


# Enabling Privacy-Preservation in Decentralized Optimization

Chunlei Zhang and Yongqiang Wang 

**Abstract**—Decentralized optimization is crucial for the design, deployment, and functionality of many distributed systems. In this paper, we address the problem of privacy-preservation in decentralized optimization. In most existing decentralized optimization approaches, participating agents exchange and disclose intermediate estimate values to neighboring agents explicitly, which may not be desirable when the estimates contain sensitive information of individual agents. The problem is more acute when adversaries exist which try to steal information from other participating agents. To address this issue, we propose a novel approach that enables privacy-preservation in decentralized optimization by combining partially homomorphic cryptography with the unique properties of the decentralized optimization. We show that cryptographic techniques can be incorporated in a fully decentralized manner to enable privacy-preservation in decentralized optimization in the absence of any third party or aggregator. This is significant in that to our knowledge, all existing cryptographic-based optimization approaches rely on the assistance of a third-party or aggregator in order to protect the privacy of all parties. The approach is also applicable to the average consensus problem, which is finding broad applications in fields as diverse as robotic networks, distributed computing, and power systems. Numerical simulations confirm the effectiveness and low computational complexity of the proposed approach.

**Index Terms**—Decentralized optimization, partially homomorphic encryption, privacy preservation, projected sub-gradient algorithm.

## I. INTRODUCTION

THE PROBLEM of decentralized optimization has attracted remarkable attention in recent years due to its wide applications in various domains, ranging from multiagent systems [1]–[3]; machine learning [4]–[6]; statistics [7], [8]; communications; and networking [1]–[3], [9], [10]; to power grids [11], [12]. In these applications, data are collected and processed in a decentralized and cooperative manner among multiple agents. Such a decentralized manner of processing provides several key advantages over its conventional computing-server-based counterpart. For example, it leads to enhanced scalability and

flexibility, and brings higher robustness to the problems of network traffic bottleneck and single point of failure.

Typical decentralized optimization algorithms include distributed (sub)gradient-based algorithms [1]–[3]; augmented Lagrangian methods (ALM) [13], [14]; and the alternating direction method of multipliers (ADMM) as well as its variants [13]–[17], etc. In (sub)gradient-based solutions, (sub)gradient computations and averaging among neighbors are conducted iteratively to achieve convergence to the minimum. In augmented Lagrangian and ADMM-based solutions, the iterative Lagrangian minimization is employed, which, coupled with dual variable update, guarantees that all agents agree on the minimization solution.

However, most of the aforementioned decentralized approaches require agents to exchange and disclose their estimates (states) explicitly to neighboring agents in every iteration [1]–[3], [13]–[17]. This brings about serious privacy concerns in many practical applications [18]. For example, in projection-based source localization, a node may infer the exact position of a neighboring node using three intermediate state values from this node [10], which is undesirable when agents do not want to reveal their position information [19]. In the rendezvous problem, where a group of individuals wants to meet at an agreed time and place [20], exchanging explicit states can reveal their initial locations to neighboring nodes, which may not be desirable [21]. Other examples include the agreement problem [22], where a group of individuals wants to reach consensus on a subject without leaking their individual opinions to others [21], and the regression problem [7], [8], where individual agents' training data may contain sensitive information (e.g., salary, medical record) and should be kept private. In addition, exchanging explicit states without encryption is susceptible to eavesdroppers, who try to intercept and steal information from exchanged messages.

To enable privacy-preservation in decentralized optimization, one commonly used approach is differential privacy [6], [23]–[27], which adds carefully designed noise to exchanged estimates to cover sensitive information. However, the added noise also unavoidably compromises the accuracy of optimization results, leading to a tradeoff between enabled privacy level and achievable computation accuracy [23], [24]. Observability-based design has also been proposed for privacy-preservation in linear multiagent networks [28], [29]. By properly designing the weights for the communication graph, the agents' information will not be revealed to non-neighboring agents. However, this approach cannot protect the privacy of the direct neighbors of

Manuscript received February 1, 2018; revised June 17, 2018; accepted September 2, 2018. Date of publication October 1, 2018; date of current version May 28, 2019. This work was supported by the National Science Foundation under Grant 1824014 and Grant 1738902. Recommended by Associate Editor Anna Scaglione. (Corresponding author: Yongqiang Wang.)

The authors are with the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC 29634 USA (e-mail: chunleiz@clemson.edu; yongqiw@clemson.edu).

Digital Object Identifier 10.1109/TCNS.2018.2873152

compromised agents and it is susceptible to external eavesdroppers. Another approach to enabling privacy-preservation is encryption. However, despite successful applications in cloud-based control and optimization [30], [31], conventional cryptographic techniques cannot be applied directly in a completely decentralized setting without the assistance of aggregators/third parties. (Note that traditional secure multiparty computation schemes such as fully homomorphic encryption [32] and Yao's garbled circuit [33] are too heavy to be computationally feasible for real-time optimization [18].)

To enable privacy-preservation without compromising the optimality of the solution, we propose a novel approach that enables privacy-preservation in decentralized optimization through incorporating partially homomorphic cryptography in existing optimization algorithms. We show that by employing the unique convergence property of decentralized optimization approaches, cryptographic techniques can be incorporated in a fully decentralized manner to enable privacy-preservation in decentralized optimization in the absence of any third party or aggregator. We consider two types of adversaries: *Honest-but-curious adversaries* are agents who follow all protocol steps correctly, but are curious and collect all intermediate and input/output data in an attempt to learn some information about other participating agents [34], [35]. In fact, any agent participating in the decentralized optimization can be honest-but-curious. *External eavesdroppers* are adversaries who steal information through wiretapping all communication channels and intercepting exchanged messages between agents.

**Contribution:** The main contribution of this paper is a novel approach that enables privacy-preservation in decentralized optimization. By combining partially homomorphic cryptography with the unique properties of the decentralized optimization, we show that cryptographic techniques can be incorporated in a fully decentralized manner to enable privacy-preservation in the decentralized optimization in the absence of any third party or aggregator. This is significant in that, to the best of our knowledge, all existing cryptographic-based optimization approaches rely on the assistance of a third-party or aggregator to protect the privacy of all parties.

The rest of this paper is organized as follows: Section II reviews the partially homomorphic cryptography, particularly the Paillier cryptosystem, and the constrained decentralized optimization problem. Based on the Paillier cryptosystem, a completely decentralized privacy-preserving approach is proposed in Section III. Rigorous analysis of the guaranteed privacy under the approach is addressed in Section IV and its implementation details are discussed in Section V. Section VI discusses its application to the average consensus problem. Numerical simulation results are given in Section VII to confirm the effectiveness and computational efficiency of the proposed approach. In the end, we draw conclusions in Section VIII.

## II. BACKGROUND

In this section, we briefly review the Paillier cryptosystem and the constrained decentralized optimization problem.

### A. Paillier Cryptosystem

Our method to enable privacy-preservation is to enable decentralized combination of the Paillier cryptosystem with optimization. The Paillier cryptosystem [36] is a public-key cryptosystem which uses a pair of keys: a public key and a private key. The public key can be disseminated publicly and used by any person to encrypt a message, but the message can only be decrypted by the private key. The Paillier cryptosystem includes three algorithms, which are as follows.

---

#### Paillier cryptosystem

---

##### Key generation:

- 1) Choose two large prime numbers  $p$  and  $q$  of equal bit-length and compute  $n = pq$ .
- 2) Let  $g = n + 1$ .
- 3) Let  $\lambda = \phi(n) = (p - 1)(q - 1)$ , where  $\phi(\cdot)$  is the Euler's totient function.
- 4) Let  $\mu = \phi(n)^{-1} \bmod n$  which is the modular multiplicative inverse of  $\phi(n)$ .
- 5) The public key  $k_p$  for encryption is  $(n, g)$ .
- 6) The private key  $k_s$  for decryption is  $(\lambda, \mu)$ .

##### Encryption ( $c = \mathcal{E}(m)$ ):

Recall the definitions of  $\mathbb{Z}_n = \{z | z \in \mathbb{Z}, 0 \leq z < n\}$  and  $\mathbb{Z}_n^* = \{z | z \in \mathbb{Z}, 0 \leq z < n, \gcd(z, n) = 1\}$ .

- 1) Choose a random  $r \in \mathbb{Z}_n^*$ .
- 2) The ciphertext is given by  $c = g^m \cdot r^n \bmod n^2$ , where  $m \in \mathbb{Z}_n, c \in \mathbb{Z}_{n^2}^*$ .

##### Decryption ( $m = D(c)$ ):

- 1) Define the integer division function  $L(\mu) = \frac{\mu-1}{n}$ .
  - 2) The plaintext is  $m = L(c^\lambda \bmod n^2) \cdot \mu \bmod n$ .
- 

A notable feature of the Paillier cryptosystem is that it is additively homomorphic, that is, the ciphertext of  $m_1 + m_2$  can be obtained from the ciphertexts of  $m_1$  and  $m_2$  directly when  $0 \leq m_1 + m_2 < n$  holds

$$\mathcal{E}(m_1, r_1) \cdot \mathcal{E}(m_2, r_2) = \mathcal{E}(m_1 + m_2, r_1 r_2) \quad (1)$$

$$\mathcal{E}(m)^k = \mathcal{E}(km), \quad k \in \mathbb{Z}^+. \quad (2)$$

Due to the existence of random  $r$ , the Paillier cryptosystem is resistant to the dictionary attack [35]. Since  $r_1$  and  $r_2$  play no role in the decryption process, (1) can be simplified as follows for the sake of readability:

$$\mathcal{E}(m_1) \cdot \mathcal{E}(m_2) = \mathcal{E}(m_1 + m_2). \quad (3)$$

### B. Constrained Decentralized Optimization

An important class of decentralized optimization problems is to minimize an objective function that is the sum of  $N$  convex functions private to  $N$  individual agents [1]–[3], [23]. Such optimization problems have found applications in domains as diverse as source localization in sensor networks [10]; spectrum sensing in cognitive networks [37]; support vector machine in machine learning [4]; cooperative control [3]; data regression in statistics [7], [8]; and the monitoring of power systems [11], [12]. Here, we consider such a decentralized problem in which

$N$  agents cooperatively solve constrained optimization over a time-varying network topology.

