Privacy-Preserving Event-Triggered Quantized Average Consensus

Apostolos I. Rikos, Themistoklis Charalambous, Karl H. Johansson, and Christoforos N. Hadjicostis

***Abstract*— In this paper, we propose a privacy-preserving event-triggered quantized average consensus algorithm that allows agents to calculate the average of their initial values without revealing to other agents their specific value. We assume that agents (nodes) interact with other agents via directed communication links (edges), forming a directed communication topology (digraph). The proposed distributed algorithm can be followed by any agent wishing to maintain its privacy (i.e., not reveal the initial value it contributes to the average) to other, possibly multiple, *curious* but not malicious agents. Curious agents try to identify the initial values of other agents, but do not interfere in the computation in any other way. We develop a distributed strategy that allows agents while processing and transmitting *quantized* information, to preserve the privacy of their initial quantized values and at the same time to obtain, after a finite number of steps, the exact average of the initial values of the nodes. Illustrative examples demonstrate the validity and performance of our proposed algorithm.**

***Index Terms*— Average consensus, quantized averaging, event-triggered, privacy preservation.**

I. INTRODUCTION

A problem of particular interest in distributed control is the *consensus* problem, in which agents communicate locally with other agents under constraints on connectivity [1]. In distributed averaging (a special case of the consensus problem), each agent is initially endowed with a numerical value, which it updates in an iterative fashion, by sending/receiving information to/from other neighboring agents, so that, it is able to eventually compute the average of all initial values. Average consensus has received significant attention recently and has been studied extensively in settings where each agent processes and transmits real-valued states with infinite precision [2]–[4].

The case where capacity-limited network links can only allow messages of certain length to be transmitted between agents, has also received significant attention recently as it effectively extends techniques for average consensus towards the direction of quantized consensus. The quantized nature of processing and communication is better suited to the available network resources (e.g., physical memories of finite capacity and/or digital communication channels of limited data rate); it also exhibits several other advantages, such as applicability to security/privacy applications where encryption is desirable [5]. As a result, a large number of works have studied the quantized average consensus problem and various probabilistic and deterministic strategies have been proposed [6]–[11].

Average consensus algorithms require each agent to exchange and disclose state information to its neighbors, which may be undesirable in cases where the state is private or contains sensitive information. In many emerging applications (e.g., health care and opinion forming/agreement in social networks) preserving the privacy of participating components is necessary for enabling cooperation between agents without requiring them to disclose sensitive information. There have been different approaches for dealing with privacy preservation in such systems. For example, [12] proposed a method in which, each node wishing to protect its privacy adds a random offset value with zero mean to its initial value, thus ensuring that its value will not be revealed to curious nodes that might be observing the exchange of values in the network. The main idea is based upon the observation that, when a large number of nodes employ the protocol, the sum of their offsets (independently chosen) will be essentially zero and therefore the nodes will converge to the true average value of the network. A related line of research is based on differential privacy [13], [14], in which agents inject uncorrelated noise into the exchanged messages so that the data associated to a particular agent cannot be inferred by a curious node during the execution of the algorithm. Nevertheless, the exact average value is not obtained due to the induced trade-off between enabled privacy and computational accuracy [14]. To overcome this trade-off and guarantee convergence to the exact average, the injection of correlated noise at each time step and for a finite period of time is proposed in [15], thus allowing a node to avoid revealing its own initial value as well as the initial values of other nodes. Once this period of time ends, each node ensures that the accumulated sum of offsets it added in the iterative commutation is removed. In [16], the nodes asymptotically subtract the initial offset values they added in the computation while in [17] each node masks its initial value with an offset such that the sum of the offsets of each agent is zero, thus guaranteeing convergence to the average. Another approach that guarantees privacy preservation is via homomorphic encryption [18]–[20]. However, this approach requires the existence of trusted nodes and imposes heavier computational requirements on the nodes.

In this paper, we build on the concepts introduced in [15], [16] and propose a novel distributed mechanism in which, each agent that would like to protect its privacy, follows

Apostolos I. Rikos and K. H. Johansson are with the Division of Decision and Control Systems, KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden. E-mails:{`rikos,kallej`}`@kth.se`.

T. Charalambous is with the Department of Electrical Engineering and Automation, Aalto University, 02150 Espoo, Finland. E-mail: `themistoklis.charalambous@aalto.fi`.

C. N. Hadjicostis is with the Department of Electrical and Computer Engineering, University of Cyprus, 1678 Nicosia, Cyprus: E-mail: `chadjic@ucy.ac.cy`.

a finite-time event-triggered quantized average consensus protocol which involves adding and subtracting offsets to its actual value according to an event-based strategy for a predefined number of steps. More specifically, when the token that triggers action arrives at a specific node for the first time, the node adds a substantial negative *quantized* offset to its initial value. Note that this is different from [15], [16] where the initial offset added is a real number. Then, the node gradually removes the initial offset, one chunk at a time (but not asymptotically as done in [16]), ensuring that by the end of a certain number of predefined steps, the total (accumulated sum of) offset is cancelled out. Note that unlike other privacy preserving protocols proposed in the literature (e.g., [12], [15], [16], [21]), the privacy strategy proposed in this paper takes full advantage of the algorithm's finite time nature. This means that consensus to the *exact* average of the initial values is achieved after a finite number of iterations, while the error, introduced from the offset initially infused in the network by the nodes following the protocol, vanishes completely.

II. Notation and Preliminaries

A. Notation

The sets of real, rational, integer, and natural numbers are denoted by $\mathbb{R}, \mathbb{Q}, \mathbb{Z}$ and $\mathbb{N}$, respectively. The symbols $\mathbb{Z}_{\geq 0}$ ($\mathbb{Z}_{>0}$) and $\mathbb{Z}_{\leq 0}$ ($\mathbb{Z}_{<0}$) denote the sets of nonnegative (positive) and nonpositive (negative) integers respectively. Vectors are denoted by small letters whereas matrices are denoted by capital letters. The transpose of a matrix A is denoted by A^T. For $A \in \mathbb{R}^{n \times n}$, A_{ij} denotes the entry at row i and column j. By $\mathbf{1}$ we denote the all-ones vector and by I we denote the identity matrix (of appropriate dimensions).

