2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Convergence Rate of Controlled Hopwise Averaging on Various Graphs

Jie Lu and Choon Yik Tang

Abstract— **This paper analyzes the convergence properties of *Ideal Controlled Hopwise Averaging* (ICHA) and *Controlled Hopwise Averaging* (CHA), two recently proposed asynchronous distributed averaging algorithms for wireless networks, which were shown via simulation to be more bandwidth/energy efficient than several existing schemes. We first derive deterministic upper bounds on the exponential convergence rate of ICHA on general graphs and specific ones (i.e., path, cycle, regular, strongly regular, and complete graphs), expressing the bounds explicitly in terms of the graph invariants. We then show that such bounds of ICHA are roughly 20% better than the average-case, stochastic convergence rate of Pairwise Averaging on path, cycle, and complete graphs. Finally, we obtain upper bounds on the convergence rate of CHA with respect to iteration and time, and show that iteration-wise CHA enjoys the same bounds as ICHA, closely mimicking its behavior while being practical.**

I. Introduction

Distributed averaging is a fundamental problem in distributed computation that finds many applications in multi-agent systems, ad hoc networks, sensor networks and the likes. Due to its significance, the problem has been widely studied (see, e.g., [1]–[18]) for different network models (e.g., wired or wireless; undirected or directed links; fixed or time-varying topologies), with different communication assumptions (e.g., without delays, errors, and quantization or with), and in different time domains (e.g., continuous- or discrete-time; synchronous or asynchronous). The research efforts have led to a growing set of algorithms, including Pairwise Averaging [1], Randomized Gossip Algorithm [6], Accelerated Gossip Algorithm [7], Distributed Random Grouping [8], and Linear Prediction-Based Accelerated Averaging [18], to name just a few.

In our recent work [19], we studied the distributed averaging problem in wireless networks with undirected links and fixed topologies. We showed that although the existing algorithms could solve the problem over such networks, they are prone to wasteful communications and, thus, are bandwidth/energy inefficient, offering room for improvement. To increase efficiency, we developed in [19] two new algorithms, referred to as *Ideal Controlled Hopwise Averaging* (ICHA) and *Controlled Hopwise Averaging* (CHA), which attempt to "make the most" out of each communication. As these two algorithms form the backbone of this paper, we briefly summarize their key features in the next paragraph.

Similar to some of the available distributed averaging algorithms (e.g., Pairwise Averaging [1] and Distributed Random Grouping [8]), both ICHA and CHA update their state variables asynchronously by forming convex combinations. Unlike all of the available algorithms, however, the state variables are assigned to the *links*, rather than to the *nodes*. Due primarily to this feature, ICHA and CHA are able to fully exploit the broadcast nature of wireless channels, requiring only one real-number transmission per iteration. Moreover, due to a suitably defined common quadratic Lyapunov function, ICHA and CHA are able to perform what we call *feedback iteration control*, using potential drops in the value of the Lyapunov function as feedback to control the order by which the asynchronous iterations occur. This last feature is also what separates the two algorithms: ICHA assumes that there is a "genie" in the network (hence the term *Ideal* in its name), who knows all the potential drops, decides to be greedy, and keeps selecting the node with the largest potential drop to initiate the next iteration, resulting in a networked dynamical system with state-dependent switching, for which the Lyapunov function value drops maximally every time (hence the term *greedy*). In contrast, CHA makes no such assumption and, instead, lets every node use its own potential drop to decentralizedly schedule when to initiate an iteration, leading to a practical discrete event system that tries to mimic the greedy behavior of ICHA. Finally, we showed via extensive simulation in [19] that CHA is markedly more efficient than Pairwise Averaging [1], Consensus Propagation [11], Algorithm A2 of [12], and Distributed Random Grouping [8] on random geometric graphs, achieving convergence with far fewer real-number transmissions.

In this paper, we study the convergence properties of ICHA and CHA. Our goal is to obtain, for the two algorithms, deterministic upper bounds $\rho(\mathcal{G})$ on the rate at which the Lyapunov function value $V(\mathbf{x}(k))$ converges exponentially to zero on various graphs $\mathcal{G}$, i.e.,

$$V(\mathbf{x}(k)) \leq \rho(\mathcal{G}) V(\mathbf{x}(k-1)), \quad \forall k = 1, 2, \ldots,$$

where k denotes iteration and $\mathbf{x}(k)$ denotes the state vector. We begin by deriving the bounds $\rho(\mathcal{G})$ of ICHA on general graphs and several specific ones, including path, cycle, regular, strongly regular, and complete graphs. We show that the bounds $\rho(\mathcal{G})$ can be expressed explicitly in terms of the graph order (i.e., the number of nodes), diameter, and degrees (i.e., the number of neighbors of each node). We then proceed to compare such bounds of ICHA—which provide hard guarantees—with the stochastic convergence rate of Pairwise Averaging [1] reported in [15]—which provides guarantees only in the *average* sense. Despite the difference in guarantees, we show that the former are roughly 20%

J. Lu and C. Y. Tang are with the School of Electrical and Computer Engineering, University of Oklahoma, Norman, OK 73019, USA (e-mail: {jie.lu-1,cytang}@ou.edu).

This work was supported by the National Science Foundation under grant CMMI-0900806.

better than the latter on three common graphs of opposing densities, namely, path, cycle, and complete graphs. Finally, we obtain the bounds $\rho(\mathcal{G})$ of CHA with respect to both iteration and time. We show that iteration-wise, CHA shares the same bounds as ICHA, suggesting that CHA does closely mimic ICHA while being practical. Time-wise, CHA converges asymptotically and perhaps exponentially, depending on its controller parameter.

