Distributed Minimum-Time Weight Balancing over Digraphs

Themistoklis Charalambous, Christoforos N. Hadjicostis, and Mikael Johansson

***Abstract*—We address the weight-balancing problem for a distributed system whose components (nodes) can exchange information via interconnection links (edges) that form an arbitrary, possibly directed, communication topology (digraph). A weighted digraph is balanced if, for each node, the sum of the weights of the edges outgoing from that node is equal to the sum of the weights of the edges incoming to that node. Weight-balanced digraphs play a key role in a variety of applications, such as coordination of groups of robots, distributed decision making, and distributed averaging which is important for a wide range of applications in signal processing. We propose a distributed algorithm for solving the weight balancing problem in a minimum number of iterations, when the weights are nonnegative real numbers. We also provide examples to corroborate the proposed algorithm.**

I. Introduction

Over the last decade, cooperative distributed control algorithms and protocols have received a surge of attention by several research communities (e.g., control, communication, signal processing, and computer science). Weight balancing is closely related to weights that form a doubly stochastic matrix, which find numerous applications in multicomponent systems (e.g., in sensor networks where one is interested in distributively averaging individual measurements at each component). In particular, the distributed average consensus problem usually concerns a linear iteration, where each node repeatedly updates its value to be a linear combination of its own value and the values of its neighboring nodes, weighted according to the edge weights. Asymptotic average consensus is guaranteed (i.e., the nodes asymptotically reach consensus to the average of their initial values) if the weights used in the linear iteration form a doubly stochastic matrix (e.g., [1]).

Finite-time consensus algorithms are, in general, more desirable than asymptotic ones; besides the fact that they converge in finite time it has been reported that closed-loop systems under finite-time control demonstrate better disturbance rejection properties [2]. Finite-time consensus was proposed in [3]–[5], while [6] proposed a method to compute the asymptotic consensus value in finite-time. Also, [7] proposed a distributed algorithm with which any arbitrarily chosen agent can compute its asymptotic consensus value in minimal time.

Balancing a weighted digraph can be accomplished via a variety of algorithms. Gharesifard *et al.* in [8] develop distributed iterative algorithms that allow the nodes to obtain (in finite time) integer weights that form a balanced graph. These algorithms achieve weight balance as long as the underlying digraph is strongly connected (or is a collection of strongly connected digraphs, which is a necessary and sufficient condition for balancing to be possible [8]). Rikos *et al.* in [9] proposed a finite time algorithm that appears to perform much faster. In this analysis, explicit bounds on the number of iterations required for the algorithm are also provided. More specifically, in the worst-case, the given digraph becomes balanced after $\mathcal{O}(n^7)$ iterations, where n is the number of nodes in the given (strongly connected) digraph.

In this paper, the results of Yuan *et al.* [7] on minimum-time consensus are adopted in a distributed algorithm for solving the weight balancing problem when the weights are nonnegative real numbers. The end result is a distributed algorithm that assigns real-valued weights in a finite number of iterations using only the *minimum* number of successive values of its own state history. To the best of the authors' knowledge, this is the first time real-valued weight balancing is achieved in finite time. The number of iterations required, in the worst-case, for the given digraph to become weight-balanced is bounded above by $\mathcal{O}(2n)$. This bound is much smaller than the one found in [9]; however, in [9] weights are restricted to be positive integers, whereas in this study we allow nonnegative real weights.

The remainder of the paper is organized as follows. In Section II we review the notation and some essential preliminaries. In Section III we present the problem formulation. In Section IV we introduce the distributed algorithm which achieves balance with real weights in a minimum number of iterations. In Section V we present simulation results and comparisons. Finally, Section VI presents concluding remarks.