**1) Optimization Model:** The problem of constrained decentralized optimization can be formulated in the following form:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \sum_{i=1}^N f_i(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X} \end{aligned} \quad (4)$$

where  $\mathcal{X} \subseteq \mathbb{R}^D$  is a closed convex set common to all agents, and function  $f_i : \mathbb{R}^D \rightarrow \mathbb{R}$  is the local objective function private to (only known to) agent  $i$ . Note that here  $f_i$  is convex but not necessarily differentiable everywhere.

**Remark 1:** In practical applications, the solutions to an unconstrained optimization problem should be finite. Therefore, we can easily reformulate an unconstrained optimization problem as a constrained optimization problem by setting a large enough constraint set  $\mathcal{X}$ .

The constrained optimization problem has been widely used. Here, we give some typical examples.

**Example 1:** In source localization,  $N$  sensors cooperatively localize the position of a source using noisy range measurements with respect to the source from distributedly deployed sensors. When the source is located in the convex hull of deployed sensors, a classical approach is to turn the localization problem into the problem of finding a point common to a set of closed convex sets  $X_i$  [10], which can be formulated as follows:

$$\min_{\mathbf{x}} \quad \frac{1}{2} \sum_{i=1}^N \|\mathbf{x} - P_{X_i}[\mathbf{x}]\|^2$$

where  $i = 1, 2, \dots, N$  is the index of deployed sensors and

$$X_i = \{\mathbf{x} \in \mathbb{R}^D \mid \|\mathbf{x} - \mathbf{p}_i\|^2 \leq r_i^2\}.$$

Here,  $\mathbf{p}_i$  is the position of sensor  $i$ , and  $r_i$  is the range measurement of the source with respect to sensor  $i$ .  $P_{X_i}[\cdot]$  denotes the projection operation onto the set  $X_i$ , that is,  $P_{X_i}[\mathbf{r}] = \operatorname{argmin}_{\mathbf{y} \in X_i} \|\mathbf{y} - \mathbf{r}\|$ .

**Example 2:** In cooperative control, a typical problem is to guarantee that a group of agents reaches a common decision or agreement formulated as follows [3], [23]:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \sum_{i=1}^N \frac{1}{2} \|\mathbf{x} - \boldsymbol{\theta}_i\|^2 \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X} \end{aligned} \quad (5)$$

where  $\boldsymbol{\theta}_i$  ( $i = 1, 2, \dots, N$ ) are known parameters, and  $\mathbf{x}$  is the unknown vector.

**Example 3:** In statistical analysis, many problems have the constrained optimization formulation in (4):

Simple linear regression

$$\min_{\gamma_0, \gamma_1} \quad \sum_{i=1}^N \|y_i - \gamma_0 - \gamma_1 x_i\|^2$$

where  $y_i$  and  $x_i$  are known parameters,  $\gamma_0$  and  $\gamma_1$  are unknown variables.

Logistic regression [8]

$$\min_{\mathbf{x}, c} \quad \frac{1}{N} \sum_{i=1}^N \log(1 + \exp(-b_i(\mathbf{x}^T \mathbf{a}_i + c))) + \tau \|\mathbf{x}\|_1$$

where  $b_i$ ,  $\mathbf{a}_i$ , and  $\tau$  are known parameters,  $c$  and  $\mathbf{x}$  are unknown variables.

Other examples can be found in power systems [12], compressive spectrum sensing [37], and machine learning [4]–[6].

**2) Decentralized Algorithm [3]:** A decentralized solution to the constrained optimization problem (4) is the projected subgradient algorithm in [3]. In the projected subgradient algorithm, each agent  $i$  updates its estimate by first fusing the estimates from its neighbors, then taking a subgradient step, and finally projecting to the closed convex set  $\mathcal{X}$ , that is,

$$\mathbf{v}_i(k) = \sum_{j=1}^N a_{ij}(k) \mathbf{x}_j(k) \quad (6)$$

$$\mathbf{x}_i(k+1) = P_{\mathcal{X}}[\mathbf{v}_i(k) - \alpha_k d_i(k)]. \quad (7)$$

Here,  $a_{ij}$  is a non-negative weight assigned to  $\mathbf{x}_j$  by agent  $i$ ,  $P_{\mathcal{X}}[\cdot]$  denotes the projection operation onto the set  $\mathcal{X}$ , that is,  $P_{\mathcal{X}}[\mathbf{r}] = \operatorname{argmin}_{\mathbf{y} \in \mathcal{X}} \|\mathbf{y} - \mathbf{r}\|$ ,  $\alpha_k > 0$  is a stepsize, and  $d_i(k)$  is a subgradient of  $f_i$  at  $\mathbf{x} = \mathbf{v}_i(k)$ .

**3) Convergence Analysis:** According to [3], the algorithm (6)–(7) is guaranteed to converge under the following three assumptions when the stepsize  $\alpha_k$  satisfies  $\sum_k \alpha_k = \infty$  and  $\sum_k \alpha_k^2 < \infty$ :

**Assumption 1:** There exists a scalar  $0 < \eta < 1$  such that for all  $k \geq 0$  and  $i = 1, 2, \dots, N$ :

- 1)  $a_{ii}(k) \geq \eta$ ;
- 2)  $a_{ij}(k) \geq \eta$  for all  $j \in \mathcal{N}_i(k)$ . Here,  $\mathcal{N}_i(k)$  denotes the set of all neighboring agents of  $i$  at time instant  $k$ ;
- 3)  $a_{ij}(k) = 0$  for all  $j \notin \mathcal{N}_i(k) \cup \{i\}$ ;
- 4)  $\sum_{j=1}^N a_{ij}(k) = 1$ ;
- 5)  $a_{ij}(k) = a_{ji}(k)$ .

Note that the  $N$  agents may form a time-varying network, that is, the neighbors of agent  $i$  may change with time.

**Assumption 2:** The graph  $(V, E_\infty)$  is strongly connected, where

$$E_\infty = \{e_{ij} \mid e_{ij} \in E_k \text{ for infinitely many indices } k\}.$$

Here,  $E_k$  denotes the set of communication links (undirected edges) at time instant  $k$ , and  $e_{ij} \in E_k$  denotes that agents  $i$  and  $j$  are neighbors (directly connected) at time instant  $k$ . Note that here we denote a communication link as  $e_{ij}$  if  $i < j$  is true or as  $e_{ji}$  otherwise.

**Assumption 3:** There exists an integer  $B \geq 1$  such that for each  $e_{ij} \in E_\infty$ , agent  $j$  sends its estimate to agent  $i$  at least once every  $B$  consecutive time slots.

A typical way to choose  $\alpha_k$  is  $\alpha_k = \frac{T_1}{k+T_2}$ , where  $0 < T_1 < \infty$  and  $0 < T_2 < \infty$ .

### III. PRIVACY-PRESERVING DECENTRALIZED OPTIMIZATION

In the algorithm (6)–(7) for the constrained optimization problem (4), to reach consensus on the final optimal solution, agents exchange and disclose estimates (states) explicitly in each



iteration to neighboring agents, which leads to privacy breaches. Such information exchange is also vulnerable to eavesdropping attacks that aim to steal the information by intercepting exchanged messages. In this section, we introduce a completely decentralized and third-party free approach to enable privacy-preservation in the constrained decentralized optimization problem. More specifically, we will propose an interaction protocol that enables easy integration of partially homomorphic cryptography with algorithm (6)–(7) to enable privacy-preservation without the assistance of any aggregator or third party.

To this end, we first rewrite (6) as follows:

$$\mathbf{v}_i(k) = \mathbf{x}_i(k) + \sum_{j=1, j \neq i}^N a_{ij}(k)(\mathbf{x}_j(k) - \mathbf{x}_i(k)). \quad (8)$$

The key idea of the privacy-preserving mechanism is to construct  $a_{ij}(k)$ ,  $i \neq j$ ,  $j \in \mathcal{N}_i(k)$  as the product of two random positive numbers, that is,  $a_{ij}(k) = b_{i \rightarrow j}(k) \times b_{j \rightarrow i}(k) = a_{ji}(k)$ , with  $b_{i \rightarrow j}(k)$  generated by and only known to agent  $i$ , and  $b_{j \rightarrow i}(k)$  generated by and only known to agent  $j$  when agent  $i$  and agent  $j$  can communicate with each other at time instant  $k$ . It is worth noting that if agent  $i$  and agent  $j$  cannot communicate with each other at time instant  $k$ ,  $a_{ij}(k)$  and  $a_{ji}(k)$  are set to 0 directly. Next, we give in detail the privacy-preserving solution to the constrained-decentralized-optimization problem in (4).

