Electronic Research Archive

https://www.aimspress.com/journal/era

ERA, 33(3): 1429–1445.
DOI: 10.3934/era.2025067
Received: 27 December 2024
Revised: 11 February 2025
Accepted: 04 March 2025
Published: 13 March 2025

Research article

Privacy-preserving distributed optimization algorithm for directed networks via state decomposition and external input

Mengjie Xu[1,2], **Nuerken Saireke**[1,2] **and Jimin Wang**[1,2,*]

[1] School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing 100083, China

[2] Key Laboratory of Knowledge Automation for Industrial Processes, Ministry of Education, Beijing 100083, China

* **Correspondence:** Email: jimwang@ustb.edu.cn.

Abstract: In this paper, we study the privacy-preserving distributed optimization problem on directed graphs, aiming to minimize the sum of all agents' cost functions and protect the sensitive information. In the distributed optimization problem of directed graphs, agents need to exchange information with their neighbors to obtain the optimal solution, and this situation may lead to the leakage of privacy information. By using the state decomposition method, the algorithm ensures that the sensitive information of the agent will not be obtained by attackers. Before each iteration, each agent decomposes their initial state into two sub-states, one sub-state for normal information exchange with other agents, and the other sub-state is only known to itself and invisible to the outside world. Unlike traditional optimization algorithms applied to directed graphs, instead of using the push-sum algorithm, we introduce the external input, which can reduce the number of communications between agents and save communication resources. We prove that in this case, the algorithm can converge to the optimal solution of the distributed optimization problem. Finally, a numerical simulation is conducted to illustrate the effectiveness of the proposed method.

Keywords: distributed optimization; privacy-preserving; state decomposition; directed graph

1. Introduction

Distributed optimization, as a core technology for solving large-scale problems in networked systems, has received widespread attention in recent years. By decomposing the global problem into local sub-problems of each agent and achieving the overall optimal solution by coordination, its algorithm design includes gradient descent, dual decomposition technology, and the alternating direction multiplier method (ADMM). These methods have been successfully applied in fields such as smart grids,

communication networks, and machine learning due to their excellent parallelism and convergence [1–3]. In addition, in order to adapt to dynamic environments, researchers have proposed dynamic consistency optimization methods to cope with changes in network topology and data distribution [4,5]. At the same time, the scalability and security issues of distributed optimization in large-scale computing have also become the focus of research [6, 7].

Distributed optimization is widely applied, especially in fields such as energy, machine learning, healthcare, and communications. In smart grids, it optimizes energy management by distributed resource allocation and load scheduling, improving system's efficiency and resilience [8–11]. In machine learning, distributed optimization provides parallelization support for large-scale model training, significantly improves computing efficiency, and especially shows great potential in the field of federated learning [12–14]. In addition, medical imaging and diagnostics have also benefited from the introduction of distributed optimization technologies that not only improve data processing capabilities but also ensure the security of patients' data with privacy-preserving algorithms [15, 16]. In the field of the Internet of Things (IoT), distributed optimization technology is widely used in resource allocation and dynamic routing decisions, providing strong support for efficient communication [17, 18]. The study of [19] alleviates the network congestion and bandwidth utilization issues caused by the quality of service queuing mechanism and denial of service attacks by introducing new compression rules, and develops an intelligent trigger controller supervised by the mini-batch machine learning algorithm. These application examples fully demonstrate that distributed optimization has wide applicability and strong theoretical support.

However, the communication characteristics of distributed optimization bring significant privacy risks, especially in scenarios involving sensitive data such as medical, financial, and user behavior modeling. Differential privacy technology effectively improves the anonymity of data by injecting noise during the optimization process while minimizing the impact on the model's performance [20,21]. Existing research has proven that using differential privacy mechanisms in distributed optimization can significantly reduce the risk of data leakage while maintaining computational efficiency [22, 23]. The authors of [24] designed a distributed algorithm based on the direction and state perturbation. This algorithm achieves differential privacy by perturbing the state variables and directions with attenuated Laplacian noise. The authors of [25] proposed a new algorithm for decentralized non-convex optimization. This algorithm can simultaneously achieve strict differential privacy and the avoidance of saddle points/maxima. Some studies also combine local differential privacy and global differential privacy methods to further optimize the privacy protection effect [26–28]. However, the added noise may prevent the agents from converging to an exact solution. Homomorphic encryption technology ensures the security of the entire optimization process by allowing calculations to be performed directly on encrypted data [29, 30]. The authors of [31] proposed a privacy-preserving decentralized optimization method based on the alternating direction method of multipliers and partial homomorphic encryption technology. The authors of [32] proposed a new algorithm through homomorphic encryption technology. This algorithm can achieve secure multi-party computation with complete correctness. Compared with differential privacy, homomorphic encryption has unique advantages in high-security scenarios, but has the problem of higher computational cost.

In addition to the two methods above, [33] developed a new privacy protection method, namely state decomposition. The main idea of the state decomposition method is to decompose the state of each agent into an actual state and a virtual state. The actual state interacts with other agents normally

to exchange information, while the virtual state is unknown to the outside world, thereby protecting the real data of the agent from being accessed. This method has been widely used in the field of privacy protection; for example, [34] proposed a state decomposition method that can privately achieve average consensus without any trustworthy neighbor agents. Moreover, [35] proposed a privacy-preserving push-sum method for directed networks. However, when using the push-sum algorithm, each agent needs to maintain two state values, which will increase the communication overhead. In this paper, on the basis of the state decomposition method, we study the privacy-preserving distributed optimization problem on a directed graph. The contributions of this paper are summarized as follows.

1) The privacy-preserving algorithm is proposed, which is based on state decomposition for the distributed optimization problem on a directed graph. In this algorithm, the initial state of each agent is decomposed into two sub-states: the actual state and the virtual state. The actual state is used for information exchange between agents, while the virtual state is only used for internal information exchange and is unknown to the outside world.

2) Many existing distributed optimization algorithms applied to directed graphs use the push-sum algorithm. They decompose the state of the agent into two states for iteration, and finally take the ratio of these two states as the true value. The problem with this is that it will increase the communication burden of the system, so we did not use the push-sum algorithm, but introduced external input during the iteration process.

3) We also demonstrate that the algorithm can protect the private information of agents from being obtained by malicious external eavesdroppers or honest but curious neighbors.

The remaining part of this paper is structured as follows. Section 2 presents some preliminaries. Section 3 is our main results, where the algorithm is proposed and the proof is given. In Section 4, a numerical example is illustrated. Finally, a brief conclusion is given in Section 5.

Notations: Let $\mathbf{1}_d$ and $\mathbf{0}_d$ be the d-dimensional all-one vector and all-zero vector, respectively. $\mathbb{R}^d$ is the set of d-dimensional real numbers. I_d and O_d denote the $d \times d$-demensional identity matrix and all-zero matrix, respectively. Given a matrix X, the element located at the i-th row and j-th column of the matrix is represented by X_{ij}. We use the notation $\rho(A)$ to stand for the spectral radius of matrix A. A matrix A is referred to as row-stochastic or column-stochastic when the sum of all elements in each row or each column amounts to 1, and all the entries of A are non-negative. The symbol $\otimes$ represents the Kronecker product, while $\|\cdot\|_2$ represents the ℓ_2-norm.