II. Notation and Preliminaries

Vectors are denoted by small letters whereas matrices are denoted by capital letters; A^{-1} denotes the inverse of matrix A. I denotes the identity matrix (of appropriate dimensions), and $\mathbf{1}$ denotes a column vector (of appropriate dimension) whose elements are all equal to one. A nonnegative matrix is denoted by $A \geq 0$, and a positive matrix, is denoted by $A > 0$. Notation $\sigma(A)$ denotes the spectrum of matrix A, $\lambda(A)$ denotes an element of the spectrum of matrix A, and $\rho(A)$ denotes its spectral radius. Notation $\mathrm{diag}(x_i)$ is used to denote the diagonal matrix with elements in the finite set $\{x_1, x_2, \ldots, x_i, \ldots\}$ on the leading diagonal and zeros elsewhere. We also denote by $e_j^{\mathrm{T}} = [0, \ldots, 0, 1_{j^{th}}, 0, \ldots, 0] \in \mathbb{R}^{1\times n}$ a row vector, where the single "1" entry is at the j^{th} position.

A distributed system whose components can exchange information via (possibly directed) interconnection links, can be captured by a digraph (directed graph). A weighted digraph

Themistoklis Charalambous and Mikael Johansson are with the Department of Automatic Control, School of Electrical Engineering and ACCESS Linnaeus Center, Royal Institute of Technology (KTH), Stockholm, Sweden. Corresponding author's address: Osquldas väg 10, 100-44 Stockholm, Sweden. E-mails: {themisc,mikaelj}@kth.se.

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

of order n ($n \geq 2$), is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, where $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ is the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} - \{(v_j, v_j) \mid v_j \in \mathcal{V}\}$ is the set of edges, and $\mathcal{W} = [w_{ji}] \in \mathbb{R}_+^{n \times n}$ is a weighted $n \times n$ adjacency matrix where w_{ji} are nonnegative values. A directed edge from node v_i to node v_j is denoted by $\varepsilon_{ji} \triangleq (v_j, v_i) \in \mathcal{E}$, implying that node v_j can receive information from node v_i. A directed edge $\varepsilon_{ji} \in \mathcal{E}$ if and only if $w_{ji} > 0$. Note that the definition of $\mathcal{G}$ excludes self-edges (though self-weights need to be added if we want to consider bistochastic digraphs [10]). The graph is undirected if and only if $\varepsilon_{ji} \in \mathcal{E}$ implies $\varepsilon_{ij} \in \mathcal{E}$ and $w_{ji} = w_{ij}$. Note that a digraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ can be thought of as a weighted digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ by defining the weighted adjacency matrix $\mathcal{W}$ with $w_{ji} = 1$ if $\varepsilon_{ji} \in \mathcal{E}$, and $w_{ji} = 0$ otherwise.

A digraph is called *strongly connected* if for each pair of vertices $(v_j, v_i) \in \mathcal{V}$, $v_j \neq v_i$, there exists a directed *path* from v_i to v_j i.e., we can find a sequence of vertices $v_i \equiv v_{l_0}, v_{l_1}, \ldots, v_{l_t} \equiv v_j$ such that $(v_{l_{\tau+1}}, v_{l_\tau}) \in \mathcal{E}$ for $\tau = 0, 1, \ldots, t-1$. 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 $\mathcal{N}_j^- = \{v_i \in V \mid \varepsilon_{ji} \in E\}$. The cardinality of $\mathcal{N}_j^-$, is called the *in-degree* of j and is denoted by $\mathcal{D}_j^-$. The nodes that receive information from node j comprise its out-neighbors and are denoted by $\mathcal{N}_j^+ = \{v_l \in \mathcal{V} \mid \varepsilon_{lj} \in \mathcal{E}\}$. The cardinality of $\mathcal{N}_j^+$ is called the *out-degree* of v_j and is denoted by $\mathcal{D}_j^+$.

Given a weighted digraph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{W})$ of order n, the total *in-weight* of node v_j is denoted by $\mathcal{S}_j^-$ and is defined as $\mathcal{S}_j^- \triangleq \sum_{v_i \in \mathcal{N}_j^-} w_{ji}$, whereas the total *out-weight* of node v_j is denoted by $\mathcal{S}_j^+$ and is defined as $\mathcal{S}_j^+ \triangleq \sum_{v_l \in \mathcal{N}_j^+} w_{lj}$. The weight *imbalance* of node v_j is denoted by $x_j \triangleq \mathcal{S}_j^- - \mathcal{S}_j^+$. The *total-imbalance* of digraph $\mathcal{G}$ is denoted by $\varepsilon = \sum_{j=1}^n |x_j|$. If the *total-imbalance* of digraph $\mathcal{G}$ is equal to zero (i.e., $\varepsilon \triangleq \sum_{j=1}^n |x_j| = 0$), then $\mathcal{G}$ is weight balanced.