B. Graph Theory

Consider a network of n ($n \geq 2$) agents communicating only with their immediate neighbors. The communication topology can be captured by a directed graph (digraph), called *communication digraph*. A digraph is defined as $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \dots, v_n\}$ is the set of nodes (representing the agents) and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} - \{(v_j, v_j) \mid v_j \in \mathcal{V}\}$ is the set of edges (self-edges excluded) whose cardinality is denoted as $m = |\mathcal{E}|$. A directed edge from node v_i to node v_j is denoted by $m_{ji} \triangleq (v_j, v_i) \in \mathcal{E}$, and captures the fact that node v_j can receive information from node v_i (but not the other way around). We assume that the given digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ is *strongly connected* (i.e., for each pair of nodes $v_j, v_i \in \mathcal{V}$, $v_j \neq v_i$, there exists a directed *path*[1] from v_i to v_j). The subset of nodes that can directly transmit information to node v_j is called the set of in-neighbors of v_j and is represented by $\mathcal{N}_j^- = \{v_i \in \mathcal{V} \mid (v_j, v_i) \in \mathcal{E}\}$, while the subset of nodes that can directly receive information from node v_j is called the set of out-neighbors of v_j and is represented by $\mathcal{N}_j^+ = \{v_l \in \mathcal{V} \mid (v_l, v_j) \in \mathcal{E}\}$. The cardinality of $\mathcal{N}_j^-$ is called the *in-degree* of v_j and is denoted by $\mathcal{D}_j^-$ (i.e., $\mathcal{D}_j^- = |\mathcal{N}_j^-|$), while the cardinality of $\mathcal{N}_j^+$ is called the *out-degree* of v_j and is denoted by $\mathcal{D}_j^+$ (i.e., $\mathcal{D}_j^+ = |\mathcal{N}_j^+|$).

[1]A directed *path* from v_i to v_j exists if we can find a sequence of vertices $v_i \equiv v_{l_0}, v_{l_1}, \dots, v_{l_t} \equiv v_j$ such that $(v_{l_{\tau+1}}, v_{l_\tau}) \in \mathcal{E}$ for $\tau = 0, 1, \dots, t-1$.

C. Agent Operation

With respect to quantization of information flow, we have that at time step $k \in \mathbb{Z}_{\geq 0}$ each node $v_j \in \mathcal{V}$ maintains the state variables $y_j^s[k], z_j^s[k], q_j^s[k]$, where $y_j^s[k] \in \mathbb{Z}$, $z_j^s[k] \in \mathbb{N}$ and $q_j^s[k] = y_j^s[k]/z_j^s[k]$, and the mass variables $y_j[k], z_j[k]$, where $y_j[k] \in \mathbb{Z}$ and $z_j[k] \in \mathbb{Z}_{\geq 0}$. The aggregate states are denoted by

$$
\begin{aligned}
y^s[k] &= [y_1^s[k] \ \dots \ y_n^s[k]]^{\mathrm{T}} \in \mathbb{Z}^n, \\
z^s[k] &= [z_1^s[k] \ \dots \ z_n^s[k]]^{\mathrm{T}} \in \mathbb{Z}_{\geq 0}^n, \\
q^s[k] &= [q_1^s[k] \ \dots \ q_n^s[k]]^{\mathrm{T}} \in \mathbb{Q}^n, \\
y[k] &= [y_1[k] \ \dots \ y_n[k]]^{\mathrm{T}} \in \mathbb{Z}^n, \\
z[k] &= [z_1[k] \ \dots \ z_n[k]]^{\mathrm{T}} \in \mathbb{Z}_{\geq 0}^n.
\end{aligned}
$$

Furthermore, each node maintains a privacy value $u_j[k]$, the number of offset adding steps $L_j \in \mathbb{N}$, the offset adding counter $l_j \in \mathbb{N}$ and its transmission counter $c_j \in \mathbb{N}$.

Transmission Policy. We assume that each node is aware of its out-neighbors and can directly transmit messages to each of them. However, it cannot necessarily receive messages (at least not directly) from them. In the proposed distributed protocol, each node v_j assigns a *unique order* in the set $\{0, 1, ..., \mathcal{D}_j^+ - 1\}$ to each of its outgoing edges m_{lj}, where $v_l \in \mathcal{N}_j^+$. More specifically, the order of link (v_l, v_j) for node v_j is denoted by P_{lj} (such that $\{P_{lj} \mid v_l \in \mathcal{N}_j^+\} = \{0, 1, ..., \mathcal{D}_j^+ - 1\}$). This unique predetermined order is used during the execution of the proposed distributed algorithm as a way of allowing node v_j to transmit messages to its out-neighbors in a *round-robin*[2] fashion.

D. Quantized Averaging via Deterministic Mass Summation

The objective of quantized average consensus problems is the development of distributed algorithms which allow nodes to process and transmit quantized information, so that they have short communication packages and eventually obtain, after a finite number of steps, a fraction q^s which is equal to the *exact* average of the initial quantized values of the nodes.

Following the recently proposed method in [9], assume that each node v_j in the network has a quantized[3] initial value $y_j[0] \in \mathbb{Z}$. At each time step k, each node $v_j \in \mathcal{V}$ maintains its mass variables $y_j[k] \in \mathbb{Z}$ and $z_j[k] \in \mathbb{Z}_{\geq 0}$ and its state variables $y_j^s[k] \in \mathbb{Z}$, $z_j^s[k] \in \mathbb{N}$ and $q_j^s[k] = y_j^s[k]/z_j^s[k]$. It

[2]When executing the deterministic protocol, each node v_j transmits to its out-neighbors, one at a time, by following the predetermined order. The next time it transmits to an out-neighbor, it continues from the outgoing edge it stopped the previous time and cycles through the edges in a round-robin fashion.

