An Event-Triggered Approach for Gradient Tracking in Consensus-Based Distributed Optimization

Lan Gao, Shaojiang Deng, Huaqing Li, *Senior Member, IEEE*, and Chaojie Li, *Member, IEEE*

***Abstract*—This paper is concerned with a communication-efficient algorithm update scheme for solving distributed convex optimization problems by introducing a distributed event-triggered approach. Compared with real-time consensus-based distributed optimization algorithms in the literature, this paper focuses on extending the real-time gradient tracking scheme and proposes a novel distributed event-triggering condition to reduce the frequency of information exchange between agents in a network. The proposed event-triggered approach for consensus-based distributed optimization algorithms not only avoids the real-time consecutive communication and the coordinated computation between agents but reduces the computation load of algorithm execution. Furthermore, the proposed event-triggering condition depends on only local states from neighbors at only their event times and does not require a global and homogeneous sampling period. In addition, this paper analytically shows that the proposed consensus-based distributed optimization algorithm based on an event-triggered approach can also converge to the exact optimal solution with a linear convergence rate even if the real-time consecutive communication between agents is replaced with sporadic communication.**

***Index Terms*—Consensus-based distributed optimization, gradient tracking, event-triggered approach, multi-agent networks.**

I. Introduction

DISTRIBUTED convex optimization problems arise in various fields such as multi-vehicle cooperative control [1], [2], distributed filtering and estimation [3], [4], smart grid implementation [5], [6], cloud computing and learning [7], [8], and large-scale machine learning [9], [10]. To solve these problems in a distributed manner, optimization algorithms need to be designed using only local available information based on local communication. That is, each agent in multi-agent networks can use only its own private objective function and can exchange state estimates only with its neighbors.

Regarding distributed optimization algorithms, gradient descent methods have attracted increasing attention due to its high efficiency and easy implementation in recent years. For gradient descent methods, there are at least two main strategies: the consensus strategy and the diffusion strategy. Existing works on consensus strategy include the subgradient method [11], [12], the stochastic subgradient projection method [13], [14], the fast subgradient method [15] and the subgradient-push method [16], [17]. Existing works on diffusion strategy can be found in [18], [19], which employ a symmetric iteration update instead of an asymmetric one compared with the consensus strategy. However, the above works on both strategies all suffer a dilemma that the iteration converges to an exact solution slowly when a diminishing step size is used. It converges faster when a fixed step size is considered but only stalls at a neighborhood of the exact solution. To overcome this dilemma, an EXTRA algorithm is proposed in [20], which can converge to the exact optimal solution with a linear rate even though a fixed and large step size is used. The EXTRA algorithm is actually a modified implementation of the consensus strategy. Instead of directly using a subgradient, EXTRA employs the difference of the subgradients at the last two iterates to approximate the global gradient and then uses the last two iterates' solution estimates to update the current solution estimate. Motivated by the idea of EXTRA, a DEXTRA algorithm for directed graphs is developed in [21] and an exact diffusion algorithm is developed in [22]. Independently, an Aug-DGM algorithm based on the Adapt-then-Combine strategy in [18], [19] and the average consensus strategy in [23] are developed in [24]. The Aug-DGM algorithm not only employs uncoordinated step sizes for different agents but can produce an exact optimal solution with a satisfactory convergence rate. Similar to [24], a DIGing algorithm based on the combination of the distributed subgradient method and the dynamic average gradient tracking technique is developed in [25]. The DIGing algorithm differs from Aug-DGM mainly in the gradient tracking update. That is, DIGing employs a direct and simpler scheme instead of the Adapt-then-Combine scheme in Aug-DGM. It is worth emphasizing that the DIGing algorithm can ensure that all agents in a network converge to the exact optimal solution linearly with a fixed step size.

Manuscript received January 13, 2021; revised August 16, 2021; accepted October 17, 2021. Date of publication October 27, 2021; date of current version March 23, 2022. This work was supported in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LQ20F030020, in part by the National Natural Science Foundation of China under Grant 62173278, in part by the Australian Research Council under Grants IH180100020, DP200101197, and DE210100274, and in part by UNSW Digital Grid Futures Institute. Recommended for acceptance by Dr. Atilla Eryilmaz. *(Corresponding author: Huaqing Li)*

Lan Gao is with the Hangzhou Innovation Institute of Beihang University, Hangzhou 310023, China (e-mail: langaouc@gmail.com).

Shaojiang Deng is with the School of Computer Science and Technology, Chongqing University, Chongqing 400044, China (e-mail: sj_deng@cqu.edu.cn).

Huaqing Li is with the Chongqing Key Laboratory of Nonlinear Circuits and Intelligent Information Processing, College of Electronic and Information Engineering, Southwest University, Chongqing 400715, PR China (e-mail: huaqingli@hotmail.com).

Chaojie Li is with the School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, NSW 2052, Australia (e-mail: cjlee.cqu@163.com).

Digital Object Identifier 10.1109/TNSE.2021.3122927

The aforementioned standard optimization algorithms, however, have some common limitations as follows. To begin with, the information exchange between agents is real-time and frequent. That is, each agent has to collect and broadcast solution estimates from/to all its neighbors at every point of sampling time, which is obviously inefficient and has to possess much expensive communication bandwidth. Furthermore, it is supposed that all agents are able to complete their local computations synchronously and then synchronously transmit local results to their neighbors at each time instant. However, this assumption is not realistic because the data flow, communication status, and computation abilities of agents often vary in a heterogeneous network. As a result, the synchronization of the computation and communication abilities between agents might not be accomplished in a real network environment. Actually, the overall efficiency of a distributed optimization algorithm mainly depends on communication efficiency rather than computation efficiency because the time consumption of communication between computing nodes is far more than that of computation in a distributed computing network system. Therefore, it is interesting to seek a communication-efficient optimization approach to make sure that the real-time consecutive communication and the coordinated computation between agents can be avoided.

The event-triggered approach provides a new perspective on how to collect and transmit information between agents in an interactive network [26], [27]. The early works introducing the event-triggered approach to the field of convex optimization can be found in [28], [29], where the communication cost is reduced effectively by employing an event-triggering condition to determine whether the local state information should be transmitted or not. However, the proposed event-triggered optimization approaches are actually based on a centralized idea because the augmented Lagrangian method in [28] and the gradient descent method in [29] are both centralized and both require that each agent know all other agents' state estimates to update its own state estimate even though the computation load can be distributed among different computing nodes. More recently, some distributed asynchronous optimization algorithms based on the randomized alternating direction method of multipliers (ADMM) are proposed in [30], [31]. But the computation cost of the proposed algorithms is expensive since there is at least one optimal solution that needs to be solved in the execution of each iteration. Furthermore, some works including [32]–[34] investigated event-triggered optimization schemes based on (sub)gradient methods with time-varing or constant stepsizes, however, they all inevitably suffer a dilemma between convergence rate and convergence accuracy. Also, event-triggered communication schemes for particle swarm optimization, communication-constrained optimization and optimal control can be found in [35]–[37]. In particular, the work in [38] extended the consensus-based distributed optimization algorithm in [25] with a distributed event-triggered optimization approach, which utilizes the state estimates of the event times instead of that of the real time to update local results. However, the proposed event-triggering condition is not satisfactory because it depends on a global time-varying threshold and the key parameters not only depend on the global communication topology but they are sensitive to all agents' initial states and measurement errors.

The aim of this paper is to develop a communication-efficient optimization approach for the consensus-based distributed optimization algorithm (also named as DIGing algorithm) in [25] based on an event-triggered control strategy to avoid the real-time consecutive communication and the synchronous state fusion among different agents effectively. Compared with existing works in [25], [38], we still focus on an extended DIGing algorithm based on event times instead of the real time in the communication network. However, a significant challenge is how to design a proper event-triggering condition that relies on only local states from neighbors at only their sporadic event times instead of the real time such that the real-time information collection of the network can also be avoided in the implementation of the event-triggering condition. Moreover, how to give a theoretical analysis to show the feasibility of the proposed event-triggering condition is another main challenge.

The contribution of this paper is to develop an event-triggered communication-efficient control strategy for the consensus-based distributed optimization algorithm and design a novel distributed event-triggering condition. The proposed control strategy gives an event-triggered solution on the tracking of both gradient and state aggregations so that the information exchange between agents can be non-real time and more efficient. The proposed event-triggering condition depends on only local states from neighbors at only their sporadic event times, which implies that the real-time information collection and the coordinated computation between agents are avoided not only in the update of the iteration algorithm but in the implementation of the event-triggering condition. Furthermore, the selections of the key parameters are simple and their upper bounds are independent of the initial states and the measurement errors. In addition, the convergence analysis is given to show that the extended DIGing algorithm can still converge to the exact optimal solution with a linear convergence rate under the proposed event-triggering condition.

This paper is organized as follows: Section II introduces some necessary preliminary knowledge about notation, graph theory, and the DIGing algorithm; Section III introduces a distributed event-triggered approach to the DIGing algorithm; Section IV presents the main theorem and convergence analysis; Section V provides some numerical simulations to illustrate the main theorem; Section VI concludes the paper.

II. Preliminaries and Background

A. Notation

The notations throughout this paper are defined as follows. The set of all real numbers and real vectors with p dimensions are denoted by $\mathbb{R}$ and $\mathbb{R}^p$, respectively. The 1 vector and 0 vector containing N entries are denoted by $\mathbf{1}_N$ and $\mathbf{0}_N$, respectively. Let I be a unit matrix. Let $||v||$ and $||A||$ represent the Euclidean norms of a vector v and a matrix A, respectively.

The transposes of a vector v and a matrix A are denoted by v^T and A^T, respectively. For a matrix A, the maximum singular value is denoted by $\sigma_{max}(A)$. For any function $f(x)$, the gradient of f at x is denoted by $\nabla f(x)$. For an infinite sequence $s_i = \{s_i(0), s_i(1), s_i(2), \ldots\}$, denote $||s_i||^{\lambda,K} = \max_{t=0,1,\ldots,K} \frac{1}{\lambda^t}||s_i(t)||$ and $||s_i||^{\lambda} = \sup_{t\geq 0} \frac{1}{\lambda^t}||s_i(t)||$ with $\lambda \in (0,1)$.