III. Problem Formulation

Given a strongly connected digraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, we want to develop a finite-time distributed algorithm that allows the nodes to iteratively adjust the weights on their outgoing edges so that they eventually obtain a weight matrix $\mathcal{W} = [w_{ji}]$ that satisfies the following:

1) $w_{ji} > 0$ for each edge $(v_j, v_i) \in \mathcal{E}$;
2) $w_{ji} = 0$ if $(v_j, v_i) \notin \mathcal{E}$;
3) $\mathcal{S}_j^+ = \mathcal{S}_j^-$ for every $v_j \in \mathcal{V}$.

In this paper we introduce and analyze a distributed algorithm that assigns real-valued weights. Specifically, we make use of an asymptotic weight balancing algorithm and find the final values of the weights on each link in a minimum number of iterations (the minimum number of step is associated with the minimal polynomial of a state [7]). Weight balancing in finite-time was considered in the literature (e.g., [8], [9]) but only for integer-valued weights. When real-valued weights are used, the proposed approach allows us to find a much lower bound (as discussed also in the introduction) on the number of steps required to balance the digraph.

IV. Minimum-time weight balancing

In [10] a distributed iterative algorithm is proposed that asymptotically reaches weight balancing. The algorithm achieves weight-balance as long as the underlying digraph is strongly connected. The rate of convergence of the algorithm is geometric and depends exclusively on the structure of the given digraph and some constant parameters chosen by the nodes, in a way that we precisely describe below. Each node observes (but cannot set) the weights of its incoming links and determines the weights on its outgoing links. It is worth pointing out that each node maintains equal weights on all of its outgoing links, which makes implementation particularly easy in settings where broadcasting is possible.

Given a strongly connected digraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, the distributed algorithm has each node v_j initialize the weights of all of its outgoing links to unity, i.e., $w_{lj}[0] = 1$, $\forall v_l \in \mathcal{N}_j^+$. Then, each node v_j enters an iterative stage where it performs the following steps:

1) It computes its weight imbalance defined by
$$x_j[k] \triangleq \mathcal{S}_j^-[k] - \mathcal{S}_j^+[k]. \tag{1}$$
2) If $x_j[k]$ is positive (resp. negative), all the weights of its outgoing links are increased (resp. decreased) by an equal amount and proportionally to $x_j[k]$.

Note that 2) above implies that w_{lj} for $v_l \in \mathcal{N}_j^+$ will always have the same (nonnegative) value.

Algorithm 1 Distributed balancing with real weights

Input: A strongly connected digraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges (and no self-loops).
Initialization: Each node $v_j \in \mathcal{V}$
1) Sets $w_{lj}[0] = 1$, $\forall v_l \in \mathcal{N}_j^+$.
2) Chooses $\beta_j \in (0, 1)$.
Iteration: For $k = 0, 1, 2, \ldots$, each node $v_j \in \mathcal{V}$ updates the weights of each of its outgoing links w_{lj}, $\forall v_l \in \mathcal{N}_j^+$, as
$$w_{lj}[k+1] = w_{lj}[k] + \beta_j \left(\frac{\mathcal{S}_j^-[k]}{\mathcal{D}_j^+} - w_{lj}[k] \right) \tag{2}$$

The intuition behind the algorithm in [10] is that we compare $\mathcal{S}_j^-[k]$ with $\mathcal{S}_j^+[k] = \mathcal{D}_j^+ w_{lj}[k]$. If $\mathcal{S}_j^+[k] > \mathcal{S}_j^-[k]$ (resp. $\mathcal{S}_j^+[k] < \mathcal{S}_j^-[k]$), then the algorithm reduces (resp. increases) the weights on the outgoing links.