Although the main contribution of this paper is the convergence rate analysis of ICHA and CHA, we have also included their development, which is more complete than in [19]. Moreover, due to space limitations, we have omitted all the proofs from this paper and refer the reader to [20].

II. Ideal Controlled Hopwise Averaging

A. Algorithm Development

Consider a multi-hop wireless network consisting of $N \geq 2$ nodes, connected by L bidirectional links in a fixed topology. The network is modeled as a connected, undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \ldots, N\}$ represents the set of N nodes and $\mathcal{E} \subset \{\{i, j\} : i, j \in \mathcal{V}, i \neq j\}$ represents the set of L links. Any two nodes $i, j \in \mathcal{V}$ are one-hop neighbors and can communicate if and only if $\{i, j\} \in \mathcal{E}$. The set of one-hop neighbors of each node $i \in \mathcal{V}$ is denoted as $\mathcal{N}_i = \{j \in \mathcal{V} : \{i, j\} \in \mathcal{E}\}$, and the communications are assumed to be delay- and error-free, with no quantization. Each node $i \in \mathcal{V}$ observes a scalar $y_i \in \mathbb{R}$, and all the N nodes wish to determine the network-wide average $x^* \in \mathbb{R}$ of their individual observations, given by

$$x^* = \frac{1}{N} \sum_{i \in \mathcal{V}} y_i. \tag{1}$$

To solve this problem, consider a networked dynamical system, defined on the graph $\mathcal{G}$ as follows: associated with each link $\{i, j\} \in \mathcal{E}$ are a parameter $c_{\{i,j\}} > 0$ and a state variable $x_{\{i,j\}} \in \mathbb{R}$ of the system. In addition, associated with each node $i \in \mathcal{V}$ is an output variable $\hat{x}_i \in \mathbb{R}$, which represents its estimate of the unknown average x^* in (1). Since the graph $\mathcal{G}$ has L links and N nodes, the system has L parameters $c_{\{i,j\}}$'s, L state variables $x_{\{i,j\}}$'s, and N output variables $\hat{x}_i$'s. To describe the system dynamics, let $x_{\{i,j\}}(0)$ and $\hat{x}_i(0)$ represent the initial values of $x_{\{i,j\}}$ and $\hat{x}_i$, and $x_{\{i,j\}}(k)$ and $\hat{x}_i(k)$ their values upon completing each iteration $k \in \mathbb{P}$, where $\mathbb{P}$ denotes the set of positive integers. With these notations, the state and output equations governing the system dynamics may be stated as

$$x_{\{i,j\}}(k) = \begin{cases} \dfrac{\sum_{\ell \in \mathcal{N}_{u(k)}} c_{\{u(k),\ell\}} x_{\{u(k),\ell\}}(k-1)}{\sum_{\ell \in \mathcal{N}_{u(k)}} c_{\{u(k),\ell\}}} \\ \qquad \text{if } u(k) \in \{i, j\}, \\ x_{\{i,j\}}(k-1), \text{ otherwise}, \end{cases} \\ \forall k \in \mathbb{P},\ \forall \{i, j\} \in \mathcal{E}, \tag{2}$$

$$\hat{x}_i(k) = \frac{\sum_{j \in \mathcal{N}_i} c_{\{i,j\}} x_{\{i,j\}}(k)}{\sum_{j \in \mathcal{N}_i} c_{\{i,j\}}}, \quad \forall k \in \mathbb{N},\ \forall i \in \mathcal{V}, \tag{3}$$

where $\mathbb{N}$ denotes the set of nonnegative integers and $u(k) \in \mathcal{V}$ is the node that initiates iteration $k \in \mathbb{P}$, so that the sequence $(u(k))_{k=1}^{\infty}$ fully dictates how the asynchronous iteration (2) takes place.

For the system (2) and (3) to solve the distributed averaging problem, the $\hat{x}_i(k)$'s must satisfy

$$\lim_{k \to \infty} \hat{x}_i(k) = x^*, \quad \forall i \in \mathcal{V}. \tag{4}$$

Due to (3), condition (4) is met if the $x_{\{i,j\}}(k)$'s satisfy

$$\lim_{k \to \infty} x_{\{i,j\}}(k) = x^*, \quad \forall \{i, j\} \in \mathcal{E}. \tag{5}$$

To ensure (5), the parameters $c_{\{i,j\}}$'s and initial states $x_{\{i,j\}}(0)$'s must satisfy a condition. To derive the condition, observe from (2) that no matter what $u(k)$ is,

$$\sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}} x_{\{i,j\}}(k) = \sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}} x_{\{i,j\}}(k-1), \quad \forall k \in \mathbb{P}. \tag{6}$$

Therefore, as it follows from (6) and (1), (5) holds only if the $c_{\{i,j\}}$'s and $x_{\{i,j\}}(0)$'s satisfy

$$\frac{\sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}} x_{\{i,j\}}(0)}{\sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}}} = \frac{\sum_{i \in \mathcal{V}} y_i}{N}. \tag{7}$$

To achieve (7), notice that the expressions $\sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}}$ and $\sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}} x_{\{i,j\}}(0)$ each has L terms, of which $|\mathcal{N}_i|$ terms are associated with links incident to node i, for every $i \in \mathcal{V}$, where $|\cdot|$ denotes the cardinality of a set. Hence, by letting each node $i \in \mathcal{V}$ evenly distribute the number 1 to the $|\mathcal{N}_i|$ terms in $\sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}}$, i.e.,

$$c_{\{i,j\}} = \frac{1}{|\mathcal{N}_i|} + \frac{1}{|\mathcal{N}_j|}, \quad \forall \{i, j\} \in \mathcal{E}, \tag{8}$$

we get $\sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}} = N$. Similarly, by letting each node $i \in \mathcal{V}$ evenly distribute its observation y_i to the $|\mathcal{N}_i|$ terms in $\sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}} x_{\{i,j\}}(0)$, i.e.,

$$x_{\{i,j\}}(0) = \frac{\frac{y_i}{|\mathcal{N}_i|} + \frac{y_j}{|\mathcal{N}_j|}}{c_{\{i,j\}}}, \quad \forall \{i, j\} \in \mathcal{E}, \tag{9}$$

we get $\sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}} x_{\{i,j\}}(0) = \sum_{i \in \mathcal{V}} y_i$. Thus, (8) and (9) together ensure (7), which is necessary for achieving (5).

