Walk Proximal Gradient: An Energy-Efficient Algorithm for Consensus Optimization

Xianghui Mao, *Graduate Student Member, IEEE*, Yuantao Gu, *Senior Member, IEEE*, and Wotao Yin

***Abstract*—Decentralized computing is widely used for multiagent systems since it works without a central computing node. In this paper, we develop a first-order algorithm for decentralized consensus optimization that is more energy efficient than the current state-of-the-art. Our algorithm is suitable for application scenarios such as networks of wireless sensors and Internet of Things, where some agents have limited (battery) energy. We call our algorithm walk proximal gradient (WPG), which passes a token through a walk (a succession of nodes) in the graph. The agents that are visited during the walk compute the gradients of their private functions and update the token. We analyze WPG where the walk is the repetition of a Hamiltonian cycle and show that the token converges to the consensual solution faster (in terms of energy consumption) than existing gradient-based decentralized methods. We also generalize the analysis to the non-Hamiltonian graphs. Numerical experiments are presented to validate the energy efficiency of our algorithm.**

***Index Terms*—Consensus optimization, decentralized computation, deterministic routing, energy efficiency, Hamiltonian cycle.**

I. Introduction

CONSIDER a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{v_1, \ldots, v_n\}$ is the set of agents and $\mathcal{E}$ is the set of m edges. We aim to solve the consensus optimization

$$\underset{x\in\mathbb{R}^p}{\text{minimize}} \ \frac{1}{n}\sum_{j=1}^{n} f_j(x) \qquad (1)$$

where each local function f_j is privately known to the corresponding agent v_j and x is a p-dimensional optimization variable.

Manuscript received May 1, 2018; revised August 4, 2018 and September 26, 2018; accepted September 29, 2018. Date of publication October 9, 2018; date of current version May 8, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61571263 and Grant 61531166005 and in part by the National Key Research and Development Program of China under Project 2016YFE0201900 and Project 2017YFC0403600. The work of W. Yin was supported by the National Science Foundation under Grant DMS-1720237 and Grant ONR N0001417121. *(Corresponding author: Yuantao Gu.)*

X. Mao and Y. Gu are with the Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China, and also with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: maoxh14@mails.tsinghua.edu.cn; gyt@tsinghua.edu.cn).

W. Yin is with the Mathematics Department, University of California at Los Angeles, Los Angeles, CA 90095 USA (e-mail: wotaoyin@math.ucla.edu).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org., provided by the author. The material includes the MATLAB code and the datasets for the experiments in this paper. This material is 47 kB in size.

Digital Object Identifier 10.1109/JIOT.2018.2875057

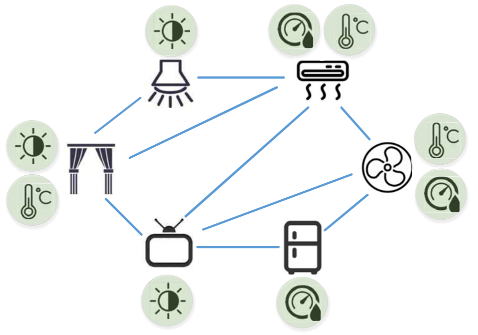


Fig. 1. Smart home system composed of light, air conditioner, curtain, fan, TV, and freezers. Each device is capable of sensing some local environmental parameters, including lightness, humidity, and temperature. The devices are also equipped with communication capabilities (indicated by the connections), which allow the agents to share information with their neighboring device.

A. Motivation and Background

Consensus optimization is broadly applied to model problems over networks, including network routing [1], cooperative control [2], data collection [3], and smart grid [4]. The network utility optimization problem [5] is an application with the utility function of each agent j corresponds to the local function $f_j(\cdot)$ in problem (1). A special case is the average consensus problem, which calculates the average of n distributedly stored numbers. Problems such as network synchronization [6], distributed coordination of mobile autonomous agents [7], and load balancing [8] can be formulated as consensus problems.

We study decentralized algorithms, which use the local computation at each agent and perform communications between one-hop neighbors. There is no central server to collect data or perform computation. The existing decentralized algorithms for (1) will be briefly reviewed in Section II.

In many multiagent systems, not having a central server leads advantages on data privacy, resilience, communication mode and cost, and so on. The architecture of the smart home system as shown in Fig. 1 is an example of decentralized multiagent system, where we can utilize a decentralized algorithm for problem (1) to maximize the comfort level. In this application, x is a common control variable, and $-f_j(x)$, $j = 1, \ldots, n$, are local comfort levels sensed by agent j. By solving problem (1), we find the optimal variable that maximizes the overall comfort level of the smart home.

How to communicate is an important aspect of decentralized computation. Deterministic routing tables are prevalently utilized to define the communication paths among the network of agents. When the network is strongly connected, the simple way of relaying packets along paths to destinations can deliver information between any pair of agents in the network. The network routing strategies can be based on Hamiltonian cycle [9], [10], the shortest path [11], [12], and so on.

Agents in wireless networks, including but not limited to wireless sensor networks and Internet of Things (IoT), are typically driven by energy limited power sources, e.g., batteries. In order for them to function properly in their lifetime, energy efficiency has been a major issue to address [13], [14]. In the literature, main approaches to reduce energy consumption include: 1) reducing the amount or frequency of information collection and processing and putting idle agents into their sleeping mode in effective scheduling schemes [15] and 2) improving routing and MAC protocols in communication [16]. The above solutions are proposed in the presence of a central computing server. The study of energy efficiency for decentralized optimization is limited but exists. Ling and Tian [3] investigated energy-efficient computing strategy for sparse signal recovery, in which the vertices are allowed to enter their sleeping mode in a random manner. Apart from [3], it seems little has been done.

B. Contributions and Organizations

This papers proposes the following decentralized algorithm to solve problem (1). Introduce the local variable z_j for each agent j, $j = 1, \dots, n$. Create a token with initial value $x^0 = (1/n)\sum_{j=1}^n z_j^0$ (we can simply set $x^0 = z_1^0 = \cdots = z_n^0 = 0$). The token travels through a walk i_t, $t = 0, 1, 2, \dots$, which is a succession of agents with every (i_t, i_{t+1}) being an edge. When the token arrives at agent i_t, it become active and computes

$$z_j^{t+1} = \begin{cases} x^t - \alpha\nabla f_{i_t}(x^t), & \text{active agent } j = i_t \\ z_j^t, & \text{inactive agents } j \neq i_t \end{cases}$$
$$x^{t+1} = x^t + \frac{1}{n}\left(z_{i_t}^{t+1} - z_{i_t}^t\right)$$

and then passes x^{k+1} to the next agent. We call this algorithm walk proximal gradient (WPG). Later in this paper, we derive it by modifying the proximal gradient algorithm, where the modification is inspired by our previous work W-ADMM [17]. However, [17] uses a random walk and solves a subproblem at each agent instead of computing the gradient. The deterministic walk is suitable to utility optimization and smart control of IoT, which uses deterministic routing.

For the ease of derivation, we first introduce the WPG method on a Hamiltonian network, and then generalize it to suit the non-Hamiltonian networks by occasionally activating an agent if it is repeatedly visited in a routing path.

We prove the linear convergence of WPG under strong-convexity assumptions. The energy efficiency of WPG is verified both theoretically and numerically. Consumption of energy includes computation and communication. As a first-order algorithm, the computation of WPG is dominated by gradient evaluation. The communication in WPG is spent on passing the token x among the agents. Assume each gradient evaluation and each token passing take $O(1)$. Then, if a Hamiltonian cycle of the network is known *a priori*, to get a solution x within ϵ to the optimal one, WPG needs $O(\ln(1/\epsilon)\cdot(n(\kappa^2+2\kappa)))$ iterations or units of energy, where κ is the condition number of the local functions. If the Hamiltonian cycle is not provided, $O(\ln(1/\epsilon)\cdot(n(\kappa^2+2\kappa))\Delta)$ units of energy is required, where Δ denotes the diameter of the network.

When we apply WPG to solve the average consensus problem, it converges after the walk covers all the agents, that is a finite number of iterations. We also discuss how to adapt WPG to dynamic network topologies.

C. Organization

The rest of this paper is organized as follows. In Section II, we briefly review relevant decentralized optimization algorithms. In Section III, we give the preliminaries on Hamiltonian cycle and proximal gradient method. Then, WPG is derived Section IV and analyzed in Section V. The energy consumptions of WPG and alternative algorithms are compared in Section VI. Extensions of WPG are discussed in Section VII, and numerical experiments are presented in Section VIII.[1] Finally, Section IX concludes this paper.

II. Related Work

Most of the existing decentralized algorithms belong to either the incremental type or the gossip type.

Incremental decentralized algorithms activate one agent and one edge in each step, keeping all the other agents and edges idle. The ordering of activation can be deterministic or random.

Among the *deterministic* incremental algorithms, [18] assumes the network $\mathcal{G}$ is a Hamiltonian network, which leads to the periodic activation of agents. The first incremental gradient method (IG) was proposed in [19], where gradients of local functions are computed in a cyclic order. To converge to the exact solution, IG must use diminishing step sizes, which leads to a sublinear convergence rate. The incremental aggregated gradient method (IAG) proposed in [20] maintains an aggregated gradient while updating the optimization variable, and both quantities are communicated when the method is implemented in a network of agents. In spite of the doubled amount of communication per step, IAG can converge linearly when the objective function is strongly convex [21]. In the setting, WPG converges linearly yet uses half of the IAG communication per step.

Random (walk) incremental algorithms activate the agents successively following a random walk over the network. To guarantee convergence to the exact solution of (1), the algorithms in [22]–[24] use diminishing step sizes, which lead to sublinear convergence rates. W-ADMM [17] manages to achieve the same with a constant step size, which improves the convergence speed. A linear rate of convergence was established for decentralized least squares. As mentioned above, W-ADMM solves a subproblem at each step instead of using a gradient, which is typically cheaper to compute.