[3]Following [8], [11] we assume that the state of each node is integer valued. This abstraction subsumes a class of quantization effects (e.g., uniform quantization).

updates the values of the mass variables as

$$y_j[k+1] = y_j[k] + \sum_{v_i \in \mathcal{N}_j^-} \mathbb{1}_{ji}[k] y_i[k], \tag{1a}$$

$$z_j[k+1] = z_j[k] + \sum_{v_i \in \mathcal{N}_j^-} \mathbb{1}_{ji}[k] z_i[k], \tag{1b}$$

where

$$\mathbb{1}_{ji}[k] = \begin{cases} 1, & \text{if a message is received at } v_j \text{ from } v_i \text{ at } k, \\ 0, & \text{otherwise.} \end{cases}$$

If *any* of the following event-triggered conditions:
(C1): $z_j[k+1] > z_j^s[k]$,
(C2): $z_j[k+1] = z_j^s[k]$ and $y_j[k+1] \geq y_j^s[k]$,
is satisfied, node v_j updates its state variables as follows:

$$z_j^s[k+1] = z_j[k+1], \tag{2a}$$

$$y_j^s[k+1] = y_j[k+1], \tag{2b}$$

$$q_j^s[k+1] = \frac{y_j^s[k+1]}{z_j^s[k+1]}. \tag{2c}$$

Then, it transmits its mass variables $y_j[k+1]$, $z_j[k+1]$ to an out-neighbor $v_l \in \mathcal{N}_j^+$ chosen according to the unique order it assigned to its out-neighbors during initialization and sets its mass variables equal to zero (i.e., $y_j[k+1] = 0$ and $z_j[k+1] = 0$).

Definition 1: The system is able to achieve quantized average consensus if, for every $v_j \in \mathcal{V}$, there exists k_0 so that for every $k \geq k_0$ we have

$$y_j^s[k] = \frac{\sum_{l=1}^n y_l[0]}{\alpha} \quad \text{and} \quad z_j^s[k] = \frac{n}{\alpha}, \tag{3}$$

for some $\alpha \in \mathbb{N}$. This means that

$$q_j^s[k] = \frac{(\sum_{l=1}^n y_l[0])/\alpha}{n/\alpha} = \frac{\sum_{l=1}^n y_l[0]}{n} =: q, \tag{4}$$

i.e., for $k \geq k_0$ every node v_j has calculated q as the ratio of two integer values.

The following result from [9] provides a worst case upper bound regarding the number of time steps required for quantized averaging to be achieved.

Theorem 1 ([9]): The iterations in (1) and (2) allow each node $v_j \in \mathcal{V}$ to reach quantized average consensus (i.e., v_j's state variables fulfil (3) and (4)) after a finite number of steps $\mathcal{S}_t$, bounded by $\mathcal{S}_t \leq nm^2$, where n is the number of nodes and m is the number of edges in the network.

III. Problem Formulation

Consider a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$, where each node $v_j \in \mathcal{V}$ has an initial quantized value $y_j[0]$ (for simplicity, we take $y_j[0] \in \mathbb{Z}$). Agents aim to agree on the average of their initial quantized values in a distributed way through local exchange of information only, namely the information exchange takes place only between agents which are neighbors with respect to $\mathcal{G}_d$ representing the system communication architecture. There exists a set of nodes, however, that wish to preserve their privacy by not revealing their initial values to the other nodes. On the other hand, some nodes are curious and try to identify the initial values of all or a subset of nodes in the network.

The problem we consider in this paper is to develop a strategy for the nodes that wish to prevent their privacy (i.e., not reveal their initial value to the other nodes) when they exchange information with neighboring nodes for reaching consensus to the average. This strategy should work seamlessly along with a strategy that is not privacy-preserving.

As aforementioned, we assume that curious nodes try to identify the initial values of other nodes but do not interfere in the computation in any other way. We also assume that curious nodes may collaborate arbitrarily and that they know the topology of the network and the predefined algorithm, followed by nodes that would like to preserve their privacy, but not the actual parameters chosen by these nodes.

Our contribution is a variant of the quantized averaging strategy (described in Section II-D), followed by nodes that wish to preserve their privacy (the remaining nodes simply follow the original strategy in Section II-D).

IV. Proposed Strategy and Convergence Analysis

In this section, we present and analyze a distributed iterative strategy that allows nodes (while processing and transmitting *quantized* information via available communication links between nodes) to preserve the privacy of their initial quantized values and to obtain, after a finite number of steps, a fraction q^s which is equal to the exact average q of the initial values of the nodes.

A. Initialization for Quantized Privacy Strategy

The primary objective in our system is to calculate the average of the initial values of the nodes in the network while preserving the privacy of at least the nodes following the protocol. Our strategy is based on the event-triggered deterministic algorithm (1)–(2) with some modifications (since the event-triggered deterministic algorithm (1)–(2) is not privacy- preserving). The main difference is that a mechanism is deployed that incorporates an offset to the mass variable of each node following the protocol, effectively preserving the privacy of its initial value.

In previous works (see, for example, [12], [15], [16], [21] and references therein) node v_j sets its initial value to $\widetilde{y}_j[0] = y_j[0] + u_j$, $u_j \in \mathbb{R}$. However, in this case, we require that the initial offset u_j is a negative integer number, i.e., $u_j \in \mathbb{Z}_{<0}$, so that the event-triggered conditions (C1) and (C2) are guaranteed to lead to the calculation of the initial average after a finite number of time steps. Furthermore, the absolute value of the initial offset u_j and the number of offset adding steps L_j need to be greater than the number of out-neighbors $\mathcal{D}_j^+$ of node v_j. Specifically, at initialization, each node v_j chooses the variables L_j, u_j, and $u_j[l_j]$ for $l_j \in [0, L_j]$, to satisfy the following constraints.

$$L_j \geq \mathcal{D}_j^+, \tag{5a}$$

$$u_j = -\sum_{l_j=0}^{L_j} u_j[l_j], \tag{5b}$$

$$u_j[l_j] = 0, \ \forall \ l_j \notin [0, L_j], \tag{5c}$$

$$u_j[l_j] > 0, \ \forall \ l_j \in [0, L_j]. \tag{5d}$$

Constraints (5a)–(5d) are explicitly analyzed below:

1) In (5a) the offset adding steps L_j of every node v_j need to be greater than or equal to node v_j's out-degree so that every out-neighbor $v_i \in \mathcal{N}_j^+$ will receive at least one value of $u_j[l_j]$ from node v_j. This has to do with privacy preservation guarantees as discussed in Section IV-D.
2) Eq. (5b) is imposed to that the accumulated offset infused in the computation by node v_j is equal to zero and the exact quantized average of the initial values can be calculated without any error.
3) Eq. (5c) means that node v_j does not need to continue injecting nonzero offsets in the network and the exact quantized average of the initial values can be calculated without any error.
4) In (5d) the offset $u_j[l_j]$, for $l_j \in [0, L_j]$, which is injected by each node v_j to the network each time its event-triggered conditions hold, needs to be nonnegative so that the event-triggered conditions (C1) and (C2) hold for every node after a finite number of steps and the exact quantized average of the initial values can be calculated.

