2018 European Control Conference (ECC)
June 12-15, 2018. Limassol, Cyprus

When to stop iterating in digraphs of unknown size? An application to finite-time average consensus

Themistoklis Charalambous and Christoforos N. Hadjicostis

Abstract— **In multi-agent systems, existing distributed algorithms for finite-time average consensus allow the agents to calculate the exact average in finite time, but typically require the agents to continue the iterative process indefinitely. The problem is that it is impossible for one agent to be certain that all other agents have also computed the average (at least not without *a priori* bounds on the network size or diameter). In this paper, we enhance an existing finite-time distributed algorithm with a distributed termination mechanism that allows the nodes to agree on terminating their iterations when they have all computed the exact average. This is accomplished by exploiting the fact that the distributed algorithm allows each node to compute, in a finite number of steps, an upper bound of its eccentricity in the network. The proposed distributed termination mechanism also facilitates the computation, in a finite number of steps, of an upper bound on the diameter of the network, which can be used by several algorithms that require such global information. Illustrative examples demonstrate the validity and performance of the proposed algorithm.**

Index Terms— **Distributed coordination; process termination; finite-time average consensus.**

I. Introduction

The emergence of a wide variety of applications that rely on multi-agent coordination has resulted in tremendous interest in pertinent research topics during the last two decades (see, e.g., [1]–[3] and references and applications therein). A multi-agent system consists of a set of agents (nodes) that can share information via connection links (edges), forming a (generally) directed communication topology (a directed graph or digraph). Distributed coordination algorithms have become very popular due to their simplicity and distributed nature, i.e., nodes are able to coordinate and perform complex tasks using relatively simple rules and local communication.

In distributed coordination, nodes use algorithms in order to compute a desired value. In particular, in distributed average consensus which is the focus of this paper, each node has some initial value and the goal is to compute the average among all participating nodes (see [4] for a recent survey on the topic). However, most existing algorithms cannot be applied to real-world coordination and control applications due to the fact that they can only guarantee asymptotic convergence, whereas algorithms completing in finite-time are, in general, more desirable. Besides finite-time convergence, it is reported that closed-loop systems under finite-time control usually demonstrate better disturbance rejection properties [5]. The ability to terminate in finite time can also prove important in applications where, for example, the averaging operation is a first step towards more involved control or coordination tasks.

Since nodes are interconnected and affect each other within the multi-agent system, if a node computes (or estimates within an acceptable precision) its desired value, it can stop updating and communicating with other nodes. However, such an action might cause the multi-agent system to become disconnected and prevent other nodes in the system (which may not yet be in position to compute their desired value) from completing their task. As a result, the distributed coordination algorithm will not work for most of the nodes in the multi-agent system. Hence, a mechanism is required to determine when an agreement between nodes in the system has been reached so that nodes can terminate the distributed process because all of them have provably computed their desired value.

In a typical consensus problem, each node possesses an initial value and the nodes need to follow a strategy to distributively compute the same function of these initial values. When this function is the average of the initial values we say that the nodes reach average consensus. Asymptotic average consensus, in which the nodes (interconnected via communication links that form an arbitrary strongly connected digraph) asymptotically converge to the average of their initial values, has been extensively studied in the literature (see, for example, [6]–[13]). Finite-time average consensus has recently attracted more attention due to its properties and applicability to practical settings. Typically, finite-time average consensus is handled by pre-determining the number of steps that all nodes in the network will perform, so that by the end of these iterations the nodes have values sufficiently close to the average [14]–[16]. This approach, however, requires some prior knowledge about the network and the convergence rate of the iteration process.

Recently, approaches that do not require such prior network knowledge and still allow the nodes to reach either *exact* or *approximate* convergence to the average have been proposed. For finite-time approximate average consensus there have been two main approaches: (i) synchronous, in which nodes assume knowledge of the diameter of the

The work of C. N. Hadjicostis has been supported in part by the European Commission (EC) through a grant for KIOS Center of Excellence in Research and Innovation.

T. Charalambous is with the Department of Electrical Engineering and Automation, School of Electrical Engineering, Aalto University, Espoo, Finland. E-mail: themistoklis.charalambous@aalto.fi.

C. N. Hadjicostis is with the Department of Electrical and Computer Engineering at the University of Cyprus, Nicosia, Cyprus, and with the Department of Electrical and Computer Engineering at the University of Illinois, Urbana, Illinois, USA. E-mail: chadjic@ucy.ac.cy.

network and terminate the process all together at a multiple of the diameter [17], [18], or, (ii) event-triggered, in which nodes terminate their process provided their value is close enough to the desired one [19], [20]. The main disadvantage of the synchronous approach is that it requires global information, and the main disadvantage of the event-triggered one is that the magnitude of the error is proportional to the diameter. For finite-time exact average consensus, nodes do not require any global parameter and compute the exact value [21]–[25]. However, in the algorithms proposed for exact finite-time consensus when no centralized action is taken or when no prior knowledge about the network is required, nodes never stop the iterations because it is impossible for them to know whether other nodes have computed the average.

In this work, we propose a distributed method with which nodes agree on terminating their iterations when they have all computed the exact average. The proposed method is based on the fact that the finite-time consensus algorithm proposed in [22]–[24] allows nodes to compute an upper bound of their eccentricity and use this information for deciding when to terminate the process. A byproduct of our proposed algorithm is that it allows nodes to distributively compute an upper bound of the diameter of the network in a finite number of steps, which can be useful in several algorithms that require such global information. Illustrative examples demonstrate the validity and performance of our algorithm.

The remainder of the paper is organized as follows. In Section II, we review necessary notation and background. Section III presents our main results accompanied with illustrative examples to demonstrate the proposed algorithm. Finally, Section IV presents concluding remarks.