B. Graph Theory

Define $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\}$ as an undirected graph composed of N nodes, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ is the node set, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set and $A = (a_{ij}) \in \mathbb{R}^{N\times N}$ is the adjacency matrix of $\mathcal{G}$. An edge $e_{ji} = (v_j, v_i)$ represents that the information from node j can arrive to node i directly. Here the adjacency matrix A is symmetric. That is, the communication channels between nodes in a network are two-way. If there is a two-way communication channel between node i and node j, then they are called neighbors of each other and accordingly $a_{ij} = a_{ji} = 1$; otherwise, $a_{ij} = a_{ji} = 0$. The set of neighbors of node i is denoted by $\mathcal{N}_i$. Let N and $N_i = |\mathcal{N}_i|$ denote the number of all nodes and the number of neighbors of node i in a network, respectively. The Laplacian matrix of A is defined as $L = (l_{ij}) \in \mathbb{R}^{N\times N}$, where $l_{ij} = -a_{ij}, i \neq j$ and $l_{ii} = \sum_{j=1,j\neq i}^{N} a_{ij}$. Here L is a symmetric positive semidefinite matrix. A square matrix $W = (w_{ij})$ with nonnegative real numbers is called doubly stochastic if the sums of each row and each column both equal to 1, i.e., $\sum_i w_{ij} = \sum_j w_{ij} = 1$.

C. DIGing Algorithm

This subsection briefly reviews the consensus-based distributed optimization algorithm proposed in [25], which is also named as DIGing algorithm by [25]. Consider the following distributed unconstrained optimization problem

$$\min_{x\in\mathbb{R}^p} \quad f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x), \tag{1}$$

where the global cost function $f(x)$ is separable and each local cost function $f_i(x)$ is convex, differentiable and only available to agent i.

To solve the problem (1) with a distributed computation method, the DIGing algorithm exhibits good performance on iteration complexity, convergence accuracy and convergence rate because DIGing is not only a simple first-order algorithm but it can converge to the exact optimal solution with a linear convergence rate. The DIGing algorithm executes the following updates at each iteration [25]:

$$\begin{aligned} x_i(t+1) &= x_i(t) + hu_i(t) - \alpha y_i(t), y_i(t+1) \\ &= y_i(t) + hv_i(t) + \nabla f_i(x_i(t+1)) - \nabla f_i(x_i(t)), \end{aligned} \tag{2}$$

where $x_i(t) \in \mathbb{R}^p$ is the solution estimate, $y_i(t) \in \mathbb{R}^p$ is the average gradient estimate, h is a positive control gain, α is a fixed step size and the auxiliary variables $u_i(t)$ and $v_i(t)$ are defined as follows:

$$\begin{aligned} u_i(t) &= \sum_{j\in\mathcal{N}_i} a_{ij}\big(x_j(t) - x_i(t)\big), \\ v_i(t) &= \sum_{j\in\mathcal{N}_i} a_{ij}\big(y_j(t) - y_i(t)\big). \end{aligned} \tag{3}$$

At each iteration time $t+1$, each agent i first calculates the auxiliary variables $u_i(t)$ and $v_i(t)$ utilizing its own previous $x_i(t)$ and $y_i(t)$ and the received $x_j(t)$ and $y_j(t)$ from its neighbors to complete the state fusion. Then, each agent i updates the current solution estimate $x_i(t+1)$ utilizing the previous $x_i(t)$ and auxiliary variable $u_i(t)$ at the direction of $-y_i(t)$. Also, each agent i updates the current average gradient estimate $y_i(t+1)$ utilizing the previous $y_i(t)$, auxiliary variable $v_i(t)$ and the gradient-difference term $\nabla f_i(x_i(t+1)) - \nabla f_i(x_i(t))$. Finally, each agent i broadcasts its current solution estimate $x_i(t+1)$ and average gradient estimate $y_i(t+1)$ to all its neighbors and meanwhile receives the solution estimate $x_j(t+1)$ and the average gradient estimate $y_j(t+1)$ from its neighbors.

III. Problem Statement and Algorithm Design

This section mainly focuses on how to introduce the event-triggered approach to the DIGing algorithm. Though the proposed DIGing algorithm in [25] exhibits good performance on iteration complexity, convergence accuracy and convergence rate, DIGing has to face a fact that the iteration is based on the real-time communication and the coordinated state fusion. That is, each agent has to broadcast its own state estimate and collect all its neighbors' state estimates at each time instant, and then complete the state fusion at the same time with other agents, which implies that each agent has to deal with heavy communication and computation load. Thus, it is interesting to design a more efficient control approach for DIGing to avoid the real-time communication and the coordinated state fusion.

A. Extended DIGing Based on an Event-Triggered Approach

The event-triggered approach provides a new perspective for the iteration update of the consensus-based distributed optimization algorithm. Before introducing the event-triggered approach, we first define a sequence $\{0 = t_0^i, t_1^i, t_2^i, \ldots\}$ for the sporadic event times of agent i. Besides, we suppose that each agent in a network can only receive the state estimates from its neighbors at their event times. As a result, the real-time consecutive state exchange is not available in a network. Therefore, the auxiliary variables $u_i(t)$ and $v_i(t)$ in (3) are redesigned by using the state estimates at only event times as follows:

$$\begin{aligned} u_i(t) &= \sum_{j\in\mathcal{N}_i} a_{ij}\Big(x_j(t_{k_j}^j) - x_i(t_{k_i}^i)\Big), v_i(t) \\ &= \sum_{j\in\mathcal{N}_i} a_{ij}\Big(y_j(t_{k_j}^j) - y_i(t_{k_i}^i)\Big), t \in [t_{k_i}^i, t_{k_i+1}^i), \end{aligned} \tag{4}$$

where $k_i = \arg\min_{l\in\mathbb{N}, t>t_l^i}\{t - t_l^i\}$ denotes agent i's latest event number. $x_i(t_{k_i}^i)$ and $y_i(t_{k_i}^i)$ represent the solution estimate and the average gradient estimate of agent i at its latest

Algorithm 1. The communication and control processes of the event-triggered DIGing algorithm

Initializes and broadcasts $x_i(t_0^i), y_i(t_0^i)$ to neighbors
repeat
  Detects if a message is received or not
  if a message from neighbor j is received**then**
    updates the state estimates $x_j(t_{k_j}^j), y_j(t_{k_j}^j)$
    updates auxiliary variables $u_i(t)$ and $v_i(t)$

$$u_i(t) = \sum_{j\in\mathcal{N}_i} a_{ij}(x_j(t_{k_j}^j) - x_i(t_{k_i}^i)),$$
$$v_i(t) = \sum_{j\in\mathcal{N}_i} a_{ij}(y_j(t_{k_j}^j) - y_i(t_{k_i}^i)).$$

else
  keeps local memory constant
end if
Updates state estimates $x_i(t+1)$ and $y_i(t+1)$

$$x_i(t+1) = x_i(t) + hu_i(t) - \alpha y_i(t),$$
$$y_i(t+1) = y_i(t) + hv_i(t) + \nabla f_i(x_i(t+1)) - \nabla f_i(x_i(t)).$$

Judges if an event occurs or not
if an event occurs at agent i**then**
  updates the latest event time: $t_{k_i}^i = t+1$
  updates state estimates $x_i(t_{k_i}^i), y_i(t_{k_i}^i)$
  updates auxiliary variables $u_i(t)$ and $v_i(t)$
  broadcasts $x_i(t_{k_i}^i), y_i(t_{k_i}^i)$ to neighbors
else
  keeps local memory constant
  keeps silent
end if
$t = t+1$
until stop condition satisfies

event time, respectively. Also, $x_j(t_{k_j}^j)$ and $y_j(t_{k_j}^j)$ represent the solution estimates and the average gradient estimates received from the neighbors of agent i at their latest event times, respectively.

Compared with the original forms in Eq. (3), the redesigned auxiliary variables $u_i(t)$ and $v_i(t)$ in (4) mainly depend on the state estimates at the latest event time instead of the real time and are responsible for state fusion with less communication and computation cost. Specifically, in time interval $t \in [t_{k_i}^i, t_{k_i+1}^i)$, if there is no event occurring from itself or its neighbors, the computation of $u_i(t)$ and $v_i(t)$ still utilizes the existing state estimate $x_i(t), y_i(t), x_j(t), y_j(t)$ at the latest event time $t = t_{k_i}^i$ and $t = t_{k_j}^j$. Once an event occurs at agent i, the agent broadcasts the state estimates $x_i(t_{k_i}^i)$ and $y_i(t_{k_i}^i)$ to its neighbors; otherwise, the agent remains silent. In other words, the auxiliary variables $u_i(t)$ and $v_i(t)$ will remain constant in the time interval $[t_{k_i}^i, t_{k_i+1}^i)$ until a new event time $t_{k_i+1}^i$ comes at agent i or agent i receives at least one new state estimate from its neighbors.

To clearly describe the communication process of the extended DIGing with the event-triggered approach, a detailed communication algorithm is provided in Algorithm 1. Suppose that each agent i is equipped with a local memory, which is used to store the solution estimates $x_i(t), x_i(t_{k_i}^i)$, the average gradient estimates $y_i(t), y_i(t_{k_i}^i)$ and the received latest state estimates $x_j(t_{k_j}^j), y_j(t_{k_j}^j)$ from its neighbors. Furthermore, the initial states of all agents are given by $x(0) = (x_1(0), \ldots, x_N(0))^T$, $y(0) = (y_1(0), \ldots, y_N(0))^T = (\nabla f_1(x_1(0)), \ldots, \nabla f_N(x_N(0)))^T$ and the initial event time t_0^i for each agent i is initialized to 0. After the initialization is completed at each agent i, the agent begins to execute Algorithm 1.

Note that the event-triggered strategy plays an important role in avoiding real-time consecutive communication and the frequent updates of auxiliary variables. As we can see, agent i broadcasts its own state estimate to its neighbors only at its event time. That is to say, if there is no event at the current time t, the current real-time state estimates $x_i(t), y_i(t)$ will not to be broadcast even though their computations have been completed at agent i. Furthermore, agent i updates the latest state estimates and auxiliary variables using its own or its neighbors' latest state estimates only if the agent detects that an event occurs or some messages from its neighbors are received. Consequently, the extended DIGing with an event-triggered approach avoids the real-time consecutive communication and the coordinated calculation compared with the existing real-time DIGing algorithm in [25]. It is worth noting that the extended DIGing algorithm still needs to depend on global time coordination even though the information exchange between agents is asynchronous since the iteration of the algorithm is based on a global time clock. Next, we will discuss how to judge if an event occurs or not in Algorithm 1 by introducing a novel distributed event-triggering condition.