To show that the system (2), (3), (8), (9) enables (5) and (4), consider a quadratic Lyapunov function candidate $V : \mathbb{R}^L \to \mathbb{R}$, defined as

$$V(\mathbf{x}(k)) = \sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}} (x_{\{i,j\}}(k) - x^*)^2, \tag{10}$$

where $\mathbf{x}(k) \in \mathbb{R}^L$ denotes the state vector. Clearly, V in (10) is positive definite with respect to $(x^*, x^*, \ldots, x^*) \in \mathbb{R}^L$, so that $\lim_{k \to \infty} V(\mathbf{x}(k)) = 0$ implies (5) and (4). The following lemma shows that $V(\mathbf{x}(k))$ is always non-increasing and quantifies its changes:

Lemma 1 ([19])**.** *Consider the system described by* (2)*,* (3)*,* (8)*, and* (9)*. Then, for any sequence* $(u(k))_{k=1}^{\infty}$*, the sequence* $(V(\mathbf{x}(k)))_{k=0}^{\infty}$ *is non-increasing and satisfies*

$$V(\mathbf{x}(k)) - V(\mathbf{x}(k-1)) = -\sum_{j\in\mathcal{N}_{u(k)}} c_{\{u(k),j\}} \times (x_{\{u(k),j\}}(k-1) - \hat{x}_{u(k)}(k-1))^2, \quad \forall k \in \mathbb{P}. \quad (11)$$

Lemma 1 says that V in (10) is a *common* quadratic Lyapunov function for the linear switched system. Moreover, since $V(\mathbf{x}(k))$ is nonnegative, it implies that for any $(u(k))_{k=1}^{\infty}$, $\lim_{k\to\infty} V(\mathbf{x}(k))$ exists and is nonnegative. Indeed, by using Corollary 3.2 of [15], it can be shown that almost any $(u(k))_{k=1}^{\infty}$ ensures $\lim_{k\to\infty} V(\mathbf{x}(k)) = 0$ and thus (5) and (4).

Although almost any $(u(k))_{k=1}^{\infty}$ can drive all the $\hat{x}_i(k)$'s to any neighborhood of x^*, certain sequences may require fewer iterations (and, hence, fewer real-number transmissions) to do so than others, yielding better bandwidth/energy efficiency. Therefore, it is desirable to produce such an efficient $(u(k))_{k=1}^{\infty}$, perhaps by means of *control*. To this end, for each $i \in \mathcal{V}$, let $\Delta V_i : \mathbb{R}^L \to \mathbb{R}$ be a positive semidefinite quadratic function, defined as

$$\Delta V_i(\mathbf{x}(k)) = \sum_{j\in\mathcal{N}_i} c_{\{i,j\}}(x_{\{i,j\}}(k) - \hat{x}_i(k))^2. \quad (12)$$

Notice that $\Delta V_i(\mathbf{x}(k))$ depends entirely on parameters and variables associated with links attached to node i or with node i itself. Also note from (11) and (12) that

$$V(\mathbf{x}(k)) - V(\mathbf{x}(k-1)) = -\Delta V_{u(k)}(\mathbf{x}(k-1)), \quad \forall k \in \mathbb{P}. \quad (13)$$

Thus, every node $i \in \mathcal{V}$ at any time knows by how much the value of V would drop if it suddenly initiates an iteration (i.e., by $\Delta V_i(\mathbf{x}(\cdot))$). This suggests a simple way to produce an efficient $(u(k))_{k=1}^{\infty}$ that drives $(V(\mathbf{x}(k)))_{k=0}^{\infty}$ quickly to zero: (A1) each node $i \in \mathcal{V}$ uses $\Delta V_i(\mathbf{x}(\cdot))$, which it always knows, as feedback to control, on its own, when to initiate an iteration; (A2) the larger $\Delta V_i(\mathbf{x}(\cdot))$ is, the sooner node i initiates an iteration; and (A3) whenever $\Delta V_i(\mathbf{x}(\cdot)) = 0$, node i refrains from initiating an iteration.

Statement A1 describes a fully decentralized feedback control architecture that requires zero communication cost to realize. With this architecture, A2 and A3 may be accomplished if nodes with larger $\Delta V_i(\mathbf{x}(\cdot))$'s would rush to initiate, while nodes with smaller or zero $\Delta V_i(\mathbf{x}(\cdot))$'s would wait longer or forever. Collectively, A1–A3 describe a greedy, decentralized approach to *feedback iteration control*, where potential drops $\Delta V_i(\mathbf{x}(\cdot))$'s in the value of V are used to drive the asynchronous iterations. This approach may be viewed as a greedy approach because the nodes seek to make the value of V drop as much as possible at each iteration, without considering the future. Because the nodes also seek to fully exploit the broadcast nature of every wireless transmission, this approach strives to "make the most" out of each iteration and represents a new way to apply Lyapunov stability theory.