II. Notation and Preliminaries

A. Notation

The set of real (integer) numbers is denoted by $\mathbb{R}$ ($\mathbb{Z}$) and the set of nonnegative numbers (integers) is denoted by $\mathbb{R}_+$ ($\mathbb{Z}_+$). $\mathbb{R}_+^n$ denotes the nonnegative orthant of the n-dimensional real space $\mathbb{R}^n$. Vectors are denoted by small letters whereas matrices are denoted by capital letters. The transpose of a matrix A is denoted by A^{T}. For $A \in \mathbb{R}^{n\times n}$, A_{ij} denotes the entry in row i and column j. The i^{th} component of a vector x is denoted by x_i, and the notation $x \geq y$ implies that $x_i \geq y_i$ for all components i. By $\mathbb{1}$ we denote the all-ones vector and by I we denote the identity matrix (of appropriate dimensions). We also denote the n dimensional unit vector by $\mathbf{e}_j^{\mathsf{T}} = [0, \ldots, 0, 1_{j^{th}}, 0, \ldots, \ldots, 0] \in \mathbb{R}^{1\times n}$, where the single "1" entry is at the j^{th} position.

We use $|A|$ to denote the element-wise absolute value of matrix A (i.e., $|A| \triangleq [|A_{ij}|]$), $A \leq B$ ($A < B$) to denote the (strict) element-wise inequality between matrices A and B. A matrix whose elements are nonnegative, called a nonnegative matrix, is denoted by $A \geq 0$, and a matrix whose elements are positive, called positive matrix, is denoted by $A > 0$. We use $\mathrm{diag}(x_i)$ to denote the matrix with elements x_1, x_2 , ... on the leading diagonal and zeros elsewhere.

In multi-component systems with fixed communication links (edges), the exchange of information between components (nodes) can be conveniently captured by a directed graph (digraph) $\mathcal{G}(\mathcal{N}, \mathcal{E})$ of order n ($n \geq 2$), where $\mathcal{N} = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes and $\mathcal{E} \subseteq \mathcal{N}\times\mathcal{N}$ is the set of edges. A directed edge from node v_i to node v_j is denoted by $\varepsilon_{ji} = (v_j, v_i) \in \mathcal{E}$ and represents a communication link that allows node v_j to receive information from node v_i. A graph is said to be undirected if and only if $\varepsilon_{ji} \in \mathcal{E}$ implies $\varepsilon_{ij} \in \mathcal{E}$. In this paper, links are not required to be bidirectional, i.e., we deal with digraphs; for this reason, we use the terms "graph" and "digraph" interchangeably. A digraph is called *strongly* connected if there exists a path from each vertex v_i in the graph to each vertex v_j ($v_j \neq v_i$). In other words, for any $v_j, v_i \in \mathcal{N}$, $v_j \neq v_i$, one can find a sequence of nodes $v_i = v_{l_1}$, v_{l_2}, v_{l_3}, ..., $v_{l_t} = v_j$ such that link $(v_{l_{m+1}}, v_{l_m}) \in \mathcal{E}$ for all $m = 1, 2, \ldots, t-1$. The eccentricity $\epsilon(v_j)$ of a node v_j is the greatest distance between v_j and any other vertex, i.e., it shows how far node v_j is from the node most distant from it in the graph. The diameter d of a graph is the maximum eccentricity of any node in the graph, i.e., $d = \max_{v_j\in\mathcal{N}} \epsilon(v_j)$.

All nodes that can transmit information to node v_j directly are said to be in-neighbors of node v_j and belong to the set $\mathcal{N}_j^- = \{v_i \in \mathcal{N} \mid \varepsilon_{ji} \in \mathcal{E}\}$. The cardinality of $\mathcal{N}_j^-$, is called the *in-degree* of v_j and is denoted by $\mathcal{D}_j^- = \left|\mathcal{N}_j^-\right|$. The nodes that receive information from node v_j belong to the set of out-neighbors of node v_j, denoted by $\mathcal{N}_j^+ = \{v_l \in \mathcal{N} \mid \varepsilon_{lj} \in \mathcal{E}\}$. The cardinality of $\mathcal{N}_j^+$, is called the *out-degree* of v_j and is denoted by $\mathcal{D}_j^+ = \left|\mathcal{N}_j^+\right|$.

In the algorithms we consider, we associate a positive weight p_{ji} with each edge $\varepsilon_{ji} \in \mathcal{E} \cup \{(v_j, v_j) \mid v_j \in \mathcal{N}\}$. The nonnegative matrix $P = [p_{ji}] \in \mathbb{R}_+^{n\times n}$ (with p_{ji} as the entry at its jth row, ith column position) is a weighted adjacency matrix (also referred to as weight matrix) that has zero entries at locations that do not correspond to directed edges (or self-edges) in the graph. In other words, apart from the main diagonal, the zero-nonzero structure of the adjacency matrix P matches exactly the given set of links in the graph.

We use $x_j[k] \in \mathbb{R}$ to denote the information state of node v_j at time t_k. In the synchronous setting we consider, each node v_j updates and sends its information to its neighbors at discrete times $t_0, t_1, t_2, \ldots$. We index nodes' information states and any other information at time t_k by k. Hence, we use $x_j[k]$ to denote the state of node v_j at time t_k.