B. A Distributed Event-Triggering Condition

Define two measurement errors associated with the solution estimate and the average gradient estimate as follows

$$e_i^x(t) = x_i(t_{k_i}^i) - x_i(t), \quad e_i^y(t) = y_i(t_{k_i}^i) - y_i(t). \tag{5}$$

Actually, (5) basically describes the degree that the state estimates at the latest event time deviate from the state estimates at the current time. In practice, some thresholds are usually specified for the measurement errors $e_i^x(t)$ and $e_i^y(t)$ in advance. Once the threshold is reached, an event is triggered and the measurement errors $e_i^x(t)$ and $e_i^y(t)$ are both reset to zero because $e_i^x(t) = x_i(t_{k_i}^i) - x_i(t_{k_i}^i) = 0, e_i^y(t) = y_i(t_{k_i}^i) - y_i(t_{k_i}^i) = 0$ at the event time $t = t_{k_i}^i$.

Substituting (4) and (5) into (2), the iteration (2) can be rewritten in a compact form as follows:

$$x(t+1) = Wx(t) - hLe^x(t) - \alpha y(t), y(t+1) = Wy(t) - hLe^y(t) + \nabla f(x(t+1)) - \nabla f(x(t)), \tag{6}$$

where $W = I - hL$, $x(t) = (x_1(t), \ldots, x_N(t))^T, y(t) = (y_1(t), \ldots, y_N(t))^T$, $e^x(t) = (e_1^x(t), \ldots, e_N^x(t))^T, e^y(t) =$

$(e_1^y(t), \ldots, e_N^y(t))^T$, and $\nabla f(x(t)) = (\nabla f_1(x_1(t)), \ldots, \nabla f_N(x_N(t)))^T$.

Before introducing our event-triggering condition, we first give the existing event-triggering condition proposed in [38] as follows:

$$t_{k_i+1}^i = \inf\Big\{t \in \mathbb{N}, t > t_{k_i}^i \big| ||e_i^x(t)|| + ||e_i^y(t)|| \geq C\lambda^t\Big\}. \quad (7)$$

Although this condition can avoid the real-time communication and synchronous state fusion, there exist several issues in practical implementation: i) the event-triggering threshold on the right of (7) is a time-varying diminishing function, which implies that this threshold heavily depends on the time parameter; ii) the selection of the parameter λ depends on not only the global communication topology but the objective functions of all agents; iii) it is quite challenging to choose an appropriate parameter C because C not only depends on all agents' objective functions but it is also sensitive to agents' initial states and measurement errors. In a word, the event-triggering condition proposed in [38] has some limitations on the selections of the key parameters and might not apply to various network environments effectively.

Next, our goal is to develop a novel distributed event-triggering condition such that it has good performance on robustness and easy implementation for various network environments. Since the available states to each agent are those state estimates from neighbors at only their event times, the ideal event-triggering condition should be computed by using only these sporadic state estimates. Thus, we propose an alternative event-triggering condition as follows

$$t_{k_i+1}^i = \inf\Big\{t \in \mathbb{N}, t > t_{k_i}^i \big| \max\big(g_1(e_i^x), g_2(e_i^y)\big) \geq 0\Big\}, \quad (8)$$

where

$$g_1(e_i^x) = ||e_i^x(t)||^2 - \gamma_1 h \sum_{j\in\mathcal{N}_i} ||x_j(t_{k_j}^j) - x_i(t_{k_i}^i)||^2,$$
$$g_2(e_i^y) = ||e_i^y(t)||^2 - \gamma_2 h \sum_{j\in\mathcal{N}_i} ||y_j(t_{k_j}^j) - y_i(t_{k_i}^i)||^2. \quad (9)$$

It is worth emphasizing that the event-triggering condition (8) is time invariant and its computation depends on only local states available to each agent. Furthermore, the computation of the second term on the right of (9) is piecewise constant even though the computation of the measurement errors is real time, which implies that the implementation of the event-triggering condition does not need to rely on the real-time consecutive states from neighbors. As will be shown in the next section, the selections of the parameters γ_1 and γ_2 are simple and their upper bounds are independent of agents' initial states and measurement errors. In addition, we should emphasize that Zeno behaviors will not happen in the extended DIGing with the event-triggered approach. As for the Zeno behavior, it describes a unique phenomenon for a hybrid system, where an infinite number of discrete transitions occur in a finite length of time. As we can see, the extended DIGing algorithm (2) with (4) is based on a discrete-time iteration, which implies that the system we study is a discrete-time one rather than a hybrid one. Even if in the worst-case scenario (the event occurs on every point of sampling time), the time interval between two adjacent events is actually one sampling time interval. That is, the number of events in a finite length of time is always bounded in the proposed event-triggered DIGing algorithm.

Note that the storage requirements are very limited in practical terms. To be specific, the variables $x_i(t_{k_i}^i), y_i(t_{k_i}^i), e_i^x(t), e_i^y(t), x_j(t_{k_j}^j), y_j(t_{k_j}^j), j \in \mathcal{N}_i$ for each agent i need additional storage space compared with the traditional DIGing algorithm. In other words, the total storage requirements for each agent increase about two times compared with original storage requirements involving variables $x_i(t), y_i(t), x_j(t), y_j(t), j \in \mathcal{N}_i$. It is worth pointing out that the required storage space for each variable is trivial since it only takes 8 Bytes (64 bits) space even though each variable uses a double-precision format in a computer. Therefore, it is more worthy to focus on the problems of computation and communication instead of storage problems.

IV. Main Results

In this section, we first give some assumptions regarding coupling matrices and objective functions and then provide our main result that the extended DIGing algorithm (2) with the auxiliary variable (4) can converge to the exact optimal solution linearly under the proposed event-triggering condition (8).

Assumption 1 (Connectivity): Suppose that the undirected graph $\mathcal{G}$ is connected.

Assumption 2 (Smoothness): Each objective function f_i is ℓ_i-smooth where $\ell_i > 0$. That is, f_i is differentiable and the gradient is ℓ_i-Lipschitz continuous, i.e.,

$$||\nabla f_i(x) - \nabla f_i(y)|| \leq \ell_i ||x - y||, \forall x, y \in \mathbb{R}.$$

Define $\bar{\ell} = \frac{1}{N}\sum_{i=1}^N \ell_i$ and $\hat{\ell} = \max_{i=1}^N\{\ell_i\}$, which will be used later.

Assumption 3 (Strong 100.onvexity): Each objective function f_i satisfies

$$f_i(x) - f_i(y) \geq \nabla f_i(y)^T(x - y) + \frac{\mu_i}{2}||x - y||^2$$

for some $\mu_i > 0$ and any $x, y \in \mathbb{R}$. Define $\bar{\mu} = \frac{1}{N}\sum_{i=1}^N \mu_i$ and $\hat{\mu} = \max_{i=1}^N\{\mu_i\}$, which will be used later.

To obtain the main result, we first give some key lemmas so that the main result can be derived clearly. Before giving the lemmas, we first state some necessary notations that will be used frequently in the next analysis. Define

$$\bar{x}(t) = \frac{1}{N}\sum_{i=1}^N x_i(t), \quad \bar{y}(t) = \frac{1}{N}\sum_{i=1}^N y_i(t),$$
$$\hat{x}(t) = x(t) - \mathbf{1}_N\bar{x}(t), \quad \hat{y}(t) = y(t) - \mathbf{1}_N\bar{y}(t),$$
$$z(t) = \nabla f(x(t)) - \nabla f(x(t-1)), \quad J = \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^T.$$

Define x^* as the globally optimal solution of the problem (1) and let

$$q(t) = x(t) - \mathbf{1}_N x^*.$$

Lemma 1 (Double Stochasticity): [20] Let Assumption 1 hold and let $0 < h < 1/d^*$, where $d^* = \max_{i=1}^N\{d_i\}$ and $d_i = \sum_{j=1}^N a_{ij}$. Then matrix $W = I - hL$ is doubly stochastic and

$$\delta = \sigma_{max}(W - J) < 1.$$

Lemma 2 (Small Gain Theorem): [25] Suppose that $s_1, s_2, \ldots, s_m$ are sequences such that for all positive integers K and for each $i = 1, \ldots, m$, we have

$$||s_{(i \bmod m)+1}||^{\lambda,K} \le \varphi_i ||s_i||^{\lambda,K} + \omega_i,$$

where $\varphi_1, \ldots, \varphi_m$ and $\omega_1, \ldots, \omega_m$ are nonnegative constants and satisfy $0 \le \prod_{i=1}^m \varphi_i < 1$. Then,

$$||s_1||^{\lambda} \le \left(\frac{1}{1 - \Pi_{i=1}^m \varphi_i}\right) \sum_{i=1}^m \omega_i \prod_{j=i+1}^m \varphi_i.$$

Similarly, one can get the bound of $||s_i||^{\lambda}$ for each $i = 2, \ldots, m$.

Lemma 3: Considering the event-triggering condition in (8), we have for any $\gamma_1, \gamma_2 \in (0, \frac{1}{4\ N^2 h})$ that

$$||e^x(t)|| \le \beta_1 ||\hat{x}(t)||,\ \beta_1 = 2\ N\sqrt{\frac{h\gamma_1}{1 - 4\ N^2 h\gamma_1}}, \tag{10a}$$

$$||e^y(t)|| \le \beta_2 ||\hat{y}(t)||,\ \beta_2 = 2\ N\sqrt{\frac{h\gamma_2}{1 - 4\ N^2 h\gamma_2}}. \tag{10b}$$

Proof: According to the event-triggering condition (8), once the measurement error $e_i^x(t)$ or $e_i^y(t)$ reaches a threshold defined on the right of (9), an event is triggered and the measurement errors $e_i^x(t)$ and $e_i^y(t)$ are both reset to zero automatically. Thus, the following inequalities can be guaranteed in the implementation of the event-triggered approach

$$||e_i^x(t)||^2 \le \gamma_1 h \sum_{j\in\mathcal{N}_i} ||x_j(t_{k_j}^j) - x_i(t_{k_i}^i)||^2, \tag{11a}$$

$$||e_i^y(t)||^2 \le \gamma_2 h \sum_{j\in\mathcal{N}_i} ||y_j(t_{k_j}^j) - y_i(t_{k_i}^i)||^2. \tag{11b}$$