Proposition 1 ([10]). *Given a strongly connected digraph,* Algorithm 1 *reaches a steady state weight matrix W^* that forms a weight-balanced digraph, with geometric convergence rate equal to $R_\infty(P) = -\ln \delta(P)$, where*
$$P_{ji} \triangleq \begin{cases} 1 - \beta_j, & \textit{if } i = j, \\ \beta_j / \mathcal{D}_j^+, & \textit{if } v_i \in \mathcal{N}_j^-, \end{cases} \tag{3}$$
and $\delta(P) \triangleq \max\{|\lambda| : \lambda \in \sigma(P)), \lambda \neq 1\}$.

We define $w = (w_1 \;\; w_2 \;\; \ldots \;\; w_n)^T$, $w_j = w_{lj}$ ($v_l \in \mathcal{N}_j^+$). The evolution of the weights in matrix form can be written as
$$w[k+1] = Pw[k], w[0] = w_0 \;, \tag{4}$$

where $w_0 = \mathbb{1}$. It should be clear from the above update equation that the weights remain nonnegative during the execution of the algorithm. In what follows, we propose another algorithm with which every node v_j can compute its final outgoing weight w_j^* in a minimum number of steps.

Definition 1. *(Minimal polynomial of a matrix pair) The minimal polynomial associated with the matrix pair $[P, e_j^{\mathrm{T}}]$, denoted by $q_j(t) = t^{M_j+1} + \sum_{i=0}^{M_j} \alpha_i^{(j)} t^i$, is the monic polynomial of minimum degree $M_j + 1$ that satisfies $e_j^{\mathrm{T}} q_j(P) = 0$.*

It is easy to show, for M_j+1 iterations of the form of (4), that $\sum_{i=0}^{M_j+1} \alpha_i^{(j)} w_j[k+i] = 0$, $\forall k \in \mathbb{N}$, where $\alpha_{M_j+1}^{(j)} = 1$. Let us now denote the z-transform of $w_j[k]$ as $W_j(z) \triangleq \mathbb{Z}(w_j[k])$. Using the time-shift property of the $z-$transform, we obtain,

$$\underbrace{\sum_{i=0}^{M_j+1} \alpha_i^{(j)} z^i}_{\triangleq q_j(z)} W_j(z) - \underbrace{\sum_{i=1}^{M_j+1} \alpha_i^{(j)} \sum_{\ell=0}^{i-1} w_j[\ell] z^{i-\ell}}_{\triangleq H(z)} = 0 \ .$$

Equivalently,

$$W_j(z) = \frac{\sum_{i=1}^{M_j+1} \alpha_i^{(j)} \sum_{\ell=0}^{i-1} w_j[\ell] z^{i-\ell}}{q_j(z)} = \frac{H(z)}{q_j(z)}. \qquad (5)$$

If the network is strongly connected, the minimal polynomial $q_j(t)$ does not have any unstable poles apart from one at 1 (see, e.g., [11]); we can then define the following polynomial:

$$p_j(z) \triangleq \frac{q_j(z)}{z-1} \triangleq \sum_{i=0}^{M_j} \beta_i^{(j)} z^i \ . \qquad (6)$$

Assume that the minimal-polynomial of $[P, e_j^{\mathrm{T}}]$ is $q_j(z)$; the application of the final value theorem yields:

$$\phi_w(j) = \lim_{k\to\infty} w_j[k] = \lim_{z\to 1} (z-1) W_j(z) = \lim_{z\to 1} (z-1) \frac{H(z)}{q_j(z)}$$
$$= \lim_{z\to 1} (z-1) \frac{H(z)}{(z-1)p_j(z)} = \frac{H(1)}{p_j(1)} = \frac{w_{M_j}^{\mathrm{T}} \boldsymbol{\beta}_j}{\mathbb{1}^{\mathrm{T}} \boldsymbol{\beta}_j} \ , \qquad (7)$$

where $w_{M_j}^{\mathrm{T}} = (w_j[0], w_j[1], \ldots, w_j[M_j])$ and $\boldsymbol{\beta}_j$ is a vector comprising of linear combinations of coefficients of the polynomial $p_j(z)$.