2. Preliminaries

2.1. Graphs

We consider a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$ which comprises n agents. Here, $\mathcal{V} = \{1, 2, \ldots, n\}$ stands for the set of agents and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ represents the set of edges. The adjacency matrix $A = \left[a_{ij}\right]$ indicates the coupling weights, where $a_{ij} > 0$ if there is an edge from i to j, that is, $(i, j) \in \mathcal{E}$, and $a_{ij} = 0$ otherwise. When we use $(j, i) \in \mathcal{E}$, it implies that there is a communication link enabling agent i to transmit information to Agent j. The agents capable of directly sending information to Agent i are referred to as the in-neighbors of Agent i, and the collection of such agents is denoted as $\mathcal{N}_i^- = \{j \in \mathcal{V} | (i, j) \in \mathcal{E}\}$. Likewise, the agents that can directly receive messages from agent i are called the out-neighbors of Agent i, and the set of these agents is denoted as $\mathcal{N}_i^+ = \{j \in \mathcal{V} | (i, j) \in \mathcal{E}\}$.

2.2. Problem formulation

Consider an optimization issue within a multiagent system that consists of n agents. Every single agent possesses a private cost function f_i, and this function is known solely to Agent i. The common objective of all the agents involved is to minimize a global objective function

$$\min_{x\in\mathbb{R}^d} \sum_{i=1}^{n} f_i(x), \tag{2.1}$$

where x is the global decision variable.

The subsequent assumptions concerning the objective function and the graph are provided, which will be utilized for deriving the main results.

Assumption 1. *(Connectivity): The directed graph $\mathcal{G}$ is strongly connected.*

Assumption 2. *(Strong convexity): The global objective function, f, is μ-strongly convex, i.e.,*

$$f(\mathbf{x}) \geqslant f(\mathbf{y}) + \nabla f(\mathbf{y})^T(\mathbf{x}-\mathbf{y}) + \frac{\mu}{2}\|\mathbf{x}-\mathbf{y}\|_2^2,$$

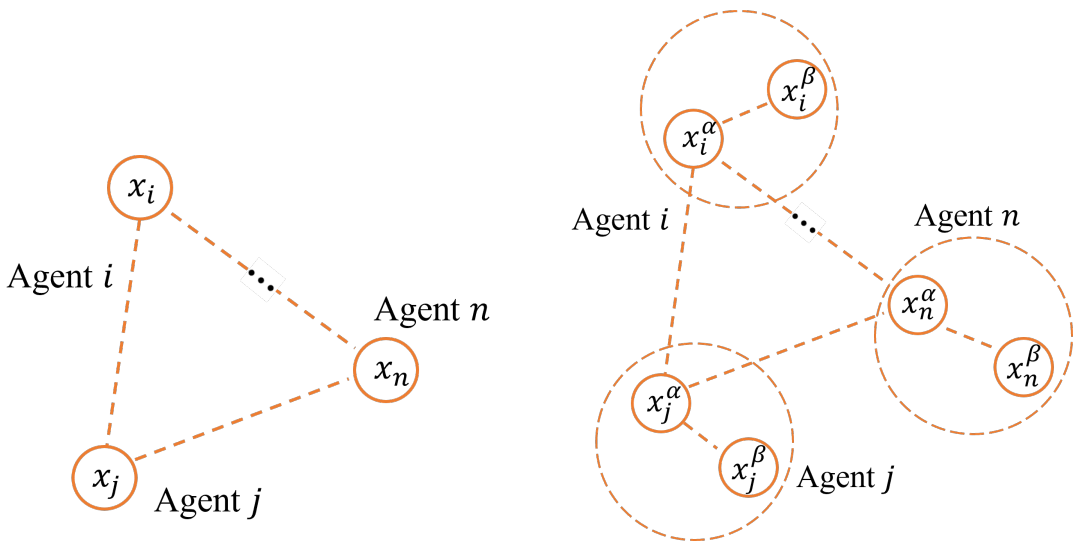
for any $\mathbf{x}$ *and* $\mathbf{y} \in \mathbb{R}^d$, *where* $\mu > 0$.

Given Assumption 2, Problem (2.1) possesses a sole optimal solution, $x^* \in \mathbb{R}^d$.

Assumption 3. *(Smoothness): The gradient of each f_i is L-Lipschitz continuous, i.e., for any* $\mathbf{x}$ *and* $\mathbf{y} \in \mathbb{R}^d$,

$$\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\|_2 \leqslant L\|\mathbf{x}-\mathbf{y}\|_2.$$

2.3. State decomposition method



(a) Before state decomposition **(b)** After state decomposition

Figure 1. State decomposition method.

The state decomposition method was initially proposed in [33]. The core concept of this method is to decompose the initial state $x_{i,0}$ of each agent into two sub-states, denoted $x_{i,0}^{\alpha}$ and $x_{i,0}^{\beta}$ respectively, as shown in Figure 1. The values of the sub-states $x_{i,0}^{\alpha}$ and $x_{i,0}^{\beta}$ can be any real numbers, yet they must satisfy the equation $x_{i,0}^{\alpha} + x_{i,0}^{\beta} = 2x_{i,0}$. After decomposition, the sub-state $x_{i,0}^{\alpha}$ assumes the role that the original state $x_{i,0}$ played in the interactions among agents. The other sub-state $x_{i,0}^{\beta}$ does not engage in the information exchange among neighboring agents. Instead, it only communicates with $x_{i,0}^{\alpha}$. For instance, consider Agent i in Figure 1(b). In the interactions between agents, x_i^{α} behaves exactly like x_i, while x_i^{β} is hidden from all agents except Agent i, even though it has an impact on the evolution of x_i^{α}.

3. Main results

Within this part, we put forward a privacy-preserving algorithm for directed graphs via state decomposition and external input. In addition, we also provide the convergence analysis and privacy analysis.

3.1. Privacy-preserving algorithm

After the state decomposition, the update equation becomes

$$\begin{cases} x_{i,k+1}^{\alpha} = a_{ii}x_{i,k}^{\alpha} + \sum\limits_{j\in\mathcal{N}_i^-} a_{ij}x_{j,k}^{\alpha} + \alpha_i x_{i,k}^{\beta}, \\ x_{i,k+1}^{\beta} = p_{ii}x_{i,k}^{\beta} + \beta_i x_{i,k}^{\alpha}, \end{cases} \tag{3.1}$$

The coupling weights between the two substates $x_{i,k}^{\alpha}$ and $x_{i,k}^{\beta}$ are not symmetric. They are denoted α_i and β_i respectively. The update weight for the substate $x_{i,k}^{\beta}$ is designated as p_{ii}. The weight of the outgoing link from Agent j to Agent i is represented by a_{ij}.

Definition 1. *[36] (Minimal polynomial of a matrix): For matrix P, its minimal polynomial is denoted as $Q(t)$, which is expressed as $Q(t) = t^{D+1} + \sum_{i=0}^{D} \omega_i t^i$. Here, $Q(t)$ is a monic polynomial. It has the minimum degree of $D+1$ and satisfies the condition that when we substitute the matrix P into $Q(t)$, we get $Q(P) = 0_n$. In this polynomial, ω_i represents the coefficients.*

Definition 2. *[36] (Minimal polynomial of a matrix pair): For the object $[P, e_i^T]$, there is an associated minimal polynomial, which we denote as $Q_i(t)$. It is given by the expression $Q_i(t) = t^{D_i+1} + \sum_{i=0}^{D_i} \omega_{i,j} t^i = 0$, where the coefficients $\omega_{i,j}$ belong to the set of real numbers $\mathbb{R}$. This $Q_i(t)$ is a monic polynomial. It has the minimum degree of $D_i + 1$ and fulfills the condition that when we perform the operation $e_i^T Q_i(P)$, the result is 0.*

On the basis of the analysis of [36], we let $x_{2k}^{\alpha} = \left[x_{i,1}^{\alpha}, x_{i,2}^{\alpha}, \ldots, x_{i,2k+1}^{\alpha}\right]^T$. In addition, we define the Hankel matrix and the difference vectors as follows:

$$\Gamma\left\{(x_{2k}^{\alpha})^T\right\} \triangleq \begin{bmatrix} x_{i,1}^{\alpha} & x_{i,2}^{\alpha} & \cdots & x_{i,k+1}^{\alpha} \\ x_{i,2}^{\alpha} & x_{i,3}^{\alpha} & \cdots & x_{i,k+2}^{\alpha} \\ \vdots & \vdots & \ddots & \vdots \\ x_{i,k+1}^{\alpha} & x_{i,k+2}^{\alpha} & \cdots & x_{i,2k+1}^{\alpha} \end{bmatrix},$$

$$\bar{x}_{2k}^{\alpha} \triangleq \left[x_{i,2}^{\alpha} - x_{i,1}^{\alpha}, \ldots, x_{i,2k+2}^{\alpha} - x_{i,2k+1}^{\alpha}\right]^T.$$

In [36], a distributed termination mechanism was put forward. This mechanism enables all agents to reach an agreement on when to end their iterations, provided that they have all finished computing the average. The specific steps are as follows.