As a result of the above choices, the initial offset u_j every node v_j injects in the network satisfies $u_j \leq -\mathcal{D}_j^+$. This is important to ensure that, during the operation of the proposed algorithm, the event-triggered conditions (C1) and (C2) hold for every node after a finite number of steps. If $u_j \geq 0$, the event-triggered conditions (C1) and (C2) may not hold and the proposed protocol may fail to calculate the correct average of the initial values.

B. Algorithm Description

The proposed algorithm is a quantized value transfer process in which every node in a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$, performs operations and transmissions according to a set of event-triggered conditions. The intuition behind the algorithm is as follows. Each node v_j that would like to preserve its privacy performs the following steps:

- As explained in the previous section, it initializes a counter l_j to zero (i.e., $l_j = 0$), and chooses the total number of offset adding steps L_j such that $L_j \geq \mathcal{D}_j^+$ and the set of $(L_j + 1)$ positive offsets $u_j[l_j] > 0$ where $l_j \in \{0, \ldots, L_j\}$. Finally it sets the initial negative offset u_j ($u_j \leq -\mathcal{D}_j^+$) that it injects to its initial value $y_j[0]$ to $u_j = -\sum_{l_j=0}^{L_j} u_j[l_j]$. For example, suppose that node v_j has four out-neighbors ($\mathcal{D}_j^+ = 4$). This means that it can choose $L_j = 6$, and then set $u_j[1] = 3$, $u_j[2] = 2$, $u_j[3] = 4$, $u_j[4] = 1$, $u_j[5] = 2$, $u_j[6] = 5$; finally it sets $u_j = -17$.
- It chooses an out-neighbor $v_l \in \mathcal{N}_j^+$ according to the unique order P_{lj} (initially, it chooses $v_l \in \mathcal{N}_j^+$ such that $P_{lj} = 0$) and transmits $z_j[0]$ and $\widetilde{y}_j[0]$ to this out-neighbor. Then, it sets $\widetilde{y}_j[0] = 0$ and $z_j[0] = 0$.
- During the execution of the algorithm, at every step k, node v_j may receive a set of mass variables $\widetilde{y}_i[k]$ and $z_i[k]$ from each in-neighbor $v_i \in \mathcal{N}_j^-$. Then, node v_j updates its values according to (1a)–(1b) and checks whether any of its event-triggered conditions hold. If so, it injects an offset $u_j[l_j]$ to the value of $y_j[k+1]$ and increases its offset increasing counter l_j by one. Then, it sets its state variables $y_j^s[k+1]$ and $z_j^s[k+1]$ equal to $\widetilde{y}_j[k+1] = y_j[k+1] + u_j[l_j]$ and $z_j[k+1]$, respectively, and transmits them to an out-neighbor according to the predetermined unique order. If none of the conditions (1a)–(1b) hold, then node v_j stores $y_j[k+1]$ and $z_j[k+1]$. Note that if no message is received from any of the in-neighbors, the mass variables remain the same.

The proposed algorithm is summarized in Algorithm 1.

Algorithm 1 Privacy-Preserving Event-Triggered Quantized Average Consensus

```
Input: A strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. Each node $v_j \in \mathcal{V}$ has an initial value $y_j[0] \in \mathbb{Z}$.
Initialization: Each node $v_j \in \mathcal{V}$ does the following:
1) It assigns a unique order $P_{lj}$ in the set $\{0, 1, ..., \mathcal{D}_j^+ - 1\}$ to each of its out-neighbors $v_l \in \mathcal{N}_j^+$.
2) It sets counter $c_j$ to 0 and priority index $e_j$ to $c_j$.
3) It sets counter $l_j$ to 0, chooses $L_j \in \mathbb{N}$, where $L_j \geq \mathcal{D}_j^+$, and $u_j[k] \geq 0$ for $k \in \{0, \ldots, L_j\}$, and $u_j[k'] = 0$ for $k' > L_j$. It also sets $u_j = -\sum_{l_j=0}^{L_j} u_j[l_j]$.
4) It sets $\widetilde{y}_j[0] = y_j[0] + u_j$, $z_j[0] = 1$, $z_j^s[0] = 1$ and $y_j^s[0] = \widetilde{y}_j[0]$ (which means that $q_j^s[0] = \widetilde{y}_j[0]/1$).
5) It selects out-neighbor $v_l \in \mathcal{N}_j^+$ such that $P_{lj} = e_j$ and transmits $z_j[0]$ and $\widetilde{y}_j[0]$ to this out-neighbor. Then, it sets $\widetilde{y}_j[0] = 0$ and $z_j[0] = 0$.
6) It sets $c_j = c_j + 1$ and $e_j = c_j \mod \mathcal{D}_j^+$.
Iteration: For $k = 0, 1, 2, \ldots$, node $v_j \in \mathcal{V}$, that follows the protocol, does the following:
  • if it receives $y_i[k]$ and $z_i[k]$ from every in-neighbor $v_i \in \mathcal{N}_j^-$ then it updates its values according to (1a)-(1b).
    ◦ if any of conditions (C1) and (C2) hold then
      - it sets $\widetilde{y}_j[k+1] = u_j[l_j] + y_j[k+1]$ and $l_j \leftarrow l_j + 1$;
      - it sets $z_j^s[k+1] = z_j[k+1]$, $y_j^s[k+1] = \widetilde{y}_j[k+1]$ and $q_j^s[k+1] = \widetilde{y}_j[k+1]/z_j^s[k+1]$;
      - it transmits $z_j[k+1]$ and $\widetilde{y}_j[k+1]$ to out-neighbor $v_\lambda \in \mathcal{N}_j^+$ for which $P_{\lambda j} = e_j$ and it sets $\widetilde{y}_j[k+1] = 0$, $y_j[k+1] = 0$ and $z_j[k+1] = 0$;
      - it sets $c_j = c_j + 1$ and $e_j = c_j \mod \mathcal{D}_j^+$.
    ◦ else it stores $y_j[k+1]$ and $z_j[k+1]$.
Output: (3) and (4) hold for every $v_j \in \mathcal{V}$.
```

Remark 1: Unlike other privacy preserving protocols proposed in the literature (see, e.g., [12], [15], [16], [21]), the proposed strategy takes full advantage of the algorithm's finite time nature which means that consensus to the average of the initial values is reached after a finite number of iterations, while the error, introduced via the offset initially infused in the network by the nodes following the protocol, vanishes.