From $(11a)$, we have

$$\begin{aligned}
||e_i^x(t)||^2 &\le \gamma_1 h \sum_{j\in\mathcal{N}_i} ||x_j(t) + e_j^x(t) - x_i(t) - e_i^x(t)||^2 \\
&= \gamma_1 h \sum_{j\in\mathcal{N}_i} ||\hat{x}_j(t) - \hat{x}_i(t) + e_j^x(t) - e_i^x(t)||^2 \\
&\le 2\gamma_1 h \sum_{j\in\mathcal{N}_i} \left[2(||\hat{x}_j(t)||^2 + ||\hat{x}_i(t)||^2) + 2(||e_j^x(t)||^2 + ||e_i^x(t)||^2)\right] \\
&\le 4\ N\gamma_1 h(||\hat{x}(t)||^2 + ||e^x(t)||^2).
\end{aligned} \tag{12}$$

Using a similar derivation in (12) for $(11b)$, we have

$$||e_i^y(t)||^2 \le 4\ N\gamma_2 h(||\hat{y}(t)||^2 + ||e^y(t)||^2). \tag{13}$$

Substituting $||e_i^x(t)||^2, ||e_i^y(t)||^2$ with $||e^x(t)||^2, ||e^y(t)||^2$ in (12) and (13), we further have

$$||e^x(t)||^2 \le 4\ N^2\gamma_1 h(||\hat{x}(t)||^2 + ||e^x(t)||^2), \tag{14a}$$

$$||e^y(t)||^2 \le 4\ N^2\gamma_2 h(||\hat{y}(t)||^2 + ||e^y(t)||^2), \tag{14b}$$

which is equivalent to (10). □

Remark 1. Note that Lemma 3 not only establishes a connection between the measurement errors $e_i^x(t), e_i^y(t)$ and the auxiliary variables $\hat{x}(t), \hat{y}(t)$ but gives an upper bound for parameters γ_1, γ_2 explicitly, which will play an important role in improving the event-triggering efficiency. Generally speaking, the frequency of the triggered events becomes lower when the parameter γ_1 or γ_2 goes larger. In other words, larger parameters γ_1, γ_2 will contribute to higher communication efficiency. However, lower triggered frequency also might lead to longer convergence time, which means that a trade-off between communication efficiency and convergence time might happen in practical implementation.

Lemma 4: [25] Let Assumption 2 hold. We have for any $K \ge 0$ and $\lambda \in (0, 1)$ that

$$||z||^{\lambda,K} \le \hat{\ell}(1 + 1/\lambda)||q||^{\lambda,K}.$$

Proof: See the proof of Lemma 5 in [25]. □

Lemma 5: Let Assumption 1 hold. For $\gamma_1 \in (0, \frac{\vartheta}{4\ N^2 h(1+\vartheta)})$, where $\vartheta = \frac{(1-\delta)^2}{h^2||L||^2}$, let λ be such that $\delta + h\beta_1||L|| < \lambda < 1$. We have for any $K \ge 0$ that

$$||\hat{x}||^{\lambda,K} \le \frac{\alpha}{\lambda - \delta - h\beta_1||L||}||\hat{y}||^{\lambda,K} + \frac{\lambda}{\lambda - \delta - h\beta_1||L||}||\hat{x}(0)||. \tag{15}$$

Proof: According to Lemma 1, the matrix W is doubly stochastic. Let $\hat{J} = I - J$. Then $\hat{J}W = W\hat{J}$ and $J\hat{J} = 0$. According to (6) and Lemma 3, we have

$$\begin{aligned}
||\hat{x}(t+1)|| &= ||(I - J)x(t+1)|| \\
&= ||\hat{J}Wx(t) - hLe^x(t) - \alpha\hat{J}y(t)|| \\
&\le ||W\hat{J}x(t)|| + h||L|| \cdot ||e^x(t)|| + \alpha||\hat{J}y(t)||, \\
&= ||(W - J)\hat{x}(t)|| + h||L|| \cdot ||e^x(t)|| + \alpha||\hat{y}(t)||, \\
&\le (\delta + h\beta_1||L||)||\hat{x}(t)|| + \alpha||\hat{y}(t)||.
\end{aligned} \tag{16}$$

Multiplying $\lambda^{-(t+1)}$ on both sides of (16) yields

$$\lambda^{-(t+1)}||\hat{x}(t+1)|| \le \frac{\delta + h\beta_1||L||}{\lambda}\lambda^{-t}||\hat{x}(t)|| + \frac{\alpha}{\lambda}\lambda^{-t}||\hat{y}(t)||. \tag{17}$$

Taking the maximum over $t = 0, \ldots, K-1$ on both sides of (17) and considering $\lambda^0||\hat{x}(0)|| \le \lambda^0||\hat{x}(0)||$ yield (15).

To ensure that there exists a λ such that $\delta + h\beta_1||L|| < \lambda < 1$ holds, $\delta + h\beta_1||L|| < 1$ must be satisfied. Then we further obtain

$$\gamma_1 \in (0, \frac{\vartheta}{4\ N^2 h(1+\vartheta)}), \tag{18}$$

where $\vartheta = \frac{(1-\delta)^2}{h^2||L||^2}$. The proof is completed. □

Lemma 6: Let Assumption 1 hold. For $\gamma_2 \in (0, \frac{\vartheta}{4\ N^2 h(1+\vartheta)})$, where $\vartheta = \frac{(1-\delta)^2}{h^2||L||^2}$, let λ be such that $\delta + h\beta_2||L|| < \lambda < 1$. Then we have for any $K \geq 0$ that

$$||\hat{y}||^{\lambda,K} \leq \frac{\lambda}{\lambda - \delta - h\beta_2||L||}||z||^{\lambda,K} + \frac{\lambda}{\lambda - \delta - h\beta_2||L||}||\hat{y}(0)||. \tag{19}$$

Proof: We omit the proof since it is similar to that of Lemma 5. □

Remark 2. Note that the additional parameter constraint $\gamma_1, \gamma_2 \in (0, \frac{\vartheta}{4\ N^2 h(1+\vartheta)})$ compared with Lemma 3 implies that the global topology information is needed because the parameter ϑ depends on the Laplacian matrix L. On the one hand, the computation of the accurate upper bound still has to depend on the global Laplacian matrix L, which is really worth studying in the further work. On the other hand, the upper bound only gives a reference on the selection of the parameter γ_1 or γ_2. In practical terms, the operator might be more inclined to select a conservative parameter γ_1 or γ_2 according to the event-triggering efficiency if the global topology of a network is not available.

Lemma 7: Suppose that Assumption 1, 2 and 3 hold and the step size α and the parameter λ satisfy

$$0 < \alpha \leq \frac{1}{(1+\eta)\hat{\ell}}, \quad \sqrt{1 - \frac{\alpha\bar{\mu}\beta}{\beta+1}} \leq \lambda < 1.$$

Then we have for any $K \geq 0$ that

$$||q||^{\lambda,K} \leq \left(1 + \frac{\sqrt{N}}{\lambda}\sqrt{\frac{\hat{\ell}(1+\eta)}{\eta\bar{\mu}} + \frac{\hat{\mu}\beta}{\bar{\mu}}}\right)||\hat{x}||^{\lambda,K} + 2\sqrt{N}||\bar{x}(0) - x^*||, \tag{20}$$

where $\beta > 0$ and $\eta > 0$ are free parameters.

Proof: Since W is doubly stochastic, multiplying $\frac{1}{N}\mathbf{1}_N^T$ on both sides of the second equation of (6) yields

$$\bar{y}(t+1) - \frac{1}{N}\mathbf{1}_N^T\nabla f(x(t+1)) = \bar{y}(t) - \frac{1}{N}\mathbf{1}_N^T\nabla f(x(t)). \tag{21}$$

Considering $y(0) = \nabla f(x(0))$, we further have

$$\bar{y}(t) = \frac{1}{N}\mathbf{1}_N^T\nabla f(x(t)) = \frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i(t)). \tag{22}$$

Multiplying $\frac{1}{N}\mathbf{1}_N^T$ on both sides of the first equation of (6) further yields

$$\bar{x}(t+1) = \bar{x}(t) - \alpha\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i(t)). \tag{23}$$

Under Assumption 1, 2 and 3, applying Lemma 8 in [25] to (23) yields

$$||\bar{x} - x^*||^{\lambda,K} \leq 2||\bar{x}(0) - x^*|| + \frac{1}{\lambda}\sqrt{\frac{\hat{\ell}(1+\eta)}{\eta\bar{\mu}} + \frac{\hat{\mu}\beta}{\bar{\mu}}}||\hat{x}||^{\lambda,K}, \tag{24}$$

where $\beta > 0$ and $\eta > 0$ are free parameters.

Since

$$q(t) = x(t) - \mathbf{1}_N x^* = x(t) - \mathbf{1}_N\bar{x}(t) + \mathbf{1}_N\bar{x}(t) - \mathbf{1}_N x^* = \hat{x}(t) + \mathbf{1}_N(\bar{x}(t) - x^*),$$

it follows that

$$||q||^{\lambda,K} \leq ||\hat{x}||^{\lambda,K} + \sqrt{N}||\bar{x} - x^*||^{\lambda,K}. \tag{25}$$

Combining (24) and (25), Lemma 7 follows. □

Remark 3. Note that Lemma 7 actually provides a reference for the upper bound of the estimation error implicitly when the key parameters satisfy certain constraints. Furthermore, Lemma 7 plays a key role in implementing the small gain theorem declared in Lemma 2 because it completes the final closed circle of the small gain theorem together with the above-mentioned Lemmas 4, 5 and 6. Now it is time to give the main result more clearly as follows.