Proposition 2. *Consider a strongly connected digraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. Let $w_j[k]$ (for all $v_j \in \mathcal{V}$ and $k = 0, 1, 2, \ldots$) be the result of the iteration* (2)*, where $P = [p_{ji}] \in \mathbb{R}_+^{n\times n}$ is any set of weights that adhere to the graph structure and form a primitive column stochastic weight matrix. The solution to the iteration can be distributively obtained in finite-time, i.e., $w_j^* \triangleq \lim_{k\to\infty} w_j[k] = \phi_w(j)$, where $\phi_w(j)$ is given by* (7).

Consider the vectors of $2k+1$ successive discrete-time values at node v_j, given by $w_{2k}^{\mathrm{T}} = (w_j[0], w_j[1], \ldots, w_j[2k])$, for the iteration $w_j[k]$ (as given in (2)). Let us define its associated Hankel matrix as

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

and the vector of differences between successive values of $w_j[k]$ as $\overline{w}_{M_j}^{\mathrm{T}} = (w_j[1] - w_j[0], \ldots, w_j[2k+1] - w_j[2k])$. We now introduce an algorithm, herein called *Algorithm 2*, in which the nodes distributively compute the weights that balance the digraph in a finite number of steps.

Algorithm 2 Distributed minimum-time average weight balancing in directed graphs

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

Data: Successive observations for $w_j[k]$, $\forall v_j \in \mathcal{V}$, $k = 0, 1, 2, \ldots$, using iteration (2), with initial conditions $w[0] = \mathbb{1}$.

Step 1: For $k = 0, 1, 2, \ldots$, each node $v_j \in \mathcal{V}$ runs iteration (2) and stores the vectors of differences $\overline{w}_{M_j}^{\mathrm{T}}$ between successive values of $w_j[k]$.

Step 2: Increase the dimension k of the square Hankel matrices $\Gamma\{\overline{w}_{M_j}^{\mathrm{T}}\}$ until they loose rank and store their first defective matrix.

Step 3: The kernel $\boldsymbol{\beta}_j = \big(\beta_0, \ldots, \beta_{M_j-1}, 1\big)^T$ of the first defective matrix gives the coefficients of ϕ_w.

Step 4: The final weight w_j^* is computed by $w_j^* = \dfrac{w_{M_j}^{\mathrm{T}} \boldsymbol{\beta}_j}{\mathbb{1}^{\mathrm{T}} \boldsymbol{\beta}_j}$.

Output: Assign $w_{lj} = w_j^*$ for $l \in \mathcal{N}_+$ (zero otherwise).

V. Numerical Examples

A. Illustrative example

In this illustrative example (included in [12]), we demonstrate the validity of the proposed algorithm.

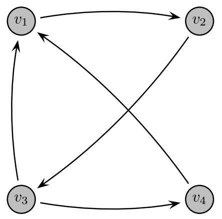

Fig. 1. A simple digraph which is weight-balanceable (included in [12]).

The final consensus vector $\boldsymbol{\phi}_w$, for iteration (2) with initial condition $w[0] = \mathbb{1}^{\mathrm{T}}$, is given by $\boldsymbol{\phi}_w = [1.4286 \ \ 1.4286 \ \ 0.7143 \ \ 0.7143]^{\mathrm{T}}$. Hence, each node can compute the exact outgoing weight $w_j^* = \boldsymbol{\phi}_w(j)$. For example, for node v_1 he have $w_1^* = \boldsymbol{\phi}_w(1) = 1.4286$. Hence, the final weight-balanced digraph $W^\star$ is given by

$$W^\star = \begin{bmatrix} 0 & 0 & 0.7143 & 0.7143 \\ 1.4286 & 0 & 0 & 0 \\ 0 & 1.4286 & 0 & 0 \\ 0 & 0 & 0.7143 & 0 \end{bmatrix}.$$

The final value for node v_1 is computed using Algorithm 2 in 10 steps (i.e., $2(M_1 + 1) = 2 \times 5 = 10$ steps). See [7] for details on how to compute the minimum number of steps.