B. Distributed linear iterations

Each node updates its information state $x_j[k]$ by combining the available information received by its neighbors $x_i[k]$ ($v_i \in \mathcal{N}_j^-$) using the positive weights $p_{ji}[k]$, that capture the weight of the information inflow from agent v_i to agent v_j at time k.In this work, we assume that each node v_j can choose its self-weight p_{jj} and the weights p_{lj} on its outgoing links (v_l, v_j), $v_l \in \mathcal{N}_j^+$ (e.g., by sending $p_{lj}v_j$ to its out-neighbour $v_l \in \mathcal{N}_j^+$. Hence, in its general form, each node updates its

information state according to the following relation:

$$x_j[k+1] = p_{jj}x_j[k] + \sum_{v_i \in \mathcal{N}_j^-} p_{ji}x_i[k], \qquad (1)$$

for $k \geq 0$, where $x_j[0] \in \mathbb{R}$ is the initial state of node v_j. Let $x[k] = (x_1[k] \;\; x_2[k] \;\; \dots \;\; x_n[k])^{\mathrm{T}}$ and $P = [p_{ji}] \in \mathbb{R}_+^{n\times n}$. Then (1) can be written in matrix form as

$$x[k+1] = Px[k], \qquad (2)$$

where $x[0] = (x_1[0] \;\; x_2[0] \;\; \dots \;\; x_n[0])^{\mathrm{T}} \triangleq x_0$. We say that the nodes asymptotically reach average consensus if

$$\lim_{k\to\infty} x_j[k] = \frac{\sum_{v_i\in\mathcal{N}} x_i[0]}{n}\ , \quad \forall v_j \in \mathcal{N}\ .$$

The necessary and sufficient conditions for (2) to reach average consensus are the following: (a) P has a simple eigenvalue at one with left eigenvector $\mathbb{1}^{\mathrm{T}}$ and right eigenvector $\mathbb{1}$, and (b) all other eigenvalues of P have magnitude less than 1. If $P \geq 0$ (as in our case), the necessary and sufficient condition is that P is a primitive doubly stochastic matrix[1]. However, in a digraph, it is not possible to set up a doubly stochastic weight matrix without exchanging information among the nodes in the network (see discussion in [27]).

C. Ratio consensus

In [28], an algorithm is suggested that solves the average consensus problem in a directed graph in which each node v_j distributively sets the weights on its self-link and outgoing-links to be $p_{lj} = \frac{1}{1+\mathcal{D}_j^+}\ \forall (v_l, v_j) \in \mathcal{E}$, so that the resulting weight matrix $P = [p_{lj}]$ is column stochastic, but not necessarily row stochastic. Asymptotic average consensus is reached by using this weight matrix to run two iterations with appropriately chosen initial conditions. The algorithm is stated below for a specific choice of weights on each link that assumes that each node knows its out-degree; note, however, that the algorithm works for any set of weights that adhere to the graph structure and form a primitive column stochastic weight matrix.

Proposition 1 ([28]): Consider a strongly connected digraph $\mathcal{G}(\mathcal{N}, \mathcal{E})$. Let $y_j[k]$ and $x_j[k]$ (for all $v_j \in \mathcal{N}$ and $k = 0, 1, 2, \dots$) be the result of the iterations

$$y_j[k+1] = p_{jj}y_j[k] + \sum_{v_i \in \mathcal{N}_j^-} p_{ji}y_i[k]\ , \qquad (3a)$$

$$x_j[k+1] = p_{jj}x_j[k] + \sum_{v_i \in \mathcal{N}_j^-} p_{ji}x_i[k]\ , \qquad (3b)$$

where $p_{lj} = \frac{1}{1+\mathcal{D}_j^+}$ for $v_l \in \mathcal{N}_j^+ \cup \{v_j\}$ (zeros otherwise), and the initial conditions are $y[0] \in \mathbb{R}^n$ and $x[0] = \mathbb{1}$. Then, the solution to the average consensus problem can be asymptotically obtained as

$$\lim_{k\to\infty} \mu_j[k] = \frac{\sum_{v_j\in\mathcal{N}} y_j[0]}{|\mathcal{N}|}\ , \forall v_j \in \mathcal{N}\ ,$$

where $\mu_j[k] = \dfrac{y_j[k]}{x_j[k]}$. □

Remark 1: Proposition 1 proposes a decentralized algorithm with which the exact average is *asymptotically* reached, even if the directed graph is not balanced. □

D. Minimum-time average consensus in digraphs

In [24] a distributed algorithm is proposed by which any arbitrarily chosen agent in a *directed* graph can compute its asymptotic final consensus value in a finite number of steps. The result hinges on the use of the concept of the minimal polynomial associated with the linear dynamics of each of (3a) and (3b), in conjunction with the final value theorem, initially proposed for undirected graphs in [22], [23]. First, we provide definitions on minimal polynomials that comprise the salient feature of the analysis.

Definition 1: (Minimal polynomial of a matrix) The minimal polynomial associated with a matrix P, denoted by

$$q(t) = t^{D+1} + \sum_{i=0}^{D} \alpha_i^{(j)} t^i, \qquad (4)$$

is the monic polynomial of minimum degree $D+1$ that satisfies $q(P) = 0$. □

Note that the minimal polynomial will have a degree at most n (i.e., $D+1 \leq n$), which means that $q(t)$, if not the same, divides the *characteristic polynomial* $\chi(t)$ of a matrix.

Definition 2: (Minimal polynomial of a matrix pair) The minimal polynomial associated with the matrix pair $[P, e_j^{\mathrm{T}}]$, denoted by

$$q_j(t) = t^{M_j+1} + \sum_{i=0}^{M_j} \alpha_i^{(j)} t^i, \qquad (5)$$

is the monic polynomial of minimum degree M_j+1 that satisfies $e_j^{\mathrm{T}} q_j(P) = 0$. □

The minimal polynomial of a matrix pair $q_j(t)$ is unique and is not necessarily the same as that of a matrix $q(t)$ (i.e., $M_j \leq D$). In fact, $q_j(t)$ divides $q(t)$ (the proof of these statements can be found in [21]). Note that M_j is an upper bound on the eccentricity of node v_j, i.e., $M_j \geq \epsilon(v_j)$ [29, Proposition 2.5.1].