Unlike incremental decentralized algorithms, *gossip-based* algorithms such as ADMM (D-ADMM) [25], [26],

[1]Simulation codes are provided to reproduce the results presented in this paper: http://gu.ee.tsinghua.edu.cn/publications. Supplemental material for the reader can be downloaded online at http://ieeexplore.ieee.org/

EXTRA [27], and exact diffusion [28] activate multiple agents in parallel and let them communicate with some or all of their neighbors in each iteration. They use more communication at each iteration and tend to require fewer total number of iterations. These algorithms are roughly derived as follows. First, problem (1) is formulated as a constrained optimization problem by introducing auxiliary variables, then a primal-dual method is applied to solve it. In [26], D-ADMM was proved to converge linearly when all the local functions are strongly convex. EXTRA [27] and then exact diffusion [28] were shown to converge linearly when the sum of local convex functions is strongly convex. The main computation of these first-order algorithms is evaluating the gradients of local functions or their proximal mappings. Scaman *et al.* [29] proposed SSDA and MSDA, which calculate the gradients of each local functions' convex conjugates. Mokhtari *et al.* [30] proposed a second-order gossip-based method, network Newton, to improve convergence speed when the Hessian of local functions are available.

III. Preliminaries

We recall the definition of Hamiltonian cycle.

Definition 1 (Hamiltonian Cycle [18]): A Hamiltonian cycle is a cycle on the graph that visits each vertex exactly once.

A graph that contains a Hamiltonian cycle is called a Hamiltonian graph. Many graphs, including complete graphs, ring graphs, and 4-connected planar graph have been proven to be Hamiltonian graphs. Ore's theorem [31] shows that any simple graph with $n \geq 3$ vertices is Hamiltonian if, for every pair of nonadjacent vertices, the sum of their degrees is no less than n. There are also many existing works studying how to find a Hamiltonian cycle in a graph [32], [33].

Proximal gradient method tackles the optimization problem

$$\underset{y}{\text{minimize}}\ \ F(y) + r(y) \tag{2}$$

where $F(\cdot)$ is a differentiable function and $r(\cdot)$ has a simple proximal mapping. In the tth iteration of the proximal gradient method, with a positive step size α, the variable is updated according to

$$y^{t+1} = \mathbf{prox}_{\alpha r}\big(y^t - \alpha\nabla F\big(y^t\big)\big) \tag{3}$$

where the proximal mapping $\mathbf{prox}_{\alpha r}$ is defined as

$$\mathbf{prox}_{\alpha r}(u) := \arg\min_x r(x) + \frac{1}{2\alpha}\|x-u\|^2. \tag{4}$$

When both F and r are convex, under mild conditions on the smoothness of function F, the sequence $(y^{t+1})_{t\geq 0}$ converges to the solution of (2).

IV. Walk Proximal Gradient

In this section, we derive the WPG algorithm.

By introducing auxiliary variables, the consensus optimization problem (1) can be rewritten as problem (2) with variable $y := \text{col}(x_1, \ldots, x_n)$ with col stacking the arguments into a long vector and each x_i, $i \in \{1, \ldots, N\}$, being a p-dimensional vector, objective functions $F(y) := \sum_{j=1}^n f_j(x_j)$ and an indicator function $r(y) := \delta_{\{x_1=\cdots=x_n\}}(y)$. Here, $r(y) = 0$ if $x_1 = \cdots = x_n$ and $= \infty$ otherwise. With

$$\mathbf{prox}_{\alpha r}(u) = \mathbf{1}_n \otimes \left(\frac{1}{n}\sum_{i=1}^n u_i\right)$$

where $\mathbf{1}_n$ denotes the all 1 vector of size n and $\otimes$ denotes the Kronecker Product, we apply the proximal gradient method (3) to get the iteration

$$y^{t+1} = \mathbf{1}_n \otimes \left[\frac{1}{n}\sum_{j=1}^n\Big(x_j^t - \alpha\nabla f_j\Big(x_j^t\Big)\Big)\right]. \tag{5}$$

We have in y^t, $x_1^t = \cdots = x_n^t$, for all $t \geq 0$. Hence, we introduce a p-dimensional vector $x^t := x_1^t = \cdots = x_n^t$ to simplify our algorithm. Remember, we have introduced the local variables $z_1^t, \ldots, z_n^t \in \mathbb{R}^p$, one for each agent. With them, we can equivalently rewrite (5) as

$$z_j^{t+1} = x^t - \alpha\nabla f_j\big(x^t\big), \quad \forall j \in \{1, \ldots, n\} \tag{6}$$

$$x^{t+1} = \frac{1}{n}\sum_{j=1}^n z_j^{t+1}. \tag{7}$$

Since (6) evaluates the gradients of all the n local functions and (7) collects information from all the agents, they are *not decentralized yet.*

To get a decentralized algorithm, we change (6) so that only z_{i_t} of the active agent i_t gets updated, keeping all the other $z_j (j \neq i_t)$ unchanged. This leads to the iteration

$$z_j^{t+1} = \begin{cases} x^t - \alpha\nabla f_j\big(x^t\big), & j = i_t \\ z_j^t, & j \neq i_t \end{cases} \tag{8}$$

$$x^{t+1} = \frac{1}{n}\sum_{j=1}^n z_j^{t+1}. \tag{9}$$

Note that the updates in (8) and (9) are not equivalent to the proximal gradient updating procedure encoded in (6) and (7). The difference is how many agents are activated to conduct the z-update at each iteration. The benefit of activating only one agent to conduct z-update in (8) and (9) is that the updating procedure can be decentralized. Next, we show that in each iteration the x-update in (9) can be accomplished by agent i_k itself under the initialization

$$x^0 = \frac{1}{n}\sum_{j=1}^n z_j^0. \tag{10}$$

Since, in the set $\{z_j\}_{j\in\{1,\ldots,n\}}$, only z_{i_t} is updated in the tth iteration, (9) can be rewritten equivalently as

$$x^{t+1} = x^t + \frac{1}{n}\Big(z_{i_t}^{t+1} - z_{i_t}^t\Big) \tag{11}$$

which can be computed solely by agent i_t since it has both the token x^t and its local variables $z_{i_t}^{t+1}, z_{i_t}^t$. By (10) and induction, steps (9) and (11) are equivalent.

The implementation of WPG is presented in Algorithm 1. For simplicity and analysis clarity, we focus only on the case where the agents are activated in a predefined circulant pattern. The agents are assigned with the numbers following their

Algorithm 1 WPG

```
**Initialization:** initialize $x^0$ and $z_j^0$ so that $x^0 = \frac{1}{n}\sum_{j=1}^{n} z_j^0$;
**Repeat** for $t = 0, 1, 2, \dots$
    agent $i_t = \text{mod}_n(t) + 1$ do:
        receive token $x^t$;
        update local variable $z_{i_t}^{t+1} = x^t - \alpha \nabla f_j(x^t)$;
        update token $x^{t+1} = x^t + \frac{1}{n}\left(z_{i_t}^{t+1} - z_{i_t}^t\right)$;
        send token $x^{t+1}$ to agent $i_{t+1}$ via edge $(i_t, i_{t+1})$;
    agents $j \neq i_t$ do nothing; hence, $z_j^{t+1} = z_j^t$;
**End**
```

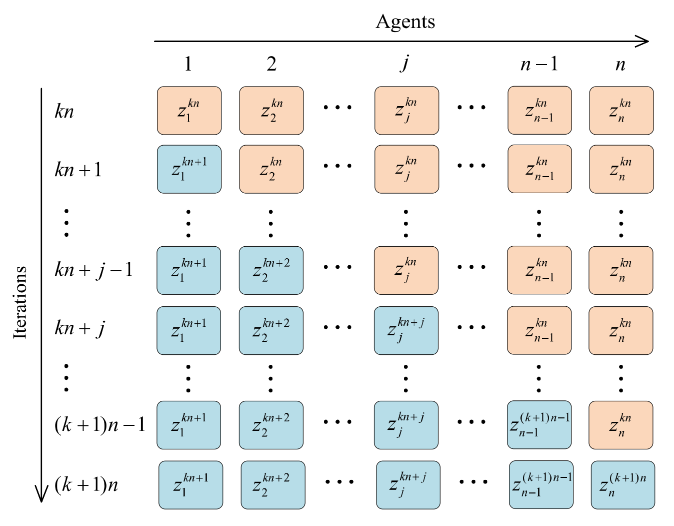


Fig. 2. Visualization of the evolution of $\{z_j^t\}_{j=1:n}$ in the period of $[kn, (k+1)n]$.

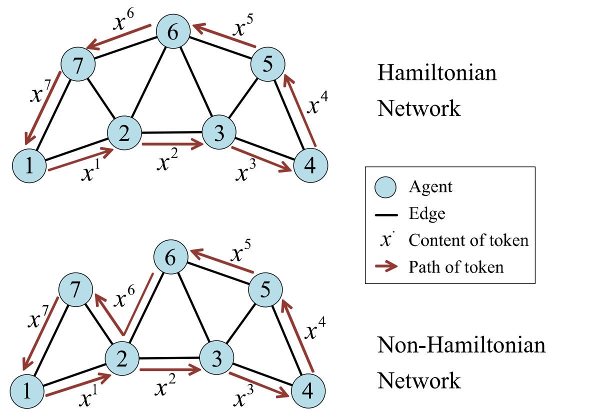


Fig. 3. Traversing pattern of the token over two different network topologies of $n = 7$ agents.

activation order. The activated agent in the tth iteration is $i_t = \text{mod}_n(t) + 1$.

The evolution of the local variables is visualized in Fig. 2. Notice that for each agent j, its local variable gets updated after the iterations $(kn + j - 1)_{k\geq 0}$, which index the visits of the token to agent j.

A. Communication Strategy

Suppose the walk $(i_t)_{t\geq 0}$ repeats a Hamiltonian cycle $(1, \dots, n)$ in WPG, Algorithm 1. Then, the agents in the following cyclic order: $1 \to 2 \to \cdots \to n \to 1 \to 2 \to \cdots$. An illustration can be found in the upper plot of Fig. 3. For Hamiltonian networks, such communications can be accomplished by a Hamiltonian-cycle-based routing strategy such as [9] and [10].

WPG can also be applied to non-Hamiltonian networks. For any strongly connected network, there exists a cycle passing every agent at least once (but possibly more than once). Our method still works if the walk repeats this cycle. However, we only analyze WPG for a Hamiltonian cycle in the next section. To apply our analysis there to a general cycle that covers every agent, if a token appears more than once in this cycle, it is only activated at its first appearance; subsequent visits are passed over without activation. As shown in Fig. 3, with the connection between agents 6 and 7 deleted, a Hamiltonian network becomes a non-Hamiltonian network. To pass the token from agent 6 to 7, a relay node, agent 2, is required. Hence, in the cycle, agent 2 is passed twice, but activated only at the first visit.