The above approach wants the nodes to try to be greedy. Thus, it is of interest to analyze an ideal scenario where, instead of just trying, the nodes actually succeed at being greedy, ensuring that every iteration $k \in \mathbb{P}$ is initiated by a node $i \in \mathcal{V}$ with the maximum $\Delta V_i(\mathbf{x}(k-1))$, i.e.,

$$u(k) \in \arg\max_{i\in\mathcal{V}} \Delta V_i(\mathbf{x}(k-1)), \quad \forall k \in \mathbb{P}, \quad (14)$$

so that $V(\mathbf{x}(k-1))$ drops maximally to $V(\mathbf{x}(k))$ for every $k \in \mathbb{P}$. Equation (14), together with (2), (3), (8), (9), and (12), defines a networked dynamical system that switches among N different dynamics, depending on where the state is in the state space, i.e., if $\mathbf{x}(k-1)$ is such that $\Delta V_i(\mathbf{x}(k-1)) > \Delta V_j(\mathbf{x}(k-1))\ \forall j \in \mathcal{V} - \{i\}$, then $u(k) = i$. To realize the system over the wireless network, for every link $\{i,j\} \in \mathcal{E}$, let nodes i and j each maintain a local copy of $x_{\{i,j\}}$, denoted as x_{ij} and x_{ji}, respectively, where they are meant to be always equal, so that the order of the subscripts is only used to indicate where they physically reside. With this notation, the system may be expressed in the form of an algorithm, referred to as *Ideal Controlled Hopwise Averaging* (ICHA), as follows:

Algorithm 1 (Ideal Controlled Hopwise Averaging [19])**.**
Initialization:

1) Each node $i \in \mathcal{V}$ transmits $|\mathcal{N}_i|$ and y_i to every node $j \in \mathcal{N}_i$.
2) Each node $i \in \mathcal{V}$ creates variables $x_{ij} \in \mathbb{R}\ \forall j \in \mathcal{N}_i$, $\hat{x}_i \in \mathbb{R}$, and $\Delta V_i \in [0,\infty)$ and initializes them sequentially:
$x_{ij} \leftarrow \frac{\frac{y_i}{|\mathcal{N}_i|}+\frac{y_j}{|\mathcal{N}_j|}}{c_{\{i,j\}}} \quad \forall j \in \mathcal{N}_i, \quad \hat{x}_i \leftarrow \frac{\sum_{j\in\mathcal{N}_i} c_{\{i,j\}}x_{ij}}{\sum_{j\in\mathcal{N}_i} c_{\{i,j\}}}$,
$\Delta V_i \leftarrow \sum_{j\in\mathcal{N}_i} c_{\{i,j\}}(x_{ij} - \hat{x}_i)^2$.

Operation: At each iteration:

3) Let $i \in \arg\max_{j\in\mathcal{V}} \Delta V_j$.
4) Node i updates $x_{ij}\ \forall j \in \mathcal{N}_i$ and ΔV_i sequentially:
$x_{ij} \leftarrow \hat{x}_i \quad \forall j \in \mathcal{N}_i, \quad \Delta V_i \leftarrow 0$.
5) Node i transmits $\hat{x}_i$ to every node $j \in \mathcal{N}_i$.
6) Each node $j \in \mathcal{N}_i$ updates x_{ji}, $\hat{x}_j$, and ΔV_j sequentially:
$x_{ji} \leftarrow \hat{x}_i, \quad \hat{x}_j \leftarrow \frac{\sum_{\ell\in\mathcal{N}_j} c_{\{j,\ell\}}x_{j\ell}}{\sum_{\ell\in\mathcal{N}_j} c_{\{j,\ell\}}}$,
$\Delta V_j \leftarrow \sum_{\ell\in\mathcal{N}_j} c_{\{j,\ell\}}(x_{j\ell} - \hat{x}_j)^2$. ■

Algorithm 1, or ICHA, requires $2N$ real-number transmissions to initialize, in Step 1, and only *one* real-number transmission per iteration, in Step 5, because it can fully exploit the broadcast nature of wireless medium. Also, each iteration is initiated by a node i experiencing the maximum ΔV_i, in Step 3, which is what makes ICHA ideal. Note that "$\Delta V_i \leftarrow 0$" in Step 4 is equivalent to "$\Delta V_i \leftarrow \sum_{j\in\mathcal{N}_i} c_{\{i,j\}}(x_{ij} - \hat{x}_i)^2$" since $x_{ij}\ \forall j \in \mathcal{N}_i$ and $\hat{x}_i$ are equal at that point. The fact that ΔV_i goes from being maximum to zero whenever node i initiates an iteration also suggests that it may be a while before ΔV_i becomes maximum again, causing node i to initiate another iteration.

B. Convergence Rate Analysis

The convergence properties of ICHA on general networks are characterized in the following theorem, in which $\mathbf{1}_n \in \mathbb{R}^n$ and $\hat{\mathbf{x}}(k) \in \mathbb{R}^N$ denote, respectively, the vectors obtained by stacking n 1's and the N $\hat{x}_i(k)$'s:

Theorem 1. *Consider the use of ICHA described in Algorithm 1. Then,*

$$V(\mathbf{x}(k)) \leq (1-\tfrac{1}{\gamma})V(\mathbf{x}(k-1)), \quad \forall k \in \mathbb{P}, \tag{15}$$

$$\|\mathbf{x}(k) - x^*\mathbf{1}_L\| \leq \sqrt{\frac{V(\mathbf{x}(0))\max\limits_{i\in\mathcal{V}}|\mathcal{N}_i|}{2}}(1-\tfrac{1}{\gamma})^{k/2}, \quad \forall k \in \mathbb{N}, \tag{16}$$

$$\|\hat{\mathbf{x}}(k) - x^*\mathbf{1}_N\| \leq \sqrt{\frac{2V(\mathbf{x}(0))\max\limits_{i\in\mathcal{V}}|\mathcal{N}_i|}{\min\limits_{i\in\mathcal{V}}|\mathcal{N}_i|+\max\limits_{i\in\mathcal{V}}|\mathcal{N}_i|}}(1-\tfrac{1}{\gamma})^{k/2}, \quad \forall k \in \mathbb{N}, \tag{17}$$