---

#### Algorithm 1

---

**Initial Setup:** Each agent initializes  $\mathbf{x}_i(0)$ .

**Input:**  $\mathbf{x}_i(k)$

**Output:**  $\mathbf{x}_i(k+1)$

1) Agent  $i$  encrypts  $-\mathbf{x}_i(k)$  with its public key  $k_{pi}$ :

$$\mathbf{x}_i(k) \rightarrow \mathcal{E}_i(-\mathbf{x}_i(k)).$$

Here, the subscript  $i$  denotes encryption using the public key of agent  $i$ .

2) Agent  $i$  sends  $\mathcal{E}_i(-\mathbf{x}_i(k))$  and its public key  $k_{pi}$  to its neighboring agents.

3) Agent  $j \in \mathcal{N}_i(k)$  encrypts  $\mathbf{x}_j(k)$  with agent  $i$ 's public key  $k_{pi}$ :

$$\mathbf{x}_j(k) \rightarrow \mathcal{E}_i(\mathbf{x}_j(k)).$$

4) Agent  $j \in \mathcal{N}_i(k)$  computes the difference directly in ciphertext:

$$\mathcal{E}_i(\mathbf{x}_j(k) - \mathbf{x}_i(k)) = \mathcal{E}_i(\mathbf{x}_j(k)) \cdot \mathcal{E}_i(-\mathbf{x}_i(k)).$$

5) Agent  $j \in \mathcal{N}_i(k)$  computes the  $b_{j \rightarrow i}(k)$ -weighted difference in ciphertext:

$$\mathcal{E}_i(b_{j \rightarrow i}(k)(\mathbf{x}_j(k) - \mathbf{x}_i(k))) = (\mathcal{E}_i(\mathbf{x}_j(k) - \mathbf{x}_i(k)))^{b_{j \rightarrow i}(k)}.$$

6) Agent  $j \in \mathcal{N}_i(k)$  sends  $\mathcal{E}_i(b_{j \rightarrow i}(k)(\mathbf{x}_j(k) - \mathbf{x}_i(k)))$  back to agent  $i$ .

7) Agent  $i$  decrypts the message received from  $j$  with its private key  $k_{si}$  and multiplies the result with  $b_{i \rightarrow j}(k)$  to get  $a_{ij}(k)(\mathbf{x}_j(k) - \mathbf{x}_i(k))$ .

8) Computing (8), agent  $i$  obtains  $\mathbf{v}_i(k)$ .

9) Computing (7), agent  $i$  obtains  $\mathbf{x}_i(k+1)$ .

10) Each agent updates  $b_{i \rightarrow j}(k)$  to  $b_{i \rightarrow j}(k+1)$  and sets  $k = k+1$ .

---

Several remarks are in order as follows.

- 1) Agent  $i$ 's state  $\mathbf{x}_i(k)$  and its intermediate communication data  $b_{j \rightarrow i}(k)(\mathbf{x}_j(k) - \mathbf{x}_i(k))$  will not be revealed to outside eavesdroppers, since they are encrypted.
- 2) The state of agent  $j \in \mathcal{N}_i(k)$  will not be revealed to agent  $i$ , because the decrypted message obtained by agent  $i$  is  $b_{j \rightarrow i}(k)(\mathbf{x}_j(k) - \mathbf{x}_i(k))$  with  $b_{j \rightarrow i}(k)$  only known to agent  $j$  and varying in each iteration.
- 3) We encrypt  $\mathcal{E}_i(-\mathbf{x}_i(k))$  because it is much easier to compute addition in ciphertext. The issue regarding encryption of signed values using Paillier will be addressed in Section V.
- 4) Paillier encryption cannot be performed on vectors directly. For vector messages  $\mathbf{x}_i(k) \in \mathbb{R}^D$ , each element of the vector has to be encrypted separately. For notational convenience, we still denote it in the same way as scalars, for example,  $\mathcal{E}_i(-\mathbf{x}_i(k))$ .
- 5) Paillier cryptosystem only works for integers, so additional steps have to be taken to convert real values in optimization to integers. This may lead to quantization errors. A common workaround is to scale a real value before quantization, as discussed in detail in Section V.
- 6) The proposed approach requires agents to update synchronously and it may fail to converge if applied to asynchronous networks directly.

In Algorithm 1, steps 1)–7) constitute the core of our approach to incorporating Paillier cryptosystem in privacy-preserving optimization in a fully decentralized manner. In fact, it can also be seen that the only net effect of our privacy-preserving mechanism is random and time-varying coefficients  $a_{ij}(k)$  in (8). Next, we show that the privacy-preserving mechanism does not affect the convergence of the algorithm to its optimal solution.

**Theorem 1:** The convergence of the privacy-preserving Algorithm 1 is guaranteed if Assumptions 2 and 3 hold, all  $b_{i \rightarrow j}(k)$  are randomly chosen from  $[\sqrt{\eta}, \sqrt{\frac{1-\eta}{N-1}}]$  with  $0 < \eta < 1/N$ , and the stepsize  $\alpha_k$  satisfies  $\sum_k \alpha_k = \infty$  and  $\sum_k \alpha_k^2 < \infty$ .

**Proof:** We show that Assumption 1 will be met if all  $b_{i \rightarrow j}(k)$  are randomly chosen from  $[\sqrt{\eta}, \sqrt{\frac{1-\eta}{N-1}}]$  with  $0 < \eta < 1/N$ . First, since  $a_{ij}(k)$  and  $a_{ji}(k)$  are set to 0 directly when  $j \notin \mathcal{N}_i(k)$ , and to  $a_{ij}(k) = b_{i \rightarrow j}(k)b_{j \rightarrow i}(k) = b_{j \rightarrow i}(k)b_{i \rightarrow j}(k) = a_{ji}(k)$  when  $j \in \mathcal{N}_i(k)$ , it is clear that conditions 3) and 5) in Assumption 1 are satisfied. (Note that  $b_{i \rightarrow j}(k)$  and  $b_{j \rightarrow i}(k)$  are unknown to agents  $j$  and  $i$ , respectively, so  $a_{ij}(k)$  and  $a_{ji}(k)$  are equal but unknown to both agent  $i$  and agent  $j$ .) Next, rewriting (8) as

$$\mathbf{v}_i(k) = \left(1 - \sum_{j=1, j \neq i}^N a_{ij}(k)\right) \mathbf{x}_i(k) + \sum_{j=1, j \neq i}^N a_{ij}(k) \mathbf{x}_j(k)$$

we have that  $a_{ii}(k) = 1 - \sum_{j=1, j \neq i}^N a_{ij}(k)$  is always true under the update rule and, hence, the condition 4) of Assumption 1 is satisfied. When  $j \in \mathcal{N}_i(k)$ ,  $b_{i \rightarrow j}(k)$  and  $b_{j \rightarrow i}(k)$  are chosen from the interval  $[\sqrt{\eta}, \sqrt{\frac{1-\eta}{N-1}}]$ , so we have  $\eta \leq a_{ij}(k) = b_{i \rightarrow j}(k)b_{j \rightarrow i}(k) \leq \frac{1-\eta}{N-1}$  and further  $a_{ii}(k) =$

$1 - \sum_{j=1, j \neq i}^N a_{ij}(k) \geq 1 - (N-1) \times \sqrt{\frac{1-\eta}{N-1}} \times \sqrt{\frac{1-\eta}{N-1}} = \eta$ , that is,  $a_{ij}(k) \geq \eta$  and  $a_{ii}(k) \geq \eta$  for  $0 < \eta < 1/N$ . Therefore, conditions 1) and 2) in Assumption 1 are also satisfied. So, we have Theorem 1. ■

**Remark 2:** It is worth noting that in Algorithm 1, if the state of agent  $i$  and agent  $j$ , that is,  $\mathbf{x}_i(k)$  and  $\mathbf{x}_j(k)$  happen to be equal to each other, then agent  $i$  will be able to know this based on the fact that the obtained  $a_{ij}(k)(\mathbf{x}_j(k) - \mathbf{x}_i(k))$  in step 7) is zero. This fact that agent  $i$  and agent  $j$  being equal is inferable from zero  $a_{ij}(k)(\mathbf{x}_j(k) - \mathbf{x}_i(k))$  can be covered by allowing  $b_{i \rightarrow j}(k)$  and  $b_{j \rightarrow i}(k)$  to be set to zero randomly. In this case, agents  $i$  and  $j$ 's mutual link is intentionally abandoned and they are not neighbors any more at this specific time instant. Because they are not neighbors at this time instant, having  $a_{ij}(k) = 0$  in this case is still consistent with conditions 2) and 3) in Assumption 1. Therefore, following a similar derivation as Theorem 1, we can easily get that the network will converge to the optimal solution as long as there exists an integer  $B \geq 1$  such that for each  $e_{ij} \in E_\infty$ ,  $b_{i \rightarrow j}(k)$  and  $b_{j \rightarrow i}(k)$  are both nonzero for at least once every  $B$  consecutive time slots.