1) When start the iteration (3.1), each Agent i simultaneously starts two counters, namely c_i and r_i, both initialized to 0. The counter c_i increases by one with each passing time step, which can be expressed as $c_{i,k+1} = c_{i,k} + 1$. The subsequent text will elaborate on how the counter c_i is updated.

2) Simultaneously with the iteration (3.1), a max-consensus algorithm is also launched, which is expressed as

$$\theta_{i,k+1} = \max_{i \in \mathcal{N}_i^- \cup i} \{\max\{\theta_{i,k}, c_{i,k}\}\}, \tag{3.2}$$

with $\theta_{i,0} = 0$. After that, the update rule for r_i is as follows:

$$r_{i,k+1} = \begin{cases} 0, & \text{if } \theta_{i,k+1} \neq \theta_{i,k}, \\ r_{i,k} + 1, & \text{otherwise.} \end{cases} \tag{3.3}$$

3) Once the Hankel matrix $\Gamma\left\{\left(\bar{x}_{D_i}^{\alpha}\right)^T\right\}$ loses rank, Agent i records the value of counter c_i at that particular time step, which we denote as k_i^0, and names it c_i^0. In other words, c_i^0 is defined as $c_i[k_i^0]$. Subsequently, Agent i halts the increment of the counter, i.e., for all $k' \geqslant k_i^0$, we have $c[k'] = c_i[k_i^0] = c_i^0$. It should be noted that $c_i^0 = 2(D_i + 1) + 1$.

4) Agent i is able to terminate the iteration (3.1) as soon as r_i reaches the value of c_i^0.

We now design Algorithms 1 and 2.

Algorithm 1 A privacy-preserving finite-time algorithm via state decomposition and external input

1: **Initialization**: Agent $i \in \mathcal{V}$ initializes the weight value of state variable x_i, which needs to satisfy $\sum_{j \in \mathcal{N}_i^- \cup \{i\}} a_{ij} + \alpha_i = 1$ and $p_{ii} + \beta_i = 1$.

2: **if** $k = 0$ **then**

3: Run the following iteration and (3.2)($u_{i,k}^{\alpha}$ and $u_{i,k}^{\beta}$ are the added external inputs, and their specific values will be given later in the text), store the vector $\left(\bar{x}_{D_i}^{\alpha}\right)^T$, increase the value of the counter $c_{i,k}$ and determine the value of the counter $r_{i,k}$ via (3.3).

$$\begin{cases} x_{i,k+1}^{\alpha} = a_{ii}\left(x_{i,k}^{\alpha} + u_{i,k}^{\alpha}\right) + \sum_{j \in \mathcal{N}_i^-} a_{ij}\left(x_{j,k}^{\alpha} + u_{i,k}^{\alpha}\right) + \alpha_i\left(x_{i,k}^{\beta} + u_{i,k}^{\beta}\right) \\ x_{i,k+1}^{\beta} = p_{ii}\left(x_{i,k}^{\beta} + u_{i,k}^{\beta}\right) + \beta_i\left(x_{i,k}^{\alpha} + u_{i,k}^{\alpha}\right) \end{cases} \tag{3.4}$$

4: Expand the dimension of the Hankel matrix $\Gamma\left\{\left(\bar{x}_{D_i}^{\alpha}\right)^T\right\}$ continuously until reaching the value of k_i^0, when it becomes rank-deficient. Once this happens, store the value $c_i^0 = 2(D_i + 1) + 1$.

5: Keep performing iteration (3.4) until the iteration reaches step $k_{i,t}$, at which point, $r_i(k_{i,t}) = c_i^0$. Then, save the value $D_{max} = (k_{i,t} - 2D_i - 2)/2 - 1$.

6: **else**

7: Run (3.4) for $k_{max} = D_{max} + 2$ steps.

8: Calculate the average value as $\hat{x}_i^{ave} = \frac{\sum_{i=1}^{n} x_i^{\alpha} + \sum_{i=1}^{n} x_i^{\beta}}{2n}$.
9: **output:** Agent $i \in \mathcal{V}$ outputs $\hat{x}_i^{ave}$.

Algorithm 2 A privacy-preserving finite-time based gradient descent (GD) algorithm

1: **Initialization**: Step size η, maximum optimization iteration number T, Agent $i \in \mathcal{V}$ initializes the value $x_i(0)$, sets $y_{i,0} = \nabla f_i(x_{i,0})$, $t = 0$.
2: **for** $t \leqslant T$ **do**
3: Take $\nabla f_i(x_{i,t})$ as the input to Algorithm 1 and obtain the output $\hat{x}_i^{ave}$. Then, define $y_{i,t} = \hat{x}_i^{ave}$.
4: Calculate $x_{i,t+1}$ with $y_{i,t}$ as follows:

$$x_{i,t+1} = \bar{a}_{ii} x_{i,t} + \sum_{j \in \mathcal{N}_i^-} \bar{a}_{ij} x_{j,t} - \eta y_{i,t}, \tag{3.5}$$

where $A = \left[\bar{a}_{ij}\right] \in \mathbb{R}^n$ is row-stochastic.
5: **output:** Agent $i \in \mathcal{V}$ obtains the solution x^*.

3.2. Convergence analysis

In this part, we present the proof regarding the convergence and accuracy of Algorithms 1 and 2.

Let $x_k^{\alpha} = [x_{1,k}^{\alpha}, x_{2,k}^{\alpha}, \ldots, x_{n,k}^{\alpha}]^T$ be the state vector of sub-agent α, $x_k^{\beta} = [x_{1,k}^{\beta}, x_{2,k}^{\beta}, \ldots, x_{n,k}^{\beta}]^T$ be the state vector of sub-agent β, and $\mathbf{x'}_k = [x_{1,k}^{\alpha}, x_{2,k}^{\alpha}, \ldots, x_{n,k}^{\alpha}, x_{1,k}^{\beta}, x_{2,k}^{\beta}, \ldots, x_{n,k}^{\beta}]^T \in \mathbb{R}^{2n}$ be the overall state vector.