Considering the iteration in (2) with weight matrix P, it is shown that

$$\sum_{i=0}^{M_j+1} \alpha_i^{(j)} x_j[k+i] = 0, \quad \forall k \in \mathbb{Z}_+\ , \qquad (6)$$

where $\alpha_{M_j+1}^{(j)} = 1$.

We denote the z-transform of $x_j[k]$ as $X_j(z) \stackrel{\triangle}{=} \mathcal{Z}(x_j[k])$.

[1] A doubly stochastic matrix A is a square matrix of nonnegative real numbers, whose rows and columns sum to 1. Matrix A is also primitive if for some $k \in \mathbb{N}$, matrix A^k has no entries equal to 0. A sufficient condition for matrix A to be primitive is for the matrix to be a nonnegative, irreducible (matrix A is irreducible if, and only if, its associated graph $\mathcal{G}$ is strongly connected) with at least one positive element on the main diagonal [26].

From (6) and the time-shift property of the $z-$transform,

$$X_j(z) = \frac{\sum_{i=1}^{M_j+1} \alpha_i^{(j)} \sum_{\ell=0}^{i-1} x_j[\ell] z^{i-\ell}}{q_j(z)}. \quad (7)$$

If the network is strongly connected, the minimal polynomial $q_j(z)$ of $[P, e_j^{\mathrm{T}}]$, does not have any unstable poles apart from one at 1; we can then define the following polynomial:

$$p_j(z) \triangleq \frac{q_j(z)}{z-1} \triangleq \sum_{i=0}^{M_j} \beta_i^{(j)} z^i \ . \quad (8)$$

The application of the final value theorem [22], [23] yields:

$$\phi_y(j) = \lim_{k\to\infty} y_j[k] = \lim_{z\to 1}(z-1)Y_j(z) = \frac{y_{M_j}^{\mathrm{T}}\boldsymbol{\beta}_j}{\mathbb{1}^{\mathrm{T}}\boldsymbol{\beta}_j} \ , \quad (9a)$$

$$\phi_x(j) = \lim_{k\to\infty} x_j[k] = \lim_{z\to 1}(z-1)X_j(z) = \frac{x_{M_j}^{\mathrm{T}}\boldsymbol{\beta}_j}{\mathbb{1}^{\mathrm{T}}\boldsymbol{\beta}_j} \ , \quad (9b)$$

where

$$y_{M_j}^{\mathrm{T}} = (y_j[0], y_j[1], \ldots, y_j[M_j]) \ ,$$
$$x_{M_j}^{\mathrm{T}} = (x_j[0], x_j[1], \ldots, x_j[M_j]) \ ,$$

and $\boldsymbol{\beta}_j$ is the vector of coefficients of the polynomial $p_j(z)$, defined in (8), i.e., $\boldsymbol{\beta}_j^{\mathrm{T}} = (\beta_0^{(j)}, \ldots, \beta_{M_j}^{(j)})$.

Theorem 1 ([24]): Consider a strongly connected graph $\mathcal{G}(\mathcal{N}, \mathcal{E})$. Let $y_j[k]$ and $x_j[k]$ (for all $v_j \in \mathcal{N}$ and $k = 0, 1, 2, \ldots$) be the result of the iterations (3a) and (3b), where $P = [p_{ji}] \in \mathbb{R}_+^{n\times n}$ is a set of weights that adhere to the graph structure and form a primitive column stochastic weight matrix. Then, the solution to the average consensus can be distributively obtained in minimum number of steps at each node v_j, by computing

$$\mu_j \triangleq \lim_{k\to\infty} \frac{y_j[k]}{x_j[k]} = \frac{\phi_y(j)}{\phi_x(j)} = \frac{y_{M_j}^{\mathrm{T}}\boldsymbol{\beta}_j}{x_{M_j}^{\mathrm{T}}\boldsymbol{\beta}_j} \ , \quad (10)$$

where $\phi_y(j)$ and $\phi_x(j)$ are given, respectively, by equations (9a) and (9b), and $\boldsymbol{\beta}_j$ is as defined in (8). □

III. Main Results

Theorem 1 proposes a distributed algorithm with which the exact average is computed in $2(M_j+1)$ time steps for each node v_j (cf. eq. (5)). However, since nodes do not know whether other nodes have computed the average yet, the iterations continue indefinitely. In this work, we propose a distributed termination process, so that nodes terminate the iterations as soon as they are certain that all nodes have finished computing their average. Distributed termination detection is a fundamental yet non-trivial problem aiming at determining whether a distributed algorithm has terminated.

Many distributed termination detection algorithms have been proposed in the literature; see discussion in [30] and references therein. In these algorithms most of the underlying assumptions are different: for example, most of them assume bidirectional communication, or, if directional communication is considered, unique IDs of nodes and/or knowledge of the out-neighbors may be required. Additionally, many of these algorithms work in an asynchronous setting without any global clock. Our algorithm differs in the sense that the finite-time consensus algorithm considered assumes synchronicity and information about each node's distance (number of hops) from all other nodes can be inferred from the procedure (as it is described below) which allows for an efficient termination process.