where $\gamma \in [\frac{N}{2}+1, N^3-2N^2+\frac{N}{2}+1]$ *is given by*

$$\gamma = \frac{N}{2} + \alpha + \frac{(N^2-\beta)(3(N-1)-D)(D+1)}{2N}, \tag{18}$$

and where $\alpha = \max_{\{i,j\}\in\mathcal{E}} \frac{b_i+b_j}{c_{\{i,j\}}} \in [1, \frac{N^2-2N+2}{2}]$, $\beta = \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i\cup\{i\}} b_i b_j \in [N+\frac{L}{2}(1+\frac{1}{N-1})^2, N^2]$, $b_i = \frac{1}{2}\sum_{j\in\mathcal{N}_i} c_{\{i,j\}}$ $\forall i \in \mathcal{V}$, *and* D *is the network diameter.*

Theorem 1 says that ICHA is exponentially convergent on any connected network, ensuring that $V(\mathbf{x}(k))$, $\|\mathbf{x}(k) - x^*\mathbf{1}_L\|$, and $\|\hat{\mathbf{x}}(k) - x^*\mathbf{1}_N\|$ all go to zero exponentially fast, at a rate that is no worse than $1-\frac{1}{\gamma}$ or $(1-\frac{1}{\gamma})^{1/2}$, so that γ in (18) represents a bound on the convergence rate. It also says that the bound γ is between $\Omega(N)$ and $O(N^3)$ and depends only on N, D, and the $|\mathcal{N}_i|$'s, making it easy to compute. The following corollary lists the bound γ for a number of common graphs:

Corollary 1. *The constant* γ *in* (18) *becomes:*

- G1) $\gamma = N^3 - 4N^2 + \frac{9}{2}N + \frac{5}{4}$ *for a path graph with* $N \geq 5$,
- G2) $\gamma = \frac{5}{8}N^3 - \frac{15}{8}N^2 - \frac{1}{8}N + \frac{31}{8}$ *if* N *is odd and* $\gamma = \frac{5}{8}N^3 - \frac{11}{8}N^2 - \frac{5}{2}N + \frac{13}{2}$ *if* N *is even for a cycle graph,*
- G3) $\gamma = \frac{N}{2} + K + \frac{(N-K-1)(3(N-1)-D)(D+1)}{2}$ *for a* K*-regular graph with* $K \geq 2$,
- G4) $\gamma = \frac{3}{2}N - 1$ *for a complete graph.*

Each bound γ in Corollary 1 is obtained by specializing (18) for arbitrary graphs to a specific one. Conceivably, tighter bounds may be obtained by working with each of these graphs individually, exploiting their particular structure. Theorem 2 below shows that this is indeed the case with path and cycle graphs (6 and 15 times tighter, respectively), besides providing additional bounds for regular and strongly regular graphs:

Theorem 2. *Consider the use of ICHA described in Algorithm 1. Then,* (15)–(17) *hold with:*

- S1) $\gamma = \frac{N^3}{6} - \frac{13}{6}N + 3$ *for a path graph with* $N \geq 4$,
- S2) $\gamma = \frac{N^3}{24} + \frac{7}{12}N - 2 + \frac{11}{8N}$ *if* N *is odd and* $\gamma = \frac{N^3}{24} + \frac{5}{6}N - 3 + \frac{4}{N}$ *if* N *is even for a cycle graph,*

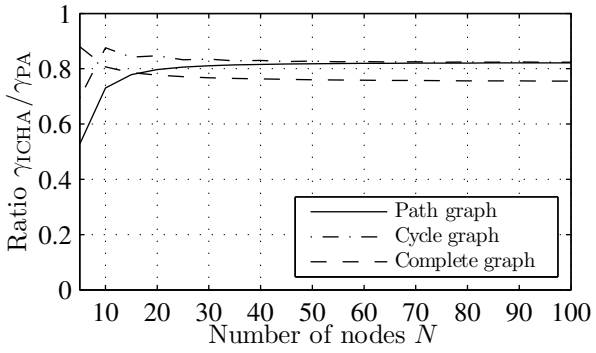

Fig. 1. Comparison between the stochastic convergence rate $1-\frac{1}{\gamma_{\text{PA}}}$ of PA and the deterministic bound $1-\frac{1}{\gamma_{\text{ICHA}}}$ on convergence rate of ICHA for path, cycle, and complete graphs.

- S3) $\gamma = \frac{N}{2} + K + \frac{KD(D+1)(N-K-1)}{2}$ *for a* K*-regular graph with* $K \geq 2$,
- S4) $\gamma = \frac{N}{2} + K + \frac{K(\mu+2)(N-K-1)}{\mu}$ *for a* (N, K, λ, μ)*-strongly regular graph with* $\mu \geq 1$.

