Private and Hot-Pluggable Distributed Averaging

Israel L. Donato Ridgley, Randy A. Freeman, *Member, IEEE*, and Kevin M. Lynch, *Fellow, IEEE*

***Abstract*—Some distributed optimization applications require privacy, meaning that the values of certain parameters local to a node should not be revealed to other nodes in the network during the joint optimization process. A special case is the problem of private distributed averaging, in which a network of nodes computes the global average of individual node reference values in a distributed manner while preserving the privacy of each reference. We present simple iterative methods that guarantee accuracy (i.e., the exact asymptotic computation of the global average) and privacy (i.e., no node can estimate another node's reference value). To achieve this, we require that the digraph modeling the communication between nodes satisfy certain topological conditions. Our method is hot-pluggable (meaning no reinitialization of the averaging process is required when the network changes or a node enters or leaves, when there is a communication or computation fault, or when a node's reference value changes); it does not require an initial scrambling phase; it does not inject noise or other masking signals into the distributed computation; it does not require random switching of edge weights; and it does not rely on homomorphic encryption.**

***Index Terms*—Optimization algorithms, distributed control, network analysis and control.**

I. Introduction

PRIVACY in distributed computation is becoming increasingly important as more systems become decentralized. Within systems such as the smart grid, smart transportation, and smart healthcare, there is an obvious need to protect user information from being revealed to other agents. In other systems such as sensor networks deployed in sensitive environments, there is a need to keep the information on each node private from other (potentially compromised) nodes. The particular example of distributed averaging studied in this letter is a basic component of many distributed computations, and the problem of maintaining privacy during distributed averaging has received significant recent attention [1]–[14].

In this letter, we consider networks of *honest-but-curious* nodes [6], [12] that cooperatively compute the global average of their reference values. This means all nodes in the network participate faithfully in the averaging algorithm but would like to reconstruct the reference values of other nodes. We allow the possibility of collusion, i.e., nodes sharing data not specified by the algorithm, possibly over side channels.

One popular approach to private distributed averaging is based on the concept of *differential privacy* from the database literature (see [4], [9], [15] and the references therein). These methods use added noise to obfuscate an individual's contribution to the computation and offer the potential to ensure some level of privacy against any amount of collusion [15]; however, they also involve a trade-off between accuracy and privacy [4]. Thus the differential privacy approach might not be suitable for applications which require high accuracy.

Another natural approach to privacy employs homomorphic encryption, wherein computations are performed on encrypted data without the need for decryption. These methods add computation and communication overhead that may not be acceptable for low-power systems [5], and they are difficult to extend to more general distributed optimization problems due to the complexity of homomorphic operations beyond addition (see [16] and the references therein).

Other approaches to distributed averaging that achieve both accuracy and privacy put topological restrictions on the digraph modeling the communication between nodes. Ours is one such approach, but our topological restrictions are significantly more relaxed than those appearing in previous work. For example, if each node has at least two out-neighbors and at least one of these is not colluding with other nodes, then our method guarantees privacy. A similar condition is used in [12] but for the more restrictive class of undirected (i.e., symmetric) graphs. To achieve privacy under our relaxed topological restrictions, we improve upon the state expansion method in [12] by using the concept of the *line digraph* of a given digraph [17]. Unlike the state expansion in [12], our line digraph expansion does not result in a slower exponential convergence rate of the iterates (relative to the less private unexpanded method).

Manuscript received March 2, 2020; revised May 5, 2020; accepted May 10, 2020. Date of publication May 22, 2020; date of current version June 11, 2020. Recommended by Senior Editor F. Dabbene. *(Corresponding author: Israel L. Donato Ridgley.)*

Israel L. Donato Ridgley is with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208 USA (e-mail: israelridgley2023@u.northwestern.edu).

Randy A. Freeman is with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208 USA, also with the Northwestern Institute on Complex Systems, Northwestern University, Evanston, IL 60208 USA, and also with the Center for Robotics and Biosystems, Northwestern University, Evanston, IL 60208 USA (e-mail: freeman@ece.northwestern.edu).

Kevin M. Lynch is with the Department of Mechanical Engineering, Northwestern University, Evanston, IL 60208 USA, also with the Northwestern Institute on Complex Systems, Northwestern University, Evanston, IL 60208 USA, and also with the Center for Robotics and Biosystems, Northwestern University, Evanston, IL 60208 USA (e-mail: kmlynch@northwestern.edu).

Digital Object Identifier 10.1109/LCSYS.2020.2996957

Many recent results are based on the standard averaging protocol introduced in [18]. This standard protocol is not private, as it requires nodes to communicate their reference values to their neighbors at the initial update step. Several methods have been proposed to modify this standard protocol so that it preserves privacy by employing an initial phase that scrambles the initial states of the nodes [6], [7], [11]. The related method in [12] involves the scrambling of initial edge weights rather than initial states. Other approaches employ masking, that is, nodes transmit carefully chosen masked or noisy versions of their states such that privacy is preserved and the effect of the mask on the average vanishes in the limit [1]–[3], [8], [10], [13]. None of these methods are *hot-pluggable*: if the network changes or nodes enter or leave, if nodes' reference values change at some point in time, or if there is some fault in communication or computation, then these algorithms must start the computation over again from the beginning (and must somehow coordinate to do so).