**Remark 3:** According to [3], when the weights  $a_{ij}(k)$  are identical and time-invariant, that is, all equal to  $1/N$ , the convergence rate of algorithm (6)–(7) is geometric, that is, there exists a  $\lambda \in (0, 1)$  and some positive constant  $C$  such that  $\|\mathbf{x}(k) - \mathbf{x}^*\| \leq C\lambda^k$  holds for all  $k$  [38]. When  $a_{ij}(k)$  are time-varying, according to [3], the convergence rate is mainly determined by the rate at which the transition matrix  $\Phi(k, s) = A(s)A(s+1)\dots A(k-1)A(k)$  converges to  $\frac{1}{N}\mathbf{1}^T\mathbf{1}$ , where  $\mathbf{1}$  represents a column vector of all ones and the  $(i, j)$ th entry of  $A(k)$  is equal to  $a_{ij}(k)$ . So, next we analyze the influence of the privacy-preserving mechanism on the convergence rate by analyzing the influence of random  $a_{ij}(k)$  on the convergence of  $\Phi(k, s)$ . Note that the convergence of the transition matrix  $\Phi(k, s)$  is established in [3]

$$\left| [\Phi(k, s)]_{ij}^j - \frac{1}{N} \right| \leq 2 \frac{1 + \eta^{-B_0}}{1 - \eta^{B_0}} (1 - \eta^{B_0})^{(k-s)/B_0}$$

where  $B_0 = (N-1)B$  with  $B$  defined in Assumption 3 and  $N$  is the total number of agents. It is clear that a greater  $\eta$  leads to higher convergence speed. However, from Theorem 1, we know that all  $a_{ij}(k)$  should be randomly chosen in  $[\eta, \frac{1-\eta}{N-1}]$  and, hence, to provide stronger privacy protection,  $\eta$  should be set smaller to ensure that weights  $a_{ij}(k)$  can randomly vary in a larger range. Therefore, there is a tradeoff in choosing  $\eta$ : a smaller  $\eta$  leads to stronger privacy protection but at lower convergence speed.

#### IV. PRIVACY ANALYSIS

In this section, we rigorously prove that each agent's private information, for example, immediate estimate (state)  $\mathbf{x}_j(k)$  and private local objective function  $f_j$ , cannot be inferred by honest-but-curious adversaries and external eavesdroppers, which are commonly used attack models in privacy studies [34], [35] (cf., definitions in Section I). It is worth noting that the form of each

agent's local objective function  $f_j$  can also be totally inaccessible to others, that is, whether it is a quadratic, exponential, or other forms of convex functions is only known to an agent itself. In addition, the distribution of  $b_{j \rightarrow i}$  is private to agent  $j$  itself.

As indicated in Section III, our approach in Algorithm 1 guarantees that the state information is not leaked to any neighbors in one iteration. However, would some information get leaked to an honest-but-curious adversary over time? More specifically, if an honest-but-curious adversary observes carefully its communications with neighbors over several steps, can it put together all of the received information to infer its neighbor's state?

We can rigorously prove that an honest-but-curious adversary cannot infer the states of its neighbors even by collecting samples from multiple steps.

**Theorem 2:** In Algorithm 1, an agent  $j$ 's state  $\mathbf{x}_j(k)$  cannot be inferred by an honest-but-curious neighboring agent  $i$ .

**Proof:** Suppose that an honest-but-curious agent  $i$  collects information from  $K$  iterations to infer the information of a neighboring agent  $j$ . From the perspective of the adversary agent  $i$ , the measurement (corresponding to neighboring agent  $j$ ) seen in each iteration  $k$  is  $\mathbf{y}(k) = b_{i \rightarrow j}(k)b_{j \rightarrow i}(k)(\mathbf{x}_j(k) - \mathbf{x}_i(k))$  ( $k = 0, 1, 2, \dots, K$ ), that is, based on received information, the adversary agent  $i$  can establish  $(K+1)D$  equations with respect to the state of agent  $j$

$$\begin{cases} \mathbf{y}(0) = b_{i \rightarrow j}(0)b_{j \rightarrow i}(0)(\mathbf{x}_j(0) - \mathbf{x}_i(0)) \\ \mathbf{y}(1) = b_{i \rightarrow j}(1)b_{j \rightarrow i}(1)(\mathbf{x}_j(1) - \mathbf{x}_i(1)) \\ \vdots \\ \mathbf{y}(K) = b_{i \rightarrow j}(K)b_{j \rightarrow i}(K)(\mathbf{x}_j(K) - \mathbf{x}_i(K)). \end{cases} \quad (9)$$

To the adversary agent  $i$ , in the system of equations (9),  $\mathbf{y}(k)$ ,  $b_{i \rightarrow j}(k)$ ,  $\mathbf{x}_i(k)$  ( $k = 0, 1, 2, \dots, K$ ) are known, but  $\mathbf{x}_j(k)$ ,  $b_{j \rightarrow i}(k)$  ( $k = 0, 1, 2, \dots, K$ ) are unknown. So, the above system of  $(K+1)D$  equations contains  $(K+1)D + K + 1$  unknown variables. It is clear that the adversary agent  $i$  cannot solve the system of equations to infer the unknowns  $\mathbf{x}_j(k)$  or  $b_{j \rightarrow i}(k)$  ( $k = 0, 1, 2, \dots, K$ ) of agent  $j$ . ■

Based on a similar line of reasoning, we can obtain that an honest-but-curious agent  $i$  cannot infer the private information of function  $f_j$  from a neighboring agent  $j$  either.

**Corollary 1:** In Algorithm 1, agent  $j$ 's private local function  $f_j$  will not be revealed to an honest-but-curious agent  $i$ .

**Proof:** Suppose that an honest-but-curious agent  $i$  collects information from  $K$  iterations to infer the function  $f_j$  of a neighboring agent  $j$ . The adversary agent  $i$  can establish  $KD$  equations with respect to  $f_j$  by making use of the fact that the update rule (7) is publicly known, that is,

$$\begin{cases} \mathbf{x}_j(1) = P_{\mathcal{X}}[\mathbf{v}_j(0) - \alpha_0 d_j(0)] \\ \mathbf{x}_j(2) = P_{\mathcal{X}}[\mathbf{v}_j(1) - \alpha_1 d_j(1)] \\ \vdots \\ \mathbf{x}_j(K) = P_{\mathcal{X}}[\mathbf{v}_j(K-1) - \alpha_{K-1} d_j(K-1)]. \end{cases} \quad (10)$$

We discuss (10) under two cases. *Case 1:* When agent  $j$  has more than one neighbor, the values of  $\mathbf{v}_j(k)$ ,  $d_j(k)$  ( $k = 0, 1, 2, \dots, K-1$ ), and  $\mathbf{x}_j(k)$  ( $k = 1, 2, \dots, K$ ) are unknown

to adversary agent  $i$ . So, the above system of  $KD$  equations contains  $3KD$  unknown variables; *Case 2*: When agent  $j$  has agent  $i$  as its only neighbor, then another set of equations  $v_j(k) = x_j(k) - y(k)$  ( $k = 1, 2, \dots, K-1$ ) is accessible to agent  $i$  and, hence, in combination with the equations in (10), agent  $i$  has access to  $(2K-1)D$  equations with  $3KD$  unknowns. In neither case, can adversary agent  $i$  infer  $f_j$ . ■

Similarly, we have that an external eavesdropper cannot infer any private information of an agent.

**Corollary 2:** Every agent's intermediate states and objective functions cannot be inferred by an external eavesdropper.

**Proof:** Since all exchanged messages are encrypted and that cracking the encryption is practically infeasible [35], an external eavesdropper cannot learn anything by intercepting exchanged messages. Therefore, it cannot infer any agents' intermediate states or objective functions. ■

From the above analysis, it is obvious that agent  $j$ 's private information cannot be uniquely derived by adversaries. However, an honest-but-curious neighbor  $i$  can still get some range information about the state  $x_j(k)$  and this estimated range will become tighter as  $x_j(k)$  converges to the optimal value as  $k \rightarrow \infty$  (cf., the simulation results in Fig. 8). We argue that this is completely unavoidable for any privacy-preserving approaches where all agents have to agree on the same final state, upon which the privacy of  $x_j(k)$  disappears. In fact, this is also acknowledged in [23], which shows that the privacy of  $x_j(k)$  will vanish as  $k \rightarrow \infty$ .

**Remark 4:** It is worth noting that an adversary agent  $i$  can combine systems of (9) and (10) to infer the information of a neighboring agent  $j$ . However, this will not enhance the ability of adversary agent  $i$  because the combination will not change the fact that the number of unknowns is greater than the number of establishable relevant equations.

**Remark 5:** From Theorem 2, we can see that in decentralized optimization, an agent's information will not be disclosed to other agents no matter how many neighbors it has. This is in distinct difference from the average consensus problem in [21], [39], and [40], where privacy cannot be protected for an agent if it has the honest-but-curious adversary as the only neighbor. This shows the disparate difference between decentralized optimization and the linear consensus problem.

## V. IMPLEMENTATION DETAILS

In this section, we discuss several technical issues that have to be addressed in the implementation of Algorithm 1.

- 1) In modern communication, a real number is represented by a floating point number, while encryption techniques only work for unsigned integers. To deal with this problem, we uniformly multiplied each element of the vector message  $x_i(k) \in \mathbb{R}^D$  (in floating point representation) by a sufficiently large number  $N_{\max}$  and round off the fractional part during the encryption to convert it to an integer. After decryption, the result is divided by  $N_{\max}$ . This process is conducted in each iteration and this quantization brings an error upper-bounded by  $\frac{1}{N_{\max}}$ .

- 2) As indicated in 1), encryption techniques only work for unsigned integers. In our implementation, all integer values are stored in fix-length integers (that is, long int in C) and negative values are left in 2's complement format. Encryption and intermediate computations are carried out as if the underlying data were unsigned. When the final message is decrypted, the overflowed bits (bits outside the fixed length) are discarded and the remaining binary number is treated as a signed integer which is later converted back to a real value.