Then each Agent i sets two new auxiliary vectors, $s_{i,k} = [s_{i1,k}, s_{i2,k}, \ldots, s_{i2n,k}]^T \in \mathbb{R}^{2n}$, $s_{n+i,k} = [s_{n+i1,k}, s_{n+i2,k}, \ldots, s_{n+i2n,k}]^T \in \mathbb{R}^{2n}$, with the initial state component $s_{ij,0} = 1$ if $i = j$, 0 otherwise. We choose the same evolution updating rule as Eq (3.1), in which case, the evolutions of these two variables can be represented as shown below.

$$\begin{cases} s_{i,k+1} = a_{ii} s_{i,k} + \sum_{j \in \mathcal{N}_i^-} a_{ij} s_{j,k}(k) + \alpha_i s_{n+i,k} \\ s_{n+i,k+1} = p_{ii} s_{n+i,k} + \beta_i s_{i,k} \end{cases} \tag{3.6}$$

The external input of Agent i is calculated as

$$\begin{cases} u_{i,k}^{\alpha} = F_{i,k}^{\alpha} - F_{i,k-1}^{\alpha} \\ u_{i,k}^{\beta} = F_{i,k}^{\beta} - F_{i,k-1}^{\beta} \end{cases}$$

with:

$$\begin{cases} F_{i,k}^{\alpha} = x_{i,0}^{\alpha} \left(\frac{1-2ns_{ii,k}}{2ns_{ii,k}}\right) \\ F_{i,k}^{\beta} = x_{i,0}^{\beta} \left(\frac{1-2ns_{ii,k}}{2ns_{ii,k}}\right) \end{cases}$$

and $F_{i,-1}^{\alpha} = 0, F_{i,-1}^{\beta} = 0$.

We now use a lemma to illustrate the convergence of $s_{i,k}$ and the specific convergence value.

Lemma 1. *[37] Assume that the initial state variable* $s_{i,0} = [s_{i1,0}, s_{i2,0}, \ldots, s_{i2n,0}]^T \in \mathbb{R}^{2n}$ *with* $s_{ij,0} = 1$ *if* $i = j$; 0 *otherwise. After updating by Eq (3.6),* $\lim_{k \to \infty} s_{i,k} = \psi = [\psi_1, \psi_2, \ldots, \psi_{2n}]^T$, *where* ψ *is the normalized left eigenvector of the matrix* $M = \begin{bmatrix} M_1 & M_2 \\ M_3 & M_4 \end{bmatrix}$, *with* $M_1 = \left[a_{ij}\right] \in \mathbb{R}^{n \times n}, M_2 =$

$diag(\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{R}^{n\times n}$, $M_3 = diag(\beta_1, \beta_2, \ldots, \beta_n) \in \mathbb{R}^{n\times n}$, $M_4 = diag(p_{11}, p_{22}, \ldots, p_{nn}) \in \mathbb{R}^{n\times n}$. *And we have* $\lim_{k\to\infty} M^k = (\rho(M))^k \mathbf{1}_{2n}\psi^T = \mathbf{1}_{2N}\psi^T$.

We now use a lemma to illustrate the properties of $u^{\alpha}_{i,k}, u^{\beta}_{i,k}$.

Lemma 2. *[37] For the external inputs $u^{\alpha}_{i,k}, u^{\beta}_{i,k}$ in the system, we have*

$$\lim_{k\to\infty}\sum_{t=0}^{k} u^{l}_{i,t} = \lim_{k\to\infty}\sum_{t}^{k}(F^{l}_{i,t} - F^{l}_{i,t-1}) = \lim_{k\to\infty} F^{l}_{i,k} = \frac{x^{l}_{i,0}}{2n\psi_i} - x^{l}_{i,0}, l = \alpha, \beta.$$

We write the iteration (3.4) in the form of matrices and vectors:

$$\mathbf{x}'_{k+1} = M(\mathbf{x}'_k + u_k), \tag{3.7}$$

where $u_k = [u^{\alpha}_1, u^{\alpha}_2, \ldots, u^{\alpha}_N, u^{\beta}_1, u^{\beta}_2, \ldots, u^{\beta}_n]^T$.

Moreover, Equation (3.5) can be rewritten as

$$\mathbf{x}_{t+1} = A\mathbf{x}_t - \eta\mathbf{y}_t, \tag{3.8}$$

where $\mathbf{x}_t = [x_{1,t}, x_{2,t}, \ldots, x_{n,t}]^T$ and $\mathbf{y}_t = [y_{1,t}, y_{2,t}, \ldots, y_{n,t}]^T$. Next, we give some necessary lemmas.

Lemma 3. *[38] Given Assumption 1, the matrix A possesses a sole non-negative left eigenvector v (corresponding to the eigenvalue* 1*) such that* $v^T\mathbf{1}_n = n$.

Lemma 4. *[38] Given Assumptions 1–3, a matrix norm* $\|\cdot\|_A$ *exists such that* $\sigma_A = \left\|A - \frac{\mathbf{1}_n u^T}{n}\right\|_A < 1$, *and* σ_A *can be made arbitrarily close to the spectral radius* $\rho(A - \frac{\mathbf{1}_n\mu^T}{n}) < 1$.

Now, we give Theorem 1.

Theorem 1. *Given Assumptions 1–3, for each Agent* $i \in \mathcal{V}$,

1) The output of Algorithm 1 is precisely the average of the initial values of all agents. That is, for every $i \in \mathcal{V}$, $\hat{x}^{ave}_i = \frac{1}{n}\sum_{i=1}^{n} x_{i,0}$.

2) When $0 < \eta < \frac{1}{\mu+L}$, *where* μ *and L are as defined in Assumptions 2 and 3, respectively, Algorithm 2 converges linearly with respect to the number of optimization iterations to the global optimum, i.e.,* $\|\mathbf{x}_t - \mathbf{1}_n \otimes x^*\|_2$ *converges to* 0 *in a linear fashion.*

Proof. (1) Expanding Eq (3.7), we have

$$\begin{aligned}
\mathbf{x}'_{k+1} =& M(\mathbf{x}'_k + u_k)\\
=& M(M(\mathbf{x}'_{k-1} + u_{k-1})) + Mu_k\\
=& \cdots\\
=& M^{k+1}\mathbf{x}'_0 + M^{k+1}u_0 + M^k u_1 + \cdots + M^2 u_{k-1} + Mu_k\\
=& M^{k+1}\mathbf{x}'_0 + \sum_{t=0}^{k} M^{k+1-t}u_t.
\end{aligned} \tag{3.9}$$

Let $F_k = [F^{\alpha}_{1,k}, F^{\alpha}_{2,k}, \ldots, F^{\alpha}_{N,k}, F^{\beta}_{1,k}, F^{\beta}_{2,k}, \ldots, F^{\beta}_{N,k}]^T$. According to Lemmas 1, 2, we have

$$\lim_{k\to\infty}(M)^{k+1} = \mathbf{1}_{2n}\psi^T = \left[\psi^T, \psi^T, \ldots, \psi^T\right]^T \in \mathbb{R}^{2n\times 2n}, \tag{3.10}$$

and

$$\begin{aligned}\lim_{k\to\infty}\sum_{t=0}^{k} M^{k+1-t}u_t &= \lim_{k\to\infty}\sum_{t=0}^{k}\psi^T u_t = \psi^T \lim_{k\to\infty} F_k \\ &= \psi^T[\frac{x^{\alpha}_{1,0}}{2n\psi_1} - x^{\alpha}_{1,0}, \ldots, \frac{x^{\alpha}_{n,0}}{2n\psi_n} - x^{\alpha}_{n,0}, \frac{x^{\beta}_{1,0}}{2n\psi_{n+1}} - x^{\beta}_{1,0}, \ldots, \frac{x^{\beta}_{n,0}}{2n\psi_{2n}} - x^{\beta}_{n,0}]^T.\end{aligned} \tag{3.11}$$