In contrast to all of this previous work, our method is hot-pluggable, it does not require an initial scrambling phase, it does not inject noise or other masking signals into the computation, it does not require random switching of the edge weights, and it does not use homomorphic encryption. It is based on a simple method for dynamic average consensus described in [19], and it builds on our previous work in [14]. For additional details about the base algorithm (including its implementation, performance, capabilities, and extensions), we refer the reader to [14], [19].

This letter uses the same base algorithm as in [14], but simplifies the expanded algorithm (which provides enhanced privacy guarantees) and improves its performance over the version in [14]. Namely, by avoiding the internal edges used in [14] and by privately initializing the references of the virtual nodes, we are able to maintain privacy while achieving the same convergence rate as in the base algorithm. In contrast, the expanded algorithm in [14] has a strictly worse convergence rate than the base algorithm. Also, we eliminate the average degree computation, which speeds up the expanded algorithm and reduces the amount of communication needed.

II. Preliminaries and Main Results

A. Notation and Terminology

Given $v \in \mathbb{R}^n$, we let $\mathrm{diag}(v)$ denote the $n \times n$ diagonal matrix whose diagonal is v. We let e_i denote the i^{th} column of the identity matrix I (with size determined by context).

We model a network of n agents participating in distributed computation as a digraph $G = (V, E)$, where $V = \{1, \ldots, n\}$ is the ordered set of n nodes (or vertices) and $E = \{E_1, \ldots, E_m\}$ is the ordered set of m directed edges (with each such edge being an ordered pair of nodes). There exists an edge $(i, j) \in E$ (with tail i and head j) if and only if node i can send information to node j, so that the edge direction corresponds to the communication direction. The digraph has no self-loops: even though a node can communicate with itself, this internal communication is not modeled by the digraph. The sets $\mathcal{N}_i^{\text{in}}, \mathcal{N}_i^{\text{out}} \subset V$ denote the sets of in- and out-neighbors of node i (respectively). The head and tail matrices H and T for G are the $n \times m$ matrices whose entries in row i and column α are given by

$$H_{i\alpha} = \begin{cases} 1 & \text{when node } i \text{ is the head of edge } \alpha \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

$$T_{i\alpha} = \begin{cases} 1 & \text{when node } i \text{ is the tail of edge } \alpha \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Note that $H^{\mathsf{T}}\mathbb{1} = T^{\mathsf{T}}\mathbb{1} = \mathbb{1}$, where $\mathbb{1}$ denotes the vector of all ones (with size determined by context), and that TH^{T} is the unweighted adjacency matrix of G. The unweighted out-degree of node i is $\delta_i^{\text{out}} = |\mathcal{N}_i^{\text{out}}|$, and the unweighted out-degree matrix is $\Delta = \mathrm{diag}(T\mathbb{1}) = \mathrm{diag}\{\delta_1^{\text{out}}, \ldots, \delta_n^{\text{out}}\}$.

We let a_{ij} denote the weight on edge (i, j), with $a_{ij} > 0$ if $(i, j) \in E$ and $a_{ij} = 0$ otherwise. The weighted out-degree of node i is $d_i^{\text{out}} = \sum_{j=1}^n a_{ij}$, and the weighted out-degree matrix is $D = \mathrm{diag}\{d_1^{\text{out}}, \ldots, d_n^{\text{out}}\}$. The weighted out-Laplacian is

$$L = D - [a_{ij}]_{n\times n} \tag{3}$$

where $[a_{ij}]_{n\times n}$ is the $n \times n$ weighted adjacency matrix. Note that L has zero row sums, i.e., $L\mathbb{1} = 0$. We use the edge order to create a weight vector $w \in \mathbb{R}^m$ such that w_k is the weight on edge E_k. If we define $W = \mathrm{diag}(w)$ then the weighted adjacency matrix for G is TWH^{T}, and we can rewrite (3) as $L = \mathrm{diag}(Tw) - TWH^{\mathsf{T}}$.

B. Assumptions and Results

We suppose each node i in the digraph G has a constant vector reference $r_i \in \mathbb{R}^p$, and we let $r_{\text{ave}} = \frac{1}{n}\sum_{i=1}^n r_i$ denote the global average.[1] Each node is honest-but-curious [6], [12], as defined in Section I; further, each node is either a *conformist* or a *colluder*. Conformists wish to preserve the privacy of their reference values and thus they only send and receive data as specified by the averaging algorithm. Colluders do not care about the privacy of their reference values and will share all information they have with other colluders (possibly over side channels not modeled by G) in order to infer the reference values of others. Our main assumptions are as follows:

(**A1**) G is a strongly connected digraph, which has at least three nodes and at least two of these are conformists.

(**A2**) Each node can send individual messages to each of its out-neighbors without the other out-neighbors hearing (in particular, each node i knows its out-neighbor set $\mathcal{N}_i^{\text{out}}$).

(**A3**) For each conformist i, either
- a) i has at least two out-neighbors, and at least one of these is a conformist, or
- b) for all $j \neq i$ there exists $k \in \{i\} \cup \mathcal{N}_i^{\text{in}}$ such that:
  - i) if j is a conformist then $k \notin \{j\} \cup \mathcal{N}_j^{\text{in}}$, and
  - ii) if j is a colluder then $k \notin \{\ell\} \cup \mathcal{N}_\ell^{\text{in}}$ $\forall$ colluders ℓ.

(**A4**) Nodes have no prior information about the reference values of other nodes.