C. Deterministic Convergence Analysis

For the development of the necessary results regarding the operation of Algorithm 1 let us consider the following setup, the analysis of which for the non privacy preserving case can be found in [22].

Setup: Consider a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. During the execution of Algorithm 1, at time step k_0, there is at least one node $v_{j'} \in \mathcal{V}$, for which

$$z_{j'}[k_0] \geq z_i[k_0], \ \forall v_i \in \mathcal{V}. \tag{6}$$

Then, among the nodes $v_{j'}$ for which (6) holds, there is at least one node v_j for which

$$y_j[k_0] \geq y_{j'}[k_0], \ v_j, v_{j'} \in \{v_i \in \mathcal{V} \mid (6) \text{ holds}\}. \tag{7}$$

For notational convenience we will call the mass variables of node v_j for which (6) and (7) hold as the "leading mass" (or "leading masses").

Now we present the following two lemmas, which are helpful in the development of our results. However, due to space limitations, we do not provide their proofs below; they will be available in an extended version of this paper.

Lemma 1 ([22]): Under the above *Setup*, the "leading mass" or "leading masses" at time step k, will always fulfill the "Event-Trigger Conditions" (C1) and (C2). This means that the mass variables of node v_j for which (6) and (7) hold at time step k_0 will be transmitted (at time step k_0) by v_j to an out-neighbor $v_l \in \mathcal{N}_j^+$.

Lemma 2: Under the above *Setup*, we have that if the event-triggered conditions of node v_j are fulfilled at least $(L_j + 1)$ instances then, from (5a)–(5d), we have that the accumulated amount of offset node v_j has injected to the network becomes equal to zero.

The following theorem states that the proposed algorithm allows all agents to reach quantized average consensus after a finite number of steps, for which we provide an upper bound.

Theorem 2: Consider a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. The execution of Algorithm 1 allows each node $v_j \in \mathcal{V}$ to reach quantized average consensus after a finite number of steps, $\mathcal{S}_t$, bounded by $\mathcal{S}_t \leq m^2(L_{\max} + 1 + n)$, where n is the number of nodes, m is the number of edges in the network and $L_{\max} = \max_{v_j \in \mathcal{V}} L_j$ is the maximum value of offset adding steps chosen from nodes in the network.

Proof: Let us assume that, at time step k_0, the mass variables of node v_j are the "leading mass" and there exists a set of nodes $\mathcal{V}^f[k_0] \subseteq \mathcal{V}$ which is defined as $\mathcal{V}^f[k_0] = \{v_i \in \mathcal{V} \mid z_i[k_0] > 0 \text{ but (6) or (7) do not hold}\}$ (i.e., it is the set of nodes which have nonzero mass variables at time step k_0 but they are not "leading masses"). Note here that if the "leading mass" reaches a node simultaneously with some other (leading or otherwise mass) then it gets "merged", i.e., the receiving node "merges" the mass variables it receives, by summing their numerators and their denominators, creating a set of mass variables with a greater denominator. Furthermore, we will say that the "leading mass", gets "obstructed" if it reaches a node whose state variables are greater than the mass variables (i.e., either the denominator of the node's state variables is greater than the denominator of the mass variables, or, if the denominators are equal, the numerator of the state variables is greater than the numerator of the mass variables). Note that if the "leading mass" we started off with at time step k_0 gets "obstructed" then its no longer the "leading mass", since it will not fulfill the event-triggered conditions of the corresponding node (from Lemma 1 we have that the "leading mass" always fulfills the event-triggered conditions).

Suppose that the "leading mass" at time step k_0 is held by node v_j and is given by $\widetilde{y}_j[k_0]$ and $z_j[k_0]$. Since this leading mass does not get merged or obstructed, during the execution of Algorithm 1, it will reach every node $v_j \in \mathcal{V}$ in at most m^2 steps, where $m = |\mathcal{E}|$ is the number of edges of the given digraph $\mathcal{G}_d$ (this follows from Proposition 3 in [23], which actually provides a bound for an unobstructed "leading mass" to reach every other node). Even if the mass gets obstructed at some node, this means that it is no longer the "leading mass"; in fact, the new "leading mass" passed by this node earlier on and has already followed the same path that the former "leading mass" was to follow.

Let us assume now that we execute Algorithm 1 for $m^2(L_{\max} + 1)$ time steps, where $L_{\max} = \max_{v_j \in \mathcal{V}} L_j$. During the $m^2(L_{\max} + 1)$ time steps each node v_j will receive at least $(L_{\max} + 1)$ times a set of nonzero mass variables from its in-neighbors that are equal to the "leading mass". Since this set of mass variables is equal to the "leading mass", from Lemma 1, we have that the event-triggered conditions of each node v_j will hold for at least $(L_{\max} + 1)$ events during these $m^2(L_{\max} + 1)$ time steps. This means that each node v_j adds the offset $u_j[l_j]$ to its mass variables for $(L_{\max} + 1)$ events. Since we have that $\sum_{l_j=0}^{L_j} u_j[l_j] = -u_j$, from Lemma 2, after $m^2(L_{\max} + 1)$ time steps, the accumulated amount of offset each node v_j has injected to the network becomes equal to zero. As a result, we have that

$$\sum_{v_j \in \mathcal{V}} \widetilde{y}_j[m^2(L_{\max} + 1)] = \sum_{v_j \in \mathcal{V}} y_j[0],$$

and

$$\sum_{v_j \in \mathcal{V}} z_j[m^2(L_{\max} + 1)] = \sum_{v_j \in \mathcal{V}} z_j[0].$$