On the basis of Eq (3.11), we can know that when $k \to \infty$, $\sum\limits_{t=0}^{k} M^{k+1-t}u_t$ is a constant. We use C to represent it, and thus we have

$$\lim_{k\to\infty}\mathbf{x}'_{k+1} = \lim_{k\to\infty}M^{k+1}\mathbf{x}'_0 + \lim_{k\to\infty}\sum_{t=0}^{k}M^{k+1-t}u_t = \left[\psi^T, \psi^T, \ldots, \psi^T\right]^T\mathbf{x}'_0 + C, \text{i.e.}$$

$$\lim_{k\to\infty}\begin{bmatrix} x^{\alpha}_{1,k+1} \\ \vdots \\ x^{\alpha}_{N,k+1} \\ x^{\beta}_{1,k+1} \\ \vdots \\ x^{\beta}_{N,k+1}\end{bmatrix} = \begin{bmatrix} \psi_1 & \cdots & \psi_{2n} \\ \vdots & \vdots & \vdots \\ \psi_1 & \cdots & \psi_{2n}\end{bmatrix}\begin{bmatrix} x^{\alpha}_{1,0} \\ \vdots \\ x^{\alpha}_{n,0} \\ x^{\beta}_{1,0} \\ \vdots \\ x^{\beta}_{n,0}\end{bmatrix} + C,$$

which can show that $\lim\limits_{k\to\infty} x^{\alpha}_{i,k+1} = \lim\limits_{k\to\infty} x^{\beta}_{i,k+1}$.

Therefore, we have

$$\begin{aligned}\lim_{k\to\infty} x^{\alpha}_{i,k+1} = \lim_{k\to\infty} x^{\beta}_{i,k+1} &= \psi^T\mathbf{x}'_0 - \psi^T\lim_{k\to\infty}F_k \\ &= \frac{x^{\alpha}_{1,0}}{2n} + \cdots\frac{x^{\alpha}_{n,0}}{2n} + \frac{x^{\beta}_{1,0}}{2n} + \cdots\frac{x^{\beta}_{n,0}}{2n} \\ &= \frac{2(x_{1,0} + \cdots + x_{n,0})}{2n} = \frac{1}{n}\sum_{i=1}^{n} x_{i,0}.\end{aligned} \tag{3.12}$$

We thus have

$$\hat{x}^{ave}_i = \frac{\sum_{i=1}^{n} x^{\alpha}_i + \sum_{i=1}^{n} x^{\beta}_i}{2n} = \frac{1}{n}\sum_{i=1}^{n} x_{i,0}. \tag{3.13}$$

(2) We write $\overline{x}_t = v^T\mathbf{x}_t/n$, $\overline{y}_t = \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(x_{i,t})$ and $g_t = \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\overline{x}_t)$. From the analysis above, we know that at the time of iteration t, each agent can obtain the average gradient at time k via Algorithm 1, i.e., $y_{i,t} = \bar{y}_t, \mathbf{y}_t = \mathbf{1}_n\bar{y}_t$. Hence, from the iteration (3.8), we can obtain

$$\bar{x}_{t+1} - x^* = \bar{x}_t - \eta\bar{y}_t - x^* = \bar{x}_t - \eta g_t - x^* - \eta(\bar{y}_t - g_t),$$

$$\mathbf{x}_{t+1} - \mathbf{1}_n\bar{x}_{t+1} = A\mathbf{x}_t - \eta\mathbf{y}_t - \mathbf{1}_n\bar{x}_t + \eta\mathbf{1}_n\bar{y}_t = (A - \mathbf{1}_n v^T/n)(\mathbf{x}_t - \mathbf{1}_n\bar{x}_t).$$

If $\eta < 1/(\mu + L)$, we have

$$\begin{aligned} \|\bar{x}_{t+1} - x^*\|_2 &\leq (1 - \eta\mu)\|\bar{x}_t - x^*\|_2 + \eta\|\bar{y}_t - g_t\| \\ &\leq (1 - \eta\mu)\|\bar{x}_t - x^*\|_2 + \frac{\eta L}{\sqrt{n}}\|\mathbf{x}_t - \mathbf{1}_n\bar{x}_t\|_2. \end{aligned} \tag{3.14}$$

Moreover, from the result of Lemma 4, we can obtain

$$\|\bar{x}_{t+1} - x^*\|_2 \leq (1 - \eta\mu)\|\bar{x}_t - x^*\|_2 + \frac{\eta q L}{\sqrt{n}}\|\mathbf{x}_t - \mathbf{1}_n\bar{x}_t\|_A.$$

Taking $V_t = [\|\bar{x}_t - x^*\|_2, \|\mathbf{x}_t - \mathbf{1}_n\bar{x}_t\|_A]^T$, we have

$$V_{t+1} \leq PV_t, \tag{3.15}$$

where the transition matrix $P = \begin{pmatrix} 1 - \eta\mu & \eta q L/\sqrt{n} \\ 0 & \sigma_A \end{pmatrix}$.

Since $0 < 1 - \eta\mu < 1$ and $0 < \sigma_A < 1$, we can see that the spectral radius of P is strictly smaller than 1 and therefore $\|\mathbf{x}_t - \mathbf{1}_n \otimes x^*\|_2$ converges to zero linearly. □

Remark 1. *The algorithm we proposed can achieve the optimal solution to the distributed optimization problem. It can be seen that, unlike the push-sum algorithm, where each agent needs to maintain two states and exchange more information during the process of information exchange with neighboring agents, our algorithm requires less information to be exchanged, which obviously reduces the communication cost.*

3.3. Privacy analysis

In this part, we provide a detailed privacy analysis of Algorithm 1 in terms of how it can resist "attacks" from curious agents (in the network) and eavesdroppers (outside the network). First, we give detailed descriptions of curious and eavesdropping attack models [25].

1) Honest-but-curious attacks refer to situations where one or more participating agents (whether they collude or not) accurately adhere to every step of the protocol. Nevertheless, they are inquisitive and gather all the received intermediate data with the intention of uncovering sensitive information about other participating agents.

2) Eavesdropping attacks are carried out by an adversary who manages to gain access to the information shared with other agents by infiltrating all the communication channels. Consequently, eavesdroppers are aware of the network's topology and all the data shared within the network. However, the state variables that are not shared will not become known to them.

Similar to the analysis in [36], we define $\Delta\mathcal{I}_{\mathcal{A}}(x_{p,i}) = \{\bar{x}_{p,i}|\mathcal{I}_{\mathcal{A}}(0:K)\}$; this set encompasses all the possible states associated with $x_{p,i}$ when the information set accessible to $\mathcal{A}$ is $\mathcal{I}_{\mathcal{A}}(0:K)$.

The diameter of $\Delta\mathcal{I}_{\mathcal{A}}(x_{p,i})$ is defined as

$$\text{Diam}\{\Delta\mathcal{I}_{\mathcal{A}}(x_{p,i})\} = \sup_{\bar{x}_{p,i},\bar{x}'_{p,i}\in\Delta\mathcal{I}_{\mathcal{A}}(x_{p,i})} |\bar{x}_{p,i} - \bar{x}'_{p,i}|,$$

where $\bar{x}_{p,i}$ and $\bar{x}'_{p,i}$ are two disparate states which are part of the set $\Delta\mathcal{I}_{\mathcal{A}}(x_{p,i})$.

Now, we present Theorem 2 in the following paragraphs.

Theorem 2. *Given Assumption 1, for every Agent $i \in \mathcal{V}$, the privacy of Agent i can be preserved under Algorithm 1:*

(1) For a set of honest-but-curious agents $\mathcal{N}$, as long as at least one neighbor of Agent i is not in the set $\mathcal{N}$, the privacy information of Agent i can be protected from being obtained by the set $\mathcal{N}$.

(2) For an eavesdropper $\mathcal{R}$, if there is an edge ε_{mi} or ε_{ji} that the eavesdropper $\mathcal{R}$ cannot eavesdrop on, then the privacy information of Agent i can be protected from being obtained by the eavesdropper $\mathcal{R}$.