Theorem 1. Suppose that Assumption 1, 2 and 3 hold. For any step size

$$\alpha \in \left(0, \frac{1.5(1-\delta)^2}{\bar{\mu}\kappa}\right), \tag{26}$$

where $\kappa = \frac{3\hat{\ell}}{\bar{\mu}}(1 + 2\sqrt{N\hat{\ell}/\bar{\mu}})$, the sequence $\{x(t)\}$ generated by (2) and (4) under the event-triggering condition (8) converges to the vector $\mathbf{1}_N x^*$ at a linear rate of $O(\lambda^t)$ if the following parameter constraint

$$\gamma_1, \gamma_2 \in \left(0, \frac{\vartheta}{4\ N^2 h(1+\vartheta)}\right) \tag{27}$$

is satisfied, where $\vartheta = \frac{(1-\delta)^2}{h^2||L||^2}, 0 < h < 1/d^*, d^* = \max_{i=1}^{N}\{d_i\}$ and $d_i = \sum_{j=1}^{N} a_{ij}$. Besides, the rate λ is given by

$$\lambda = \max\left\{\sqrt{1 - \frac{\alpha\bar{\mu}}{1.5}}, \sqrt{\frac{\alpha\bar{\mu}\kappa}{1.5}} + \delta\right\}. \tag{28}$$

Proof: Combining Lemmas 4, 5, 6 and 7, we have

$$||z||^{\lambda,K} \leq \varphi_1 ||q||^{\lambda,K} + \omega_1, \varphi_1 = \hat{\ell}(1+1/\lambda),$$
$$||\hat{x}||^{\lambda,K} \leq \varphi_2 ||\hat{y}||^{\lambda,K} + \omega_2, \varphi_2 = \frac{\alpha}{\lambda - \delta - h\beta_1 ||L||},$$
$$||\hat{y}||^{\lambda,K} \leq \varphi_3 ||z||^{\lambda,K} + \omega_3, \varphi_3 = \frac{\lambda}{\lambda - \delta - h\beta_2 ||L||},$$
$$||q||^{\lambda,K} \leq \varphi_4 ||\hat{x}||^{\lambda,K} + \omega_4, \varphi_4 = 1 + \frac{\sqrt{N}}{\lambda}\sqrt{\frac{\hat{\ell}(1+\eta)}{\eta\bar{\mu}} + \frac{\hat{\mu}\beta}{\bar{\mu}}},$$

where $\omega_1 = 0, \omega_2 = \frac{\lambda}{\lambda-\delta-h\beta_1||L||}||\hat{x}(0)||, \omega_3 = \frac{\lambda}{\lambda-\delta-h\beta_2||L||}||\hat{y}(0)||$ and $\omega_4 = 2\sqrt{N}||\bar{x}(0) - x^*||$.

Applying Lemma 2 yields that $||q||^{\lambda}$ is bounded when $0 < \varphi_1\varphi_2\varphi_3\varphi_4 < 1$, which is equivalent to

$$\hat{\ell}\left(1+\frac{1}{\lambda}\right)\frac{\alpha}{\lambda-\delta-h\beta_1||L||}\frac{\lambda}{\lambda-\delta-h\beta_2||L||} \times \left(1+\frac{\sqrt{N}}{\lambda}\sqrt{\frac{\hat{\ell}(1+\eta)}{\eta\bar{\mu}}+\frac{\hat{\mu}\beta}{\bar{\mu}}}\right) < 1, \tag{29}$$

where the parameters λ and α satisfy the following constraints

$$\delta + h\beta_1||L|| < \lambda < 1, \quad \delta + h\beta_2||L|| < \lambda < 1,$$
$$\sqrt{1-\frac{\alpha\bar{\mu}\beta}{1+\beta}} \leq \lambda < 1, \quad 0 < \alpha \leq \frac{1}{(1+\eta)\hat{\ell}}. \tag{30}$$

To obtain a concise result, we set the free parameters $\beta = 2\hat{\ell}/\hat{\mu}$ and $\eta = 1$. Then it follows that

$$\alpha \leq \frac{(\lambda-\delta)^2}{\hat{\ell}(1+\lambda)\left(\lambda + 2\sqrt{N\hat{\ell}/\bar{\mu}}\right)}. \tag{31}$$

Considering $\sqrt{1-\frac{\alpha\bar{\mu}\beta}{1+\beta}} \leq \lambda < 1$ in (30), using $1 + 1/\beta \leq 1.5$ yields

$$\alpha \geq \frac{1.5(1-\lambda^2)}{\bar{\mu}}. \tag{32}$$

Thus, combining (31) and (32) yields

$$\alpha \in \left[\frac{1.5(1-\lambda^2)}{\bar{\mu}}, \frac{(\lambda-\delta)^2}{\hat{\ell}(1+\lambda)\left(\lambda+2\sqrt{N\hat{\ell}/\bar{\mu}}\right)}\right]. \tag{33}$$

Next we discuss the validity of the interval in (33). To simplify the analysis, we here consider a smaller interval

$$\alpha \in \left[\frac{1.5(1-\lambda^2)}{\bar{\mu}}, \frac{1.5(\lambda-\delta)^2}{\bar{\mu}\kappa}\right], \tag{34}$$

where $\kappa = \frac{3\hat{\ell}}{\bar{\mu}}(1+2\sqrt{N\hat{\ell}/\bar{\mu}})$. Let $\hat{\beta} = \max\{\beta_1, \beta_2\}$. When λ increases from $\delta + h\hat{\beta}||L||$ to 1, the left-bound of the interval in (34) is monotonically decreasing while the right-bound is monotonically increasing. In particular, when we choose

$$\lambda^* = \frac{\delta + \sqrt{\kappa^2 + (1-\delta^2)\kappa}}{1+\kappa}, \tag{35}$$

the left-bound and the right-bound of the interval in (34) both equal to

$$\Delta = \frac{1.5(\sqrt{\kappa^2+(1-\delta^2)\kappa} - \delta\kappa)^2}{\bar{\mu}\kappa(1+\kappa)^2}. \tag{36}$$

Specifically, we can further verify that when λ increases from $\delta + h\hat{\beta}||L||$ to λ^*, the lower bound of the left-bound equals to the upper bound of the right-bound, that is,

$$\frac{1.5(1-\lambda^2)}{\bar{\mu}} \in \left[\Delta, \frac{1.5\left(1-(\delta+h\hat{\beta}||L||)^2\right)}{\bar{\mu}}\right],$$
$$\frac{1.5(\lambda-\delta)^2}{\bar{\mu}\kappa} \in \left[\frac{1.5(h\hat{\beta}||L||)^2}{\bar{\mu}\kappa}, \Delta\right],$$
$$\alpha \in \left[\frac{1.5(1-\lambda^2)}{\bar{\mu}}, \frac{1.5(\lambda-\delta)^2}{\bar{\mu}\kappa}\right] = \emptyset,$$

which implies that when $\lambda \in (\delta + h\hat{\beta}||L||, \lambda^*)$, the interval in (34) is invalid. On the other hand, when λ increases from λ^* to 1, the upper bound of the left-bound just equals to the lower bound of the right-bound and they both equal to Δ, that is,

$$\frac{1.5(1-\lambda^2)}{\bar{\mu}} \in (0, \Delta],$$
$$\frac{1.5(\lambda-\delta)^2}{\bar{\mu}\kappa} \in \left[\Delta, \frac{1.5(1-\delta)^2}{\bar{\mu}\kappa}\right),$$
$$\alpha \in \left[\frac{1.5(1-\lambda^2)}{\bar{\mu}}, \frac{1.5(\lambda-\delta)^2}{\bar{\mu}\kappa}\right] \neq \emptyset,$$

which implies that when $\lambda \in [\lambda^*, 1)$, the interval in (34) is valid and the corresponding interval is

$$\alpha \in \left(0, \frac{1.5(1-\delta)^2}{\bar{\mu}\kappa}\right). \tag{37}$$

Considering the constraint between α and λ in (34), once the step size α is given, the rate parameter λ can be obtained as follows

$$\lambda = \max\left\{\sqrt{1-\frac{\alpha\bar{\mu}}{1.5}}, \sqrt{\frac{\alpha\bar{\mu}\kappa}{1.5}} + \delta\right\}. \tag{38}$$

Then Theorem 1 follows and the proof is completed. □

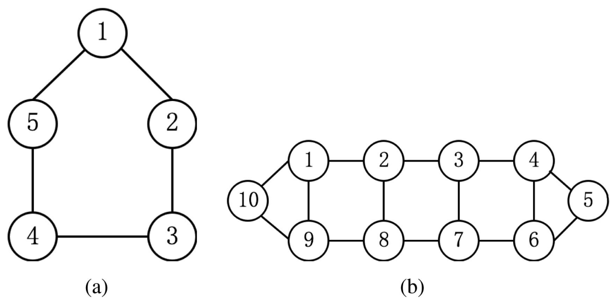


Fig. 1. The communication topology. (a) The network with 5 agents. (b) The network with 10 agents.

Remark 4. Regarding the selection of the step size α in Theorem 1, the upper bound depends on a few of parameters $\delta, \bar{\mu}, \hat{\ell}$. To begin with, it is not very hard to determine the objective parameters $\bar{\mu}, \hat{\ell}$ once the objective functions are given since the averaging and the maximum operations between neighbors can be easily preprocessed by an algorithm for each agent. Moreover, even if it is difficult to obtain the accurate value of the parameter δ without the global topology information of the network, it is possible to evaluate a conservative bound for δ. For example, in the worst case, the bound $\delta \leq 1 - 1/N^3$ can be used (see Remark 2 in [25]). Actually, some other better bounds for different topology structures can be found in the existing works [39], [40].

Remark 5. There are three points worth particular attention. First, the convergence rate λ in (28) is similar with the rate $\lambda = \max\{\sqrt{1 - \frac{\alpha\bar{\mu}}{1.5}}, \sqrt{\frac{\alpha\bar{\mu}J_1}{1.5} + \delta}\}$ in the traditional DIGing algorithm [25] and the key parameters γ_1 and γ_2 of the event-triggered control approach have no direct impact on the linear convergence rate λ. Eq. (28) shows that the linear rate λ mainly depends on the step size α, network parameters N, δ and objective function parameters $\bar{\mu}, \hat{\ell}$ instead of the event-triggering control parameters γ_1 and γ_2, which implies that the linear rate λ is not affected even though an event-triggered control approach is employed to reduce the communication frequency between agents. Second, the parameters γ_1 and γ_2 in event-triggering condition (9) play a key role in reducing communication frequency. To be specific, larger γ_1 and γ_2 will contribute to less communication because larger γ_1 and γ_2 lead to a larger event-triggering threshold which consequently produces larger time interval of communication. Thus the proposed approach is able to adjust the communication efficiency by changing the parameters γ_1 and γ_2 according to practical requirements. Third, the convergence rate λ has a direct impact on the total number of iterations, however, there is no necessary connection between the total number of iterations and the communication frequency between agents. That is to say, the iteration can be continued even if there is no communication happening between neighbors. In traditional algorithms, the iteration updates are based on real-time communication data and every iteration needs at least one time communication to exchange the state estimates. In contrast, our work proves that the real-time communication between neighbors is not necessary to drive the iteration and the reuse of some old exchanged data can effectively reduce communication frequency without compromising convergence time and convergence accuracy.