## VI. APPLICATION TO AVERAGE CONSENSUS

Average consensus addresses the distributed computation of the mathematical mean of participating agents' states. In recent years, it has found applications in domains as diverse as automatic control, social sciences, signal processing, robotics, and optimization [41]. In this section, we show that the average consensus problem can be formulated as a constrained optimization problem, which, in turn, can be solved using Algorithm 1 with privacy guarantee for participating agents.

Assume that participating agents have scalar states  $\beta_i$  for  $i = 1, 2, \dots, N$ . Then, the problem of reaching average consensus, that is,  $\bar{\beta} = \frac{1}{N} \sum_{i=1}^N \beta_i$ , on every agent, can be formulated as the following decentralized optimization problem:

$$\begin{aligned} \min_x \quad & \sum_{i=1}^N \frac{1}{2} (x - \beta_i)^2 \\ \text{subject to} \quad & x \in \mathcal{X}. \end{aligned} \quad (11)$$

Here,  $\mathcal{X}$  is assumed to be large enough to contain the average consensus value  $\bar{\beta}$ .

**Theorem 3:** A network of  $N$  agents with individual states  $\beta_i$  ( $i = 1, 2, \dots, N$ ) can distributedly compute the average  $\bar{\beta}$  by solving (11) using Algorithm 1 if Assumptions 2 and 3 hold, all  $b_{i \rightarrow j}(k)$  are randomly chosen from  $[\sqrt{\eta}, \sqrt{\frac{1-\eta}{N-1}}]$  with  $0 < \eta < 1/N$ , and the stepsize  $\alpha_k$  satisfies  $\sum_k \alpha_k = \infty$  and  $\sum_k \alpha_k^2 < \infty$ .

**Proof:** The proof can be obtained following a similar line of reasoning of Theorem 1. ■

In addition, we have that an honest-but-curious agent  $i$  cannot infer the state of any other agents.

**Theorem 4:** Agent  $j$ 's private state  $\beta_j$  cannot be inferred by an honest-but-curious neighboring agent  $i$  if the network is composed of more than two agents, that is,  $N > 2$ , and the stepsize  $\alpha_k$  satisfies  $\alpha_k \neq 1$  for all  $k \geq 0$ .

**Proof:** The proof can be obtained following a similar line of reasoning of Theorem 2. ■

**Remark 6:** The condition  $\alpha_k \neq 1$  for all  $k \geq 0$  is easy to satisfy. For example, the commonly used form of  $\alpha_k = \frac{T_1}{k+T_2}$  ( $0 < T_1 < T_2 < \infty$ ) in [42] and [43] naturally satisfies this condition.

**Remark 7:** It is worth noting that in average consensus, the update rule of  $d_j(k)$ , that is,  $d_j(k) = v_j(k) - \beta_j$ , can be known to every participating agent. This is different from the general constrained optimization problem (4), where  $f_j$  and the update rule of  $d_j(k)$  are completely private to agent  $j$ . Therefore,



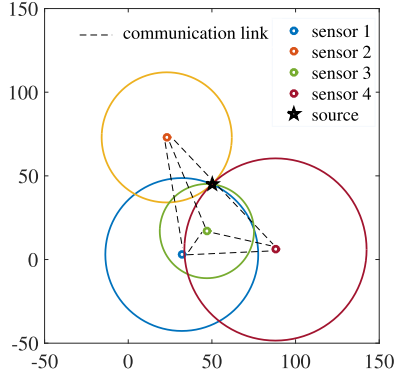


Fig. 1. Source localization setup used in one simulation run.

average consensus requires a stronger condition for privacy-preservation, that is,  $N > 2$  and  $\alpha_k \neq 1$  for all  $k \geq 0$ . However, the condition  $N > 2$  is still less restrictive than the condition of requiring at least two neighbors in existing data-obfuscation-based privacy-preserving average consensus results [21], [40]. In addition, as all exchanged messages are encrypted, our approach is also resilient to outside eavesdroppers, which will fail existing approaches in [21] and [40].

## VII. NUMERICAL SIMULATIONS

In this section, we first illustrate the efficiency of the proposed privacy-preserving approach using C/C++ implementations. Then, we compare our privacy-preserving average consensus approach with existing results in [21] and [40]. We used the open-source C implementation of the Paillier cryptosystem [44] in our simulations.

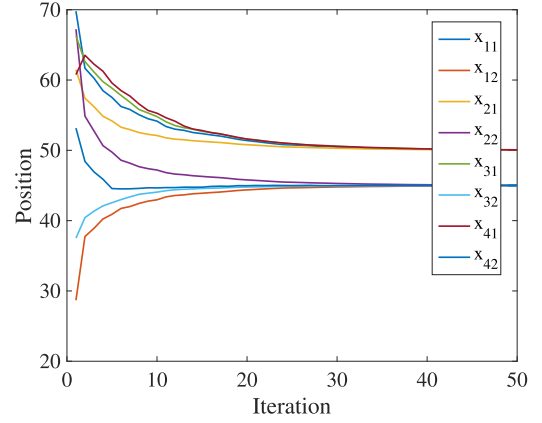
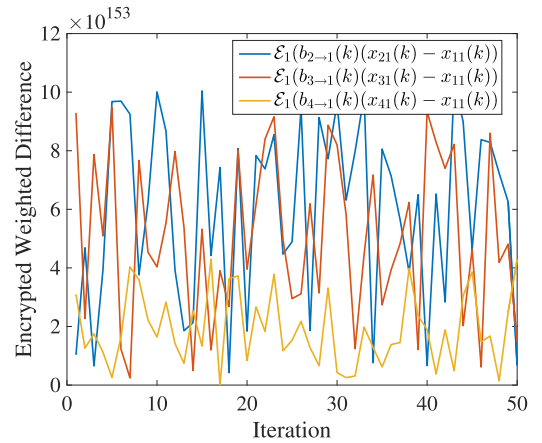
In the implementation,  $N_{\max}$  was set to  $10^6$  to convert each element in  $\mathbf{x}_i$  to a 64-bit integer during intermediate computations.  $b_{i \rightarrow j}(k)$  and  $b_{j \rightarrow i}(k)$  were also scaled up in the same way and represented by 64-bit integers. The encryption and decryption keys were chosen as 256-bit long.

### A. Evaluation of Algorithm 1

We evaluated the effectiveness of Algorithm 1 by using the source localization problem (cf., Example 1), agreement problem (cf., Example 2), and a least square problem (cf., Example 3), which are typical and important applications of the decentralized optimization problem (4).

**1) Source Localization:** We implemented Algorithm 1 under different source localization setups with sensors randomly distributed in the plane  $[0, 100] \times [0, 100]$  and a source located at  $[50; 45]$ . Simulation results confirmed that Algorithm 1 always converged to the source position when the source was located in the convex hull of all sensors. Fig. 2 visualizes the evolution of  $\mathbf{x}_i$  ( $i = 1, 2, 3, 4$ ) in one specific run where the network deployment is illustrated in Fig. 1. In Fig. 2,  $x_{ij}$  ( $i = 1, 2, 3, 4, j = 1, 2$ ) denotes the  $j$ th element of  $\mathbf{x}_i$ . All  $\mathbf{x}_i$  ( $i = 1, 2, 3, 4$ ) converged to the source position  $[50; 45]$ .

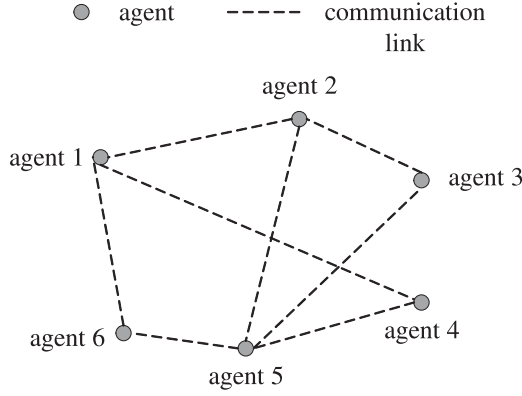
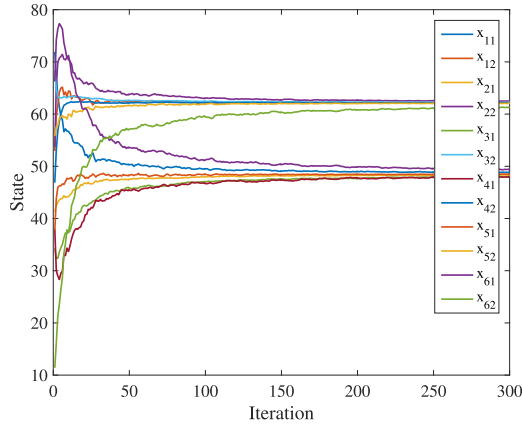
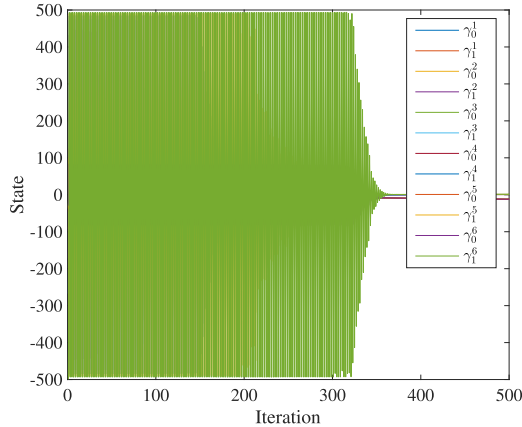
Fig. 3 visualizes the encrypted weighted differences (in ciphertext)  $\mathcal{E}_1(b_{2 \rightarrow 1}(k)(x_{21}(k) - x_{11}(k)))$ ,  $\mathcal{E}_1(b_{3 \rightarrow 1}(k)(x_{31}(k) - x_{11}(k)))$ , and  $\mathcal{E}_1(b_{4 \rightarrow 1}(k)(x_{41}(k) - x_{11}(k)))$ . It is


 Fig. 2. Evolution of  $\mathbf{x}_i$  ( $i = 1, 2, 3, 4$ ).