For strongly connected networks, there are many routing strategies to generate such a cycle. Specifically, we can implement the shortest path routing strategy [11], [12] to the network, and the shortest paths between any pair of agents are maintained by the routing tables. A possible cycle could be the concatenation of the shortest paths from agents 1 to 2, 2 to 3,$\dots$, $n-1$ to n, and n to 1. We name it the *shortest path cycle*.

B. Comparison With Similar Works

To better evaluate the novelty of WPG, we explain its differences from the existing decentralized algorithms.

1) *W-ADMM [17]:* Both WPG and W-ADMM are based on passing a token over a walk on network. While W-ADMM is a primal-dual method, WPG is a primal method. Their computations in each step are different. W-ADMM is proven to converge linearly when solving least squares problem.
2) *IAG [20]:* One can run IAG over the same walk as WPG. The updating procedure of IAG is

$$x_{\text{iag}}^{t+1} = x_{\text{iag}}^t - \alpha g_{\text{iag}}^t \tag{12}$$

$$z_{j,\text{iag}}^{t+1} = \begin{cases} \nabla f_j\left(x_{\text{iag}}^t\right), & j = i_t \\ z_{j,\text{iag}}^t, & j \neq i_t \end{cases} \tag{13}$$

$$g_{\text{iag}}^{t+1} = g_{\text{iag}}^t + \frac{1}{n}\left(z_{i_t,\text{iag}}^{t+1} - z_{i_t,\text{iag}}^t\right) \tag{14}$$

where each $z_{j,\text{iag}}^t$ is kept locally by agent j, and the variable x_{iag}^t and the aggregated gradient g_{iag}^t must be passed around as a token. That doubles the per-step communication of WPG. Both IAG and WPG are primal methods. IAG is related to variance-reduction stochastic gradient (but IAG follows a deterministic ordering), and WPG is more related to block coordinate descent (BCD) (but with differences outline in part 3 below). Under Lipschitz differentiability and strong convexity assumptions, linear convergence of IAG is established in [21].

3) *S-PPG [34]:* The similarity between S-PPG and WPG is that they are both backward-forward splitting methods using block updates. However, S-PPG chooses the update blocks in an independent identically distributed random fashion, and its analysis cannot be adapted to the cyclic block selection used in WPG. Under certain smoothness and strong convexity assumptions, S-PPG also achieves linear convergence. Both the WPG algorithm and its analysis in this paper will connect it to a circular BCD method. However, WPG is not a direct application of BCD since, in the BCD literature, the nonsmooth function $r(y)$ in (2), if present, is always separable, i.e., taking the form $r(y) = r_1(x_1) + \cdots + r_n(x_n)$. However, our $r(y) = \delta_{\{x_1=\cdots=x_n\}}(y)$ is nonsmooth, nonseparable. Under Lipschitz differentiability and strong convexity, BCD is guaranteed to converge linearly.

C. Energy Consumption Per Iteration and Storage Requirement

There are two types of energy consumption in the network: 1) computation and 2) communication. For first-order algorithms, computation is dominated by gradient evaluation. For the ease of analysis, we assume that the each gradient evaluation takes a constant amount of energy, E_1. In real-world scenario, the communication energy is tightly related to the distance between the sender and receiver. But for simplicity, we consider the case that the distances between connected pairs of agents are approximately the same, and can be upper bounded by a communication range threshold, which is independent of both the numbers of the agents n and the numbers of edges m. Specifically, we assume that each communication between a pair of agents (sending a p-dimensional variable from one to the other) consumes a constant amount of energy, E_2.

Remark 1: For WPG, each agent j only needs to store the local variable z_j^t, which requires local storage space $O(p)$. In each iteration of WPG:

1) computation energy is E_1, spent on one gradient evaluation of a local function;
2) communication energy depends on the traversing pattern of the token:
   a) E_2 when traversing via Hamiltonian cycle;
   b) $\leq \Delta E_2$ when traversing via the shortest path cycle, where Δ denotes the diameter of the network, and defined as $\Delta := \max_{i,j\in\{1,\ldots,n\}} d(i,j)$ hops, where $d(i,j)$ denotes the length of the shortest path between agents i and j.

By Remark 1, the total storage space required for the network is $O(np)$, which can hardly be further decreased. The existing incremental and gossip-based methods requires storage space on the same order.

The computation energy per iteration of WPG is independent from the size of the network, n. This property is shared by the incremental algorithms, including IAG, S-PPG, W-ADMM, etc. When implementing deterministic incremental algorithms, in either traversing pattern, the per iteration communication consumption of WPG is the least one can expect. Recall the analysis in part 2 of Section IV-B, the per iteration communication consumption of IAG is doubled.

Noticing that in each iteration of gossip-based algorithms, all the agents are activated to compute, which involves at least n times computation consumption of WPG. As for the communication consumption, in gossip algorithms, any pair of connected agents share their computation results with each other, which requires $2|\mathcal{E}|$ times communications compared to WPG.

With Remark 1 analyzing the computation and communication consumption per iteration of WPG, the total energy consumption of WPG is postponed to Section VI as it is tightly related to the convergence rate provided in Section V.

V. Convergence Analysis

In this section, we establish the linear convergence of WPG for a Hamiltonian network under smoothness and strong convexity assumptions of the local functions $\{f_j\}_{j\in\{1,\ldots,n\}}$.

Assumption 1: The activation pattern $(i_t)_{t\geq 0}$ cyclically repeats a permutation of the vertex set $\mathcal{V}$.

To be noted, the activation pattern listed in Algorithm 1 satisfies Assumption 1, where $(i_t)_{t\geq 0}$ repeats $(1, 2, \ldots, n)$. On a connected network, we will show that it is easy to generate a sequence of $(i_t)_{t\geq 0}$ satisfying Assumption 1 by briefly recalling the communication strategy in Section IV-A. When the network is Hamiltonian connected, an activation pattern satisfying Assumption 1 is achieved by letting the token traveling among the agents following a certain Hamiltonian walk and the token always activating the visited agent on its arrival. When the network is not Hamiltonian connected, the token should walk over all the nodes in the network following a (shortest path cycle), and on every repeat of the cycle, each agent is only activated when it is visited for the first time in each repeat. In other words, some agents may be visited the second, third, ... times without incurring any computation.

Below we establish convergence based on a new Lyapunov function. To begin with, we first give the assumptions that would be applied in the following convergence analysis.

Assumption 2: Each local function $f_j(\cdot)$ is convex and L-Lipschitz differentiable, that is, for any $u, v \in \mathbb{R}^p$, it holds

$$\|\nabla f_j(u) - \nabla f_j(v)\| \leq L\|u - v\|, \ \forall j \in \{1, \ldots, n\}. \quad (15)$$

Assumption 3: Each local function $f_j(\cdot)$ is μ-strongly convex, that is, for any $u, v \in \mathbb{R}^p$, we have

$$\left\langle \nabla f_j(u) - \nabla f_j(v), u - v\right\rangle \geq \mu\|u - v\|^2, \ \forall j \in \{1, \ldots, n\}. \quad (16)$$

For any optimal solution x^* of problem (1), we define

$$z_j^* := x^* - \alpha\nabla f_j\left(x^*\right) \in \mathbb{R}^p \quad (17)$$
$$z^* := \mathrm{col}\left(z_1^*, \ldots, z_n^*\right) \in \mathbb{R}^{np}. \quad (18)$$

We consider the Lyapunov function

$$\xi^t := \sum_{j=1}^{n} \frac{\tau_j^t}{n} \|z_j^t - z_j^*\|^2 \quad (19)$$

where

$$\tau_j^t := \underset{n}{\mathrm{mod}}(j - i_t) + 1 \tag{20}$$

is one plus the number of iterations beyond i_t when node j will be visited. The evolution of the local variables in Fig. 2 can provide an intuitive understanding to the parameter τ_j^t. Suppose $t = kn$, $i_t = 1$; then, for $j \in \{1, \ldots, n\}$, we have $\tau_j^t = j$ being the amount of light red blocks in the jth column. Note that $\{\tau_1^t, \ldots, \tau_n^t\}$ is a permutation of $\{1, \ldots, n\}$. For the ease of deduction, for each $t \geq 0$, we stack the local variables to form a np-dimensional vector $\mathcal{Z}^t := \mathrm{col}(z_1^t, \ldots, z_n^t)$. The Lyapunov function ξ^t can be rewritten as

$$\xi^t = \left(\mathcal{Z}^t - \mathcal{Z}^*\right)^\mathsf{T}\mathbf{H}^t\left(\mathcal{Z}^t - \mathcal{Z}^*\right) \tag{21}$$

where $\mathbf{H}^t := \mathrm{diag}((\tau_1^t/n), \ldots, (\tau_n^t/n))$ is a positive definite diagonal matrix. ξ^t is a variable measure of the distance between $\mathcal{Z}^t$ and $\mathcal{Z}^*$.

Below, we establish the sufficient descent of $(\xi^t)_{t\geq 0}$.

Lemma 1 (Descent Lemma): Under Assumptions 1 and 2, for any optimal solution x^* of problem (1) and $\mathcal{Z}^*$ and ξ^t as defined in (18) and (19), respectively, with $\alpha \leq (2/L)$, for any $k \geq 0$, $i \in \{0, \ldots, n-1\}$, we have

$$\xi^{t+1} \leq \xi^t - \left(\frac{2\alpha}{L} - \alpha^2\right)\left\|\nabla f_{i_t}(x^t) - \nabla f_{i_t}(x^*)\right\|^2. \tag{22}$$

Proof: By Assumption 1 and the definition of τ_j^t, we learn that for any $j \neq i_t$, we have

$$\tau_j^{t+1} - \tau_j^t = -1 \tag{23}$$

and

$$\tau_{i_t}^{t+1} = n, \quad \tau_{i_t}^t = 1. \tag{24}$$

Notice that $z_j^t = z_j^{t+1}$ holds for any $j \neq i_t$. By the definition of ξ_j^t in (19) and local update (8), we get

$$\begin{aligned}\xi^{t+1} - \xi^t &= \|z_{i_t}^{t+1} - z_{i_t}^*\|^2 - \frac{1}{n}\sum_{j=1}^n \|z_j^t - z_j^*\|^2 \\ &\leq \|z_{i_t}^{t+1} - z_{i_t}^*\|^2 - \|\frac{1}{n}\sum_{j=1}^n z_j^t - z_j^*\|^2 \\ &= \|z_{i_t}^{t+1} - z_{i_t}^*\|^2 - \|x^t - x^*\|^2 \end{aligned} \tag{25}$$

where the last equality follows from (9). According to (8) and the definition of z^* as in (17), we have

$$\begin{aligned}\left\|z_{i_t}^{t+1} - z_{i_t}^*\right\|^2 &= \left\|x^t - x^* - \alpha\nabla f_{i_t}(x^t) + \alpha\nabla f_{i_t}(x^*)\right\|^2 \\ &= \left\|x^t - x^*\right\|^2 + \alpha^2\left\|\nabla f_{i_t}(x^t) - \nabla f_{i_t}(x^*)\right\|^2 \\ &\quad - 2\alpha\langle x^t - x^*, \nabla f_{i_t}(x^t) - \nabla f_{i_t}(x^*)\rangle \\ &\leq \left\|x^t - x^*\right\|^2 - \left(\frac{2\alpha}{L} - \alpha^2\right)\left\|\nabla f_{i_t}(x^t) - \nabla f_{i_t}(x^*)\right\|^2 \end{aligned} \tag{26}$$

where the last inequality comes from Baillon–Haddad theorem [35]. The proof is completed by substituting (26) into (25). ■

Lemma 1 shows that the sequence $(\xi^t)_{t\geq 0}$ is a nonincreasing sequence. Next, we apply the strongly convexity Assumption 3 to get the convergence rate of WPG.