Proof. Since, under Algorithm 1, the maximum communication round equals k_1, the information set $\mathcal{I}_{\mathcal{N}}(0 : k_1)$ and $\mathcal{I}_{\mathcal{R}}(0 : k_1)$ denote all the information accessible to the adversary. From Algorithm 2, it can be seen that the private information $\nabla f_i(x_{i,t}), \forall t \geq 0$ is regarded as the input of Algorithm 1. Thus, it is enough to demonstrate that the privacy of the initial value $x_{i,0}$ of Agent i can be preserved under Algorithm 1.

(1) We use a method similar to that in [33] to prove that honest-but-curious neighbors cannot distinguish any changes in the neighbor's initial value. More specifically, under the following initial condition:

$$\begin{aligned} &\hat{x}^{\alpha}_{i,0} = x^{\alpha}_{i,0}, \hat{x}^{\beta}_{i,0} = x^{\beta}_{i,0} + 2\Delta, \\ &\hat{x}^{\alpha}_{m,0} = x^{\alpha}_{m,0}, \hat{x}^{\beta}_{m,0} = x^{\beta}_{m,0} - 2\Delta, \\ &\hat{x}^{\alpha}_{r,0} = x^{\alpha}_{r,0}, \hat{x}^{\beta}_{r,0} = x^{\beta}_{r,0}. \end{aligned} \tag{3.16}$$

We set the following weights:

$$\begin{aligned} &\hat{a}_{ii} = \hat{a}_{ii}, \hat{a}_{ij} = a_{ij}, \hat{a}_{im} = a_{im}, \hat{\alpha}_i = \frac{\alpha_i(x^{\beta}_{i,0} + u^{\beta}_{i,0})}{x^{\beta}_{i,0} + u^{\beta}_{i,0} + 2\Delta}, \\ &\hat{a}_{mm} = a_{mm}, \hat{a}_{mi} = a_{mi}, \hat{a}_{mj} = a_{mj}, \hat{\alpha}_m = \frac{\alpha_m(x^{\beta}_{m,0} + u^{\beta}_{m,0})}{x^{\beta}_{m,0} + u^{\beta}_{m,0} - 2\Delta}, \\ &\hat{p}_{ii} = \frac{p_{ii}(x^{\beta}_{i,0} + u^{\beta}_{i,0})}{2\Delta + x^{\beta}_{i,0} + u^{\beta}_{i,0}}, \hat{p}_{mm} = \frac{p_{mm}(x^{\beta}_{m,0} + u^{\beta}_{m,0})}{x^{\beta}_{m,0} + u^{\beta}_{m,0} - 2\Delta}, \\ &\hat{\beta}_i = \beta_i, \hat{\beta}_m = \beta_m, \hat{a}_{rr} = a_{rr}, \hat{a}_{rq} = a_{rq}, \hat{\alpha}_r = \alpha_r, \hat{p}_{rr} = p_{rr}, \hat{\beta}_r = \beta_r, \end{aligned}$$

where $r \in \mathcal{V}\backslash\{i, m\}, q \in \mathcal{V}$, Δ is an arbitrary real number, and "\" represents set subtraction. The external input $u^{l}_{i,k}, l = \alpha, \beta$ is invariant for $i \in \mathcal{V}$, it can be easily verified that $\hat{x}^{\alpha}_{i,1} = x^{\alpha}_{i,1}, \hat{x}^{\alpha}_{m,1} = x^{\alpha}_{m,1}$. Moreover, due to the weight values remaining identical for $k \geqslant 1$, it is easy to see that the updates of all agents are same, namely, $\hat{x}^{\alpha}_{i,k} = x^{\alpha}_{i,k}, i \in \mathcal{V}$. Thus, the information received by Agent j remains consistent even when the real initial state of Agent i changes. We now have

$$\mathrm{Diam}(\Delta_i(\mathcal{I}_{\mathcal{N}}(0 : k_1))) \geq \sup_{\Delta \in \mathbb{R}} |x_{i,0} - (x_{i,0} + \Delta)| = \infty.$$

From the definition in [36], we have proved the first statement.

(2) For the second statement, because of the topological limitations, the eavesdropper $\mathcal{R}$ is unable to eavesdrop on ε_{mi} or ε_{im}, where $m \in \mathcal{N}^{+}_{i}$ or $m \in \mathcal{N}^{-}_{i}$, i.e., a_{mi} or a_{im}, is inaccessible to $\mathcal{R}$. Moreover, since the self-weights a_{ii}, a_{mm} are not transmitted over the communication network, the eavesdropper

$\mathcal{R}$ can not obtain any information of a_{im}, a_{mm} or a_{mi}, a_{ii}. Subsequently, in a similar vein to the proof of the first statement, we have $\text{Diam}(\Delta_i(\mathcal{I}_{\mathcal{N}}(0 : k_1))) = \infty$, i.e., the privacy of $x_{i,0}$ is preserved.

□

Remark 2. *The state decomposition method we proposed enhances the privacy protection ability of distributed optimization on directed graphs, but it may also fail under certain circumstances. One potential vulnerability arises when adversaries possess highly correlated auxiliary data, which could enable inference attacks. Additionally, if an attacker continuously observes multiple decomposed states over time, they may reconstruct sensitive information through statistical analysis. Of course, the probability of this situation occurring is extremely low. To mitigate these risks, incorporating differential privacy mechanisms or applying obfuscation techniques to the decomposed states could enhance robustness. Future research will explore these strategies to further strengthen privacy guarantees in adversarial environments.*

4. Numerical example

In this part, we validate the convergence and privacy of the proposed algorithm by exemplary simulations. Consider a directed graph consisting of five agents, that is, $n = 5$, as shown in Figure 2. The task of each agent is to find the nearest gathering point x^* by cooperating with other agents without exposing her/his initial position $p_i, \forall i \in \mathcal{V}$. The objective function of Agent i is described by $f_i(x) = \frac{1}{2}\|x - p_i\|_2^2$, i.e.,

$$\min f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x) = \frac{1}{5}\sum_{i=1}^{5} \frac{1}{2}\|x - p_i\|_2^2.$$

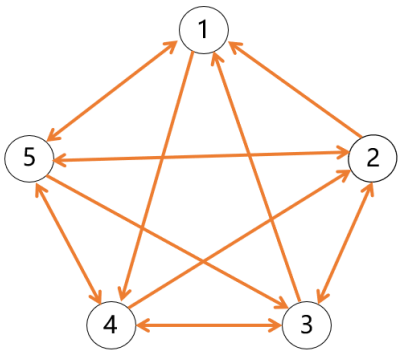


Figure 2. Digraph of five agents.

We suppose that the initial state values of the agents are $x_{1,0} = 10$, $x_{2,0} = 20$, $x_{3,0} = 30$, $x_{4,0} = 40$, and $x_{5,0} = 50$, and select the weight values between the agents reasonably.

The variation in the variable $\mathbf{x}_k = [x_{1,k}, x_{2,k}, \ldots, x_{5,k}]^T$ with respect to the iteration k is shown in Figure 3. It is observable that, with regard to these five agents, by applying Algorithms 1 and 2, $\mathbf{x}_k$ is capable of converging to the optimal solution x^* successfully.

In addition, we need to prove that our algorithm can successfully save communication resources compared with algorithms like the one using push-sum in [36]. We record the amount of data that Agent 1 needs to send to their neighbors throughout the iteration process. As shown in Figure 4, the

amount of data our algorithm needs to send during the iteration is lower. The same applies to other agents. This fully demonstrates that our algorithm can successfully save communication resources.

