the agents. It is known that the classical concept of differential privacy has issues in evaluating noisy real-valued variables [46]. Establishing a method of evaluating the degree of information leak is important.

d) Randomness in graph spectra: As discussed in Section VI-B, the origin of the non-smoothness shown in Fig. 6 and the consensus algorithm based on random graphs are still an open question.

e) Security analysis: The random chunking algorithm combined with the dynamic consensus algorithm appears to have more flexibility than traditional cryptographic methods. We need to study further the pros and cons of those methods.

f) Use-cases: Finally, we need to develop practical use-cases where the decentralized architecture is truly useful. The lightweight probabilistic privacy guarantee seems suitable in internet-of-things (IoT) applications [29], but more study is needed. We hope that, just as Nakamoto's Bitcoin changed the landscape of financial transaction management, our decentralized collaborative learning platform as the next-generation Blockchain has the potential to change the way of doing business.

ACKNOWLEDGEMENT

T.I. thank Dr. Sachiko Yoshihama for her insightful suggestions. T.I. is partially supported by the Department of Energy National Energy Technology Laboratory under Award Number DE-OE0000911. A part of this report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents

that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

REFERENCES

[1] D. Tse, B. Zhang, Y. Yang, C. Cheng, and H. Mu, "Blockchain application in food supply information security," in *Proceeding of 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM 2017)*. IEEE, 2017, pp. 1357–1361.

[2] Q. Lu and X. Xu, "Adaptable blockchain-based systems: a case study for product traceability," *IEEE Software*, vol. 34, no. 6, pp. 21–27, 2017.

[3] K. Toyoda, P. T. Mathiopoulos, I. Sasase, and T. Ohtsuki, "A novel blockchain-based product ownership management system (poms) for anti-counterfeits in the post supply chain," *IEEE Access*, vol. 5, pp. 17 465–17 477, 2017.

[4] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[5] O. Scekic, S. Nastic, and S. Dustdar, "Blockchain-supported smart city platform for social value co-creation and exchange," *IEEE Internet Computing*, vol. 23, no. 1, pp. 19–28, 2019.

[6] S. Seebacher and R. Schüritz, "Blockchain technology as an enabler of service systems: A structured literature review," in *International Conference on Exploring Services Science*, 2017, pp. 12–23.

[7] G. Kondrateva, E. de Boissieu, C. Ammi, and E. Seulliet, "The potential use of blockchain technology in co-creation ecosystems," *Journal of Innovation Economics Management*, pp. I104–27, 2021.

[8] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Preprint. https://bitcoin.org/bitcoin.pdf*, 2008.

[9] T. Idé, R. Raymond, and D. T. Phan, "Efficient protocol for collaborative dictionary learning in decentralized networks." in *Proceeding of the 28th International Joint Conference on Artificial Intelligence (IJCAI-19)*, 2019, pp. 2585–2591.

[10] C. W. Wu, "Agreement and consensus problems in groups of autonomous agents with linear dynamics," in *Proceedings of the 2005 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2005, pp. 292–295.

[11] W. Ren, R. W. Beard, and E. M. Atkins, "A survey of consensus problems in multi-agent coordination," in *Proceedings of the 2005 American Control Conference*. IEEE, 2005, pp. 1859–1864.

[12] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[13] S. P. Vadhan *et al.*, "Pseudorandomness," *Foundations and Trends in Theoretical Computer Science*, vol. 7, no. 1–3, pp. 1–336, 2012.

[14] A. Lubotzky, *Discrete groups, expanding graphs and invariant measures*. Springer Science & Business Media, 2010.

[15] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 19–38.

[16] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017, pp. 1273–1282.

[17] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[18] N. Agarwal, A. T. Suresh, F. Yu, S. Kumar, and H. B. McMahan, "cpSGD: Communication-efficient and differentially-private distributed SGD," in *Advances in Neural Information Processing Systems*, 2018, pp. 7575–7586.

[19] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, p. 12, 2019.

[20] M. Xu, B. Lakshminarayanan, Y. W. Teh, J. Zhu, and B. Zhang, "Distributed Bayesian posterior sampling via moment sharing," in *Advances in Neural Information Processing Systems*, 2014, pp. 3356–3364.

[21] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proceedings of the 36th International Conference on Machine Learning (ICML 2019)*, 2019, pp. 4615–4625.

[22] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4424–4434.

[23] G. Bernstein and D. R. Sheldon, "Differentially private Bayesian inference for exponential families," in *Advances in Neural Information Processing Systems*, 2018, pp. 2924–2934.

[24] L. Xie, I. M. Baytas, K. Lin, and J. Zhou, "Privacy-preserving distributed multi-task learning with asynchronous updates," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 1195–1204.

[25] M. Heikkilä, E. Lagerspetz, S. Kaski, K. Shimizu, S. Tarkoma, and A. Honkela, "Differentially private Bayesian learning on distributed data," in *Advances in neural information processing systems*, 2017, pp. 3226–3235.

[26] B. Ding, H. Nori, P. Li, and J. Allen, "Comparing population means under local differential privacy: with significance and power," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, 2018.

[27] Y. Liu, Y. Kang, C. Xing, T. Chen, and Q. Yang, "A secure federated transfer learning framework," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 70–82, 2020.

[28] M. Ruan, M. Ahmad, and Y. Wang, "Secure and privacy-preserving average consensus," in *Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and Privacy*. ACM, 2017, pp. 123–129.

[29] T. Idé, "Collaborative anomaly detection on blockchain from noisy sensor data," in *Proceeding of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2018, pp. 120–127.

[30] E. L. Lehmann and G. Casella, *Theory of point estimation*. Springer Science & Business Media, 2006.

[31] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[32] C.-J. Hsieh, M. A. Sustik, I. S. Dhillon, and P. Ravikumar, "QUIC: quadratic approximation for sparse inverse covariance estimation," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2911–2947, 2014.

[33] G. Strang, *Linear Algebra and its Applications*. Academic Press, 1976.

[34] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[35] D. Boneh and V. Shoup, "A graduate course in applied cryptography," *Draft of a book, version 0.3, December*, 2016.

[36] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 1432–1465, 2020.

[37] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," in *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, 2003.* IEEE, 2003, pp. 113–127.

[38] N. Gupta, J. Katz, and N. Chopra, "Privacy in distributed average consensus," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 9515–9520, 2017.

[39] G. Davidoff, P. Sarnak, and A. Valette, *Elementary number theory, group theory and Ramanujan graphs*. Cambridge University Press, 2003.

[40] A. E. Brouwer and W. H. Haemers, *Spectra of graphs*. Springer Science & Business Media, 2011.

[41] J. Friedman, "A proof of Alon's second eigenvalue conjecture," in *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing (STOC '03)*, 2003, p. 720–724.

[42] J. Kim and V. Vu, "Generating random regular graphs," *Combinatorica*, vol. 26, no. 6, pp. 683–708, 2006.

[43] B. Narasimhan, "homomorpheR," in *CRAN (The Comprehensive R Archive Network)*, 2019.

[44] M. Ruan, H. Gao, and Y. Wang, "Secure and privacy-preserving consensus," *IEEE Transactions on Automatic Control*, 2019.

[45] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. the 2017 Intl. Conf. the Theory and Application of Cryptology and Information Security (ASIACRYPT 2017)*. Springer, 2017, pp. 409–437.

[46] K. Minami, H. Arai, I. Sato, and H. Nakagawa, "Differential privacy without sensitivity," in *Advances in Neural Information Processing Systems*, 2016, pp. 956–964.