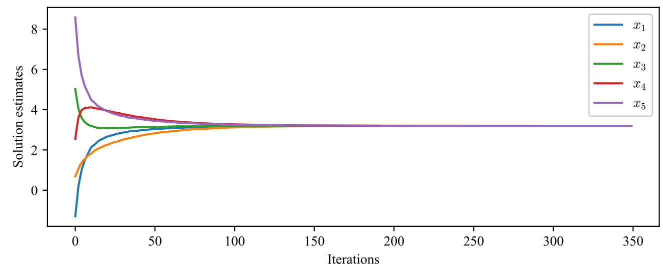


Fig. 2. The evolution of solution estimates.

V. Simulations

A. Decentralized Least Squares

Considering the following least squares problem

$$\min_{x \in \mathbb{R}^p} \quad f(x) = \frac{1}{2}\sum_{i=1}^{N} ||A_i x - b_i||^2.$$

For each cost function $f_i(x) = \frac{1}{2}||A_i x - b_i||^2$, we can obtain that $\nabla f_i(x)$ is Lipschitz continuous and $f(x) = \sum_{i=1}^{N} f_i(x)$ is strongly convex. We first consider the connected communication topology with $N = 5$ agents shown in Fig. 1(a). For simplicity, we choose $x \in \mathbb{R}$ as a scalar and $b_i = 2$ for $i = 1, \ldots, 3$, $b_i = 5$ for $i = 4, 5$ and $A_i = 1$ for all i. Then we can easily obtain the globally optimal solution $x^* = 3.2$. As for control parameters, we set $h = 0.05, \alpha = 0.06, \gamma_1 = \gamma_2 = 0.18$ according to Theorem 1.

Fig. 2 shows the evolution of solution estimates of all agents, from which we can see that the solution estimates in different agents achieve an agreement asymptotically as the iteration goes on. To show the convergence of the solution estimates, we define the relative error $||\epsilon(t)|| = ||x(t) - \mathbf{1}_N x^*||$ to represent the relative distance between the solution estimates and the globally optimal solution. Fig. 3 shows that the relative error converges to zero asymptotically, which implies that the exact optimal solution can be achieved as the iteration goes on. In Fig. 4, the events of all agents are marked in time interval $[0, 350]$, which shows that the sampling of the event time is sporadic rather than consecutive. In particular, the step size of each event is plotted in Fig. 5, which shows that the event step size of each agent contains multiple sampling time intervals.

To show the performance improvement of the extended DIGing compared to the real-time DIGing, we run both algorithms with the same parameters under the same stop condition $||\epsilon(t)|| = ||x(t) - \mathbf{1}_N x^*|| \leq 0.0001$ and obtain the statistical data shown in Table I. We can conclude that the extended DIGing has a large advantage on communication rate (communication times divided by iteration times) compared to the real-time DIGing because it can achieve the same accuracy with lower communication rate (16.9% versus 100%) under similar total iterations (327 versus 359).

To show the impact of the parameters γ_1 and γ_2, we give some statistical data in Table II (assume $\gamma_1 = \gamma_2$). It can be

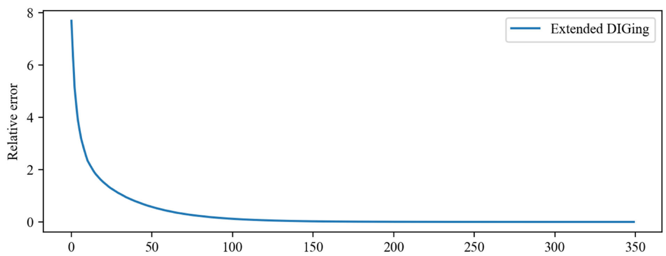


Fig. 3. The error between solution estimates and the globally optimal solution.

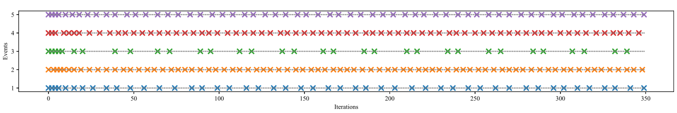


Fig. 4. The event time of each agent $i, i \in 1, \ldots, 5$. Each X mark denotes an event and each little dot denotes a sampling time.

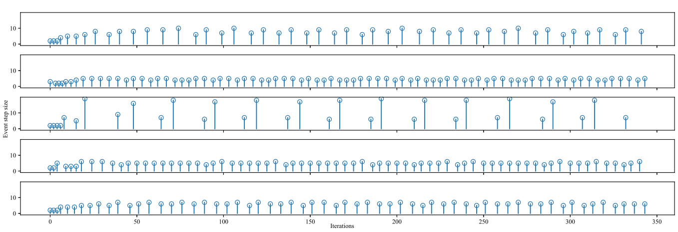


Fig. 5. The event step size of each agent $i, i \in 1, \ldots, 5$. The height of each vertical line denotes the number of sampling time intervals of an event step size.

seen that the larger parameters γ_1 and γ_2 lead to lower communication frequency (higher communication efficiency). Besides, the parameter γ_1 or γ_2 can be greater than the upper bound in Theorem 1, which implies that the theoretical upper bound in Theorem 1 is more conservative than practical bound. However, the agreement convergence cannot be achieved once the parameters γ_1 and γ_2 are much larger than the upper bound, which can be seen in Figs. 6 and 7. As for the step size α, the performance impact of different α is shown in Table III. We can see that the step size α mainly has an impact on the total number of iterations that is related to the convergence rate λ.

We also consider a larger communication network composed of 10 agents as shown in Fig. 1(b). In this case, we set $b_i = 2$ for $i = 1, \ldots, 6$, $b_i = 5$ for $i = 7, \ldots, 10$ and $A_i = 1$ for all i. Then we can obtain the globally optimal solution $x^* = 3.2$. As for control parameters, we set $\gamma_1 = \gamma_2 = 0.045$ according to Theorem 1, h and α are the same as the case of 5 agents. Then we get the simulation results as shown in Figs. 8–11. Note that the upper bound of γ_1 or γ_2 is inversely proportional to the network scale N according to Theorem 1, which implies that larger network scale will lead to a smaller upper bound.

The performance comparison between the real-time DIGing and the extended DIGing in the case of 10 agents is shown in Table IV. It can be seen that the extended DIGing still has a large advantage on communication rate compared to the real-time DIGing because it can achieve the same accuracy with lower communication rate (16.4% versus 100%) under similar total iterations (2504 versus 2541). We can also conclude that the larger network scale will delay the final convergence especially when the connectivity of the network is not large enough. However, the proposed event-triggered control strategy is able to play a key role in reducing the communication frequency even if the large network scale leads to increasing iteration times.

TABLE I
PERFORMANCE COMPARISON BETWEEN DIGING AND EXTENDED DIGING

Performance	DIGing	Extend DIGing
Agreement Convergence	Yes	Yes
Globally Optimal Solution	Yes	Yes
Iteration Times	359	327
Communication Times (Avg)	359	55.2
Communication Rate	100%	**16.9%**

TABLE II
THE PERFORMANCE IMPACT OF DIFFERENT PARAMETERS γ_1 AND γ_2

Performance	0.06	0.09	**0.18**	0.36	1.8	9	18
Agreement	Yes	Yes	**Yes**	Yes	Yes	Yes	No
Iteration Times	338	334	**327**	319	295	268	Infinity
COMM Times	76.2	66.4	**55.2**	45.2	28.8	26	Infinity
COMM Rate	22.5%	19.9%	**16.9%**	14.2%	9.8%	9.7%	None

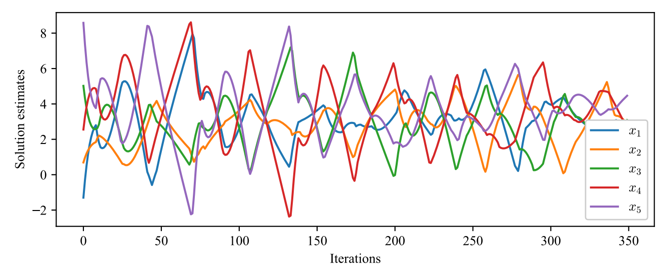


Fig. 6. The agreement cannot be achieved when $\gamma_1 = \gamma_2 = 18.0$.

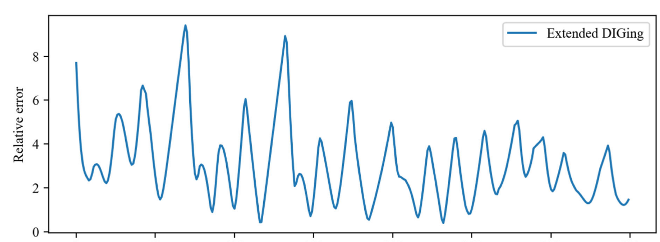


Fig. 7. The error between solution estimates and the globally optimal solution when $\gamma_1 = \gamma_2 = 18.0$.

TABLE III
THE PERFORMANCE IMPACT OF DIFFERENT STEP SIZE α

Performance	0.01	0.02	0.04	**0.06**	0.12	0.18	0.24
Agreement	Yes	Yes	Yes	**Yes**	Yes	Yes	No
Iteration Times	759	378	279	**327**	465	597	Infinity
COMM Times	349	141	51	**55.2**	69.2	81.2	Infinity
COMM Rate	46.0%	37.3%	18.3%	**16.9%**	14.9%	13.6%	None

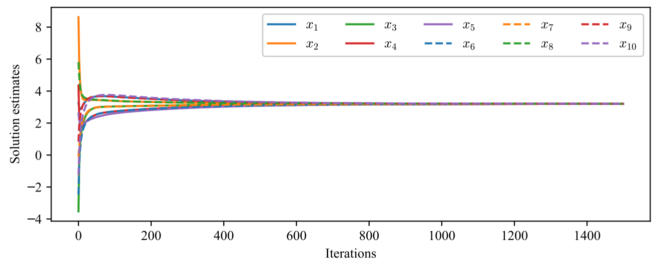


Fig. 8. The evolution of solution estimates in the case of 10 agents.

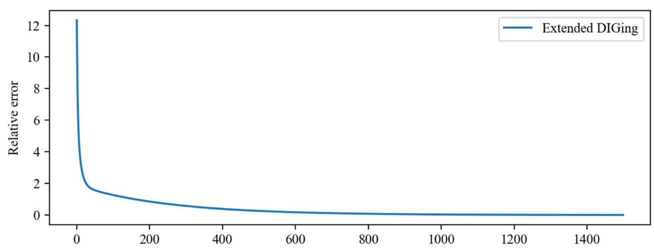


Fig. 9. The error between solution estimates and the globally optimal solution in the case of 10 agents.