Without assumption (**A1**) it is either trivial or impossible to achieve both privacy and accuracy. Assumptions (**A3**)(a)

[1]Like many of the algorithms in [19], our algorithms are also able to approximately track averages of slowly-varying references.

and (**A3**)(b) describe two different local topological restrictions on the digraph G, but only one of these restrictions needs to be satisfied for a given conformist i (there is an "or" between these conditions, not an "and"). Also, some conformists might satisfy (**A3**)(a) while others satisfy (**A3**)(b), i.e., the choice between these two options need not be uniform. Note that in [1], [3], [13] every conformist must satisfy (**A3**)(b) (so that (**A3**)(a) is not an option); thus our topological restriction (**A3**) is considerably weaker. Furthermore, (**A3**)(a) can be verified locally if a given node trusts that at least one of its out-neighbors is not a colluder. An example of a digraph G satisfying (**A3**)(a) for each conformist is any complete digraph with $n \geq 3$ having at least two conformists. An example of a digraph G satisfying (**A3**)(b) for each conformist is any directed cycle graph with $n \geq 3$ for which every conformist is either in- or out-adjacent to another conformist.

In the next sections we present distributed, synchronous, discrete-time algorithms to be run on each node such that under assumptions (**A1**)–(**A4**) we achieve the following:

Accuracy: each node's output signal converges exponentially to the exact global average r_{ave}.

Privacy: if i is a conformist and j is some other node, then for every vector $\rho \in \mathbb{R}^p$ there exist two different overall algorithm trajectories that produce the same data at node j but for which the associated reference values r_i differ by ρ.

We present two different algorithms, a simple *base algorithm* that only guarantees privacy when each conformist satisfies (**A3**)(b) and a slightly more complex *expanded algorithm* that requires more communication between nodes but that allows both options (**A3**)(a) and (**A3**)(b).

III. Distributed Averaging With Privacy

A. The Base Algorithm

Each node i runs a local discrete-time system with input u_i, output y_i, and internal state x_i, all taking values in $\mathbb{R}^{p+1}$. These will all be functions of the discrete-time value $t = 0, 1, 2, \ldots$. At each time t, node i sends the vector $a_{ij}x_i[t]$ individually to each of its out-neighbors j; likewise, it receives the vector $a_{ki}x_k[t]$ from each of its in-neighbors k. It then uses this received data to update its local signals as follows:

$$y_i[t] = u_i[t] - d_i^{\text{out}} x_i[t] + \sum_{k \in \mathcal{N}_i^{\text{in}}} a_{ki} x_k[t] \tag{4}$$

$$x_i[t+1] = x_i[t] + \gamma y_i[t]. \tag{5}$$

Here γ is a global constant gain and all nodes in the digraph G choose their outgoing weights so that the weighted out-degrees satisfy the following:

(**A5**) $\gamma d_i^{\text{out}} < 1$ for every node i (or $\gamma d_i^{\text{out}} \leq 1$ if it is known that G is aperiodic).

These weight choices need not be private (see Remarks 3 and 4).

If we stack the vector signals by defining

$$u[t] = \begin{bmatrix} u_1[t] \\ \vdots \\ u_n[t] \end{bmatrix} \quad y[t] = \begin{bmatrix} y_1[t] \\ \vdots \\ y_n[t] \end{bmatrix} \quad x[t] = \begin{bmatrix} x_1[t] \\ \vdots \\ x_n[t] \end{bmatrix}, \tag{6}$$

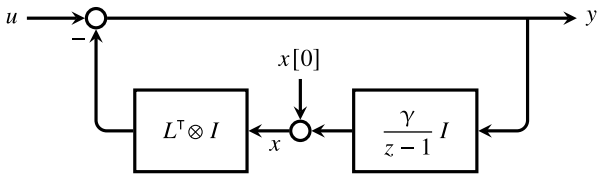

Fig. 1. The base algorithm for distributed averaging. Here $\frac{1}{z-1}$ represents the transfer function of the delayed discrete-time integrator, i.e., the system whose current output is the sum of all of its past inputs.

then we can write the collection of all such updates (4)–(5) as

$$y[t] = u[t] - (L^{\mathsf{T}} \otimes I)x[t] \tag{7}$$

$$x[t+1] = x[t] + \gamma y[t]. \tag{8}$$

Here L denotes the out-Laplacian in (3). A block diagram of the global update equations (7)–(8) is shown in Fig. 1. Before we specify how we choose the inputs u_i, we give a generic convergence result for this algorithm in Theorem 1. In the statement of this result, the vector $v \in \mathbb{R}^n$ denotes the unique vector satisfying $L^{\mathsf{T}}v = 0$ and $\mathbb{1}^{\mathsf{T}}v = 1$. The Perron-Frobenius theorem, together with (**A1**) and (**A5**), guarantees both the existence and uniqueness of v and the property that all entries in v are positive.

Theorem 1: *If $u[t]$ is a constant u, then*

$$\lim_{t \to \infty} y[t] = y^\star = (v \otimes \mathbb{1}^{\mathsf{T}} \otimes I)u. \tag{9}$$

Furthermore, $y[\cdot]$ converges exponentially with rate $|\lambda|$, which is the second-largest magnitude eigenvalue of $I - \gamma L^{\mathsf{T}}$.