Lemma 2: Under Assumptions 1–3, with $\alpha \leq (2/L)$, we have that the sequence $(\mathcal{Z}^t)_{t\geq 0}$ generated by WPG converges linearly to $\mathcal{Z}^*$. Specifically, for any $t \geq 0$, we have

$$\xi^{t+n} \leq \frac{1}{1+\theta(\alpha)}\xi^t \tag{27}$$

where $\theta(\alpha) := ([(2\alpha)/L] - \alpha^2)/(\alpha + (1/\mu))^2$.

Proof: By Lemma 1, we have

$$\xi^{t+n} \leq \xi^t - \left(\frac{2\alpha}{L} - \alpha^2\right)\sum_{s=t}^{t+n-1}\|\nabla f_{i_s}(x^s) - \nabla f_{i_s}(x^*)\|^2. \tag{28}$$

By Assumption 3, for any $\lambda > 0$, we have

$$\begin{aligned}\left\|z_{i_t}^{t+1} - z_{i_t}^*\right\|^2 &= \left\|x^t - \alpha\nabla f_{i_t}(x^t) - \left(x^* - \alpha\nabla f_{i_t}(x^*)\right)\right\|^2 \\ &= \left\|x^t - x^*\right\|^2 + \alpha^2\left\|\nabla f_{i_t}(x^t) - \nabla f_{i_t}(x^*)\right\|^2 \\ &\quad - 2\alpha\langle x^t - x^*, \nabla f_{i_t}(x^t) - \nabla f_{i_t}(x^*)\rangle \\ &\overset{(a)}{\leq} (1+\alpha\lambda)\left\|x^t - x^*\right\|^2 \\ &\quad + \left(\alpha^2 + \frac{\alpha}{\lambda}\right)\left\|\nabla f_{i_t}(x^t) - \nabla f_{i_t}(x^*)\right\|^2 \\ &\overset{(b)}{\leq} \left(\frac{1+\alpha\lambda}{\mu^2} + \alpha^2 + \frac{\alpha}{\lambda}\right)\left\|\nabla f_{i_t}(x^t) - \nabla f_{i_t}(x^*)\right\|^2 \end{aligned} \tag{29}$$

where (a) follows from Young's inequality ($\pm 2ab \leq \lambda a^2 + \lambda^{-1}b^2$) and (b) follows from Assumption 3. By taking $\lambda = \mu$, the parenthesis in (29) is minimized, which leads to

$$\left\|z_{i_t}^{t+1} - z_{i_t}^*\right\|^2 \leq \left(\alpha + \frac{1}{\mu}\right)^2\left\|\nabla f_{i_t}(x^t) - \nabla f_{i_t}(x^*)\right\|^2. \tag{30}$$

Substituting (30) into (28) with $\theta(\alpha) := ([(2\alpha)/L] - \alpha^2)/(\alpha + (1/\mu))^2$, we have

$$\xi^{t+n} \leq \xi^t - \theta(\alpha)\sum_{s=t}^{t+n-1}\left\|z_{i_s}^{s+1} - z_{i_s}^*\right\|^2. \tag{31}$$

Remember $\{i_t, i_{t+1}, \ldots, i_{t+n-1}\} = \{1, 2, \ldots, n\}$. Hence, from (31), we get

$$\sum_{j=1}^n\left(\frac{\tau_j^t}{n} + \theta(\alpha)\right)\left\|z_j^{t+n} - z_j^*\right\|^2 \leq \sum_{j=1}^n \frac{\tau_j^t}{n}\left\|z_j^t - z_j^*\right\|^2$$

and, by $(\tau_j^t/n) \leq 1$

$$\sum_{j=1}^n \frac{\tau_j^t}{n}(1+\theta(\alpha))\|z_j^{t+n} - z_j^*\|^2 \leq \sum_{j=1}^n \frac{\tau_j^t}{n}\|z_j^t - z_j^*\|^2.$$

With $\tau_j^{t+n} = \tau_j^t$ and the definition of ξ^t, we complete the proof. ■

Lemma 2 indicates that the Lyapunov function ξ^t converges linearly to 0. Recall ξ^t is a distance between the stacked local variables $\mathcal{Z}^t$ and $\mathcal{Z}^*$. Therefore, Lemma 2 implies that the local variables converges to $\mathcal{Z}^*$ in a linear rate. Finally, we give the rate at which x^t converges to x^*.

Theorem 1: Under Assumptions 1–3, with $\alpha \leq (2/L)$, the sequence $\{x^t\}_{t\geq 0}$ generated by WPG converges to x^* at a linear rate. Specifically,

$$\|x^t - x^*\| \leq \left(\frac{1}{1+\theta(\alpha)}\right)^{\lfloor \frac{t}{n} \rfloor} \|z^0 - z^*\|^2. \tag{32}$$

Proof: By Lemma 2, we have

$$\begin{aligned} \xi^{kn} &\leq \left(\frac{1}{1+\theta(\alpha)}\right)^k \xi^0 \\ &\leq \left(\frac{1}{1+\theta(\alpha)}\right)^k \|z^0 - z^*\|^2. \end{aligned} \tag{33}$$

We also have

$$\left\|x^t - x^*\right\|^2 \leq \left\|\frac{1}{n}\sum_{j=1}^{n} z_j^t - z_j^*\right\|^2 \leq \frac{1}{n}\sum_{j=1}^{n} \|z_j^t - z_j^*\|^2 \leq \xi^t. \tag{34}$$

Combining (33) and (34) and using the nonincreasing property of the sequence $\{\xi^t\}_{t\geq 0}$ complete the proof. ■

Next, we optimize the factor $[1/(1+\theta(\alpha))]$ by choosing α.

Corollary 1: Maximizing $\theta(\alpha)$ over α yields the optimal step size

$$\alpha^* := \frac{1}{L+\mu} \tag{35}$$

and the corresponding factor

$$\theta^* := \theta\left(\alpha^*\right) = \frac{1}{(2+\kappa)\kappa} \tag{36}$$

where $\kappa := L/\mu$ denotes the condition number of the local functions.

Proof: Differentiate $\theta(\alpha)$

$$\frac{d\theta(\alpha)}{d\alpha} = \frac{-2(\mu+L)\alpha+2}{L\mu\left(\alpha+\frac{1}{\mu}\right)^3}. \tag{37}$$

Notice that with $\mu, L > 0$, the parameter $\theta(\alpha)$ first increases then decreases when α increases from 0 to $+\infty$. So, the maximal value is achieved with $\alpha = \alpha^*$. It is easy to check that $\alpha^* \leq 2/L$, which admits the requirement in Theorem 1. ■

VI. Energy Consumption Analysis

In this section, we study how much energy the state-of-the-art consensus optimization algorithms spend on solving problem (1). We compared those of WPG, gossip-based algorithms, and incremental algorithms. These are gradient-based algorithms. We skip second-order algorithms (e.g., network Newton [30]) and those involving local proximal mappings, such as D-ADMM [25], [26] and W-ADMM [17]. The evaluations of second-order information or proximal mappings can be more energy-consuming than evaluating gradients.

Be aware that this comparison is based on existing bounds, some of which may not be tight.

Recall that E_1 and E_2 are the amounts of energy spent on each gradient evaluation and edge activation, respectively.

A. Energy Consumption of WPG

By Theorem 1 and Corollary 1, with α set as α^* given in (35), to achieve $\|x^t - x^*\| \leq \epsilon$, it suffices to have

$$\|z^0 - z^*\|^2 \left(\frac{1}{1+\theta^*}\right)^{\lfloor \frac{t}{n} \rfloor} \leq \epsilon. \tag{38}$$

Specifically, with $t = kn$, if k satisfies

$$k \geq \ln\left(\frac{\|z^0 - z^*\|^2}{\epsilon}\right) / \ln\left(1+\theta^*\right) \tag{39}$$

it is guaranteed to have $\|x^{kn} - x^*\|^2 \leq \epsilon$. Note that in each iteration of WPG, only one gradient evaluation is conducted, which corresponds to an E_1 amount of computation energy consumption. The per-iteration communication consumption of WPG depends on the network structure. Recall the communication strategy stated in Section IV-A under Hamiltonian and non-Hamiltonian networks. When conducting WPG on a Hamiltonian network, each iteration only activates one edge communication, and consumes E_2 amount of energy. With k satisfying (39), WPG achieves $\|x^{kn} - x^*\|^2 \leq \epsilon$ and consumes the following amount of total energy:

$$\ln\left(\frac{\|z^0 - z^*\|^2}{\epsilon}\right) \Big/ \ln\left(1+\theta^*\right) \cdot (nE_1 + nE_2) \tag{40}$$

$$= \left[\ln\left(\|z^0 - z^*\|^2\right) + \ln\left(\frac{1}{\epsilon}\right)\right] \Big/ \ln\left(1+\theta^*\right) \cdot (nE_1 + nE_2). \tag{41}$$

We consider the asymptotic case that ϵ tends to 0, which corresponds to $\ln(1/\epsilon) \to \infty$. The energy consumption of WPG on Hamiltonian network is

$$O\left(\ln\left(\frac{1}{\epsilon}\right) / \ln\left(1+\theta^*\right) \cdot (nE_1 + nE_2)\right). \tag{42}$$

As for a non-Hamiltonian but connected network, to apply WPG, the shortest path cycle should be applied, which evokes no more than Δ edges in each iteration. Hence, on a non-Hamiltonian network, WPG consumes the following amount of total energy:

$$O\left(\ln\left(\frac{1}{\epsilon}\right) / \ln\left(1+\theta^*\right) \cdot (nE_1 + n\Delta E_2)\right) \tag{43}$$

where Δ is the diameter of the network, defined in Remark 1.