After executing Algorithm 1 for an additional number of time steps equal to nm^2, we have that the convergence analysis of our protocol becomes identical to the analysis presented in [22, Proposition 1] where during the additional nm^2 we have that either (a) the "leading masses" never merge (because they all move simultaneously) or (b) there are at most $n-1$ "mergings" of mass variables (each merging occurring after at most m^2 time steps). As a result, we have that after $\mathcal{S}_t$ iterations, where $\mathcal{S}_t \leq m^2(L_{\max}+1+n)$, we are guaranteed that the offset each node has injected to the network has become equal to zero, and sufficient "mergings" (at most $n-1$) have occurred, so that the nodes will be able to calculate the average of their initial values. ■

Remark 2: The proof of Theorem 2 shows us that if we execute the proposed distributed protocol for a finite number of time steps equal to $m^2 L_{\max}$, then every node in the network will receive a set of nonzero mass variables that are equal to the leading mass for at least $L_{\max}$ instances. This means that the event-triggered conditions will be fulfilled for each node in the network for at least $L_{\max}$ instances and, from Lemma 2, the accumulated amount of offset injected in the network from each node in the network is equal to zero. As a result, by executing the proposed protocol for an additional number of time steps equal to nm^2, we have that every nonzero mass in the network will merge to one leading mass (or multiple leading masses that are equal) that is (are) equal to the average of the initial values, and then this (these) leading mass (masses) will update the state variables of each node in the network, setting them equal to the average of the initial values.

D. Topological conditions for privacy preservation

We establish topological conditions that ensure privacy for the nodes following the proposed protocol despite the presence of possibly colluding curious nodes in the network.

Proposition 1: Consider a fixed strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes. Assume that a set of nodes P follow the predefined privacy-preserving protocol, as described in Algorithm 1, with offsets chosen as in (5a)-(5c). Curious node v_c will not be able to identify the initial value $y_j[0]$ of $v_j \in P$, as long as v_j has

a) at least one other node (in- or out-neighbor) $v_\ell \in P$ connected to it, or
b) has a non-curious in-neighbor $v_i \notin P$ which first transmits to node v_j at initialization.

In other words, if one of the conditions in Proposition 1 is satisfied, the network will reach average consensus and the privacy of the initial values of the nodes following the privacy-preserving protocol will be preserved.

Proof: Let us assume that node v_j follows the privacy-preserving protocol. We consider the following simple scenarios, which constitute the building blocks of the directed network, due to the token-based nature of the privacy-preserving protocol.

1) It is easy to observe that if all the in- and out-neighbors of node v_j are curious and they communicate with each other, it is not possible for this node to keep its privacy. At initialization, the curious nodes will know $\widetilde{y}_j[0]$. At every step, they will know what node v_j has received and they will be able to extract the offset added. Hence, after (L_j+1) updates from node v_j, the curious nodes will be able to compute the initial offset, since the initial offset satisfies (5b); hence, privacy of the initial value is not preserved. As a result, at least one neighbor that is not curious is needed.
2) Suppose that there exists at least one in-neighbor, say v_i, that is not curious, but it does not follow the privacy-preserving protocol; all other in- and out-neighbors of node v_j and node v_i are assumed (as a worst-case assumption) to be curious. If at initialization, node v_i first transmits to node v_j, then the curious nodes will not be able to infer its initial value, since they cannot distinguish the initial values of v_i and v_j; even if at the end the curious nodes are able to obtain all the offsets, they can only infer the sum of the initial values of node v_j and node v_i, but not their individual values, i.e., *both* nodes preserve their privacy. If, however, v_j is not contacted by node v_i at initialization, then the curious nodes will be able to extract the initial condition and know all the inputs (and hence outputs) of node v_i (recall that node v_i does not follow the privacy-preserving protocol). Thus, curious nodes will be able to infer all the values that node v_i transmits to v_j and, as a consequence, *none* of the nodes will preserve its privacy. This suggests that, if a node v_i trusts that an out-neighbor v_j is not curious and follows the privacy-preserving protocol, then this out-neighbor should be prioritized, i.e., $P_{ji} = 0$.
3) Let us consider the case for which there exists one out-neighbor of node v_j, say v_l, that is neither curious nor following the privacy-preserving protocol, and all other in- and out-neighbors of both nodes are curious. Since curious nodes can infer the input of node v_l from its output, then they will be able to extract the messages of node v_j in the same way as if a curious node was directly connected to node v_j. Hence, privacy of node v_j cannot be preserved.
4) If we assume that there exists at least one in-neighbor, say v_i, that follows the privacy-preserving protocol, the curious nodes will not be able to infer the initial value of node v_j due to the offsets node v_i transmits to v_j and, therefore, both v_i and v_j retain their privacy.

From these discussions, we can deduce that it is *sufficient* that conditions a) and b) in Proposition 1 are satisfied. In such cases, the initial values of both nodes are protected (though the sum of these values may be exposed). Note that it is also sufficient if a node that does not follow the privacy-preserving protocol first selects an out-neighbor that does follow the protocol to transmit its values; see item 5) in the initialization stage of Algorithm 1. ■

Note here that a set of curious nodes could also attempt to "estimate" the initial values of some other nodes (e.g., by taking into account any available statistics about the initial values, u_j and L_j). However, in our analysis in this section, we are interested in whether the curious nodes can exactly infer the value of another node. The case where curious nodes attempt to "estimate" the initial value of other nodes will be considered as a future direction.

V. Simulation Results

In this section, we present simulation results to illustrate the behavior of our proposed distributed protocol. Specifically, we analyze the cases of:

A) a ring digraph of 20 nodes with the average of the initial values of the nodes turning out to be equal to $q = 191/20 = 9.55$,

B) a randomly generated digraph of 20 nodes with the average of the initial values of the nodes turning out to be equal to $q = 220/20 = 11$,

C) 1000 randomly generated digraphs of 20 nodes each where, for convenience, the initial quantized value of each node remained the same (for each one of the 1000 randomly generated digraphs); this means that the average of the nodes initial quantized values also remained equal to $q = 210/20 = 10.5$.