Recently, [15] studied, among other things, the convergence rate of Pairwise Averaging (PA) [1]. The results in [15] are different from those above in three notable ways: first, the convergence rate of PA is defined in [15] as the decay rate of the *expected value* of a Lyapunov-like function $d(k)$. Although this stochastic measure captures the average behavior of PA, it offers little guarantee on the decay rate of each realization $(d(k))_{k=0}^{\infty}$. In contrast, the bounds γ on convergence rate of ICHA above are deterministic, providing hard guarantees on the decay rate of $(V(\mathbf{x}(k)))_{k=0}^{\infty}$. Second, even if the first difference is disregarded, the bounds of ICHA are still roughly 20% better than the convergence rate of PA for a few common graphs. To justify this claim, let $1-\frac{1}{\gamma_{\text{PA}}}$ denote the convergence rate of PA. Since PA requires two real-number transmissions per iteration while ICHA requires only one, to enable a fair comparison we introduce a two-iteration bound γ_{ICHA} for ICHA, defined as $\gamma_{\text{ICHA}} = \frac{\gamma^2}{2\gamma-1}$ so that $1-\frac{1}{\gamma_{\text{ICHA}}} = (1-\frac{1}{\gamma})^2$. Figure 1 plots the ratio $\frac{\gamma_{\text{ICHA}}}{\gamma_{\text{PA}}}$ versus N for path, cycle, and complete graphs, where γ_{PA} is computed according to [15], while γ_{ICHA} is computed using γ in S1, S2, and G4. Observe that for $N > 50$, γ_{ICHA} is 18% smaller than γ_{PA} for path and cycle graphs, and 25% so for complete graphs. The latter can also be shown analytically: since $\gamma_{\text{PA}} = N-1$ and $\gamma_{\text{ICHA}} = \frac{(\frac{3}{2}N-1)^2}{2(\frac{3}{2}N-1)-1}$, $\lim_{N\to\infty}\frac{\gamma_{\text{ICHA}}}{\gamma_{\text{PA}}} = \frac{3}{4}$. This justifies the claim. Finally, unlike γ and γ_{ICHA}, γ_{PA} in general cannot be expressed in a form that explicitly reveals its dependence on the graph invariants. Indeed, it generally can only be computed by numerically finding the spectral radius of an invariant subspace of an N^2-by-N^2 matrix, which may be prohibitive for large N.

III. Controlled Hopwise Averaging

A. Algorithm Development

The strong convergence properties of ICHA suggest that its greedy behavior may be worthy of emulating. In this

subsection, we derive a practical algorithm that closely mimics such behavior.

Reconsider the system (2), (3), (8), (9) and suppose this system evolves in a discrete event fashion, according to the following description: associated with the system is global *time*, which is real-valued, nonnegative, and denoted as $t \in [0, \infty)$, where $t = 0$ represents the time instant at which the nodes have observed the y_i's but have yet to execute an iteration. In addition, associated with each node $i \in \mathcal{V}$ is an *event*, which is scheduled to occur at time $\tau_i \in (0, \infty]$ and is marked by node i initiating an iteration, where $\tau_i = \infty$ means the event will not occur. Each event time τ_i is a *variable*, which is initialized at time $t = 0$ to $\tau_i(0)$, is updated only at each iteration $k \in \mathbb{P}$ from $\tau_i(k-1)$ to $\tau_i(k)$, and is no less than t at any time t, so that no event is scheduled to occur in the past. Starting from $t = 0$, time advances to $t = \min_{i\in\mathcal{V}} \tau_i(0)$, at which an event, marked by node $u(1) \in \arg\min_{i\in\mathcal{V}} \tau_i(0)$ initiating iteration 1, occurs, during which $\tau_i(1)$ $\forall i \in \mathcal{V}$ are determined. Time then advances to $t = \min_{i\in\mathcal{V}} \tau_i(1)$, at which a subsequent event, marked by node $u(2) \in \arg\min_{i\in\mathcal{V}} \tau_i(1)$ initiating iteration 2, occurs, during which $\tau_i(2)$ $\forall i \in \mathcal{V}$ are determined. In the same way, time continues to advance toward infinity, while events continue to occur one after another, except if $\tau_i(k) = \infty$ $\forall i \in \mathcal{V}$ for some $k \in \mathbb{N}$, for which the system terminates.

Having described how the system evolves, we now specify how $\tau_i(k)$ $\forall k \in \mathbb{N}$ $\forall i \in \mathcal{V}$ are recursively determined. First, consider the time instant $t = 0$, at which $\tau_i(0)$ $\forall i \in \mathcal{V}$ need to be determined. To behave greedily, nodes with the maximum $\Delta V_i(\mathbf{x}(0))$'s should have the minimum $\tau_i(0)$'s. This may be accomplished by letting

$$\tau_i(0) = \Phi(\Delta V_i(\mathbf{x}(0))), \quad \forall i \in \mathcal{V}, \tag{19}$$

where $\Phi : [0, \infty) \to (0, \infty]$ is a continuous and strictly decreasing function satisfying $\lim_{v\to 0} \Phi(v) = \infty$ and $\Phi(0) = \infty$. Although, mathematically, (19) ensures that $V(\mathbf{x}(0))$ drops maximally to $V(\mathbf{x}(1))$, in reality it is possible that multiple nodes have the same minimum $\tau_i(0)$'s, leading to wireless collisions. To address this issue, we insert a little randomness into (19), rewriting it as

$$\tau_i(0) = \Phi(\Delta V_i(\mathbf{x}(0))) + \varepsilon(\Delta V_i(\mathbf{x}(0))) \cdot \mathrm{rand}(), \quad \forall i \in \mathcal{V}, \tag{20}$$

where $\varepsilon : [0, \infty) \to (0, \infty)$ is a continuous function meant to take on small positive values and each call to $\mathrm{rand}()$ returns a uniformly distributed random number in $(0, 1)$. With (20), with high probability iteration 1 is initiated by a node i with the maximum, or a near-maximum, $\Delta V_i(\mathbf{x}(0))$.