Proof: The matrix $I - \gamma L^{\mathsf{T}}$ has a simple eigenvalue at 1 with eigenvector v. It follows from (**A5**) and the Perron-Frobenius theorem that all other eigenvalues of $I - \gamma L^{\mathsf{T}}$ lie strictly inside the unit circle in the complex plane. From (7)–(8) we get

$$y[t+1] = ((I - \gamma L^{\mathsf{T}}) \otimes I)y[t]. \tag{10}$$

If we express $y[0]$ in terms of the eigenvectors of $(I - \gamma L^{\mathsf{T}}) \otimes I$, then those not associated with eigenvalue 1 vanish exponentially and there exists some vector η such that $y[t] \to v \otimes \eta$. Furthermore, since $\mathbb{1}^{\mathsf{T}}$ is a left eigenvector of L^{T}, from (7) we have that $(\mathbb{1}^{\mathsf{T}} \otimes I)y[t] = (\mathbb{1}^{\mathsf{T}} \otimes I)u$. Thus, $\eta = (\mathbb{1}^{\mathsf{T}} \otimes I)u$ and the result follows. ∎

Now suppose each node i chooses a constant input as

$$u_i[\,\cdot\,] \equiv \begin{bmatrix} r_i \\ 1 \end{bmatrix} \in \mathbb{R}^{p+1}. \tag{11}$$

Then we conclude from Theorem 1 that its output y_i will converge exponentially to the constant

$$y_i^\star = \begin{bmatrix} v_i \sum_{j=1}^n r_j \\ v_i n \end{bmatrix}, \tag{12}$$

where v_i denotes the i^{th} component of the vector v (which satisfies $v_i > 0$). Thus by taking the ratio of the appropriate values in $y_i^\star$, node i can compute the global average r_{ave}. This approach is related to the "push-sum" or "ratio consensus" methods for distributed averaging over directed graphs (see [20] and the references therein).

Remark 1: *According to Theorem 1, the final value of the output y is independent of the initial state $x[0]$. This is a key feature of the algorithm and is the mechanism by*

which privacy is possible without using any noise/masking, weight switching, initial scrambling, or homomorphic encryption. This feature also makes the algorithm hot-pluggable, in the sense that the algorithm will recover the correct average after a transient due to changes to the network.

Remark 2: *The base algorithm is not internally stable: even though the output y converges, the state x does not. Indeed, we see from the update equation* (8) *that x grows linearly in t as y converges to a constant. Such linear growth may cause numerical problems if the algorithm is run over long time periods. As explained in [19], it is possible to modify this base algorithm so that the output y still converges to the desired value while x remains bounded, but at the price of slower convergence.*

B. Privacy for the Base Algorithm

In this section we show that the base algorithm guarantees the privacy of the conformists' reference values when each conformist satisfies (**A3**)(b).

Each node can choose an arbitrary initial state $x[0]$ in the base algorithm, but this choice should not be made public and it should be difficult for other nodes to obtain a good prior estimate of it. For example, it might be drawn at random from a distribution having infinite or undefined moments.

To analyze the privacy of the base algorithm, we rewrite the dynamics (7)–(8) as

$$\begin{bmatrix} x[t+1] \\ u[t+1] \end{bmatrix} = A \begin{bmatrix} x[t] \\ u[t] \end{bmatrix}, \quad A = \begin{bmatrix} (I - \gamma L^{\mathsf{T}}) \otimes I & \gamma I \\ 0 & I \end{bmatrix}. \tag{13}$$

It is easy to show that for any k and any $\varphi \in \mathbb{R}^q$, the vector

$$\begin{bmatrix} e_k \otimes \varphi \\ L^{\mathsf{T}} e_k \otimes \varphi \end{bmatrix} \tag{14}$$

is an eigenvector of A associated with the unit eigenvalue.

Suppose i is a conformist, let j be some other node, and suppose (**A3**)(b) holds. Let ζ_j be the collection of all signals available to j. We now construct a matrix C_j such that

$$\zeta_j[t] = C_j \begin{bmatrix} x[t] \\ u[t] \end{bmatrix} \tag{15}$$

for all t. First suppose j is a conformist; then since j can measure its own signals x_j and u_j, the top part of C_j looks like

$$\begin{bmatrix} e_j^{\mathsf{T}} \otimes I & 0 \\ 0 & e_j^{\mathsf{T}} \otimes I \end{bmatrix}. \tag{16}$$

Also, if $h \in \mathcal{N}_j^{\text{in}}$, then C_j includes blocks of the form

$$\begin{bmatrix} a_{hj} e_h^{\mathsf{T}} \otimes I & 0 \end{bmatrix} \tag{17}$$

stacked underneath each other and underneath (16). Let k be as in (**A3**)(b) so that $k \in \{i\} \cup \mathcal{N}_i^{\text{in}}$ and $k \notin \{j\} \cup \mathcal{N}_j^{\text{in}}$, i.e., so that $L_{ki} \neq 0$ but $L_{kj} = 0$. It is straightforward to show that the eigenvector (14) is in the null spaces of the matrices in (16) and (17) and thus also in the unobservable subspace of the pair (A, C_j). In addition, this eigenvector satisfies

$$\begin{bmatrix} 0 & e_i^{\mathsf{T}} \otimes I \end{bmatrix} \begin{bmatrix} e_k \otimes \varphi \\ L^{\mathsf{T}} e_k \otimes \varphi \end{bmatrix} = L_{ki} \varphi. \tag{18}$$

We also have

$$u_i[t] = \begin{bmatrix} 0 & e_i^{\mathsf{T}} \otimes I \end{bmatrix} \begin{bmatrix} x[t] \\ u[t] \end{bmatrix}, \tag{19}$$

and it follows that there exist two trajectories of (13) that produce the same measurements $\zeta_j[\,\cdot\,]$ but for which the inputs $u_i[\,\cdot\,]$ differ by the constant offset $L_{ki}\varphi$. Since $L_{ki} \neq 0$ and φ is arbitrary, we conclude that node i keeps its input u_i private from node j.