For each of the above cases we analyze the scenarios where each node $v_j \in \mathcal{V}$ (i) initially infuses in the network a randomly chosen offset $u_j \in [-100, -50]$ with randomly chosen offset adding steps $L_j \in [20, 40]$ and (ii) initially does not infuse any offset in the network i.e., $u_j = 0$ (which means that it does not want to preserve the privacy of its initial value). Note that for cases B) and C) the digraphs were randomly generated by creating, independently for each ordered pair (v_j, v_i) of two nodes v_j and v_i ($v_j \neq v_i$), a directed edge from node v_i to node v_j with $p = 0.3$.

A. Execution of Algorithm 1 over a Ring Digraph

In Fig. 1, we illustrate Algorithm 1 over a ring digraph of 20 nodes, where the average of the initial values of the nodes is equal to $q = 191/20 = 10.1$. We analyze the operation of our algorithm for the scenarios where each node $v_j \in \mathcal{V}$: (i) initially infuses in the network an offset $u_j \in [-100, -50]$ with offset adding steps $L_j \in [20, 40]$ (see top of Fig. 1) and (ii) initially does not infuse any offset in the network i.e., $u_j = 0$ (see bottom of Fig. 1). Here, we observe that for the case where each node v_j does not attempt to preserve the privacy of its initial value, Algorithm 1 is able to converge after 70 time steps. However, for the case where each node v_j wants to preserve the privacy of its initial value (by infusing the offset u_j with offset adding steps L_j as mentioned above) we observe that Algorithm 1 converges after 400 time steps, while being able to calculate the *exact* average of the initial values of the nodes without introducing any error due to the utilized privacy preserving strategy.

B. Execution of Algorithm 1 over a Random Digraph of 20 Nodes

In Fig. 2, we present the same cases as in Fig. 1, with the difference being that Algorithm 1 is executed over a random directed graph of 20 nodes, where the average of the initial values of the nodes is equal to $q = 205/20 = 10.25$. The main results do not change due to the network structure, which means that for the case where each node v_j preserves the privacy of its initial value we have that Algorithm 1 requires 450 time steps (in comparison to 95 time steps for the case where each node v_j does not preserve the privacy of its initial value) to converge to the *exact* average of the initial values.

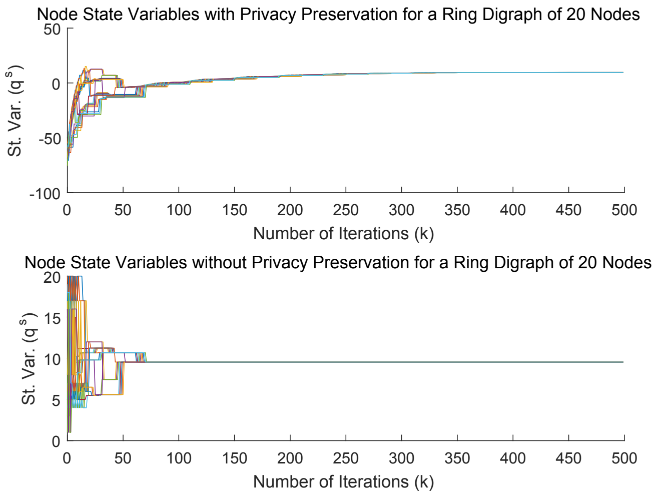

Fig. 1. Execution of Algorithm 1 for a ring digraph of 20 nodes. *Top figure:* Node state variables with privacy preservation plotted against the number of iterations. *Bottom Figure:* Node state variables without privacy preservation plotted against the number of iterations.

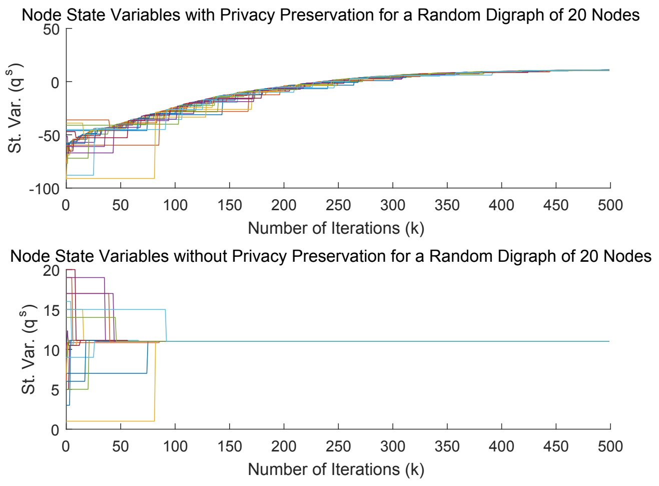

Fig. 2. Execution of Algorithm 1 for a random digraph of 20 nodes. *Top figure:* Node state variables with privacy preservation plotted against the number of iterations. *Bottom Figure:* Node state variables without privacy preservation plotted against the number of iterations.

C. Execution of Algorithm 1 Averaged over 1000 Random Digraphs of 20 Nodes

In Fig. 3 we present the same cases as in Fig. 1 and Fig. 2 with the difference being that they are averaged over 1000 randomly generated digraphs of 20 nodes. The initial quantized value of each node remained the same for each one of the 1000 randomly generated digraphs (which means that the average of the initial values of the nodes is equal to $q = 205/20 = 10.25$). However, the initial offset $u_j \in [-100, -50]$ and the offset adding steps $L_j \in [20, 40]$, of every node v_j, were randomly chosen for each digraph. We can see that the main results resemble those in Fig. 2, and Algorithm 1 converges after 450 time steps, while being

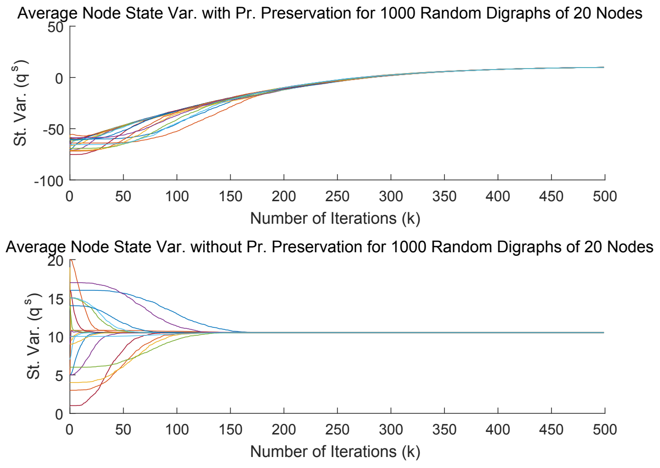

Fig. 3. Execution of Algorithm 1 averaged over 1000 random digraphs of 20 nodes. *Top figure:* Average values of node state variables with privacy preservation plotted against the number of iterations (averaged over 1000 random digraphs of 20 nodes). *Bottom Figure:* Average values of node state variables without privacy preservation plotted against the number of iterations (averaged over 1000 random digraphs of 20 nodes).

able to calculate the *exact* average of the initial values of the nodes without introducing any error due to our proposed privacy preserving strategy.