 Fig. 3. Evolution of the encrypted weighted differences (in ciphertext)  $\mathcal{E}_1(b_{2 \rightarrow 1}(k)(x_{21}(k) - x_{11}(k)))$ ,  $\mathcal{E}_1(b_{3 \rightarrow 1}(k)(x_{31}(k) - x_{11}(k)))$ , and  $\mathcal{E}_1(b_{4 \rightarrow 1}(k)(x_{41}(k) - x_{11}(k)))$ .

worth noting that although the estimates of all agents have converged after about 30 iterations, the encrypted weighted differences (in ciphertext) still appeared random to an outside eavesdropper. For Paillier cryptosystem, if the encryption/decryption key length is  $n$ -bit, the size of ciphertexts will be  $2n$  [45]. Since we use the 256-bit key, the ciphertext is 512-bit, that is, as large as  $2^{512}$ .

**2) Agreement Problem:** We next implemented Algorithm 1 under different network topologies to solve the agreement problem in Example 2. Simulation results confirmed that Algorithm 1 always converged to the optimal solution  $\sum_{i=1}^N \theta_i$ . Fig. 5 visualizes the evolution of  $\mathbf{x}_i$  ( $i = 1, 2, \dots, 6$ ) in one specific run where the network communication topology is given in Fig. 4 and  $\alpha_k$  was set to  $\alpha_k = \frac{1}{k+2}$ . In Fig. 5,  $x_{ij}$  ( $i = 1, 2, \dots, 6, j = 1, 2$ ) denotes the  $j$ th element of  $\mathbf{x}_i$ . All  $\mathbf{x}_i$  ( $i = 1, 2, \dots, 6$ ) converged to the optimal value  $[48.5; \frac{373}{6}]$ .

**3) Least Square Problem:** We also implemented Algorithm 1 to solve the simple linear regression problem in Example 3. Simulation results confirmed that Algorithm 1 always converged to the optimal solution. However, the convergence speed was lower than the source localization problem and the agreement problem. Simulation results also suggested that the convergence rate was sensitive to the stepsize

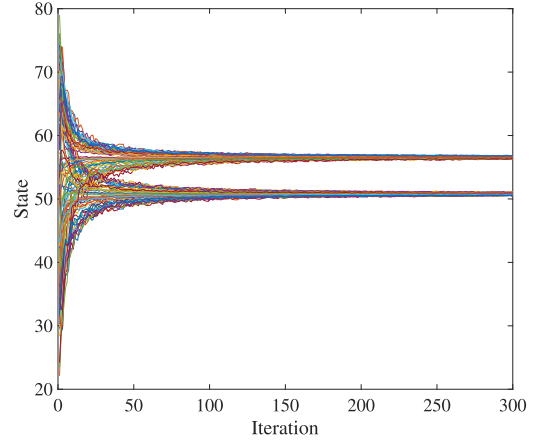
Fig. 4. Network of six agents ( $N = 6$ ).Fig. 5. Evolution of  $x_i$  ( $i = 1, 2, \dots, 6$ ).Fig. 6. Evolution of  $\gamma_0^i$  and  $\gamma_1^i$  ( $i = 1, 2, \dots, 6$ ).

$\alpha_k$ . Fig. 6 visualizes the evolution of  $\gamma_0^i$  and  $\gamma_1^i$  ( $i = 1, 2, \dots, 6$ ) in one specific run where the network communication topology is given in Fig. 4,  $\alpha_k$  was set to  $\alpha_k = \frac{10}{k+20}$ ,  $x_i$  was set to  $x_i = i$ ,  $y_i$  was set to  $y_i = 2 \times i - 16$ , and the constrained set was set to  $\gamma_0^2 + \gamma_1^2 \leq 500^2$ .  $\gamma_0^i$  and  $\gamma_1^i$  ( $i = 1, 2, \dots, 6$ ) denote the intermediate states  $\gamma_0$  and  $\gamma_1$  of agent  $i$ , respectively. All  $\gamma_0^i$  and  $\gamma_1^i$  ( $i = 1, 2, \dots, 6$ ) converged to their respective optimal values, that is, 2 and  $-16$ .

**4) Effect of Encryption and Decryption Key-Length and Network Size:** We also considered the influence of encryption

TABLE I  
AVERAGE ENCRYPTION AND DECRYPTION COMPUTATION TIME

network size	key-length (bit)	average time (s)
6	256	0.005
	2048	0.193
51	256	0.042
	2048	1.763

Fig. 7. Evolution of  $x_i$  ( $i = 1, 2, \dots, 51$ ).

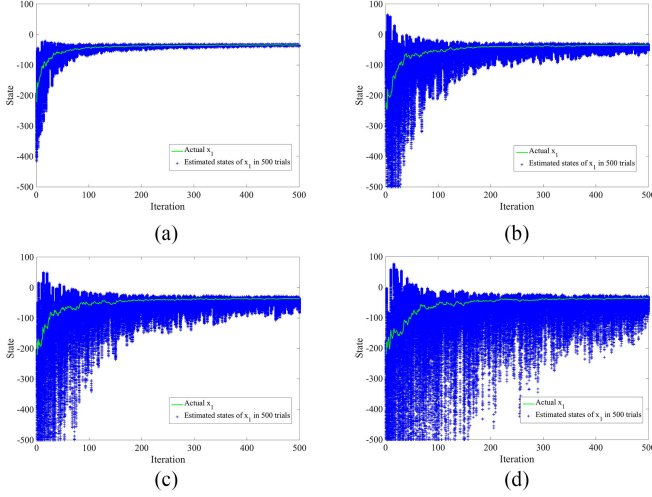
and decryption key-length and network size on Algorithm 1 (based on the agreement problem). We simulated two all-to-all networks with 6 and 51 agents, respectively. Both 256-bit and 2048-bit keys are evaluated. Table I gives the average computation time of encryption and decryption for each agent to communicate with all of its neighbors in each iteration on a 3.6-GHz CPU with 15.6-GB RAM. Fig. 7 visualizes the evolution of  $x_i$  ( $i = 1, 2, \dots, 51$ ) in the all-to-all network with 51 agents using 2048-bit keys. We can see that the average computation time increased with increased key-length and network size.

#### 5) Tradeoff Between Convergence Speed and Privacy:

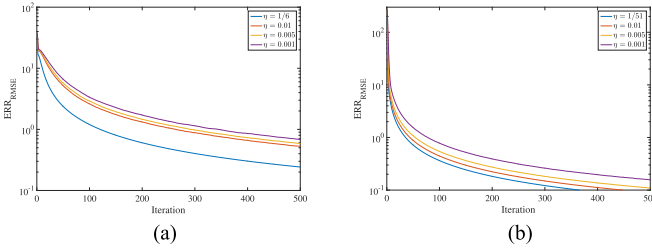
In this part, we first simulated an honest-but-curious adversary who tried to estimate its neighbors' intermediate states under different  $\eta$  values to illustrate the strengths of enabled privacy under different values of  $\eta$ . Then, we simulated the convergence speed of Algorithm 1 under different  $\eta$  values. All simulation results were obtained based on the agreement problem.

Assume that agent 2 in Fig. 4 is an honest-but-curious adversary who intends to estimate the intermediate states of agent 1. The individual local objective functions are the same as in (5) with  $\theta_i \in \mathbb{R}$ . Because agent 2 knows the constraints on agent 1's generation of  $b_{1 \rightarrow 2}$ , that is,  $b_{1 \rightarrow 2}$  is randomly chosen from  $[\sqrt{\eta}, \sqrt{\frac{1-\eta}{N-1}}]$  with  $0 < \eta < 1/N$  (cf., Theorem 1), it generated estimates of  $b_{1 \rightarrow 2}$  by using a guessed stochastic distribution of  $b_{1 \rightarrow 2}$ . We conservatively assume that agent 2 knows the probability distribution of  $b_{1 \rightarrow 2}$ , which gives it an edge in estimating  $b_{1 \rightarrow 2}$ . Then, agent 2 obtained a series of estimated  $x_1(k)$  according to (9). For example, after agent 2 obtained  $y(k) = b_{2 \rightarrow 1}(k)b_{1 \rightarrow 2}(k)(x_1(k) - x_2(k))$  at iteration  $k$ , it generated an estimate of  $b_{1 \rightarrow 2}(k)$  [denoted as  $\bar{b}_{1 \rightarrow 2}(k)$ ], and then it estimated  $x_1(k)$  as  $x_1(k) = x_2(k) + \frac{y(k)}{\bar{b}_{1 \rightarrow 2}(k)b_{2 \rightarrow 1}(k)}$ .





**Fig. 8.** Adversary's estimation of the intermediate state of agent 1. The green line is the actual intermediate state  $x_1$  of agent 1, the blue "+" are estimated states of  $x_1$  by agent 2 in 500 trials. (a)  $\eta = 0.05$ . (b)  $\eta = 0.01$ . (c)  $\eta = 0.005$ . (d)  $\eta = 0.001$ .