Next suppose j is a colluder. Then the matrix C_j has vertically stacked blocks of the form

$$\begin{bmatrix} e_\ell^{\mathsf{T}} \otimes I & 0 \\ 0 & e_\ell^{\mathsf{T}} \otimes I \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} a_{h\ell} e_h^{\mathsf{T}} \otimes I & 0 \end{bmatrix} \tag{20}$$

for any colluder ℓ (including $\ell = j$) and any $h \in \mathcal{N}_\ell^{\text{in}}$. Let k be as in (**A3**)(b) so that $k \in \{i\} \cup \mathcal{N}_i^{\text{in}}$ and $k \notin \{\ell\} \cup \mathcal{N}_\ell^{\text{in}}$ for every colluder ℓ, i.e., so that $L_{ki} \neq 0$ but $L_{k\ell} = 0$ for every colluder ℓ. Again, it is straightforward to show that the eigenvector (14) is in the null space of matrices of the form (20), and the proof of privacy follows using the same argument as above. This unobservability-based notion of privacy assumes nodes have no prior information about the state being inferred, and thus we require (**A4**). We can say more about the case in which (**A4**) does not hold, but that is beyond the scope of this letter.

Remark 3: *Since the privacy argument is based on a standard unobservability analysis for the LTI system* (13)*, knowledge of the matrix A provides no advantage to curious nodes wishing to reconstruct the unobservable state. In particular, the privacy result still holds when all nodes know the entire Laplacian L.*

Remark 4: *If assumption (**A3**)(b) is violated for a node i by a curious node j, then j can observe i's reference value r_i with only knowledge of $\mathcal{N}_i^{\text{in}}$ and no knowledge of the relevant edge weights. Node j can infer both the weights and r_i in $O(\beta)$ iterations, where β is the number of unknown weights, by exploiting the known form of the update equations in* (4) *and* (5)*.*

IV. Enhanced Privacy Using the Line Digraph

A. The Line Digraph

The *line digraph* of a digraph $G = (V, E)$ is the digraph $\overline{G}$ given by $\overline{G} = (\overline{V}, \overline{E})$, where the node set is $\overline{V} = E$ and the edge set $\overline{E} \subset E \times E$ is the set of all ordered pairs of edges in E such that the head of the first edge is the tail of the second one [17]. Fig. 2 depicts a digraph G in blue along with its line digraph $\overline{G}$ in pink. If we assign a linear order to $\overline{E}$, then we can define the head and tail matrices $\overline{H}$ and $\overline{T}$ for $\overline{G}$ in a manner analogous to equation (1). One can show that the unweighted adjacency matrix $\overline{T}\overline{H}^{\mathsf{T}}$ for $\overline{G}$ satisfies $\overline{T}\overline{H}^{\mathsf{T}} = H^{\mathsf{T}}T$ [21]. Given a weight vector w for G, one way to define a weight vector $\overline{w}$ for $\overline{G}$ is $\overline{w} = \overline{H}^{\mathsf{T}} w$, that is, the weight we assign to an edge in $\overline{E}$ is the w-weight of its head in E. We will call such weights for $\overline{G}$ the *head-induced weights*. If we define $\overline{W} = \operatorname{diag}(\overline{w})$, then the head-induced weighted adjacency matrix for $\overline{G}$ satisfies $\overline{T}\,\overline{W}\overline{H}^{\mathsf{T}} = H^{\mathsf{T}}TW$, and thus its weighted out-Laplacian is $\overline{L} = \operatorname{diag}(H^{\mathsf{T}}Tw) - H^{\mathsf{T}}TW$.

If G is strongly connected, we say its weights are *stochastic* when $D\mathbb{1} = Tw = \mathbb{1}$, that is, when $d_i^{\text{out}} = 1$ for each i so that

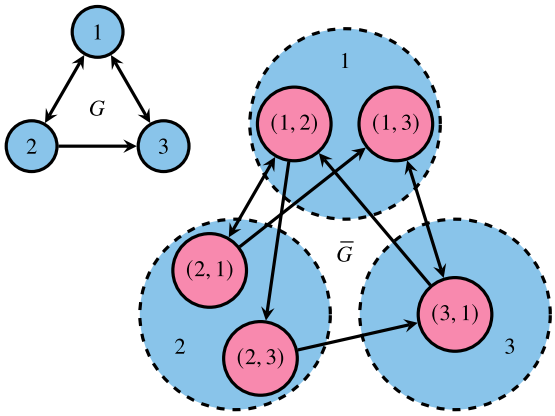

Fig. 2. A digraph G together with its line digraph expansion $\overline{G}$.