We first explain how a node v_j can compute $2(M_j+1)$ and at the same time obtain the coefficient vector $\boldsymbol{\beta}_j$ in the computation of final values, e.g., eqs. (9a) and (9b). Consider the vectors of $2k+1$ successive discrete-time values at node v_j, given by

$$y_{2k}^{\mathrm{T}} = (y_j[0], y_j[1], \ldots, y_j[2k]),$$
$$x_{2k}^{\mathrm{T}} = (x_j[0], x_j[1], \ldots, x_j[2k]),$$

for the two iterations $y_j[k]$ and $x_j[k]$ at node v_j (as given in (3a) and (3b)), respectively. Let us define their associated Hankel matrices as

$$\Gamma\{y_{2k}^{\mathrm{T}}\} \triangleq \begin{bmatrix} y_j[0] & y_j[1] & \ldots & y_j[k] \\ y_j[1] & y_j[2] & \ldots & y_j[k+1] \\ \vdots & \vdots & \ddots & \vdots \\ y_j[k] & y_j[k+1] & \ldots & y_j[2k] \end{bmatrix},$$

$$\Gamma\{x_{2k}^{\mathrm{T}}\} \triangleq \begin{bmatrix} x_j[0] & x_j[1] & \ldots & x_j[k] \\ x_j[1] & x_j[2] & \ldots & x_j[k+1] \\ \vdots & \vdots & \ddots & \vdots \\ x_j[k] & x_j[k+1] & \ldots & x_j[2k] \end{bmatrix}.$$

We also consider the vector of differences between successive values of $y_j[k]$ and $x_j[k]$:

$$\overline{y}_{2k}^{\mathrm{T}} = (y_j[1]-y_j[0], \ldots, y_j[2k+1]-y_j[2k]),$$
$$\overline{x}_{2k}^{\mathrm{T}} = (x_j[1]-x_j[0], \ldots, x_j[2k+1]-x_j[2k]).$$

It has been shown in [23] that after $2(M_j+1)$ the kernel of the Hankel matrices $\Gamma\{\overline{y}_{2k}^{\mathrm{T}}\}$ and $\Gamma\{\overline{x}_{2k}^{\mathrm{T}}\}$ for arbitrary initial conditions y_0 and x_0 except a set of initial conditions with *Lebesgue measure zero* become defective.

Remark 2: Note that the number of steps required for the Hankel matrix to lose rank are less than $2n$. Thus, an upper bound on the network size immediately yields an upper bound on the storage requirements. However, in many cases, nodes do not need to store as many values: as soon as the square Hankel matrix loses rank[2], the node can stop storing information. Throughout this work we assume that nodes can store at least as many values as necessary to obtain the resulting defective matrix.

A. Distributed termination algorithm

In what follows we describe a distributed termination algorithm, which allows all nodes in the network running iterations (3a) and (3b) to decide when to terminate these processes. The procedure is as follows:

[2]The special structure of Hankel matrices allows for efficient methods to compute their rank. For example, in [31] a modular method is proposed with complexity $\mathcal{O}(r^2)$, where r is the order of the matrix.

- Once iterations (3a) and (3b) are initiated, each node v_j also initiates two counters c_j, $c_j[0]=0$, and r_j, $r_j[0]=0$. Counter c_j increments by one at every time step, i.e., $c_j[k+1]=c_j[k]+1$. The way counter r_j updates is described next.
- Alongside iterations (3a) and (3b) a max-consensus algorithm [32]–[36] is initiated as well, given by

$$\theta_j[k+1]=\max_{v_i\in\mathcal{N}_j\cup\{v_j\}}\big\{\max\{\theta_i[k],c_i[k]\}\big\}, \qquad (11)$$

with $\theta_j[0]=0$. Every time step k for which $\theta_j[k+1]=\theta_j[k]$, counter r_j increments by one, but if, however, at any step k', $\theta_j[k'+1]\neq\theta_j[k']$, then r_j is set to zero, i.e.,

$$r_j[k+1]=\begin{cases}0, & \text{if } \theta_j[k+1]\neq\theta_j[k],\\ r_j[k]+1, & \text{otherwise.}\end{cases} \qquad (12)$$

- Once the square Hankel matrices $\Gamma\{\overline{y}^{\mathrm{T}}_{M_j}\}$ and $\Gamma\{\overline{x}^{\mathrm{T}}_{M_j}\}$ for node v_j lose rank, node v_j saves the count of the counter c_j at that time step, denoted by k^o_j, as c^o_j, i.e., $c^o_j\triangleq c_j[k^o_j]$, and it stops incrementing the counter, i.e., $\forall k'\geq k^o_j, c[k']=c_j[k^o_j]=c^o_j$. Note that $c^o_j=2(M_j+1)$.
- Node v_j can terminate iterations (3a) and (3b) when r_j reaches c^o_j.

The main idea is that, except for a set of initial values of Lebesque measure zero, $2(M_j+1)$ is an upper bound of the maximum distance of any other node to node v_j (which is *a priori* unknown). Once this distance becomes known, via the finite-time consensus algorithm [24], node v_j can use it to decide when everybody is done. This is done by observing whether the value of node v_j in a max-consensus algorithm has changed in $2(M_j+1)$ time steps. If it does, then all other nodes have also computed their final value and the node can terminate the iterations. Otherwise, if a node v_i has not finished computing its average, its value participating in the max-consensus algorithm keeps increasing and so does the maximum value of all the other nodes v_j in the network, whose parameter is changed within $2(M_j+1)$ time steps. The procedure is summarized in Algorithm 1.

Example 1: We consider a digraph consisting of 6 nodes, as shown in Fig. 1.

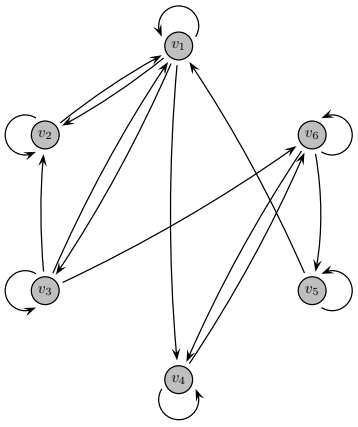

Fig. 1. A digraph consisting of 6 nodes.

Algorithm 1 Distributed termination algorithm for exact minimum-time average consensus in digraphs

Input: A strongly connected digraph $\mathcal{G}(\mathcal{N},\mathcal{E})$ with $n=|\mathcal{N}|$ nodes and $m=|\mathcal{E}|$ edges.