VI. Conclusions and Future Directions

A. Conclusions

In this paper, we proposed an event-triggered quantized privacy-preserving strategy which allows the agents of a multi-agent system to calculate the average of their initial values after a finite number of time steps without revealing to other agents their initial value. Specifically, our protocol consists of initially adding a negative quantized offset (determined by the node and the number of times certain conditions are satisfied) and then, for a predetermined number of events each node injects a set of positive offsets in the network such that the total offset is canceled out. We combine our proposed privacy preserving protocol with our quantized averaging algorithm in [9], and we show that it takes full advantage of the algorithm's finite time nature which means that consensus to the *exact* average of the initial values is achieved after a finite number of iterations which we explicitly calculated, while the error, introduced from the offset initially infused in the network by the nodes following the protocol, vanishes completely (a characteristic not present in [15], [16]). Finally, we have demonstrated the performance of our proposed protocol via illustrative examples.

B. Future Directions

The point-to-point communication protocol and the quantized nature of the packets used in this algorithm facilitate the use of cryptographic primitives for setting up secure channels and preventing eavesdropping, while harvesting the benefits of event-triggered and finite-time operation of the distributed privacy-preserving protocol proposed. These are directions that we plan to exploit further in future work.

References

[1] R. Olfati-Saber and R. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, September 2004.

[2] C. N. Hadjicostis, A. D. Domínguez-García, and T. Charalambous, "Distributed averaging and balancing in network systems, with applications to coordination and control," *Foundations and Trends® in Systems and Control*, vol. 5, no. 3–4, 2018.

[3] S. Sundaram and C. N. Hadjicostis, "Distributed function calculation and consensus using linear iterative strategies," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 650–660, May 2008.

[4] J. Tsitsiklis, "Problems in decentralized decision making and computation," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, Cambridge, 1984.

[5] A. B. Alexandru, M. S. Darup, and G. J. Pappas, "Encrypted cooperative control revisited," *Proceedings of the IEEE 58^{th} Conference on Decision and Control (CDC)*, pp. 7196–7202, 2019.

[6] T. C. Aysal, M. Coates, and M. Rabbat, "Distributed average consensus using probabilistic quantization," *IEEE/SP Workshop on Statistical Signal Processing*, pp. 640–644, 2007.

[7] J. Lavaei and R. M. Murray, "Quantized consensus by means of Gossip algorithm," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 19–32, January 2012.

[8] A. Kashyap, T. Basar, and R. Srikant, "Quantized consensus," *Automatica*, vol. 43, no. 7, pp. 1192–1203, 2007.

[9] A. I. Rikos and C. N. Hadjicostis, "Distributed average consensus under quantized communication via event-triggered mass summation," *Proceedings of the IEEE 57^{th} Conference on Decision and Control (CDC)*, pp. 894–899, 2018.

[10] M. E. Chamie, J. Liu, and T. Basar, "Design and analysis of distributed averaging with quantized communication," *IEEE Transactions on Automatic Control*, vol. 61, no. 12, pp. 3870–3884, December 2016.

[11] K. Cai and H. Ishii, "Quantized consensus and averaging on gossip digraphs," *IEEE Transactions on Automatic Control*, vol. 56, no. 9, pp. 2087–2100, September 2011.

[12] M. Kefayati, M. S. Talebi, B. H. Khalaj, and H. R. Rabiee, "Secure consensus averaging in sensor networks using random offsets," in *IEEE International Conference on Telecommunications and Malaysia International Conference on Communications*, May 2007, pp. 556–560.

[13] J. Cortés, G. E. Dullerud, S. Han, J. L. Ny, S. Mitra, and G. J. Pappas, "Differential privacy in control and network systems," *Proceedings of the IEEE 55^{th} Conference on Decision and Control (CDC)*, pp. 4252–4272, 2016.

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

[15] N. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," *Proceedings of the European Control Conference (ECC)*, pp. 760–765, 2013.

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

[17] N. Gupta, J. Katz, and N. Chopra, "Privacy in distributed average consensus," *IFAC-PapersOnLine*, vol. 50, pp. 9515–9520, 2017.

[18] C. N. Hadjicostis, "Privary preserving distributed average consensus via homomorphic encryption," in *IEEE Conference on Decision and Control (CDC)*, Dec. 2018, pp. 1258–1263.

[19] M. Ruan, H. Gao, and Y. Wang, "Secure and privacy-preserving consensus," *IEEE Transactions on Automatic Control*, vol. 64, no. 10, pp. 4035–4049, Oct. 2019.

[20] C. N. Hadjicostis and A. D. Dominguez-Garcia, "Privacy-preserving distributed averaging via homomorphically encrypted ratio consensus," *IEEE Transactions on Automatic Control*, vol. 65, no. 9, pp. 3887–3894, September 2020.

[21] T. Charalambous, N. E. Manitara, and C. N. Hadjicostis, "Privacy-preserving average consensus over digraphs in the presence of time delays," *Proceedings of the 57^{th} Annual Allerton Conference on Communication, Control, and Computing*, 2019.

[22] A. I. Rikos and C. N. Hadjicostis, "Event-triggered quantized average consensus via ratios of accumulated values," *IEEE Transactions on Automatic Control*, April 2020 (early access).

[23] A. I. Rikos, T. Charalambous, and C. N. Hadjicostis, "Distributed weight balancing over digraphs," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 2, pp. 190–201, June 2014.