B. Energy Comparisons

Next, we list the energy consumptions of some existing decentralized algorithms for consensus optimization, based on the reported analyses of convergence rates. Note that the energy comparison is made in the asymptotic sense, that is, the amount of agents n tends to infinity and the solution precision ϵ tends to zero, whereas the function related parameters L, μ and κ are constants. The total energy consumption of IAG [19] depends on whether the network is Hamiltonian or not. With a Hamiltonian cycle, IAG consumes

$$O\left(\ln\left(\frac{1}{\epsilon}\right) / \ln\left(1 + \frac{2}{25n(2n+1)(\kappa+1)^2 - 2}\right) \cdot (E_1 + 2E_2)\right). \tag{44}$$

Using a shortest path cycle, it consumes

$$O\left(\ln\left(\frac{1}{\epsilon}\right)/\ln\left(1+\frac{2}{25n(2n+1)(\kappa+1)^2-2}\right)\cdot(E_1+2\Delta E_2)\right). \tag{45}$$

Here, the factor 2 before E_2 signifies the communication of both the variable and aggregated gradient in IAG; see Section **IV-C**.

In gossip-based algorithms, each agent j keeps a copy of local variables y_j, and shares y_j with its neighbors by linear combination. This combination can be represented by multiplying $\mathcal{Y} = [{y_1}^{\mathsf{T}}, \ldots, {y_n}^{\mathsf{T}}]^{\mathsf{T}}$ with a n-by-n mixing matrix $\mathbf{W}$, where w_{j_1,j_2} is positive when agents j_1 and j_2 are connected by an edge, otherwise it takes the value of 0. One choice of $\mathbf{W}$ is a symmetric doubly stochastic matrix, that is, $\mathbf{W}^{\mathsf{T}} = \mathbf{W}$ and $\mathbf{W1} = \mathbf{1}$. Existing convergence analyses have shown that the convergence speed of this type of algorithms depends on the second largest eigenvalue $\hat{\lambda}(\mathbf{W}) = \max\{|\lambda_i(\mathbf{W})| : \lambda_i(\mathbf{W}) \neq 1\}$. The energy consumption of EXTRA [27] for reaching an ϵ-precise solution is

$$O\left(\ln\left(\frac{1}{\epsilon}\right)/\ln\left(2-\hat{\lambda}(\mathbf{W})\right)\cdot(nE_1+mE_2)\right). \tag{46}$$

That of exact diffusion [36] is

$$O\left(\ln\left(\frac{1}{\epsilon}\right)/\ln\left(1+\frac{1-\hat{\lambda}(\mathbf{W})}{\hat{\lambda}(\mathbf{W})+C}\right)\cdot(nE_1+mE_2)\right) \tag{47}$$

where C is a constant only depending on (L/μ). Turning to single-step dual accelerated method (SSDA) [29], in each of its each iteration, the gradient of the conjugate of each $f_j(\cdot)$ is evaluated. We assume it consumes E_1 energy. Then, the energy consumption for SSDA is

$$O\left(\ln\left(\frac{1}{\epsilon}\right)\sqrt{\kappa}/\sqrt{\gamma}\cdot(nE_1+mE_2)\right) \tag{48}$$

where $\kappa = (L/\mu)$ is the condition number of the local functions, γ is the normalized spectral gap of a gossip matrix, which can be taken as $\mathbf{I} - \mathbf{W}$. With this gossip matrix, $\gamma \geq 1 - \hat{\lambda}(\mathbf{W})$, so the energy consumption of SSDA equals

$$O\left(\ln\left(\frac{1}{\epsilon}\right)\sqrt{\kappa}/\sqrt{1-\hat{\lambda}(\mathbf{W})}\cdot(nE_1+mE_2)\right). \tag{49}$$

In the same work, another method named MSDA is proposed, which conducts multiple gossips in each iteration. At the sacrifice of increasing the communication cost in each iteration, the number of iterations for convergence is reduced. It is also proved in [29] that in the sense of iterations for convergence, MSDA is the optimal algorithm among the class of gossip-based first-order algorithms. The energy consumption for MSDA is

$$O\left(\ln\left(\frac{1}{\epsilon}\right)\sqrt{\kappa}\cdot\left(nE_1+\frac{m}{\sqrt{1-\hat{\lambda}(\mathbf{W})}}E_2\right)\right). \tag{50}$$

Noticing that a connected graph means $m \geq n$, we compare the energy consumption of the above algorithms to find an ϵ-precise solution in Table **I**. For the ease of comparison, we take κ as a constant independent of n, m, and study the asymptotic case that is $n \to \infty$. Besides, we apply the approximation $\ln(1+x) \approx x$ when x is close to 0. Further, for better comparison between incremental algorithms and gossip-based algorithms, we apply the approximation $\Delta \approx (1/\sqrt{\gamma}) \leq [1/(\sqrt{1-\hat{\lambda}(\mathbf{W})})]$ [29], [37].

TABLE I
ENERGY CONSUMPTION OF VARIOUS ALGORITHMS WHEN $\hat{\lambda}(\mathbf{P})$ IS CLOSE TO 1 AND n TENDS TO ∞

Algorithm	Energy consumption	
	Hamiltonian cycle	shortest path cycle
WPG	$O\left(\ln\left(\frac{1}{\epsilon}\right)\cdot n\right)$	$O\left(\ln\left(\frac{1}{\epsilon}\right)\cdot\frac{n}{\sqrt{1-\hat{\lambda}(\mathbf{W})}}\right)$
IAG[19]	$O\left(\ln\left(\frac{1}{\epsilon}\right)\cdot n^2\right)$	$O\left(\ln\left(\frac{1}{\epsilon}\right)\cdot\frac{n^2}{\sqrt{1-\hat{\lambda}(\mathbf{W})}}\right)$
EXTRA[27]	$O\left(\ln\left(\frac{1}{\epsilon}\right)\cdot\frac{m}{1-\hat{\lambda}(\mathbf{W})}\right)$	
exact diffusion[28]	$O\left(\ln\left(\frac{1}{\epsilon}\right)\cdot\frac{m}{1-\hat{\lambda}(\mathbf{W})}\right)$	
SSDA[29]	$O\left(\ln\left(\frac{1}{\epsilon}\right)\cdot\frac{m}{\sqrt{1-\hat{\lambda}(\mathbf{W})}}\right)$	
MSDA[29]	$O\left(\ln\left(\frac{1}{\epsilon}\right)\cdot\frac{m}{\sqrt{1-\hat{\lambda}(\mathbf{W})}}\right)$	

We observe that WPG is more energy efficient than IAG under either communication cycle. This indicates that WPG is more scalable from the perspective of energy consumption. When the network is sparsely connected, i.e., $O(m) = O(n)$, the energy consumption of WPG with the shortest path cycle is in the same order with SSDA and MSDA. In other cases, WPG with the shortest path cycle is more energy efficient than the gossip-based algorithms.

Intuitively, incremental algorithms using a Hamiltonian cycle are more communication efficient and hence more energy efficient than using the shortest path cycle. Next, we analyze how much improvement will be achieved by knowing a Hamiltonian cycle *a priori*. Specifically, we need to analyze the relationship between m, $1 - \hat{\lambda}(\mathbf{W})^2$ and n. We study two different graph topologies.

1) In a ring graph where the graph itself is a Hamiltonian cycle and each agent in the network connects with two other agents, $1 - \hat{\lambda}(\mathbf{W})$ is $O(1/n^2)$, and $m = n$. In such a case, EXTRA and exact diffusion exhibit energy consumption of $O(n^3)$, while SSDA and MSDA consume $O(n^2)$ amount of energy.
2) Consider a complete graph, where each pair of agents is connected by an edge, which corresponds to $m = O(n^2)$ and $1 - \hat{\lambda}(\mathbf{W}) = 1$. Under this setting, EXTRA, exact diffusion, SSDA, and MSDA all consume $O(n^2)$ amount of energy.

In the above two extreme cases, WPG with a Hamiltonian cycle is better than all the compared algorithms.

VII. DISCUSSION

In this section, we first analyze a specific application of WPG on solving the consensus problem, and further show that WPG converges to the optimal solution after one pass of the

$^2 1 - \hat{\lambda}(\mathbf{W})$ labels the consensual speed achieved by repeatedly multiplying the mixing matrix, $\mathbf{W}$.

Algorithm 2 WPG Solving (51) With $\alpha = 1$

```
Initialization: initialize $x^0$ and $z_j^0$ so that $x^0 = \frac{1}{n}\sum_{j=1}^n z_j^0$ holds;
Repeat for $t = 0, 1, 2, \ldots$
    agent $i_t = \text{mod}_n(t) + 1$ do:
        receive token $x^t$;
        update local variable $z_{i_t}^{t+1} = y_{i_t}$;
        update token $x^{t+1} = x^t + \frac{1}{n}\left(z_{i_t}^{t+1} - z_{i_t}^t\right)$;
        send token $x^{t+1}$ to agent $i_{t+1}$ via edge $(i_t, i_{t+1})$;
    agents $j \neq i_t$ do nothing, i.e., $z_j^{t+1} = z_j^t$;
End
```

agents. Next, we discuss the application of WPG to dynamic topologies.

A. Convergence With One Pass for Consensus Problem

As a special case of consensus optimization problem, the average consensus problem is

$$\underset{x\in\mathbb{R}^p}{\text{minimize}} \ \frac{1}{2n}\sum_{j=1}^n \|x - y_j\|^2 \tag{51}$$

where each agent j stores a p-dimensional vector y_j. The optimal solution to problem (51) is $x^* = (\sum_{j=1}^n y_j)/n$, i.e., the average of all the locally stored y_j's. Next, we apply WPG to solve (51) with the step size $\alpha = 1$ (see Algorithm 2) and analyze its finite convergence.