G corresponds to an irreducible Markov chain. In this case the out-Laplacians L and $\overline{L}$ (using the head-induced weights) become

$$L = I - TWH^{\mathsf{T}} \quad \text{and} \quad \overline{L} = I - H^{\mathsf{T}}TW. \tag{21}$$

From [22] it is known that, if matrices A and B are the same size, then the products AB^{T} and $B^{\mathsf{T}}A$ have the same nonzero eigenvalues. As a direct corollary, we conclude:

Lemma 1: *If a strongly connected digraph G has stochastic weights and if its line digraph $\overline{G}$ has the associated head-induced weights, then the eigenvalues of their Laplacian matrices L and $\overline{L}$ are identical (including multiplicities), except possibly for the multiplicities of any eigenvalues at one.*

An example of a stochastic weighting scheme is *reciprocal-degree weighting*, where each edge $(i, j) \in E$ has the weight $a_{ij} = 1/\delta_i^{\text{out}}$. These weights are such that $TW = \Delta^{-1}T$. Recall that a digraph $G = (V, E)$ is *symmetric* when $(i, j) \in E$ if and only if $(j, i) \in E$, or equivalently when its unweighted adjacency matrix TH^{T} is symmetric. Likewise, a weighted digraph is *weight-symmetric* when its weighted adjacency matrix TWH^{T} is symmetric. Note that a symmetric digraph with reciprocal-degree weights is generally not weight-symmetric; nevertheless, its weighted out-Laplacian L has all real eigenvalues (we observe that $L = \Delta^{-1}(\Delta - TH^{\mathsf{T}})$ from plugging $TW = \Delta^{-1}T$ into (21), and then use the fact that the product of any positive semidefinite real matrix and any symmetric real matrix has real eigenvalues).

B. The Expanded Algorithm

In the expanded algorithm, we first use G to construct the line digraph $\overline{G}$ with head-induced weights and then apply the base algorithm to $\overline{G}$. We refer to the nodes and edges of $\overline{G}$ as *virtual nodes* and *virtual edges* as they will exist within the nodes and edges of G itself. All virtual nodes having tail i are *internal* to i in the sense that their updates will run on i itself. The virtual edges of $\overline{G}$ are implemented externally over an existing edge in G, so that the expanded algorithm uses precisely those communication links modeled by G itself. Here, we make use of (**A2**) so that the virtual topology of $\overline{G}$ can run on top of the physical topology of G. Fig. 2 shows an example of how the expanded digraph $\overline{G}$ fits within G. Note that $\overline{G}$ remains strongly connected.

If the weights of G satisfy (**A5**), then those of $\overline{G}$ also satisfy (**A5**) and thus Theorem 1 holds for $\overline{G}$. If we choose stochastic weights for G so that $d_i^{\text{out}} = 1$ for each i, then by Lemma 1 the matrices $I - \gamma L$ and $I - \gamma\overline{L}$ have the same spectra, except for multiplicities at $1 - \gamma$. If $1 - \gamma \leq |\lambda|$ for all non-unity eigenvalues λ of $I - \gamma L$ (which will always be the case if we choose γ to be sufficiently close to one), then the update (7)–(8) will have the same exponential convergence rate when applied either to G or to $\overline{G}$. Thus the expanded algorithm will incur no penalty in its convergence rate relative to the base algorithm.

Remark 5: *If G is symmetric and we assign reciprocal-degree weights, then L and $\overline{L}$ have real eigenvalues and we can solve for γ in closed form to optimize convergence speed and implement accelerated versions of the algorithm (as described in [19]). These techniques, however, require some knowledge of bounds on the largest and second smallest eigenvalues of L.*

Next, we assign an input value to each virtual node of $\overline{G}$ so that the base algorithm running on $\overline{G}$ produces the correct global average r_{ave} while maintaining privacy. We randomly and privately initialize $r_{(i,j)}$ such that $\sum_{j\in\mathcal{N}_i^{\text{out}}} r_{(i,j)} = r_i$ for all i. Thus, the reference value of each node is encoded in the sum over its virtual nodes' reference values, and no individual virtual node contains any private information. The input value $u_{(i,j)}$ that we assign to a virtual node (i, j) of $\overline{G}$ and the resulting limit of the outputs $y^\star_{(i,j)}$ are

$$u_{(i,j)}[\,\cdot\,] \equiv \begin{bmatrix} r_{(i,j)} \\ 1/\delta_i^{\text{out}} \end{bmatrix} \quad \text{and} \quad y^\star_{(i,j)} = \begin{bmatrix} v_{(i,j)} \sum_{j=1}^n r_i \\ v_{(i,j)} n \end{bmatrix}. \tag{22}$$

The outputs $y_{(i,j)}$ will converge exponentially to the constant $y^\star_{(i,j)}$ analogous to (12), and we again take the ratio to recover the global average.

C. Privacy for the Expanded Algorithm

In this section we show that the expanded algorithm guarantees the privacy of the conformists' reference values when each conformist satisfies either (**A3**)(a) or (**A3**)(b).

The expanded algorithm also has dynamics of the form (13), but now we replace L with $\overline{L}$ and label the stacked vector components of the signals $x[\cdot]$ and $u[\cdot]$ as $x_{(i,j)}[\cdot]$ and $u_{(i,j)}[\cdot]$ using the virtual node index (i, j).