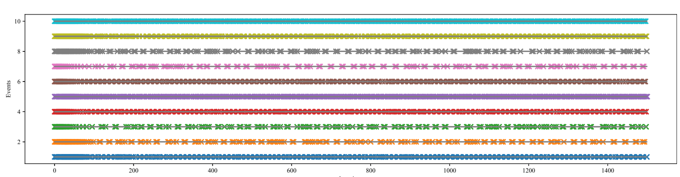


Fig. 10. The event time of each agent $i, i \in 1, \ldots, 10$. Each X mark denotes an event and each little dot denotes a sampling time.

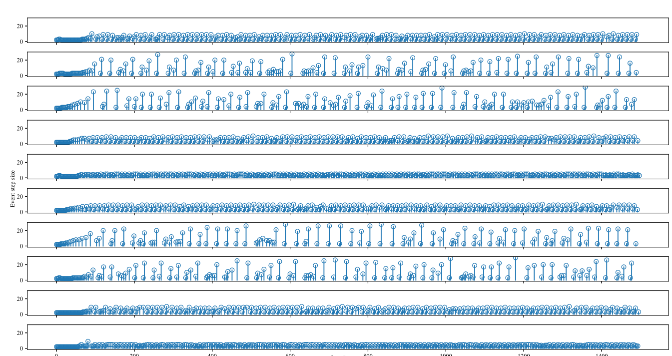


Fig. 11. The event step size of each agent $i, i \in 1, \ldots, 10$. The height of each vertical line denotes the number of sampling time intervals of an event step size.

B. Huber Loss Function

Considering the following decentralized cost function:

$$\min_{x \in \mathbb{R}^p} \quad f(x) = \frac{1}{N} \sum_{i=1}^{N} \left\{ \sum_{j=1}^{m_i} H_\delta (M_{(i)j} x - y_{(i)j}) \right\},$$

where $M_{(i)j}$ is the j-th row of matrix $M_{(i)}$ and $y_{(i)j}$ is the j-th row of vector $y_{(i)}$. The Huber loss function is defined as follows:

$$H_\delta(a) = \begin{cases} \frac{1}{2} a^2, & \text{for } |a| < \delta, \\ \delta(|a| - \frac{1}{2}\delta), & \text{otherwise.} \end{cases}$$

TABLE IV
PERFORMANCE COMPARISON BETWEEN DIGING AND EXTENDED DIGING IN CASE OF 10 AGENTS

Performance	DIGing	Extend DIGing
Agreement Convergence	Yes	Yes
Globally Optimal Solution	Yes	Yes
Iteration Times	2541	2504
Communication Times (Avg)	2541	409.8
Communication Rate	100%	**16.4%**

We first consider the network topology composed of 5 agents shown in Fig. 1 (a). For simplicity, we also choose and $y_{(i)} \in \mathbb{R}$ as a scalar respectively and set $m_i = 1, M_{(i)j} = 1$ for all $i \in \{1, \ldots, 5\}$, $y_{(i)} = 2$ for $i = 1, \ldots, 3$, $y_{(i)} = 5$ for $i = 4, 5$. Before we run the proposed extended DIGing algorithm, we first obtain the practical optimal solution $x^* = 2.6667$ by running a centralized optimization algorithm (e.g., the gradient descent algorithm). As for control parameters, we also set $h = 0.05, \alpha = 0.06$. But we set $\gamma_1 = \gamma_2 = 1.8$ that is greater than the theoretical upper bound in Theorem 1 because this upper bound is much more conservative than the practical bound which can be inferred from Table II. Figs. 12 and 13 show that the solution estimates of all agents asymptotically converge to the global optimal solution. The event time and the event step size of each agent are shown in Figs. 14 and 15, from which we can see that the event step size is larger than that in Figs. 4 and 5. The performance comparison between DIGing and Extended DIGing is given in Table V, from which we can see that the communication efficiency is improved obviously (14.5% versus 100%).

We then consider the network topology composed of 10 agents shown in Fig. 1(b). We set $m_i = 1, M_{(i)j} = 1$ for all $i \in \{1, \ldots, 10\}$, $y_{(i)} = 2$ for $i = 1, \ldots, 6$, $y_{(i)} = 5$ for $i = 7, \ldots, 10$. We first obtain the practical optimal solution $x^* = 2.6667$ by running a centralized optimization algorithm. As for control parameters, we set $h = 0.05, \alpha = 0.06, \gamma_1 = \gamma_2 = 0.18$, of which $\gamma_1 = \gamma_2 = 0.18$ is also larger than the theoretical upper bound in Theorem 1. The results are shown in Figs. 16–19. The performance comparison between DIGing and Extended DIGing for Huber loss function in case of 10 agents is given in Table VI, from which we can see that the communication efficiency is improved obviously (9.0% versus 100%). It is worth nothing that the larger γ_1 and γ_2 in Table V and Table VI lead to higher communication efficiency compared with that in Table I and Table IV, which can be also verified in Table II.

VI. CONCLUSION

We studied the distributed convex optimization problem by extending the real-time DIGing algorithm to the event-triggered DIGing algorithm in this paper. We developed a novel distributed event-triggering condition to schedule the information exchange between agents. This condition relies on only local states from neighbors at only their event times, which implies that the real-time consecutive communication and the coordinated computation between agents are avoided

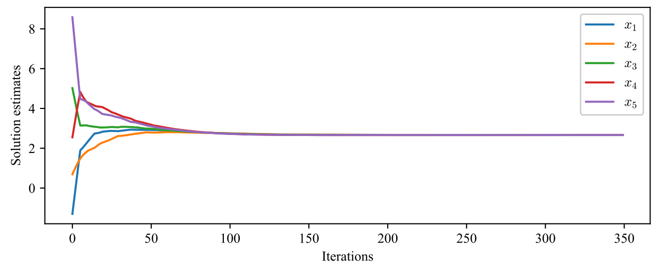


Fig. 12. The evolution of solution estimates for Huber loss function.

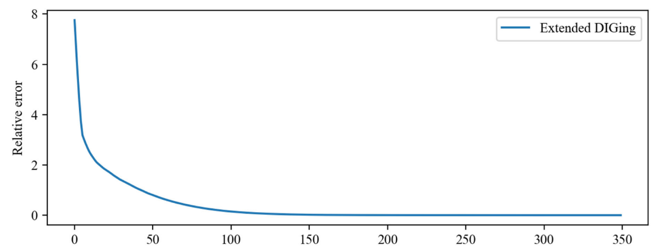


Fig. 13. The evolution of relative error for Huber loss function.

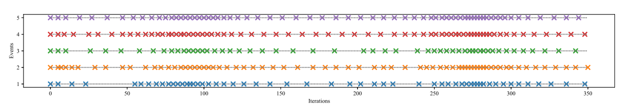


Fig. 14. The event time of each agent $i, i \in 1, \ldots, 5$. Each X mark denotes an event and each little dot denotes a sampling time.

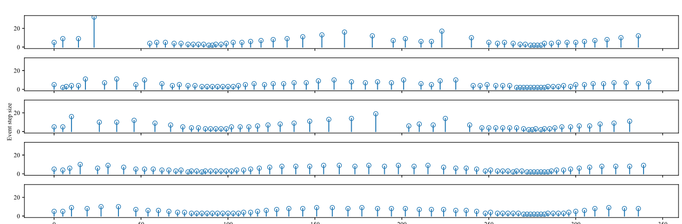


Fig. 15. The event step size of each agent $i, i \in 1, \ldots, 5$. The height of each vertical line denotes the number of sampling time intervals of an event step size.

TABLE V
PERFORMANCE COMPARISON BETWEEN DIGING AND EXTENDED DIGING FOR HUBER LOSS FUNCTION

Performance	DIGing	Extend DIGing
Agreement Convergence	Yes	Yes
Globally Optimal Solution	Yes	Yes
Iteration Times	336	304
Communication Times (Avg)	336	44.2
Communication Rate	100%	**14.5%**

not only in the update of the algorithm iteration but in the implementation of the event-triggering condition. Furthermore, the selections of the key parameters are simple and their upper bounds are independent of agents' initial states and measurement errors. In addition, the convergence analysis was given to show that the extended event-triggered DIGing

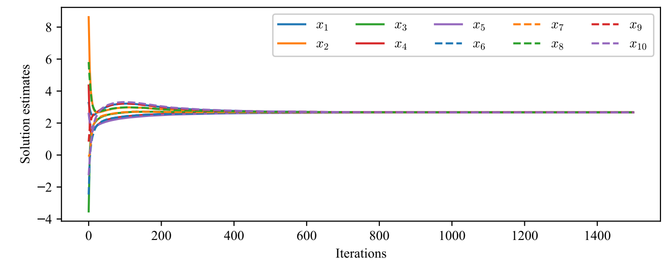


Fig. 16. The evolution of solution estimates for Huber loss function with 10 agents.

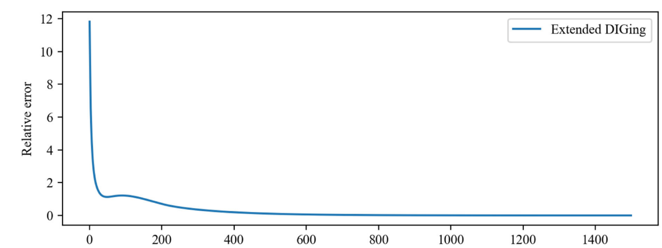


Fig. 17. The evolution of relative error for Huber loss function with 10 agents.

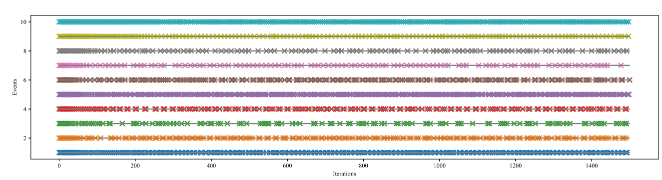


Fig. 18. The event time of each agent $i, i \in 1, \ldots, 5$. Each X mark denotes an event and each little dot denotes a sampling time.