Proposition 1: The sequence $(x^t, z_1^t, \ldots, z_n^t)_{t\geq 0}$ generated by Algorithm 2 converges after one pass of all the vertices. Specifically, we have

$$x^t = x^*, z_j^t = y_j, \quad \forall t \geq n+1, \quad \forall j \in \{1, \ldots, n\}. \tag{52}$$

Proof: Intuitively, the updating pattern of Algorithm 2 can be rephrased as follows.

1) Once the token passes agent j, its local variable z_j^t is assigned with the value of y_j.
2) The token x^t always maintains the average of the all the n local variables $\{z_j^t\}_{j\in\{1,\ldots,n\}}$.

By point 1, once the token has passed all the n agents, each local variable z_j^t is set as y_j and will not be changed, which proves the convergence of z_j^t. Point 2 guarantees the convergence of the token, x^t. ■

Remark 2: Note that Proposition 1 is not a simple application of Theorem 1. Because to derive Proposition 1, we utilize the special form of consensus problem and the step size $\alpha = 1$. If $\alpha \neq 1$, the one pass convergence property is not guaranteed, but the linear convergence rate given in Theorem 1 still holds.

B. Extension to Dynamic Network

As shown in Section IV-A, the communication strategy of the proposed WPG method only relies on a cycle of the network. In each iteration, only one edge on the cycle is activated for communication. No matter whether the other edges are capable of communication, the proposed WPG method works. This property indicates the applicability WPG to some periodically dynamic network.

For a general dynamic network, we have: 1) if the dynamic network always protects a stable Hamiltonian cycle, then WPG works and 2) if the dynamic network is possibly non-Hamiltonian, the routing protocol guarantees that an agent is able to find the shortest path toward any other agent.

VIII. Numerical Experiments

In this section, numerical experiments on three different consensus optimization problems are conducted, including least squares problem, logistic regression problem, and consensus problem. To verify the correctness of the proposed WPG and test its energy efficiency, we compare the proposed WPG with state-of-the-art consensus optimization methods. The compared methods are those compared with WPG theoretically in Section VI, including the deterministic incremental method, IAG [19], and gossip-based ones, MSDA [29], SSDA [29], EXTRA [27], and exact diffusion [28].

As for the incremental algorithms, both Hamiltonian cycle-based and shortest path cycle-based token traversing pattern are tested. Correspondingly, we generate two different network topologies for test, named as $\mathcal{G}_{\text{Hc}}(\mathcal{V}, \mathcal{E}_{\text{Hc}})$ and $\mathcal{G}_{\text{spc}}(\mathcal{V}, \mathcal{E}_{\text{spc}})$, respectively.

The possibly non-Hamiltonian topology $\mathcal{G}_{\text{spc}}(\mathcal{V}, \mathcal{E}_{\text{spc}})$ is generated from the random geometric graph model. Specifically, $n = 50$ agents are placed uniformly at random in a 50×50 square, an edge is connected between any pair of agents if their distance is not greater than 15. Based on $\mathcal{G}_{\text{spc}}(\mathcal{V}, \mathcal{E}_{\text{spc}})$, we generate a Hamiltonian network $\mathcal{G}_{\text{Hc}}(\mathcal{V}, \mathcal{E}_{\text{Hc}})$ by first generating a ring graph with the 50 agents, and then taking $\mathcal{E}_{\text{Hc}}$ as the union of the edge set of the ring graph and $\mathcal{E}_{\text{spc}}$.

For the completeness of our experimental results, we also compared the run times of different first-order decentralized algorithms. Time consumptions of decentralized algorithms depend on multiple factors including both computation and communication. In real-world networks, communication makes more contributions to total run time than computation, and communication is commonly modeled with the exponential distribution in [38] and [39]. For simplicity, we assume that each communication requires a time independently following the exponential distribution with parameter 1. Each iteration of gossip-based algorithms, such as EXTRA, exact diffusion, and SSDA, involves one gossip round (all the m edges to communicate bidirectionally), and thus needs to wait for the slowest communication (out of the $2m$ communications) to complete. This determines the time spent on each round of communication. In contrast, incremental algorithms including IAG and WPG only evoke one communication per iteration. There is no coordination and thus no wait.

For the simplicity of evaluating energy consumption, in each experiment we assume that the gradient evaluation of each local function takes $E_1 = 1$ unit of energy, while each communication between a pair of connected agents takes $E_2 = 1$ unit of energy.

A. Least Squares

Consider a sensing problem in a network of n agents. Each agent $i \in \{1, \ldots, n\}$ captures its own linear measurements

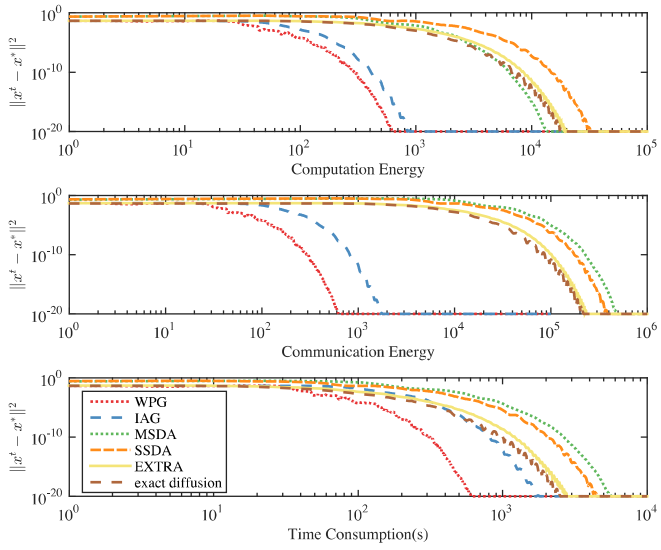


Fig. 4. Performance of different consensus optimization methods on least squares on the network $\mathcal{G}_{\text{Hc}}(\mathcal{V}, \mathcal{E}_{\text{Hc}})$.

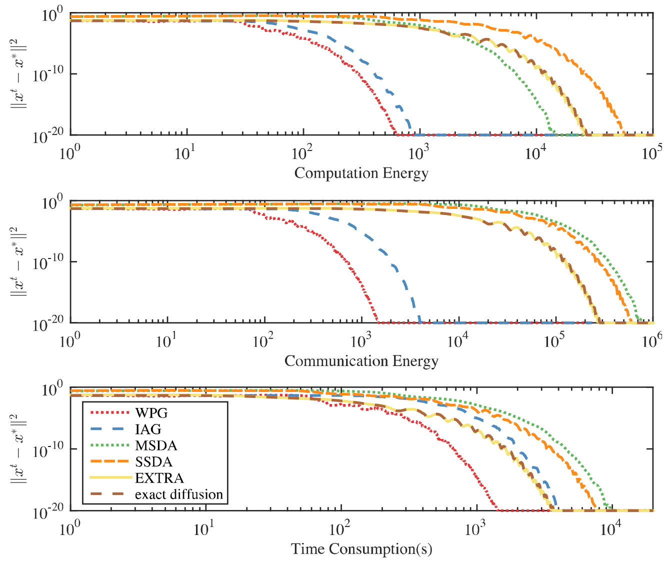


Fig. 5. Performance of different consensus optimization methods on least squares on the network $\mathcal{G}_{\text{spc}}(\mathcal{V}, \mathcal{E}_{\text{spc}})$.

of the unknown signal $x \in \mathbb{R}^p$ by $y_i = \mathbf{M}_i x + e_i$, where the measurement $y_i \in \mathbb{R}^{d\times 1}$, $\mathbf{M}_i \in \mathbb{R}^{d\times p}$ and $e_i \in \mathbb{R}^{d\times 1}$ is unknown noise. To estimate y, the n agents cooperate to solve

$$\underset{x\in\mathbb{R}^p}{\text{minimize}} \ \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{M}_i x - y_i\|^2. \tag{53}$$

In this experiment, we fix $d = 1, p = 3$, and take $\mathbf{M}_i \sim \mathcal{N}(0, \mathbf{I})$, $y_i \sim \mathcal{N}(0, 1)$ as independent Gaussian random variables.

The experimental result on the network $\mathcal{G}_{\text{Hc}}(\mathcal{V}, \mathcal{E}_{\text{Hc}})$ is shown in Fig. 4, while the result on $\mathcal{G}_{\text{spc}}(\mathcal{V}, \mathcal{E}_{\text{spc}})$ can be found in Fig. 5. In the upper two subplots of each figure, we compare the computation energy consumption, communication energy consumption separately, while the third subplot compares the time consumption. In both Figs. 4 and 5, it can

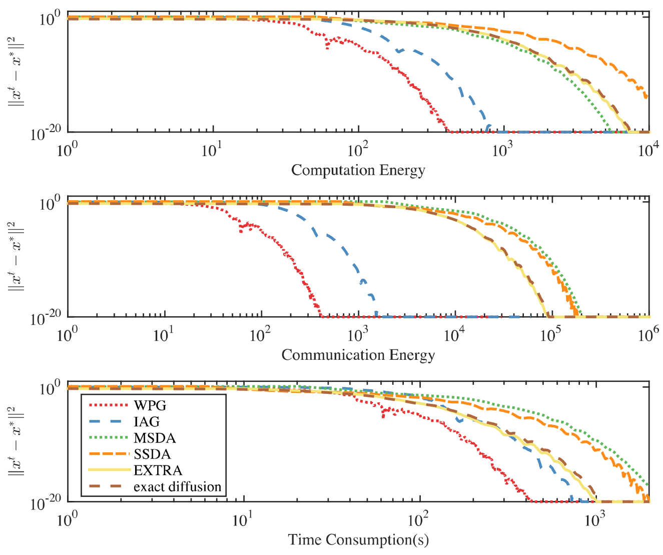


Fig. 6. Performance of different consensus optimization methods on training logistic classifier on the network $\mathcal{G}_{\text{Hc}}(\mathcal{V}, \mathcal{E}_{\text{Hc}})$.

be observed that the incremental algorithms including IAG and WPG is more energy efficient than the gossip-based ones, and the proposed WPG is the most energy efficient, no matter when utilizing Hamiltonian cycle on the network $\mathcal{G}_{\text{Hc}}(\mathcal{V}, \mathcal{E}_{\text{Hc}})$ or the shortest path cycle on the network $\mathcal{G}_{\text{spc}}(\mathcal{V}, \mathcal{E}_{\text{spc}})$. In terms of time consumption, WPG is more efficient than IAG and the gossip-based algorithms.

Comparing Figs. 4 and 5, one can observe that the trends of the curves are very similar. The main difference is the convergence speed.