Suppose i is a conformist, let j be some other node, and let ζ_j in (15) be the collection of all signals available to j in the expanded algorithm. First suppose j is a conformist; then the matrix C_j consists of vertically stacked blocks of the form

$$\begin{bmatrix} e_{(j,s)}^{\mathsf{T}} \otimes I & 0 \\ 0 & e_{(j,s)}^{\mathsf{T}} \otimes I \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} e_{(h,j)}^{\mathsf{T}} \otimes I & 0 \end{bmatrix} \tag{23}$$

for each virtual node (j, s) of $\overline{G}$ and for all $h \in \mathcal{N}_j^{\text{in}}$. Suppose (**A3**)(a) holds; then i has at least two out-neighbors, which means there exists a node $k \in \mathcal{N}_i^{\text{out}}$ such that $k \neq j$. It is possible that j can observe $r_{(i,j)}$; however, since $(i, k) \notin \{(j, s)\} \cup \mathcal{N}_{(j,s)}^{\text{in}}$ for any virtual node (j, s) of $\overline{G}$, the vector

$$\begin{bmatrix} e_{(i,k)} \otimes \varphi \\ \overline{L}^{\mathsf{T}} e_{(i,k)} \otimes \varphi \end{bmatrix} \tag{24}$$

is in the null space of the matrices in (23). Since it is an eigenvector of A with the unit eigenvalue, it is in the unobservable subspace of the pair (A, C_j). In addition, this eigenvector satisfies

$$\begin{bmatrix}0 & \sum_\sigma e_{(i,\sigma)}^\mathsf{T} \otimes I\end{bmatrix}\begin{bmatrix} e_{(i,k)} \otimes \varphi \\ \overline{L}^\mathsf{T} e_{(i,k)} \otimes \varphi\end{bmatrix} = \overline{L}_{(i,k)(i,k)} \cdot \varphi \neq 0 \quad (25)$$

Thus j cannot determine r_i to within any arbitrary offset. Next suppose (**A3**)(b) holds; then there exists $k \in \{i\} \cup \mathcal{N}_i^{\text{in}}$ such that $k \notin \{j\} \cup \mathcal{N}_j^{\text{in}}$. Then for any $\varphi \in \mathbb{R}^q$ and any virtual node (k, τ) of $\overline{G}$, the vector

$$\begin{bmatrix} e_{(k,\tau)} \otimes \varphi \\ \overline{L}^\mathsf{T} e_{(k,\tau)} \otimes \varphi\end{bmatrix} \quad (26)$$

is an eigenvector of A associated with the unit eigenvalue, and again it is straightforward to show that (26) is in the null spaces of the matrices in (23) and thus also in the unobservable subspace of the pair (A, C_j). In addition, this eigenvector satisfies

$$\begin{bmatrix}0 & e_{(i,\sigma)}^\mathsf{T} \otimes I\end{bmatrix}\begin{bmatrix} e_{(k,\tau)} \otimes \varphi \\ \overline{L}^\mathsf{T} e_{(k,\tau)} \otimes \varphi\end{bmatrix} = \overline{L}_{(k,\tau)(i,\sigma)} \cdot \varphi \quad (27)$$

for all virtual nodes (i, σ) of $\overline{G}$. If $k \neq i$ then we can choose $\tau = i$ so that $\overline{L}_{(k,\tau)(i,\sigma)} < 0$ for each σ, and it follows that the sum of $\overline{L}_{(k,\tau)(i,\sigma)}$ over σ is nonzero. If $k = i$ then we choose $\tau = \sigma$, $\overline{L}_{(k,\tau)(i,\sigma)} > 0$; moreover, their sum over σ is also nonzero. Thus, j again cannot determine any of the inputs $u_{(i,\sigma)}$ individually, nor their sum over σ, to within any arbitrary offset.

Next suppose j is a colluder; then the matrix C_j has vertically stacked blocks of the form

$$\begin{bmatrix} e_{(\ell,s)}^\mathsf{T} \otimes I & 0 \\ 0 & e_{(\ell,s)}^\mathsf{T} \otimes I\end{bmatrix} \text{ and } \begin{bmatrix} e_{(h,\ell)}^\mathsf{T} \otimes I & 0\end{bmatrix} \quad (28)$$

for any colluder ℓ (including $\ell = j$), for any virtual node (ℓ, s) of $\overline{G}$, and for any $h \in \mathcal{N}_\ell^{\text{in}}$. Suppose (**A3**)(a) holds; then i has a conformist out-neighbor k, and it follows that $(i, k) \notin \{(\ell, s)\} \cup \mathcal{N}_{(\ell,s)}^{\text{in}}$ for any virtual node (ℓ, s) of $\overline{G}$ internal to any colluder ℓ. Like before, $r_{(i,j)}$ may not be private, but it is straightforward to show that the eigenvector (24) is in the null spaces of the matrices in (28) and that it again satisfies (25). The proof of privacy follows using the same argument as above. Finally, suppose (**A3**)(b) holds; then there exists $k \in \{i\} \cup \mathcal{N}_i^{\text{in}}$ such that $k \notin \{\ell\} \cup \mathcal{N}_\ell^{\text{in}}$ for every colluder ℓ. It is straightforward to show that the eigenvector (26) is in the null spaces of the matrices in (28), and the proof of privacy follows using the same argument as those following (27).

V. Summary and Future Work

In this letter, we showed that a simple dynamic average consensus algorithm guarantees accuracy and preserves privacy under relatively weak topological restrictions on the communication digraph. In contrast to previous work, our method is hot-pluggable, it does not require an initial scrambling phase, it does not inject noise or other masking signals into the computation, it does not require random switching of the edge weights, and it does not use homomorphic encryption.