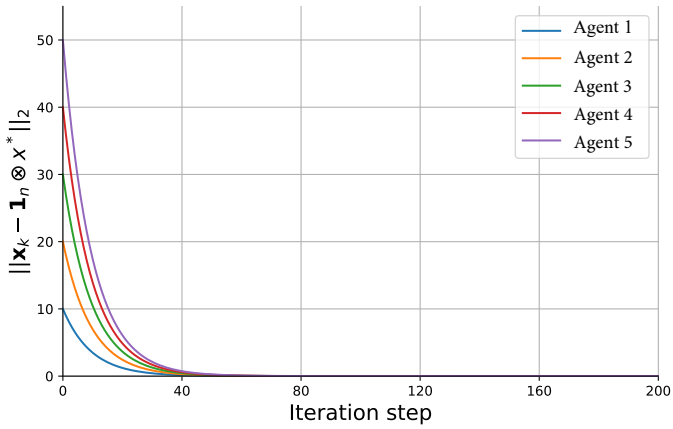


Figure 3. The variation in $\|\mathbf{x}_k - \mathbf{1}_n \otimes x^*\|_2$ with respect to the iteration k.

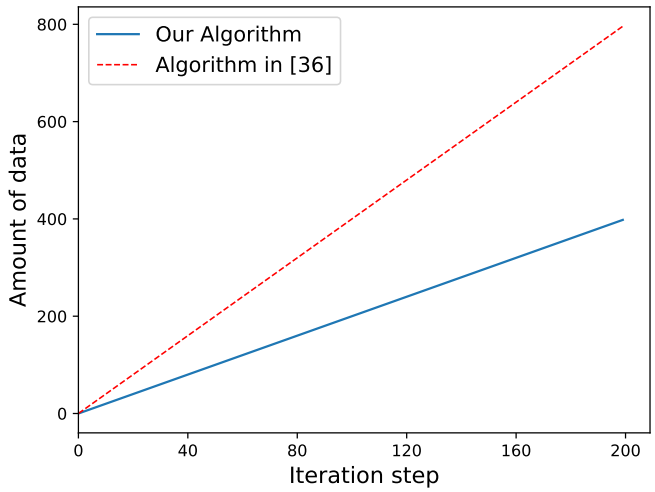
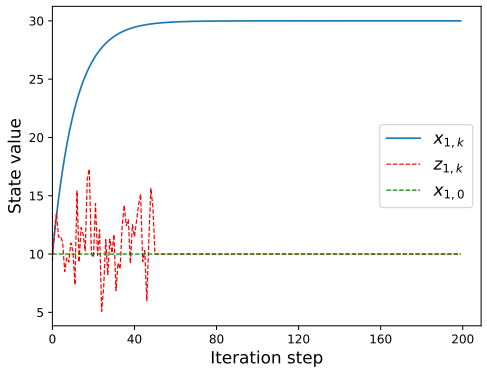
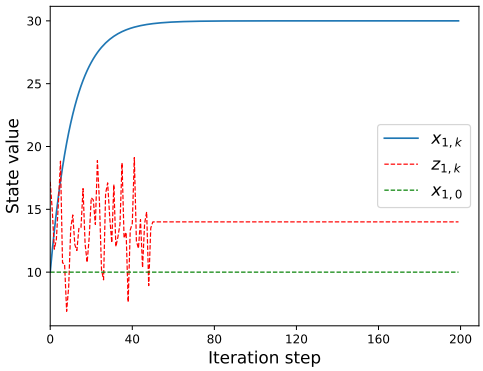


Figure 4. Record of the amount of data.

In the end, we assume that there is an eavesdropper that wants to obtain the information of Agent 1, where $z_{1,k}$ represents the information of Agent 1 obtained by the eavesdropper. When the privacy-preserving algorithm is not used, the result is shown in Figure 5(a). Although the convergence effect can eventually be achieved, the eavesdropper can successfully obtain the information of Agent 1. After using Algorithm 1, the convergence situation is shown in Figure 5(b). It is observable that not only can it converge, but the eavesdropper cannot steal the initial value information of Agent 1, and thus the privacy of Agent 1 is protected.



(a) The observed result without privacy-preserving theory



(b) The observed result under our algorithm

Figure 5. The observation results of external eavesdroppers.

5. Conclusions

In this paper, we proposed a novel privacy-preserving distributed optimization algorithm for directed graphs. By leveraging a state decomposition method, our approach ensures the protection of sensitive information while enabling agents to collaboratively minimize the sum of cost functions. Unlike traditional methods, the introduction of external inputs effectively reduces the communication overhead, enhancing efficiency without compromising convergence to the optimal solution. Numerical simulations demonstrate the effectiveness and practicality of the proposed method, confirming its potential for addressing privacy concerns in distributed optimization problems on directed graphs. Future work involves enhancing the optimization accuracy and addressing constrained optimization problems.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The work was supported by the National Natural Science Foundation of China under Grants 62203045 and 62433020. The material in this paper was not presented at any conference.

Conflict of interest

The authors declare there are no conflicts of interest.

References

1. A. Priolo, A. Gasparri, E. Montijano, C. Sagues, A distributed algorithm for average consensus on strongly connected weighted digraphs, *Automatica*, **50** (2014), 946–951. https://doi.org/10.1016/j.automatica.2013.12.026

2. B. Houska, J. Frasch, M. Diehl, An augmented Lagrangian based algorithm for distributed nonconvex optimization, *SIAM J. Optim.*, **26** (2016), 1101–1127. https://doi.org/10.1137/140975991

3. R. Mohebifard, A. Hajbabaie, Distributed optimization and coordination algorithms for dynamic traffic metering in urban street networks, *IEEE Trans. Intell. Transp. Syst.*, **20** (2019), 1930–1941. https://doi.org/10.1109/TITS.2018.2848246

4. G. Chen, Q. Yang, Distributed constrained optimization for multiagent networks with nonsmooth objective functions, *Syst. Control Lett.*, **124** (2019), 60–67. https://doi.org/10.1016/j.sysconle.2018.12.005

5. M. Tajalli, A. Hajbabaie, Distributed optimization and coordination algorithms for dynamic speed optimization of connected and autonomous vehicles in urban street networks, *Transp. Res. Part C Emerging Technol.*, **95** (2018), 497–515. https://doi.org/10.1016/j.trc.2018.07.012

6. J. Zhou, R. Q. Hu, Y. Qian, Scalable distributed communication architectures to support advanced metering infrastructure in smart grid, *IEEE Trans. Parallel Distrib. Syst.*, **23** (2012), 1632–1642. https://doi.org/10.1109/TPDS.2012.53

7. F. Lai, Y. Dai, S. Singapuram, J. Liu, X. Zhu, H. Madhyastha, et al., Fedscale: benchmarking model and system performance of federated learning at scale, in *Proceedings of the 39th International Conference on Machine Learning*, **162** (2022), 11814–11827. https://doi.org/10.48550/arXiv.2105.11367

8. Z. Chen, Y. Wang, Privacy-preserving distributed optimization and learning, preprint, arXiv:2403.00157, 2024. https://doi.org/10.48550/arXiv.2403.00157

9. P. Braun, L. Grüne, C. M. Kellett, S. R. Weller, K. Worthmann, A distributed optimization algorithm for the predictive control of smart grids, *IEEE Trans. Autom. Control*, **61** (2016), 3898–3911. https://doi.org/10.1109/TAC.2016.2525808

10. W. Chen, Z. Wang, Q. Liu, D. Yue, G. P. Liu, A new privacy-preserving average consensus algorithm with two-phase structure: Applications to load sharing of microgrids, *Automatica*, **167** (2024), 111715. https://doi.org/10.1016/j.automatica.2024.111715