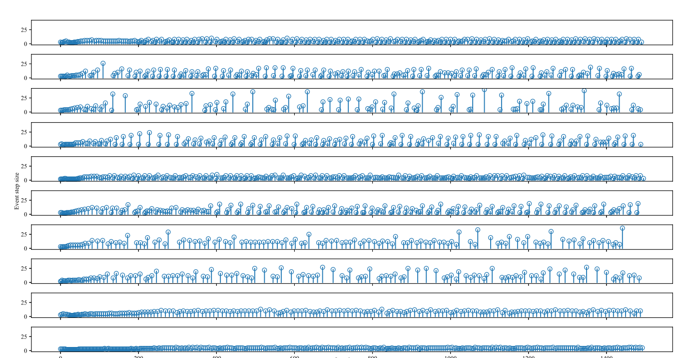


Fig. 19. The event step size of each agent $i, i \in 1, \ldots, 5$. The height of each vertical line denotes the number of sampling time intervals of an event step size.

algorithm can also converge to the exact optimal solution with a linear convergence rate under the proposed event-triggering condition.

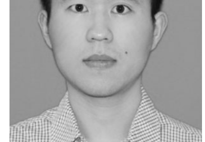
REFERENCES

[1] W. Ren, R. W. Beard, and E. M. Atkins, "Information consensus in multivehicle cooperative control," *IEEE Control Syst.*, vol. 27, no. 2, pp. 71–82, Apr. 2007.

TABLE VI
PERFORMANCE COMPARISON BETWEEN DIGING AND EXTENDED DIGING FOR HUBER LOSS FUNCTION IN CASE OF 10 AGENTS

Performance	DIGing	Extend DIGing
Agreement Convergence	Yes	Yes
Globally Optimal Solution	Yes	Yes
Iteration Times	1751	1676
Communication Times (Avg)	1751	151.5
Communication Rate	100%	**9.0%**

[2] W. Ren and E. Atkins, "Distributed multi-vehicle coordinated control via local information exchange," *Int. J. Robust Nonlinear Control: IFAC-Affiliated J.*, vol. 17, no. 10-11, pp. 1002–1033, 2007.
[3] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," in *Proc. 46th IEEE Conf. Decis. Control*, 2007, pp. 5492–5498.
[4] W. Ren, R. W. Beard, and D. B. Kingston, "Multi-agent Kalman consensus with relative uncertainty," in *Proc. Amer. Control Conf.*, 2005, pp. 1865–1870.
[5] S. Yang, S. Tan, and J.-X. Xu, "Consensus based approach for economic dispatch problem in a smart grid," *IEEE Trans. Power Syst.*, vol. 28, no. 4, pp. 4416–4426, Nov. 2013.
[6] C. Li, X. Yu, T. Huang, and X. He, "Distributed optimal consensus over resource allocation network and its application to dynamical economic dispatch," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2407–2418, Jun. 2018.
[7] C. Yu, M. Zhang, F. Ren, and G. Tan, "Multiagent learning of coordination in loosely coupled multiagent systems," *IEEE Trans. Cybern.*, vol. 45, no. 12, pp. 2853–2867, Dec. 2015.
[8] G.-P. Liu, "Predictive control of networked multiagent systems via cloud computing," *IEEE Trans. Cybern.*, vol. 47, no. 8, pp. 1852–1859, Aug. 2017.
[9] S. Boyd *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
[10] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst., Man, Cybern., Part B.*, vol. 42, no. 2, pp. 513–529, Apr. 2012.
[11] 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.
[12] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *SIAM J. Optim.*, vol. 26, no. 3, pp. 1835–1854, 2016.
[13] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *J. Optim. Theory Appl.*, vol. 147, no. 3, pp. 516–545, 2010.
[14] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, no. Jul, pp. 2121–2159, 2011.
[15] D. Jakovetić, J. Xavier, and J. M. Moura, "Fast distributed gradient methods," *IEEE Trans. Autom. Control*, vol. 59, no. 5, pp. 1131–1146, May 2014.
[16] A. Nedić and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Trans. Autom. Control*, vol. 60, no. 3, pp. 601–615, Mar. 2015.
[17] A. Nedić and A. Olshevsky, "Stochastic gradient-push for strongly convex functions on time-varying directed graphs," *IEEE Trans. Autom. Control*, vol. 61, no. 12, pp. 3936–3947, Dec. 2016.
[18] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4289–4305, Aug. 2012.
[19] A. H. Sayed, "Diffusion adaptation over networks," *Academic Press Library in Signal Processing*, vol. 3, Amsterdam, Netherlands: Elsevier, 2014, pp. 323–453.
[20] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *SIAM J. Optim.*, vol. 25, no. 2, pp. 944–966, 2015.
[21] C. Xi and U. A. Khan, "DEXTRA: A fast algorithm for optimization over directed graphs," *IEEE Trans. Autom. Control*, vol. 62, no. 10, pp. 4980–4993, Oct. 2017.
[22] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact diffusion for distributed optimization and learning-part I: Algorithm development," *IEEE Trans. Signal Process.*, vol. 67, no. 3, pp. 708–723, Oct. 2018.
[23] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.
[24] J. Xu, S. Zhu, Y. C. Soh, and L. Xie, "Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes," in *Proc. IEEE 54th Annu. Conf. Decis. Control*, 2015, pp. 2055–2060.
[25] A. Nedić, 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.
[26] P. Tabuada, "Event-triggered real-time scheduling of stabilizing control tasks," *IEEE Trans. Autom. Control*, vol. 52, no. 9, pp. 1680–1685, Sep. 2007.
[27] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson, "Distributed event-triggered control for multi-agent systems," *IEEE Trans. Autom. Control*, vol. 57, no. 5, pp. 1291–1297, May 2012.
[28] M. Lemmon, "Event-triggered feedback in control, estimation, and optimization," *Networked Control Systems*. New York, USA: Springer, 2010, pp. 293–358.
[29] M. Zhong and C. G. Cassandras, "Asynchronous distributed optimization with event-driven communication," *IEEE Trans. Autom. Control*, vol. 55, no. 12, pp. 2735–2750, Dec. 2010.
[30] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem, "Asynchronous distributed optimization using a randomized alternating direction method of multipliers," in *Proc. IEEE 52nd Annu. Conf. Decis. Control*, 2013, pp. 3671–3676.
[31] P. Bianchi, W. Hachem, and F. Iutzeler, "A coordinate descent primal-dual algorithm and application to distributed asynchronous optimization," *IEEE Trans. Autom. Control*, vol. 61, no. 10, pp. 2947–2957, Oct. 2016.
[32] W. Chen and W. Ren, "Event-triggered zero-gradient-sum distributed consensus optimization over directed networks," *Automatica*, vol. 65, pp. 90–97, 2016.
[33] Y. Kajiyama, N. Hayashi, and S. Takai, "Distributed subgradient method with edge-based event-triggered communication," *IEEE Trans. Autom. Control*, vol. 63, no. 7, pp. 2248–2255, Jul. 2018.
[34] C. Liu, H. Li, Y. Shi, and D. Xu, "Distributed event-triggered gradient method for constrained convex minimization," *IEEE Trans. Autom. Control*, vol. 65, no. 2, pp. 778–785, Feb. 2020.
[35] K. Ishikawa, N. Hayashi, and S. Takai, "Consensus-based distributed particle swarm optimization with event-triggered communication," *IEICE Trans. Fundamental Electron., Commun. Comput. Sci.*, vol. 101, no. 2, pp. 338–344, 2018.
[36] H. Li, S. Liu, Y. C. Soh, and L. Xie, "Event-triggered communication and data rate constraint for distributed optimization of multiagent systems," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 48, no. 11, pp. 1908–1919, Nov. 2018.
[37] X. Han, X. Zhao, T. Sun, Y. Wu, N. Xu, and G. Zong, "Event-triggered optimal control for discrete-time switched nonlinear systems with constrained control input," *IEEE Trans. Syst., Man, Cybern.: Syst.*, to be published, doi: 10.1109/TSMC.2020.2987136.
[38] H. Li, S. Liu, Y. C. Soh, L. Xie, and D. Xia, "Achieving linear convergence for distributed optimization with zeno-like-free event-triggered communication scheme," in *Proc. 29th Chin. Control Decis. Conf.*, 2017, pp. 6224–6229.
[39] A. K. Chandra, P. Raghavan, W. L. Ruzzo, R. Smolensky, and P. Tiwari, "The electrical resistance of a graph captures its commute and cover times," *Comput. Complexity*, vol. 6, no. 4, pp. 312–340, 1996.
[40] D. A. Levin and Y. Peres, *Markov Chains and Mixing Times*, vol. 107, American Mathematical Society, Providence, Rhode Island, 2017.



Lan Gao received the Ph.D. degree from the School of Computer Science and Technology, Chongqing University, Chongqing, China, in 2018. He is currently working as a Postdoctoral Fellow with the Hangzhou Innovation Institute of Beihang University, Hangzhou, China. His current research interests include cooperative control of multi-agent networks, differential privacy and distributed optimization.

Shaojiang Deng received the B.S. degree from the School of Computer Science and Technology, Chongqing Jianzhu University, Chongqing, China, in 1993, and the Ph.D. degree from the School of Computer Science and Technology, Chongqing University, Chongqing, China, in 2005. He is currently a Professor with the School of Computer Science and Technology, Chongqing University, Chongqing, China. His research interests include neural networks, wireless sensor networks and wireless body area networks.

Huaqing Li (Senior Member, IEEE) received the B.S. degree in information and computing science from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2009, and the Ph.D. degree in computer science and technology from Chongqing University, Chongqing, China, in 2013. From September 2014 to September 2015, he was a Postdoctoral Researcher with the School of Electrical and Information Engineering, The University of Sydney, and with the School of Electrical and Electronic Engineering, Nanyang Technological University from November 2015 to November 2016. From July 2018, he was a Professor with the College of Electronic and Information Engineering, Southwest University. His main research interests include nonlinear dynamics and control, multi-agent systems, and distributed optimization. He is a Regional Editor for *Neural Computing & Applications* and the Editorial Board Member for IEEE ACCESS.

Chaojie Li (Member, IEEE) received the B.Eng. degree in electronic science and technology and the M.Eng. degree in computer science from Chongqing University, Chongqing, China, in 2007 and 2011, respectively, and the Ph.D. degree from RMIT University, Melbourne, VIC, Australia, in 2017, where he was a Research Fellow for one and a half years. He was a Senior Algorithm Engineer with Alibaba Group. He is a Senior Research Associate with the School of Electrical Engineering and Telecommunications, UNSW. His current research interests include graph representation learning, distributed optimization and control in smart grid, neural networks, and their application. He was the recipient of ARC Discovery Early Career Researcher Award in 2020.