1) The convergence speed of the gossip-based algorithms shown in Fig. 5 are slower than Fig. 4, which is because the $\mathcal{G}_{\text{spc}}(\mathcal{V}, \mathcal{E}_{\text{spc}})$ is less densely connected than $\mathcal{G}_{\text{Hc}}(\mathcal{V}, \mathcal{E}_{\text{Hc}})$, which affects the gossiping speed over the network.
2) Regarding the incremental algorithms (WPG and IAG), these two figures share the same computation energy curves, but communication energy curves and time consumption curves are different. The reason is that the intrinsic computations of the incremental algorithms on these two networks coincide with each other, whereas the communication is different due to the relays involved in the shortest path cycle.

B. Training Logistic Classifier

With the training data including features and labels stored distributedly over the network, the decentralized regression problem refers to learn a classifier x by solving

$$\underset{x\in\mathbb{R}^p}{\text{minimize}} \ \frac{1}{n}\sum_{i=1}^{n}\frac{1}{b}\left[\sum_{j=1}^{b}\ln\Big(1+\exp\Big(-y_{ij}v_{ij}{}^{\mathsf{T}}x\Big)\Big)\right] + \lambda\|x\|^2 \tag{54}$$

where $y_{ij} \in \{-1, 1\}$ denotes the label of the jth sample kept by the ith agent, $v_{ij} \in \mathbb{R}^p$ represents its feature, and there are b samples kept by each agent.

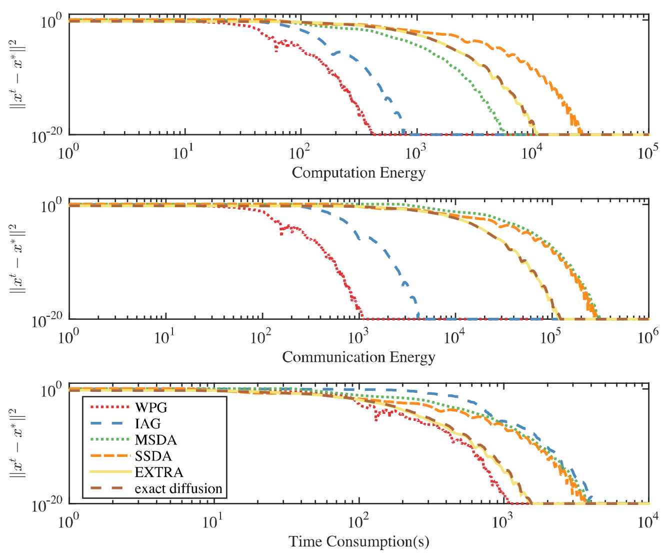


Fig. 7. Performance of different consensus optimization methods on training logistic classifier on the network $\mathcal{G}_{\text{spc}}(\mathcal{V}, \mathcal{E}_{\text{spc}})$.

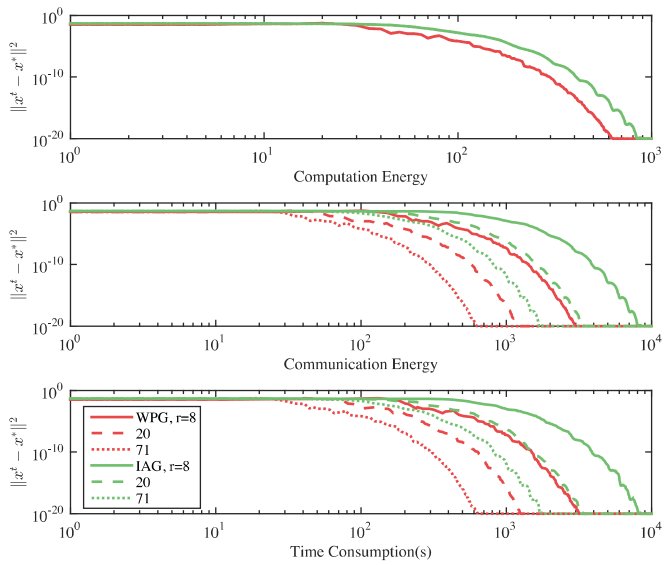


Fig. 9. Performance of WPG and IAG on least squares problem on $\mathcal{G}_{\text{spc}}(\mathcal{V}, \mathcal{E}_{\text{spc}})$ networks generated with different covering radii r.

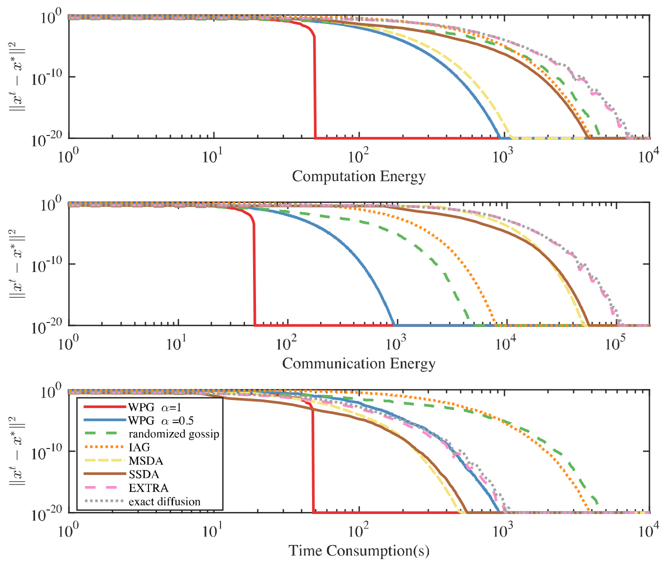


Fig. 8. Performance of different consensus optimization methods on consensus problem on the network $\mathcal{G}_{\text{Hc}}(\mathcal{V}, \mathcal{E}_{\text{Hc}})$.

In this experiment, we fix $b = 5, p = 3$, and take the sample features $v_{ij} \sim \mathcal{N}(0, 1)$. To generate y_{ij}, we first generate a random vector $x^0 \in \mathbb{R}^3 \sim \mathcal{N}(0, \mathbf{I})$. Then we generate a uniformly distributed variable $z_{ij} \sim \mathcal{U}(0, 1)$, and if $z_{ij} \le 1/[1 + \exp(-{v_{ij}}^\top x^0)]$, y_{ij} is taken as 1; otherwise y_{ij} is set as -1.

The numerical results on the two different topologies are shown in Figs. 6 and 7. Similar phenomena can be observed as decentralized least squares experiments. The energy efficiency and time efficiency of the proposed WPG method is also verified in this experiment.

C. Consensus Problem

In order to test the one pass convergence of the WPG method on solving consensus problem, in this experiment on the network $\mathcal{G}_{\text{Hc}}(\mathcal{V}, \mathcal{E}_{\text{Hc}})$ we implement Algorithm 2, i.e., WPG with the optimal step size value $\alpha = 1$, to solve problem (51) with $p = 1$. The locally cached vectors y_j's are independent identically distributed random variables following the uniform distribution in the range of [0, 1]. Besides, we test a nonoptimal choice of step size, $\alpha = 0.5$, and compare it with the state-of-the-art consensus optimization algorithms. For the completeness of comparison, we also conduct a method named randomized gossip [40], which is a random incremental method specifically designed for consensus problem. The numerical result is shown in Fig. 8. It can be observed that, the WPG method with $\alpha = 1$ converges after 50 computations, and 50 communications, which labels one pass of the 50 agents. This coincides with the analysis in Section VII-A. Even when the step size of WPG is not taken as the optimal value 1, WPG with $\alpha = 0.5$ still shows competitive energy efficiency over the compared methods.

D. Effects of Edge Connection Density

This experiment tests the efficiency of the proposed WPG under different edge connection densities. To generate network topologies with different edge connection densities, we vary the covering radius of each vertex r of the SPC network. The connecting rule is that any pair of vertices are connected if their distance is less than r, and not connected otherwise. Hence, increasing r would increase the edge connection density. With r taken as 8, 20, and 71 (generating complete graph), respectively, the compared consensus optimization methods are applied to solve a same least squares problem. For ease of display, we compare the proposed WPG with IAG and a representative of gossip algorithms, EXTRA in Figs. 9 and 10, respectively.

It can be read from the first subplot of Fig. 9 that the computation energy curves of WPG under different r perfectly coincide with each other. The reason is that as long as the

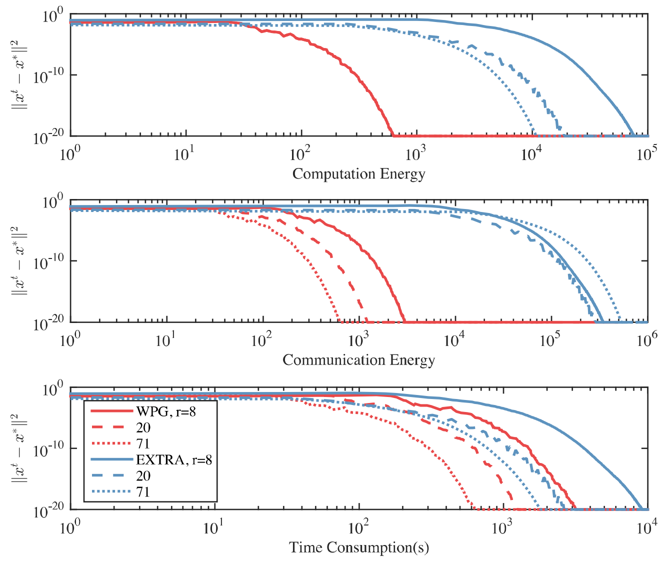


Fig. 10. Performance of WPG and EXTRA on least squares problem on $\mathcal{G}_{\text{spc}}(\mathcal{V}, \mathcal{E}_{\text{spc}})$ networks generated with different covering radii r.

traversing pattern of the walking token is determined, the computing procedures of incremental algorithms are fixed and not influenced by the network topology. In the other two subplots, the gaps among the curves of each algorithm are caused by the differences of routing path's length. For both WPG and IAG, a larger r generates a more densely connected network, where one could expect that less relay nodes are required when passing the token from one vertex to another, which results in less communication consumption and less time consumption. Fig. 9 also indicates that the proposed WPG outperforms IAG under different network topologies.

In Fig. 10, the relative positions of the convergence curves of EXTRA are more complicated. The first subplot verifies the convergence property of EXTRA as in (46), that is, EXTRA converges faster on a more densely connected graph. In the second subplot, one could read that when increasing r, the communication consumption of EXTRA first decreases then increases. The reason is that when r increases the number of iterations for convergence decreases, whereas the per-iteration communication cost increases. According to the last subplots, the time efficiency of EXTRA increases with r increasing, which implies that in the experimental time model, the gain on convergence rate dominates the increase of per-iteration time consumption.