Next, pick any $k \in \mathbb{P}$ and consider the time instant $t = \min_{i\in\mathcal{V}} \tau_i(k-1)$, at which node $u(k) \in \arg\min_{i\in\mathcal{V}} \tau_i(k-1)$ initiates iteration k, during which $\tau_i(k)$ $\forall i \in \mathcal{V}$ need to be determined. Again, to be greedy, nodes with the maximum $\Delta V_i(\mathbf{x}(k))$'s should have the minimum $\tau_i(k)$'s. At first glance, this may be approximately accomplished following ideas from (20), i.e., by letting

$$\tau_i(k) = \Phi(\Delta V_i(\mathbf{x}(k))) + \varepsilon(\Delta V_i(\mathbf{x}(k))) \cdot \mathrm{rand}(), \quad \forall i \in \mathcal{V}. \tag{21}$$

However, with (21), it is possible that $\tau_i(k)$ turns out to be smaller than t, causing an event to be scheduled in the past. Moreover, nodes who are two or more hops away from node $u(k)$ are unaware of the ongoing iteration k and, thus, are unable to perform an update. Fortunately, these issues may be overcome by slightly modifying (21) as follows:

$$\tau_i(k) = \begin{cases} \max\{\Phi(\Delta V_i(\mathbf{x}(k))), t\} + \varepsilon(\Delta V_i(\mathbf{x}(k))) \cdot \mathrm{rand}(), \\ \qquad \text{if } i \in \mathcal{N}_{u(k)} \cup \{u(k)\}, \\ \tau_i(k-1), \text{ otherwise}, \end{cases} \quad \forall i \in \mathcal{V}. \tag{22}$$

Using (20) and (22) and by induction on $k' \in \mathbb{P}$, it can be shown that $\tau_i(k')$ satisfies

$$\max\{\Phi(\Delta V_i(\mathbf{x}(k'))), t'\} \le \tau_i(k') \le \max\{\Phi(\Delta V_i(\mathbf{x}(k'))), t'\} + \varepsilon(\Delta V_i(\mathbf{x}(k'))), \quad \forall k' \in \mathbb{P}, \ \forall i \in \mathcal{V},$$

where $t' = \min_{j\in\mathcal{V}} \tau_j(k'-1)$. Hence, with (22), it is highly probable that iteration $k+1$ is initiated by a node i with the maximum or a near-maximum $\Delta V_i(\mathbf{x}(k))$. It follows that with (20) and (22), the nodes closely mimic the greedy behavior of ICHA. Note that (20) and (22) represent a *feedback iteration controller*, which uses architecture A1 and follows the spirit of A2 (since Φ is strictly decreasing and ε is small) and A3 (since $\Phi(0) = \infty$). Also, Φ and ε represent the *controller parameters*, which may be selected based on practical wireless networking considerations (e.g., all else being equal, $\Phi(v) = \frac{1}{v}$ and $\varepsilon(v) = 0.001$ yield faster convergence time than $\Phi(v) = \frac{10}{v}$ and $\varepsilon(v) = 0.01$ but higher collision probability).

The above description defines a discrete event system, which can be realized via a distributed asynchronous algorithm, referred to as *Controlled Hopwise Averaging* (CHA) and stated as follows:

Algorithm 2 (Controlled Hopwise Averaging [19])**.**
Initialization:

1) Let time $t = 0$.
2) Each node $i \in \mathcal{V}$ transmits $|\mathcal{N}_i|$ and y_i to every node $j \in \mathcal{N}_i$.
3) Each node $i \in \mathcal{V}$ creates variables $x_{ij} \in \mathbb{R}$ $\forall j \in \mathcal{N}_i$, $\hat{x}_i \in \mathbb{R}$, $\Delta V_i \in [0, \infty)$, and $\tau_i \in (0, \infty]$ and initializes them sequentially:
$x_{ij} \leftarrow \frac{\frac{y_i}{|\mathcal{N}_i|} + \frac{y_j}{|\mathcal{N}_j|}}{c_{\{i,j\}}}$ $\forall j \in \mathcal{N}_i$, $\hat{x}_i \leftarrow \frac{\sum_{j\in\mathcal{N}_i} c_{\{i,j\}} x_{ij}}{\sum_{j\in\mathcal{N}_i} c_{\{i,j\}}}$,
$\Delta V_i \leftarrow \sum_{j\in\mathcal{N}_i} c_{\{i,j\}} (x_{ij} - \hat{x}_i)^2$,
$\tau_i \leftarrow \Phi(\Delta V_i) + \varepsilon(\Delta V_i) \cdot \mathrm{rand}()$.

Operation: At each iteration:

4) Let $t = \min_{j\in\mathcal{V}} \tau_j$ and $i \in \arg\min_{j\in\mathcal{V}} \tau_j$.
5) Node i updates x_{ij} $\forall j \in \mathcal{N}_i$, ΔV_i, and τ_i sequentially:
$x_{ij} \leftarrow \hat{x}_i$ $\forall j \in \mathcal{N}_i$, $\Delta V_i \leftarrow 0$, $\tau_i \leftarrow \infty$.
6) Node i transmits $\hat{x}_i$ to every node $j \in \mathcal{N}_i$.

7) Each node $j \in \mathcal{N}_i$ updates x_{ji}, $\hat{x}_j$, ΔV_j, and τ_j sequentially:
$x_{ji} \leftarrow \hat{x}_i, \quad \hat{x}_j \leftarrow \frac{\sum_{\ell \in \mathcal{N}_j} c_{\{j,\ell\}} x_{j\ell}}{\sum_{\ell \in \mathcal{N}_j} c_{\{j,\ell\}}}$,
$\Delta V_j \leftarrow \sum_{\ell \in \mathcal{N}_j} c_{\{j,\ell\}} (x_{j\ell} - \hat{x}_j)^2$,
$\tau_j \leftarrow \max\{\Phi(\Delta V_j), t\} + \varepsilon(\Delta V_j) \cdot \mathrm{rand}()$. ■

Algorithm 2, or CHA, is similar to ICHA in Algorithm 1 except that each node i maintains an additional variable τ_i, in Steps 3, 5, and 7, and that each iteration is initiated, in a discrete event fashion, by a node i having the minimum τ_i, in Step 4. Note that "$\tau_i \leftarrow \infty$" in Step 5 is due to "$\Delta V_i \leftarrow 0$" and to $\Phi(0) = \infty$. Moreover, every step of CHA is implementable in a fully decentralized manner, making it a practical algorithm.