Data: Successive observations for $y_j[k]$ and $x_j[k]$, $\forall v_j\in\mathcal{N}$, $k=0,1,2,\ldots$, using iterations (3a) and (3b), with initial conditions $y[0]=y_0$ and $x[0]=\mathbb{1}$, respectively. Observations for counters $c_j[k], r_j[k]$, $c_j[0]=r_j[0]=0$, and observation of the state $\theta_j[k]$ of the max-consensus algorithm (11), with initial condition $\theta_j[0]=0$.

For $k=0,1,2,\ldots$, each node $v_j\in\mathcal{N}$ does the following:

Step 1: Runs the ratio consensus algorithm (3) and the max-consensus algorithm (11), and (a) stores the vectors of differences $\overline{y}^{\mathrm{T}}_{M_j}$ and $\overline{x}^{\mathrm{T}}_{M_j}$ between successive values of $y_j[k]$ and $x_j[k]$, respectively; (b) stores the values $\theta_j[k]$ from the max-consensus; (c) increments the value of the counter $c_j[k]$ and finds the value of the counter $r_j[k]$ via (12).

Step 2: Increases the dimension k of the square Hankel matrices $\Gamma\{\overline{y}^{\mathrm{T}}_{M_j}\}$ and $\Gamma\{\overline{x}^{\mathrm{T}}_{M_j}\}$ for each of the iterations until k^o_j at which they lose rank. Once this happens, each node stores its first defective matrix and the value $c^o_j=2(M_j+1)$. At this point, nodes also stop incrementing $c_j[k]$.

Step 3: The kernel $\boldsymbol{\beta}_j=\big(\beta_0,\ldots,\beta_{M_j-1},1\big)^{\mathrm{T}}$ of the first defective matrix gives the values ϕ_y and ϕ_x, via (9a) and (9b), respectively, and the average consensus value is computed as $\mu_j=\phi_y(j)/\phi_x(j)=y^{\mathrm{T}}_{M_j}\boldsymbol{\beta}_j/x^{\mathrm{T}}_{M_j}\boldsymbol{\beta}_j$.

Step 4: Continue iterations (3a) and (3b) until time step k_t, at which $r_j[k_t]=c^o_j$.

Each node v_j chooses its weight and the weight of its outgoing links to be $(1+\mathcal{D}^+_j)^{-1}$ (such that the sum of all weights assigned by each node v_j is equal to 1). When each node updates its information states $y_j[k]$ and $x_j[k]$ using equations (3a) and (3b), respectively, the information states for the whole network is given by $y[k+1]=Py[k]$ and $x[k+1]=Px[k]$ (as in equation (2)), where

$$P=\begin{pmatrix}1/4 & 1/2 & 1/4 & 0 & 1/2 & 0\\ 1/4 & 1/2 & 1/4 & 0 & 0 & 0\\ 1/4 & 0 & 1/4 & 0 & 0 & 0\\ 1/4 & 0 & 0 & 1/2 & 0 & 1/3\\ 0 & 0 & 0 & 0 & 1/2 & 1/3\\ 0 & 0 & 1/4 & 1/2 & 0 & 1/3\end{pmatrix}.$$

The asymptotic convergence time and the termination time are shown in Fig. 2. By running our algorithm, each node can compute the exact average value in a minimum number of steps (column 2 of Table I) and then terminate the process in a finite number of steps (column 3 of Table I).

Note that each node v_j needed $2(M_j+1)-1$ extra steps to terminate its process after the last node(s) finished computing its (their) average needed, as expected, since its (their) maximum does not change for $2(M_j+1)$ steps. Once a node terminates its process before its out-neighbors, it stops sending messages and its out-neighbors use the last values

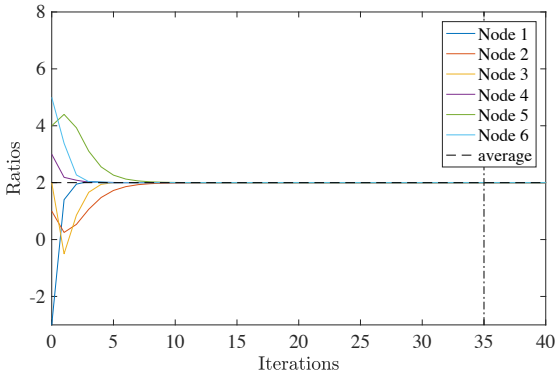

Fig. 2. Average consensus is asymptotically reached for the ratio $y_j[k]/x_j[k]$ of each node v_j for a digraph of size 6.

Node number	# of steps (average)	# of steps (termination)
1	12	18+12-1=29
2	14	18+14-1=31
3	12	18+12-1=29
4	18	18+18-1=35
5	14	18+14-1=31
6	12	18+12-1=29

TABLE I

NUMBER OF STEPS NEEDED BY EACH NODE TO COMPUTE THE AVERAGE (COLUMN 2) AND TO TERMINATE THE ITERATIONS (COLUMN 3).

it sent. In general, the number of steps, T_j, needed for node v_j to terminate a process is

$$T_j = \max_{v_i \in \mathcal{N}} \{2(M_i + 1)\} + 2(M_j + 1) - 1. \quad (13)$$

As a result, the maximum time needed for all nodes to terminate their process, $T_{\max}$, is given by

$$\begin{aligned} T_{\max} &= \max \Big\{ \max_{v_i \in \mathcal{N}} \{2(M_i + 1)\} + 2(M_j + 1) - 1 \Big\} \\ &= 2 \max_{v_i \in \mathcal{N}} \{2(M_i + 1)\} - 1. \end{aligned} \quad (14)$$

Remark 3: Note that once a node has completed the termination algorithm, it can deduce $\max_{v_i \in \mathcal{N}} \{2(M_i+1)\}$ and hence find an upper bound on the diameter of the network. In Example 1, the upper bound is $\max_{v_i \in \mathcal{N}} M_i = 8$.