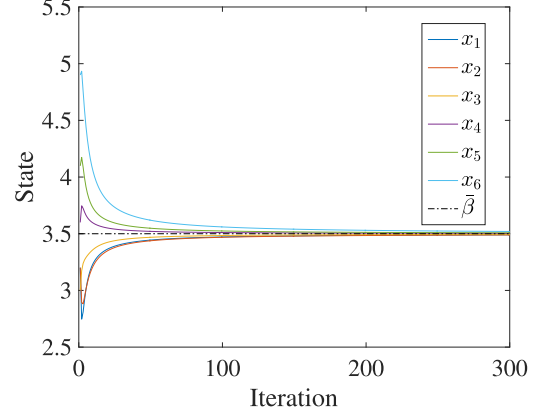
**Fig. 9.** Evolution of  $\text{ERR}_{\text{RMSE}}$  under different  $\eta$  values. (a) Evolution of  $\text{ERR}_{\text{RMSE}}$  under different  $\eta$  values when the network topology is given in Fig. 4. (b) Evolution of  $\text{ERR}_{\text{RMSE}}$  under different  $\eta$  values in an all-to-all network with 51 agents.

Fig. 8 shows the estimated  $x_1$  in 500 trials under different  $\eta$  values when  $b_{1 \rightarrow 2}$  follows uniform distribution. It can be seen that a smaller  $\eta$  leads to less accurate estimation and, hence, better privacy protection, confirming the statement in Remark 3. In addition, it can be seen that agent 2 cannot accurately estimate  $x_1$  initially. However, as  $x_1$  converges to the optimal value, agent 2 will be able to estimate the value that every agent agrees on, confirming the statement right above Remark 4.

We use the root-mean-square error (RMSE) to quantify the error between intermediate states and the optimal value, which is denoted as  $\text{ERR}_{\text{RMSE}}$

$$\text{ERR}_{\text{RMSE}} = \sqrt{\frac{\sum_{j=1}^L \sum_{i=1}^N \|x_{ij} - x^*\|^2}{LN}}$$

where  $L$  is the number of Monte Carlo trials,  $N$  is the number of agents,  $x_{ij}$  is the intermediate state of agent  $i$  in the  $j$ th Monte Carlo trial, and  $x^*$  is the optimal value. Fig. 9(a) visualizes the evolution of  $\text{ERR}_{\text{RMSE}}$  under different  $\eta$  values when the network topology is given in Fig. 4 ( $L = 500$ ) and Fig. 9(b) visualizes the evolution of  $\text{ERR}_{\text{RMSE}}$  under different  $\eta$  values in an all-to-all network with 51 agents ( $L = 500$ ). It can be seen that to reach the same  $\text{ERR}_{\text{RMSE}}$ , a smaller  $\eta$  in-



**Fig. 10.** Evolution of  $x_i$  ( $i = 1, 2, \dots, 6$ ) under the proposed privacy-preserving average consensus approach when  $\alpha_k$  was set to  $\alpha_k = \frac{1}{k+2}$ .

curs more iterations for convergence, confirming the statement in Remark 3. Because our approach requires a small  $\eta$  to enable strong privacy protection, it sacrifices the convergence speed in this sense.

## B. Privacy-Preserving Average Consensus

Using the network communication topology in Fig. 4, we compared our privacy-preserving average consensus approach with the algorithms in [21] and [40]. We set the states  $\beta_i$  of the six agents to  $\{1, 2, 3, 4, 5, 6\}$ , respectively. The weights were set as follows:

$$a_{ij} = \begin{cases} 0.2 & j \in \mathcal{N}_i \\ 0 & j \notin \mathcal{N}_i \cup \{i\} \\ 1 - \sum_{j \in \mathcal{N}_i} a_{ij} & i = j. \end{cases} \quad (12)$$

The internal state and the exchanged state are denoted as  $x_i(k)$  and  $x_i^+(k)$  for the algorithms in [21] and [40].

Fig. 10 visualizes the evolution of  $x_i$  ( $i = 1, 2, \dots, 6$ ) under the proposed approach in one specific run, where  $\alpha_k$  was set to  $\alpha_k = \frac{1}{k+2}$ . It can be seen that all  $x_i$  converged to the exact average value 3.5, confirming the effectiveness of the proposed approach.

It is worth noting that the convergence speed of the privacy-preserving average consensus approach can be increased by judiciously designing the stepsize  $\alpha_k$ . For example, simulation results suggested that using the  $\alpha_k$  below, convergence to the average can be made much faster (cf., the evolution of  $x_i$  in Fig. 11)

$$\alpha_k = \begin{cases} \frac{10}{k+20} & k < 30 \\ \frac{1}{k+1000} & k \geq 30. \end{cases} \quad (13)$$

Since our approach encrypts all exchanged messages, an outside eavesdropper cannot learn anything by intercepting these messages. In contrast, the algorithms in [21] and [40] cannot protect the privacy of participating agents against an external eavesdropper that can intercept all exchanged messages, as confirmed by our numerical simulation results below. Without loss of generality, we assume that an outside eavesdropper is interested in learning the state  $\beta_1$  of agent 1 and builds the following

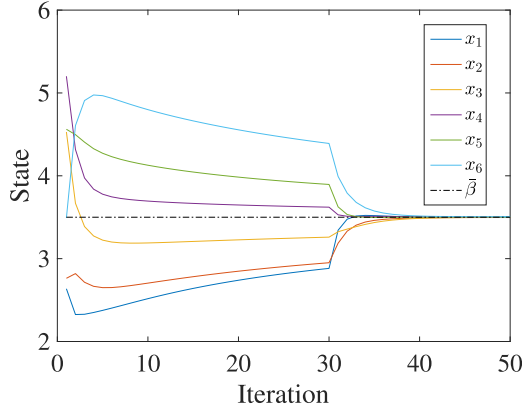


Fig. 11. Evolution of  $x_i$  ( $i = 1, 2, \dots, 6$ ) under the proposed privacy-preserving average consensus approach when  $\alpha_k$  was set according to (13).

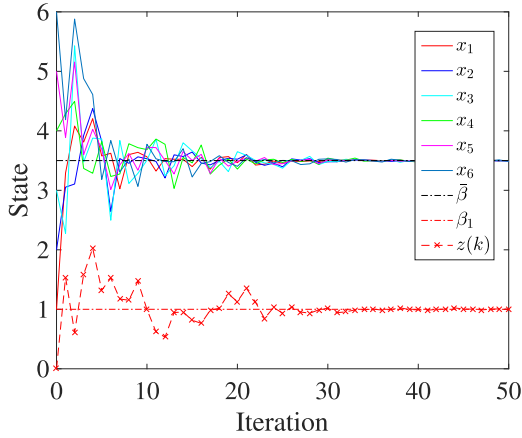


Fig. 12. Evolution of  $x_i$  ( $i = 1, 2, \dots, 6$ ) and  $z(k)$  under the algorithm in [21].

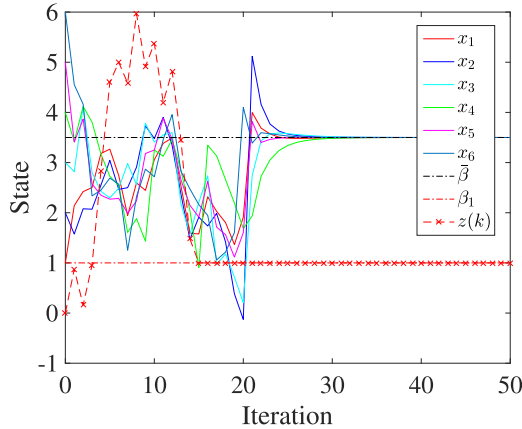


Fig. 13. Evolution of  $x_i$  ( $i = 1, 2, \dots, 6$ ) and  $z(k)$  under the algorithm in [40].

observer to estimate  $\beta_1$ :

$$z(k+1) = z(k) + x_1^+(k+1) - \left( a_{11}x_1^+(k) + \sum_{j \in \mathcal{N}_1} a_{1j}x_j^+(k) \right) \quad (14)$$

with the initial value of  $z$  set to  $z(0) = x_1^+(0)$ . As mentioned earlier, in (14),  $x_1$  and  $x_1^+$  denote the internal and exchanged states, respectively. Fig. 12 visualizes the evolution of  $x_i$  ( $i = 1, 2, \dots, 6$ ) as well as the eavesdropper's observer state  $z(k)$  under the approach in [21]. It can be seen that the eavesdropper can accurately estimate the internal state  $x_1$ . The same conclusion can be drawn for the approach in [40], which is confirmed vulnerable to eavesdropping attacks (cf., Fig. 13).

## VIII. CONCLUSION

In this paper, we proposed a novel approach to enabling privacy-preservation in decentralized optimization based on the integration of partially homomorphic cryptography with an existing decentralized optimization algorithm. By leveraging the Paillier cryptosystem and the unique convergence properties of the decentralized optimization, that is, robustness to random coupling weights, our approach provides a privacy guarantee without compromising the optimality of optimization in the absence of an aggregator or third party. Theoretical analysis confirms that an honest-but-curious adversary cannot infer the information of neighboring agents even by recording and analyzing the information exchanged in multiple iterations. The approach is also applicable to average consensus which has found extensive applications in fields as diverse as distributed computing, robotic networks, and power grids. Numerical simulation results confirmed the effectiveness and low computational complexity of the proposed approach.