B. Convergence Rate Analysis

To analyze the behavior of CHA, recall that ε is meant to take on small positive values, creating just a little randomness so that the probability of wireless collisions is zero. For the purpose of analysis, we turn this feature off (i.e., set $\varepsilon(v) = 0$ $\forall v \in [0, \infty)$) and let the symbol "$\in$" in Step 4 take care of the randomness (i.e., randomly pick an element i from the set $\arg\min_{j \in \mathcal{V}} \tau_j$ whenever it has multiple elements). We also allow Φ to be arbitrary (but satisfy the conditions stated when it was introduced). With this setup, the following convergence properties of CHA can be established:

Theorem 3. *Theorems 1 and 2, intended for ICHA described in Algorithm 1, hold verbatim for CHA described in Algorithm 2 with any Φ and with ε satisfying $\varepsilon(v) = 0$ $\forall v \in [0, \infty)$. In addition, $\lim_{k \to \infty} t(k) = \infty$ and $V(\mathbf{x}(k)) \leq (\gamma - 1)\Phi^{-1}(t(k))$ $\forall k \in \mathbb{P}$, where $t(0) = 0$ and $t(k)$ is the time instant at which iteration k occurs.*

Theorem 3 characterizes the convergence of CHA in two senses: *iteration* and *time*. Iteration-wise, it says that CHA converges exponentially and shares the same bounds γ on convergence rate as ICHA, regardless of Φ. This result suggests that CHA does closely emulate ICHA. Time-wise, the theorem says that CHA converges asymptotically and perhaps exponentially, depending on Φ. For example, $\Phi(v) = \frac{1}{v}$ does not guarantee exponential convergence in time (since $\Phi^{-1}(v) = \frac{1}{v}$), but $\Phi(v) = W(\frac{1}{v})$, where W is the Lambert W function, does (since $\Phi^{-1}(v) = \frac{1}{v}e^{-v}$). Therefore, the controller parameter Φ may be used to shape the temporal convergence of CHA.

IV. Conclusion

In this paper, we have presented closed-form formulas describing upper bounds on the convergence rates of ICHA and CHA on various graphs, including both arbitrary and structured ones. We have also indicated how the bounds are different from the convergence rate of PA, and for which wirelessly connected graphs are they better. The results obtained provide a theoretical justification for the benefit of feedback iteration control in distributed averaging. In particular, they confirm the effectiveness of CHA's Lyapunov-based, greedy decentralized approach to feedback iteration control.

References

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

[2] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *Proc. IEEE Symposium on Foundations of Computer Science*, Cambridge, MA, 2003, pp. 482–491.

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

[4] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[5] D. S. Scherber and H. C. Papadopoulos, "Distributed computation of averages over ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 776–787, 2005.

[6] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[7] M. Cao, D. A. Spielman, and E. M. Yeh, "Accelerated gossip algorithms for distributed computation," in *Proc. Allerton Conference on Communication, Control, and Computing*, Monticello, IL, 2006, pp. 952–959.

[8] J.-Y. Chen, G. Pandurangan, and D. Xu, "Robust computation of aggregates in wireless sensor networks: Distributed randomized algorithms and analysis," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 9, pp. 987–1000, 2006.

[9] J. Cortés, "Finite-time convergent gradient flows with applications to network consensus," *Automatica*, vol. 42, no. 11, pp. 1993–2000, 2006.

[10] D. B. Kingston and R. W. Beard, "Discrete-time average-consensus under switching network topologies," in *Proc. American Control Conference*, Minneapolis, MN, 2006, pp. 3551–3556.

[11] C. C. Moallemi and B. Van Roy, "Consensus propagation," *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 4753–4766, 2006.

[12] M. Mehyar, D. Spanos, J. Pongsajapan, S. H. Low, and R. M. Murray, "Asynchronous distributed averaging on communication networks," *IEEE/ACM Transactions on Networking*, vol. 15, no. 3, pp. 512–520, 2007.

[13] A. Tahbaz-Salehi and A. Jadbabaie, "Small world phenomenon, rapidly mixing Markov chains, and average consensus algorithms," in *Proc. IEEE Conference on Decision and Control*, New Orleans, LA, 2007, pp. 276–281.

[14] L. Xiao, S. Boyd, and S.-J. Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33–46, 2007.

[15] F. Fagnani and S. Zampieri, "Randomized consensus algorithms over large scale networks," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 634–649, 2008.

[16] M. Zhu and S. Martínez, "Dynamic average consensus on synchronous communication networks," in *Proc. American Control Conference*, Seattle, WA, 2008, pp. 4382–4387.

[17] A. Olshevsky and J. N. Tsitsiklis, "Convergence speed in distributed consensus and averaging," *SIAM Journal on Control and Optimization*, vol. 48, no. 1, pp. 33–55, 2009.

[18] B. N. Oreshkin, M. J. Coates, and M. G. Rabbat, "Optimization and analysis of distributed averaging with short node memory," *IEEE Transactions on Signal Processing*, vol. 58, no. 5, pp. 2850–2865, 2010.

[19] C. Y. Tang and J. Lu, "Controlled hopwise averaging: Bandwidth/energy-efficient asynchronous distributed averaging for wireless networks," in *Proc. American Control Conference*, St. Louis, MO, 2009, pp. 1561–1568.

[20] J. Lu, "Distributed computation and optimization over networks," Ph.D. Thesis, University of Oklahoma, Norman, OK, 2011.