Example 2: Next, we consider an example of a larger network, for which asymptotic convergence to the average takes a considerable amount of time. More specifically, we consider a Leslie (a discrete, age-structured model of population growth) matrix consisting of 100 nodes. The adjacency matrix $P \in \mathbb{R}_+^{100 \times 100}$ is as follows.

$$P = \begin{pmatrix} 1/2 & 1/3 & 1/3 & 1/3 & \cdots & 1/3 & 1/2 \\ 1/2 & 1/3 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1/3 & 1/3 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1/3 & 1/3 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1/3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1/3 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 1/3 & 1/2 \end{pmatrix}.$$

Due to the structure of the network, the asymptotic convergence time is large. By running our distributed termination algorithm, all nodes terminate their process in 75 steps, while the (asymptotic) ratio-consensus algorithm seems to need more than 160 steps for the error to be less than 0.1 (see Fig. 3).

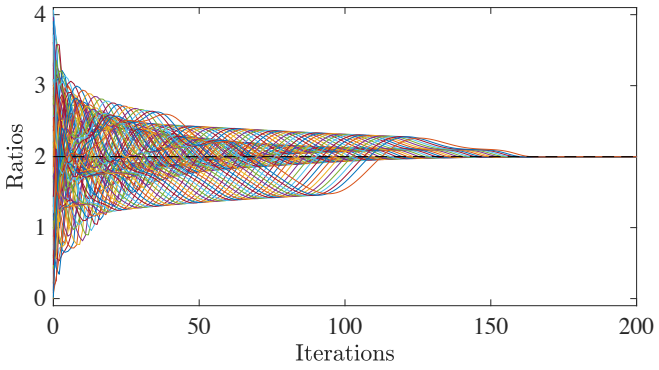

Fig. 3. Average consensus is asymptotically reached for the ratio $y_j[k]/x_j[k]$ of each node v_j for a Leslie matrix of size 100.

More specifically, the minimum and maximum number of steps for computing the average are 28 and 38, respectively. Hence, based on (14) the maximum time needed for all nodes to terminate their process is

$$T_{\max} = 2 \max_{v_i \in \mathcal{N}} \{2(M_i + 1)\} - 1 = 2 \times 38 - 1 = 75.$$

IV. CONCLUSIONS AND FUTURE DIRECTIONS

A. Conclusions

In this paper, we proposed a distributed algorithm with which nodes can determine when to terminate their iterations when they have all computed the exact average. To the best of our knowledge, this is the first finite-time exact average consensus algorithm that computes the average in a finite number of steps and terminates the process without requiring knowledge of any global parameter (such as, the number of nodes or the diameter of the network). A byproduct of our proposed algorithm is that it allows nodes to distributively compute an upper bound of the diameter of the network in a finite number of steps, which can be useful in several algorithms that require such global information. We demonstrated the performance of our proposed algorithm via illustrative examples.

B. Future Directions

From the proposed algorithm it is evident that the nodes in the network with the largest eccentricity can be identified in a distributed fashion. While distributed computation of the network diameter has been considered in the literature and algorithms will low complexity have been proposed (see, for example, [37], [38] and references therein), these algorithms assume knowledge of the size of the network or an upper bound. With our approach we will investigate fully distributed approaches that aim at reducing also the communication exchange (and hence overhead) for computing

the diameter by having only a single node (or a few nodes) calculate their eccentricity and hence the diameter of the network.

The proposed algorithm works for any initial conditions (except for a set of Lebesgue measure zero) assuming the ability to communicate and process real values. In practice, both communication and computation will be done with finite precision; thus, another interesting future research direction is to investigate the limitations imposed by finite precision on the calculation of the size/diameter of networks of increasing size.

REFERENCES

[1] J. Shamma, *Cooperative Control of Distributed Multi-Agent Systems*. New York, NY, USA: Wiley-Interscience, 2008.

[2] F. Bullo, J. Cortes, and S. Martinez, *Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms*. Princeton, NJ, USA: Princeton University Press, 2009.

[3] W. Ren and Y. Cao, *Distributed Coordination of Multi-agent Networks: Emergent Problems, Models, and Issues*. Springer Publishing Company, Incorporated, 2013.

[4] C. N. Hadjicostis, A. D. Domínguez-García, and T. Charalambous, "Distributed averaging and balancing in network systems, with applications to coordination and control," *Foundations and Trends® in Systems and Control*, vol. 5, no. 3–4, 2018.

[5] S. Bhat and D. Bernstein, "Finite-time stability of continuous autonomous systems," *SIAM Journal on Control and Optimization*, vol. 38, no. 3, pp. 751–766, 2000.

[6] M. Franceschelli, A. Giua, and C. Seatzu, "Distributed averaging in sensor networks based on broadcast gossip algorithms," *IEEE Sensors Journal*, vol. 11, no. 3, pp. 808–817, March 2011.

[7] K. Cai and H. Ishii, "Average consensus on general strongly connected digraphs," *Automatica*, vol. 48, no. 11, pp. 2750–2761, Nov. 2012.

[8] A. Dominguez-Garcia and C. N. Hadjicostis, "Distributed strategies for average consensus in directed graphs," in *Proceedings of the* $50th$ *IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, Dec. 2011, pp. 2124–2129.

[9] A. D. Domínguez-García and C. N. Hadjicostis, "Distributed matrix scaling and application to average consensus in directed graphs," *IEEE Transactions on Automatic Control*, vol. 58, no. 3, pp. 667–681, March 2013.

[10] F. Iutzeler, P. Ciblat, and W. Hachem, "Analysis of sum-weight-like algorithms for averaging in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 61, no. 11, pp. 2802–2814, June 2013.

[11] A. Priolo, A. Gasparri, E. Montijano, and C. Sagues, "A distributed algorithm for average consensus on strongly connected weighted digraphs," *Automatica*, vol. 50, no. 3, pp. 946–951, 2014.