We have implemented the base and expanded algorithms and experimentally verified that the expanded algorithm converges at the same rate as the base algorithm over a range of graph types and sizes. As expected, the total number of iterations to converge is typically larger in the expanded algorithm due to the initial transient, but the duration of the initial transient depends heavily on graph topology. For example, the performance impact appears to be negligible on graphs with a power law degree distribution. More work is needed to fully understand the transient behavior.

References

[1] N. E. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," in *Proc. Eur. Control Conf.*, Zurich, Switzerland, 2013, pp. 760–765.

[2] S. S. Kia, J. Cortés, and S. Martínez, "Dynamic average consensus under limited control authority and privacy requirements," *Int. J. Robust Nonlinear Control*, vol. 25, no. 13, pp. 1941–1966, 2015.

[3] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Trans. Autom. Control*, vol. 62, no. 2, pp. 753–765, Feb. 2017.

[4] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private average consensus: Obstructions, trade-offs, and optimal algorithm design," *Automatica*, vol. 81, pp. 221–231, Jul. 2017.

[5] M. Ruan, M. Ahmad, and Y. Wang, "Secure and privacy-preserving average consensus," in *Proc. Workshop Cyber Phys. Syst. Security Privacy*, 2017, pp. 123–129.

[6] H. Gao, C. Zhang, M. Ahmad, and Y. Wang, "Privacy-preserving average consensus on directed graphs using push-sum," in *Proc. IEEE Conf. Commun. Netw. Security*, Beijing, China, 2018, pp. 1–9.

[7] N. Gupta, J. Katz, and N. Chopra, "Information-theoretic privacy in distributed average consensus," 2018. [Online]. Available: arXiv:1809.01794.

[8] N. Rezazadeh and S. S. Kia, "Privacy preservation in a continuous-time static average consensus algorithm over directed graphs," in *Proc. Amer. Control Conf.*, Milwaukee, WI, USA, 2018, pp. 5890–5895.

[9] D. Fiore and G. Russo, "Resilient consensus for multi-agent systems subject to differential privacy requirements," *Automatica*, vol. 106, pp. 18–26, Aug. 2019.

[10] C. Altafini, "A dynamical approach to privacy preserving average consensus," in *Proc. 58th IEEE Conf. Decis. Control*, Nice, France, Dec. 2019, pp. 4501–4506.

[11] Y. Liu, J. Wu, I. Manchester, and G. Shi, "Dynamical privacy in distributed computing—Part II: PPSC gossip algorithms," 2019. [Online]. Available: arXiv:1808.00120.

[12] Y. Wang, "Privacy-preserving average consensus via state decomposition," *IEEE Trans. Autom. Control*, vol. 64, no. 11, pp. 4711–4716, Nov. 2019.

[13] T. Charalambous, N. E. Manitara, and C. N. Hadjicostis, "Privacy-preserving average consensus over digraphs in the presence of time delays," in *Proc. 57th Annu. Allerton Conf. Commun. Control Comput.*, Monticello, IL, USA, 2019, pp. 238–245.

[14] I. D. Ridgley, R. A. Freeman, and K. M. Lynch, "Simple, private, and accurate distributed averaging," in *Proc. 57th Allerton Conf. Commun. Control Comput.*, Monticello, IL, USA, 2019, pp. 446–452.

[15] J. Cortés, G. E. Dullerud, S. Han, J. L. Ny, S. Mitra, and G. J. Pappas, "Differential privacy in control and network systems," in *Proc. 55th IEEE Conf. Decis. Control*, Las Vegas, NV, USA, 2016, pp. 4252–4272.

[16] M. Naehrig, K. Lauter, and V. Vaikuntanathan, "Can homomorphic encryption be practical?" in *Proc. 3rd ACM Workshop Cloud Comput. Security Workshop*, New York, NY, USA, 2011, pp. 113–124.

[17] F. Harary and R. Z. Norman, "Some properties of line digraphs," *Rendiconti del Circolo Matematico di Palermo*, vol. 9, no. 2, pp. 161–168, 1960.

[18] J. N. Tsitsiklis, *Problems in Decentalized Decision-Making and Computation*, Massachusetts Inst. Technol. Cambridge, MA, USA, 1984.

[19] S. S. Kia, B. Van Scoy, J. Cortés, R. A. Freeman, K. M. Lynch, and S. Martínez, "Tutorial on dynamic average consensus: The problem, its applications, and the algorithms," *IEEE Control Syst. Mag.*, vol. 39, no. 3, pp. 40–72, Jun. 2019.

[20] C. N. Hadjicostis, A. D. Domínguez-García, and T. Charalambous, *Distributed Averaging and Balancing in Network Systems* (Foundations and Trends (R) in Systems and Control), vol. 13. Hanover, MA, USA: Now, 2018.

[21] C. Balbuena, D. Ferrero, X. Marcote, and I. Pelayo, "Algebraic properties of a digraph and its line digraph," *J. Interconnect. Netw.*, vol. 4, no. 4, pp. 377–393, 2003.

[22] C. R. Johnson and E. A. Schreiner, "The relationship between AB and BA," *Amer. Math. Monthly*, vol. 103, no. 7, pp. 578–582, 1996.