Comparing WPG and EXTRA, one could read that under the tested edge connection densities, WPG outperforms EXTRA in terms of both energy consumption and time consumption. As for the energy consumption, the computation energy benefit of WPG is more obvious in a sparsely connected network, whereas the communication energy benefit is maximized in the most densely connected network (complete graph).

IX. Conclusion

Inspired by the walking token idea and deterministic routing of the network, based on proximal-gradient algorithm, we propose a first-order incremental decentralized consensus optimization method, WPG. Two different communication strategies, i.e., Hamiltonian cycle and shortest path cycle are considered. The linear convergence rate is derived under the strongly convex case, based on which, the energy efficiency of the proposed method over the existing consensus optimization methods is theoretically analyzed and numerically testified. Besides, the one-pass convergence property of WPG in solving consensus problem is also established, which has not been found for any existing first-order decentralized consensus optimization method.

References

[1] M. Leinonen, M. Codreanu, and M. Juntti, "Distributed joint resource and routing optimization in wireless sensor networks via alternating direction method of multipliers," *IEEE Trans. Wireless Commun.*, vol. 12, no. 11, pp. 5454–5467, Nov. 2013.

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

[3] Q. Ling and Z. Tian, "Decentralized sparse signal recovery for compressive sleeping wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 58, no. 7, pp. 3816–3827, Jul. 2010.

[4] V. Kekatos and G. B. Giannakis, "Distributed robust power system state estimation," *IEEE Trans. Power Syst.*, vol. 28, no. 2, pp. 1617–1626, May 2013.

[5] D. P. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1439–1451, Aug. 2006.

[6] D. Zennaro, E. Dall'Anese, T. Erseghe, and L. Vangelista, "Fast clock synchronization in wireless sensor networks via ADMM-based consensus," in *Proc. IEEE Int. Symp. Model. Optim. Mobile Ad Hoc Wireless Netw. (WiOpt)*, 2011, pp. 148–153.

[7] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 988–1001, Jun. 2003.

[8] A. N. Tantawi and D. Towsley, "Optimal static load balancing in distributed computer systems," *J. ACM*, vol. 32, no. 2, pp. 445–465, 1985.

[9] N.-C. Wang, C.-P. Yen, and C.-P. Chu, "Multicast communication in wormhole-routed symmetric networks with Hamiltonian cycle model," *J. Syst. Archit.*, vol. 51, no. 3, pp. 165–183, 2005.

[10] L. Guo, X. Wang, J. Du, and T. Wu, "A new heuristic routing algorithm with Hamiltonian cycle protection in survivable networks," *Comput. Commun.*, vol. 31, no. 9, pp. 1672–1678, 2008.

[11] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: A survey," *IEEE Wireless Commun.*, vol. 11, no. 6, pp. 6–28, Dec. 2004.

[12] J.-H. Chang and L. Tassiulas, "Maximum lifetime routing in wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 12, no. 4, pp. 609–619, Aug. 2004.

[13] C. E. Jones, K. M. Sivalingam, P. Agrawal, and J. C. Chen, "A survey of energy efficient network protocols for wireless networks," *Wireless Netw.*, vol. 7, no. 4, pp. 343–358, 2001.

[14] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy-aware wireless microsensor networks," *IEEE Signal Process. Mag.*, vol. 19, no. 2, pp. 40–50, Mar. 2002.

[15] F. Liu, C.-Y. Tsui, and Y. J. Zhang, "Joint routing and sleep scheduling for lifetime maximization of wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 9, no. 7, pp. 2258–2267, Jul. 2010.

[16] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proc. 21st Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM)*, vol. 3, 2002, pp. 1567–1576.

[17] W. Yin, X. Mao, K. Yuan, Y. Gu, and A. H. Sayed, "A communication-efficient random-walk algorithm for decentralized optimization," arXiv preprint arXiv:1804.06568, 2018.

[18] S. V. Pemmaraju and S. S. Skiena, *Computational Discrete Mathematics: Combinatorics and Graph Theory With Mathematica*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[19] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA, USA: Athena Sci., 1999.

[20] D. Blatt, A. O. Hero, and H. Gauchman, "A convergent incremental gradient method with a constant step size," *SIAM J. Optim.*, vol. 18, no. 1, pp. 29–51, 2007.

[21] M. Gürbüzbalaban, A. Ozdaglar, and P. A. Parrilo, "On the convergence rate of incremental aggregated gradient algorithms," *SIAM J. Optim.*, vol. 27, no. 2, pp. 1035–1048, 2017.

[22] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM J. Optim.*, vol. 7, no. 4, pp. 913–926, 1997.

[23] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Incremental stochastic subgradient algorithms for convex optimization," *SIAM J. Optim.*, vol. 20, no. 2, pp. 691–717, 2009.

[24] B. Johansson, M. Rabi, and M. Johansson, "A randomized incremental subgradient method for distributed optimization in networked systems," *SIAM J. Optim.*, vol. 20, no. 3, pp. 1157–1170, 2009.

[25] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Püschel, "D-ADMM: A communication-efficient distributed algorithm for separable optimization," *IEEE Trans. Signal Process.*, vol. 61, no. 10, pp. 2718–2723, May 2013.

[26] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization," *IEEE Trans. Signal Process.*, vol. 62, no. 7, pp. 1750–1761, Apr. 2014.

[27] 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.

[28] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact diffusion for distributed optimization and learning—Part I: Algorithm development," arXiv preprint arXiv:1702.05122, 2017.

[29] K. Scaman, F. Bach, S. Bubeck, Y. T. Lee, and L. Massoulié, "Optimal algorithms for smooth and strongly convex distributed optimization in networks," arXiv preprint arXiv:1702.08704, 2017.

[30] A. Mokhtari, Q. Ling, and A. Ribeiro, "Network Newton—Part I: Algorithm and convergence," arXiv preprint arXiv:1504.06017, 2015.

[31] O. Ore, "Note on Hamilton circuits," *Amer. Math. Monthly*, vol. 67, no. 1, p. 55, 1960.

[32] N. Chiba and T. Nishizeki, "The Hamiltonian cycle problem is linear-time solvable for 4-connected planar graphs," *J. Algorithms*, vol. 10, no. 2, pp. 187–211, 1989.

[33] J. S. Deogun and G. Steiner, "Polynomial algorithms for Hamiltonian cycle in cocomparability graphs," *SIAM J. Comput.*, vol. 23, no. 3, pp. 520–552, 1994.

[34] E. K. Ryu and W. Yin, "Proximal-proximal-gradient method," arXiv preprint arXiv:1708.06908, 2017.

[35] J.-B. Baillon and G. Haddad, "Quelques propriétés des opérateurs angle-bornés etn-cycliquement monotones," *Israel J. Math.*, vol. 26, no. 2, pp. 137–150, 1977.

[36] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact diffusion for distributed optimization and learning—Part II: Convergence analysis," arXiv preprint arXiv:1702.05142, 2017.

[37] N. Alon and V. D. Milman, "$\lambda 1$, isoperimetric inequalities for graphs, and superconcentrators," *J. Comb. Theory B*, vol. 38, no. 1, pp. 73–88, 1985.

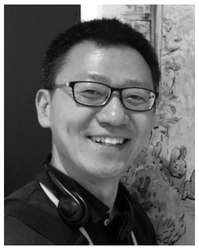
[38] L. Zhou, X. Wang, W. Tu, G.-M. Muntean, and B. Geller, "Distributed scheduling scheme for video streaming over multi-channel multi-radio multi-hop wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 3, pp. 409–419, Apr. 2010.

[39] X. Wang, W. Huang, S. Wang, J. Zhang, and C. Hu, "Delay and capacity tradeoff analysis for motioncast," *IEEE/ACM Trans. Netw.*, vol. 19, no. 5, pp. 1354–1367, Oct. 2011.

[40] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized Gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006.

Xianghui Mao (GS'18) received the B.E. degree (Hons.) from the Electronic Engineering Department, Tsinghua University, Beijing, China, in 2014, where she is currently pursuing the Ph.D. degree with the Electronic Engineering Department, under the supervision of Prof. Y. Gu.

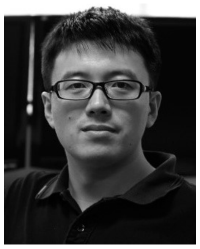
In 2017, she was a Visiting Graduate Researcher with the University of California at Los Angeles, Los Angeles, CA, USA, for six months. Her current research interests include optimization-related topics and graph signal processing.



Yuantao Gu (S'02–M'03–SM'17) received the B.E. degree in electronic engineering from Xi'an Jiaotong University, Xi'an, China, in 1998, and the Ph.D. degree (Hons.) in electronic engineering from Tsinghua University, Beijing, China, in 2003.

In 2003, he joined the faculty of Tsinghua University, where he is currently a Tenured Associate Professor with the Department of Electronic Engineering. He was a Visiting Scientist with Microsoft Research Asia, Beijing, from 2005 to 2006, the Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA, USA, from 2012 to 2013, and the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA, in 2015. His current research interests include high-dimensional statistics, sparse signal recovery, temporal-space and graph signal processing, related topics in wireless communications and information networks.

Dr. Gu was a recipient of the Best Paper Award of IEEE Global Conference on Signal and Information Processing in 2015, the Award for Best Presentation of Journal Paper of IEEE International Conference on Signal and Information Processing in 2015, and the Zhang Si-Ying (CCDC) Outstanding Youth Paper Award (with his student) in 2017. He has been an associate editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING since 2015, a Handling Editor for EURASIP Digital Signal Processing from 2015 to 2017, and an Elected Member of the IEEE Signal Processing Theory and Methods Technical Committee since 2017.



Wotao Yin received the B.S. degree in mathematics and applied mathematics from Nanjing University, Nanjing, China, in 2001, and the M.S. and Ph.D. degrees in operations research from Columbia University, New York, NY, USA, in 2003 and 2006, respectively.

From 2006 to 2013, he was an Assistant Professor and then an Associate Professor with the Department of Computational and Applied Mathematics, Rice University, Houston, TX, USA. Since 2013, he has been a Professor with the Department of Mathematics, University of California at Los Angeles, Los Angeles, CA, USA. His current research interests include large-scale decentralized/distributed optimization, operator splitting methods, and machine learning.