[12] C. N. Hadjicostis and T. Charalambous, "Average consensus in the presence of delays in directed graph topologies," *IEEE Transactions on Automatic Control*, vol. 59, no. 3, pp. 763–768, March 2014.

[13] C. N. Hadjicostis, N. H. Vaidya, and A. D. Domínguez-García, "Robust distributed average consensus via exchange of running sums," *IEEE Transactions on Automatic Control*, vol. 61, no. 6, pp. 1492–1507, June 2016.

[14] L. Wang and F. Xiao, "Finite-time consensus problems for networks of dynamic agents," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 950–955, April 2010.

[15] T.-M. D. Tran and A. Y. Kibangou, "Distributed design of finite-time average consensus protocols," in *IFAC Workshop on Distributed Estimation and Control in Networked Systems*, Sep. 2013, pp. 227–233.

[16] S. Safavi and U. A. Khan, "Revisiting finite-time distributed algorithms via successive nulling of eigenvalues," *IEEE Signal Processing Letters*, vol. 22, no. 1, pp. 54–57, Jan. 2015.

[17] S. T. Cady, A. D. Domínguez-García, and C. N. Hadjicostis, "Finite-time approximate consensus and its application to distributed frequency regulation in islanded AC microgrids," in *48th Hawaii International Conference on System Sciences*, Jan. 2015, pp. 2664–2670.

[18] N. E. Manitara and C. N. Hadjicostis, "Distributed stopping for average consensus in digraphs," *IEEE Transactions on Control of Network Systems*, vol. PP, no. 99, pp. 1–1, 2017.

[19] G. S. Seyboth, D. V. Dimarogonas, and K. H. Johansson, "Event-based broadcasting for multi-agent average consensus," *Automatica*, vol. 49, no. 1, pp. 245–252, 2013.

[20] N. E. Manitara and C. N. Hadjicostis, "Distributed stopping for average consensus in undirected graphs via event-triggered strategies," *Automatica*, vol. 70, pp. 121–127, 2016.

[21] S. Sundaram and C. N. Hadjicostis, "Finite-time distributed consensus in graphs with time-invariant topologies," in *Proceedings of the American Control Conference (ACC)*, July 2007, pp. 711–716.

[22] Y. Yuan, G.-B. Stan, L. Shi, and J. Gonçalves, "Decentralised final value theorem for discrete-time LTI systems with application to minimal-time distributed consensus," in *Proceedings of the* 48^{th} *IEEE Conference on Decision and Control (CDC)*, Dec. 2009, pp. 2664–2669.

[23] Y. Yuan, G.-B. Stan, M. Barahona, L. Shi, and J. Gonçalves, "Decentralised minimal-time consensus," *Automatica*, vol. 49, no. 5, pp. 1227–1235, 2013.

[24] T. Charalambous, Y. Yuan, T. Yang, W. Pan, C. N. Hadjicostis, and M. Johansson, "Distributed finite-time average consensus in digraphs in the presence of time delays," *IEEE Transactions on Control of Network Systems*, vol. 2, no. 4, pp. 370–381, Dec. 2015.

[25] T. Charalambous, M. G. Rabbat, M. Johansson, and C. N. Hadjicostis, "Distributed finite-time computation of digraph parameters: Left-eigenvector, out-degree and spectrum," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 2, pp. 137–148, June 2016.

[26] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, UK: Cambridge University Press, 1985.

[27] A. I. Rikos, T. Charalambous, and C. N. Hadjicostis, "Distributed weight balancing over digraphs," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 2, pp. 190–201, June 2014.

[28] A. D. Domínguez-García and C. N. Hadjicostis, "Coordination and control of distributed energy resources for provision of ancillary services," in *Proceedings of the First IEEE International Conference on Smart Grid Communications*, Oct. 2010, pp. 537–542.

[29] Y. Yuan, "Decentralised network prediction and reconstruction algorithms," Ph.D. dissertation, University of Cambridge, 2012.

[30] M. Raynal, *Distributed Termination Detection*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 367–399.

[31] J. Sendra and J. Llovet, "Rank of a hankel matrix over $\mathrm{Z}[x_1, \ldots, x_r]$," *Applicable Algebra in Engineering, Communication and Computing*, vol. 3, no. 4, pp. 245–256, 1992.

[32] A. Tahbaz-Salehi and A. Jadbabaie, "A one-parameter family of distributed consensus algorithms with boundary: From shortest paths to mean hitting times," in *Proceedings of the 45th IEEE Conference on Decision and Control*, Dec. 2006, pp. 4664–4669.

[33] J. Cortés, "Distributed algorithms for reaching consensus on general functions," *Automatica*, vol. 44, pp. 726–737, March 2008.

[34] I. Shames, T. Charalambous, C. Hadjicostis, and M. Johansson, "Distributed network size estimation and average degree estimation and control in networks isomorphic to directed graphs," in *50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Oct. 2012, pp. 1885–1892.

[35] F. Iutzeler, P. Ciblat, and J. Jakubowicz, "Analysis of max-consensus algorithms in wireless channels," *IEEE Transactions on Signal Processing*, vol. 60, no. 11, pp. 6103–6107, Nov. 2012.

[36] G. Shi, W. Xia, and K. H. Johansson, "Convergence of max-min consensus algorithms," *Automatica*, vol. 62, pp. 11–17, 2015.

[37] F. W. Takes and W. A. Kosters, "Determining the diameter of small world networks," in *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, ser. CIKM, 2011, pp. 1191–1196.

[38] S. Holzer and R. Wattenhofer, "Optimal distributed all pairs shortest paths and applications," in *Proceedings of the ACM Symposium on Principles of Distributed Computing*, ser. PODC, 2012, pp. 355–364.