## REFERENCES

- [1] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.
- [2] I. Lobel and A. Ozdaglar, "Distributed subgradient methods for convex optimization over random networks," *IEEE Trans. Autom. Control*, vol. 56, no. 6, pp. 1291–1306, Jun. 2011.
- [3] A. Nedic, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Trans. Autom. Control*, vol. 55, no. 4, pp. 922–938, Apr. 2010.
- [4] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
- [5] F. Yan, S. Sundaram, S. Vishwanathan, and Y. Qi, "Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 11, pp. 2483–2493, Nov. 2013.
- [6] T. Zhang and Q. Y. Zhu, "Dynamic differential privacy for ADMM-based distributed classification learning," *IEEE Trans. Inf. Forens. Security*, vol. 12, no. 1, pp. 172–187, Jan. 2017.
- [7] G. Mateos, J. A. Bazerque, and G. B. Giannakis, "Distributed sparse linear regression," *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 5262–5276, Oct. 2010.
- [8] J. Liu, J. Chen, and J. Ye, "Large-scale sparse logistic regression," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 547–556.
- [9] C. Zhang and Y. Wang, "Distributed event localization via alternating direction method of multipliers," *IEEE Trans. Mobile Comput.*, vol. 17, no. 2, pp. 348–361, Feb. 2018.
- [10] W. Meng *et al.*, "A projection based fully distributed approach for source localization in wireless sensor networks," *Ad Hoc Sensor Wireless Netw.*, vol. 18, no. 1–2, pp. 131–158.
- [11] G. B. Giannakis, V. Kekatos, N. Gatsis, S. J. Kim, H. Zhu, and B. F. Wollenberg, "Monitoring and optimization for power grids: A signal processing perspective," *IEEE Signal Process. Mag.*, vol. 30, no. 5, pp. 107–128, Sep. 2013.

- [12] S. Nabavi, J. H. Zhang, and A. Chakraborty, "Distributed optimization algorithms for wide-area oscillation monitoring in power systems using interregional PMU-PDC architectures," *IEEE Trans. Smart Grid*, vol. 6, no. 5, pp. 2529–2538, Sep. 2015.
- [13] B. S. He, L. S. Hou, and X. M. Yuan, "On full Jacobian decomposition of the augmented Lagrangian method for separable convex programming," *SIAM J. Optim.*, vol. 25, no. 4, pp. 2274–2312, 2015.
- [14] B. S. He, H. K. Xu, and X. M. Yuan, "On the proximal jacobian decomposition of ALM for multiple-block separable convex minimization problems and its relationship to ADMM," *J. Sci. Comput.*, vol. 66, no. 3, pp. 1204–1217, 2016.
- [15] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
- [16] Q. Ling and A. Ribeiro, "Decentralized dynamic optimization through the alternating direction method of multipliers," in *Proc. IEEE 14th Workshop Signal Process. Adv. Wireless Commun.*, 2013, pp. 170–174.
- [17] Q. Ling, Y. Liu, W. Shi, and Z. Tian, "Weighted ADMM for fast decentralized network optimization," *IEEE Trans. Signal Process.*, vol. 64, no. 22, pp. 5930–5942, Nov. 2016.
- [18] R. L. Lagendijk, Z. Erkin, and M. Barni, "Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation," *IEEE Signal Process. Mag.*, vol. 30, no. 1, pp. 82–105, Jan. 2013.
- [19] A. Al-Anwar, Y. Shoukry, S. Chakraborty, P. Martin, P. Tabuada, and M. B. Srivastava, "ProLoc: Resilient localization with private observers using partial homomorphic encryption," in *Proc. 16th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, 2017, pp. 41–52.
- [20] J. Lin, A. S. Morse, and B. Anderson, "The multi-agent rendezvous problem—the asynchronous case," in *Proc. 43rd IEEE Conf. Decis. Control*, 2004, vol. 2, pp. 1926–1931.
- [21] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Trans. Autom. Control*, vol. 62, no. 2, pp. 753–765, Feb. 2017.
- [22] M. H. DeGroot, "Reaching a consensus," *J. Amer. Stat. Assoc.*, vol. 69, no. 345, pp. 118–121, 1974.
- [23] Z. Huang, S. Mitra, and N. Vaidya, "Differentially private distributed optimization," in *Proc. Int. Conf. Distrib. Comput. Netw.*, 2015, p. 4.
- [24] S. Han, U. Topcu, and G. J. Pappas, "Differentially private distributed constrained optimization," *IEEE Trans. Autom. Control*, vol. 62, no. 1, pp. 50–64, Jan. 2017.
- [25] J. Cortés, G. E. Dullerud, S. Han, J. Le Ny, S. Mitra, and G. Pappas, "Differential privacy in control and network systems," in *Proc. IEEE 55th Conf. Decis. Control*, 2016, pp. 4252–4272.
- [26] M. T. Hale and M. Egerstedt, "Differentially private cloud-based multi-agent optimization with constraints," in *Proc. Amer. Control Conf.*, 2015, pp. 1235–1240.
- [27] E. Nozari, P. Tallapragada, and J. Cortes, "Differentially private distributed convex optimization via functional perturbation," *IEEE Trans. Control Netw. Syst.*, vol. 5, no. 1, pp. 395–408, Mar. 2016.
- [28] S. Pequito, S. Kar, S. Sundaram, and A. P. Aguiar, "Design of communication networks for distributed computation with privacy guarantees," in *Proc. IEEE 53rd Annu. Conf. Decis. Control*, 2014, pp. 1370–1376.
- [29] A. Alaeddini, K. Morgansen, and M. Mesbahi, "Adaptive communication networks with privacy guarantees," 2017, arXiv: 1704.01188.
- [30] Z. Xu and Q. Zhu, "Secure and resilient control design for cloud enabled networked control systems," in *Proc. 1st ACM Workshop Cyber-Phys. Syst.-Secur. Privacy*, 2015, pp. 31–42.
- [31] K. Kogiso and T. Fujita, "Cyber-security enhancement of networked control systems using homomorphic encryption," in *Proc. IEEE 54th Annu. Conf. Decis. Control*, 2015, pp. 6836–6843.
- [32] C. Gentry *et al.*, "Fully homomorphic encryption using ideal lattices," in *Proc. 41st Annu. ACM Sympo. Theory Comput.*, 2009, vol. 9, pp. 169–178.
- [33] A. C. Yao, "Protocols for secure computations," in *Proc. 23rd Annu. Symp. Found. Comput. Sci.*, 1982, pp. 160–164.
- [34] F. Li, B. Luo, and P. Liu, "Secure information aggregation for smart grids using homomorphic encryption," in *Proc. 1st IEEE Int. Conf. Smart Grid Commun.*, 2010, pp. 327–332.
- [35] O. Goldreich, *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge, U.K: Cambridge Univ. Press, 2009.
- [36] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 1999, pp. 223–238.
- [37] F. Zeng, C. Li, and Z. Tian, "Distributed compressive spectrum sensing in cooperative multihop cognitive networks," *IEEE J. Sel. Top. Signal Process.*, vol. 5, no. 1, pp. 37–48, Feb. 2011.
- [38] A. Nedic, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM J. Optim.*, vol. 27, no. 4, pp. 2597–2633, 2017.
- [39] M. H. Ruan, M. Ahmad, and Y. Q. Wang, "Secure and privacy-preserving average consensus," in *Proc. 3rd ACM Workshop Cyber-Phys. Syst. Security Privacy*, 2017, pp. 123–129.
- [40] N. E. Manitaras and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," in *Proc. Eur. Control Conf.*, 2013, pp. 760–765.
- [41] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.
- [42] A. Nedic and D. P. Bertsekas, "Incremental subgradient methods for non-differentiable optimization," *SIAM J. Optim.*, vol. 12, no. 1, pp. 109–138, 2001.
- [43] S. Boyd, L. Xiao, and A. Mutapcic, *Subgradient Methods (Lecture Notes EE3920)*. Stanford, CA, USA: Stanford Univ., 2003.
- [44] J. Bethencourt, "Advanced crypto software collection," [Online]. Available: <http://acsc.cs.utexas.edu/libpaillier>
- [45] D. Catalano, R. Gennaro, and N. Howgrave-Graham, "The bit security of Pailliers encryption scheme and its applications," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 2001, pp. 229–243.



**Chunlei Zhang** received the B.S. degree in electronic and information engineering, Beihang University, Beijing, China, in 2015. She is currently working toward the Ph.D. degree in electrical engineering at the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC, USA.

Her research interests include localization in mobile sensor networks and privacy preservation in decentralized optimization.



**Yongqiang Wang** was born in Shandong, China. He received the B.S. degree in electrical engineering and automation, the B.S. degree in computer science and technology from Xi'an Jiaotong University, Shaanxi, China, in 2004, and the M.Sc. and Ph.D. degrees in control science and engineering from Tsinghua University, Beijing, China, in 2009.

During 2007–2008, he was with the University of Duisburg-Essen, Germany, as a visiting student. He is currently an Assistant Professor

with the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC, USA. Before joining Clemson University, he was a Project Scientist with the University of California, Santa Barbara, CA, USA. His research interests include cooperative and networked control, clock synchronization, privacy and security in dynamical systems, and model-based fault diagnosis.

Dr. Wang is the recipient of 2008 Young Author Prize from IFAC Japan Foundation for a paper presented at the 17th IFAC World Congress in Seoul.