11. W. Chen, L. Liu, G. P. Liu, Privacy-preserving distributed economic dispatch of microgrids: A dynamic quantization-based consensus scheme with homomorphic encryption, *IEEE Trans. Smart Grid*, **14** (2022), 701–713. https://doi.org/10.1109/TSG.2022.3189665

12. W. Zhang, D. Yang, W. Wu, H. Peng, N. Zhang, H. Zhang, et al., Optimizing federated learning in distributed industrial IoT: A multi-agent approach, *IEEE J. Sel. Areas Commun.*, **39** (2021), 3688–3703. https://doi.org/10.1109/JSAC.2021.3118352

13. Z. Hu, K. Shaloudegi, G. Zhang, Y. Yu, Federated learning meets multiobjective optimization, *IEEE Trans. Network Sci. Eng.*, **9** (2022), 2039–2051. https://doi.org/10.1109/TNSE.2022.3169117

14. T. Wang, Y. Liu, X. Zheng, H. N. Dai, W. Jia, M. Xie, Edge-based communication optimization for distributed federated learning, *IEEE Trans. Network Sci. Eng.*, **9** (2021), 2015–2024. https://doi.org/10.1109/TNSE.2021.3083263

15. A. Mang, A. Gholami, C. Davatzikos, G. Biros, PDE-constrained optimization in medical image analysis, *Optim. Eng.*, **19** (2018), 765–812. https://doi.org/10.1007/s11081-018-9390-9

16. V. Nikitin, V. D. Andrade, A. Slyamov, B. J. Gould, Y. Zhang, V. Sampathkumar, et al., Distributed optimization for nonrigid nano-tomography, *IEEE Trans. Comput. Imaging*, **7** (2021), 272–287. https://doi.org/10.1109/TCI.2021.3060915

17. Y. Gao, M. Kim, C. Thapa, A. Abuadbba, Z. Zhang, S. Camtepe, et al., Evaluation and optimization of distributed machine learning techniques for internet of things, *IEEE Trans. Comput.*, **71** (2021), 2538–2552. https://doi.org/10.1109/TC.2021.3135752

18. R. Anand, D. Pandey, D. N. Gupta, M. K. Dharani, N. Sindhwani, J. V. N. Ramesh, Wireless sensor-based IoT system with distributed optimization for healthcare, in *Meta Heuristic Algorithms for Advanced Distributed Systems*, Wiley Data and Cybersecurity, (2024), 261–288. https://doi.org/10.1002/9781394188093.ch16

19. X. Cai, K. Shi, Y. Sun, J. Cao, S. Wen, C. Qiao, et al., Stability analysis of networked control systems under DoS attacks and security controller design with mini-batch machine learning supervision, *IEEE Trans. Inf. Forensics Secur.*, **19** (2023). 3857–3865. https://doi.org/10.1109/TIFS.2023.3347889

20. E. Nozari, P. Tallapragada, J. Cortés, Differentially private average consensus: obstructions, trade-offs, and optimal algorithm design, *Automatica*, **81** (2017), 221–231. https://doi.org/10.1016/j.automatica.2017.03.016

21. Z. Huang, S. Mitra, N. Vaidya, Differentially private distributed optimization, in *Proceedings of the 16th International Conference on Distributed Computing and Networking*, (2015), 1–10. https://doi.org/10.1145/2684464.268448

22. Y. Xiong, J. Xu, K. You, J. Liu, L. Wu, Privacy-preserving distributed online optimization over unbalanced digraphs via subgradient rescaling, *IEEE Trans. Control Network Syst.*, **7** (2020), 1366–1378. https://doi.org/10.1109/TCNS.2020.2976273

23. E Nozari, P. Tallapragada, J. Cortés, Differentially private distributed convex optimization via functional perturbation, *IEEE Trans. Control Network Syst.*, **5** (2016), 395–408. https://doi.org/10.1109/TCNS.2016.2614100

24. T. Ding, S. Zhu, J. He, C. Chen, Differentially private distributed optimization via state and direction perturbation in multiagent systems. *IEEE Trans. Autom. Control*, **67** (2021), 722–737. https://doi.org/10.1109/TAC.2021.3059427

25. Y. Wang, T. Başar, Decentralized nonconvex optimization with guaranteed privacy and accuracy, *Automatica*, **150** (2023), 110858. https://doi.org/10.1016/j.automatica.2023.110858

26. H. Wang, Q. Zhao, Q. Wu, S. Chopra, A. Khaitan, H. Wang, Global and local differential privacy for collaborative bandits, in *Proceedings of the 14th ACM Conference on Recommender Systems*, (2020), 150–159. https://doi.org/10.1145/3383313.3412254

27. W. Lin, B. Li, C. Wang, Towards private learning on decentralized graphs with local differential privacy, *IEEE Trans. Inf. Forensic Secur.*, **17** (2022), 2936–2946. https://doi.org/10.1109/TIFS.2022.3198283

28. Y. Zhao, J. Zhao, M. Yang, T. Wang, N. Wang, L. Lyu, et al., Local differential privacy-based federated learning for internet of things, *IEEE Internet Things J.*, **8** (2020), 8836–8853. https://doi.org/10.1109/JIOT.2020.3037194

29. C. Zhang, Y. Wang, Enabling privacy-preservation in decentralized optimization, *IEEE Trans. Control Network Syst.*, **6** (2018), 679–689. https://doi.org/10.1109/TCNS.2018.2873152

30. C. N. Hadjicostis, A. D. Domínguez-García, Privacy-preserving distributed averaging via homomorphically encrypted ratio consensus, *IEEE Trans. Autom. Control*, **65** (2020), 3887–3894. https://doi.org/10.1109/TAC.2020.2968876

31. C. Zhang, M. Ahmad, Y. Wang, ADMM based privacy-preserving decentralized optimization, *IEEE Trans. Inf. Forensics Secur.*, **14** (2018), 565–580. https://doi.org/10.1109/TIFS.2018.2855169

32. Y. Lu, M. Zhu, Privacy preserving distributed optimization using homomorphic encryption, *Automatica*, **96** (2018), 314–325. https://doi.org/10.1016/j.automatica.2018.07.005

33. Y. Wang, Privacy-preserving average consensus via state decomposition, *IEEE Trans. Autom. Control*, **64** (2019), 4711–4716. https://doi.org/10.1109/TAC.2019.2902731

34. S. Lin, F. Wang, Z. Liu, Z. Chen, Privacy-preserving average consensus via enhanced state decomposition, in *2022 41st Chinese Control Conference (CCC)*, (2022), 4484–4488. https://doi.org/10.23919/CCC55666.2022.9902777

35. X. Chen, L. Huang, K. Ding, S. Dey, L. Shi, Privacy-preserving push-sum average consensus via state decomposition, *IEEE Trans. Autom. Control*, **68** (2023), 7974–7981. https://doi.org/10.1109/TAC.2023.3256479

36. X. Chen, W. Jiang, T. Charalambous, L. Shi, A privacy-preserving finite-time push-sum based gradient method for distributed optimization over digraphs, *IEEE Control Syst. Lett.*, **7** (2023), 3133–3138. https://doi.org/10.1109/LCSYS.2023.3292463

37. J. Zhang, J. Q. Lu, C. N. Hadjicostis, Average consensus for expressed and private opinions, *IEEE Trans. Autom. Control*, **69** (2024), 5627–5634. https://doi.org/10.1109/TAC.2024.3379256

38. S. Pu, W. Shi, J. Xu, A. Nedic, Push-pull gradient methods for distributed optimization in networks, *IEEE Trans. Autom. Control*, **66** (2020), 1–16. https://doi.org/10.1109/TAC.2020.2972824

AIMS Press