Fig. 2 shows the total imbalance of Algorithm 1 when $\beta_j = 0.5$ for all $v_j \in \mathcal{V}$ as it evolves during the execution of the algorithm, and the maximum number of iterations required among all nodes to compute the final value via Algorithm 2.

While Algorithm 1 forms a weight-balanced digraph with geometric convergence rate, Algorithm 2 is able to compute the final values in minimum number of iterations.

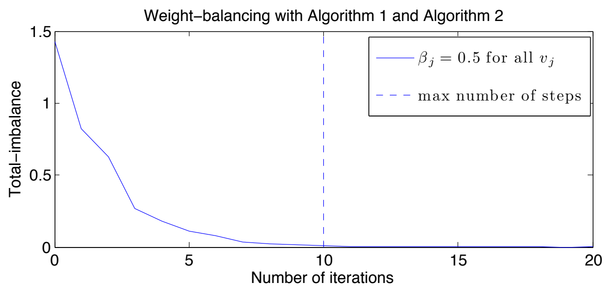

Fig. 2. Total imbalance for weight-balancing algorithm (*Algorithm 1*) for the digraph depicted in Fig. 1 and the maximum number of iterations required among all nodes to compute the final value of the outgoing weights.

B. Comparisons

Here we run the algorithm for larger graphs (of size $n = 10, 20, \ldots, 200$) and we compare the performance of our algorithm against (i) the *imbalance correcting algorithm* in [8], in which every node v_j adds all of its weight imbalance x_j to the outgoing node with the lowest weight w, and (ii) the *distributed integer balancing* in [9], in which the nodes iteratively adjust the positive integer weights of their outgoing edges, such that the digraph becomes weight-balanced after a finite number of iterations.

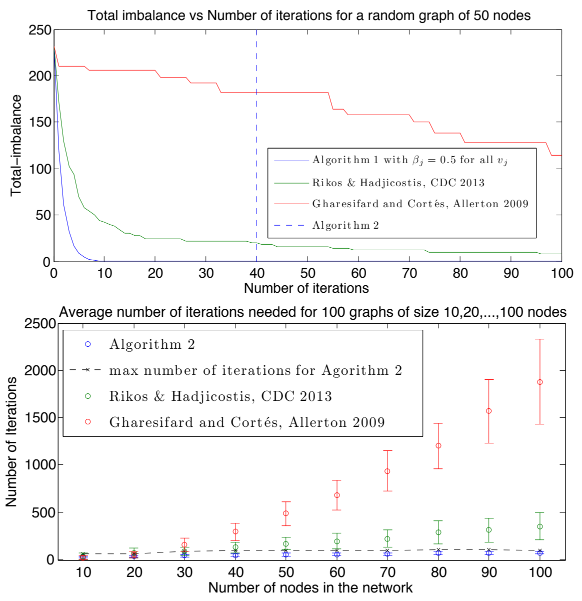

Fig. 3. *Top figure:* Total imbalance plotted against the number of iterations for a random graph of 50 nodes for the distributed weight-balancing (Algorithm 1 with $\beta_j = 0.5$), the algorithms in [8], [9] and the distributed minimum-time weight balancing algorithm (Algorithm 2). *Bottom figure:* Average number of iterations (and standard deviations) against the number of iterations for 100 random graphs of $10, 20, ..., 100$ nodes for the minimum-time weight-balancing (Algorithm 2) and the algorithms in [8], [9]. For large networks, the number of iterations needed scales very fast for [8], [9] with the number of nodes, compared to our method.

The proposed weight-balancing algorithm, Algorithm 2, computes the weights that establish weight balancing in the digraph in minimum number of iterations and outperforms the algorithms suggested in [8], [9], as shown in Fig. 3. It worths pointing out, however, that we consider real-valued weights, whereas [8], [9] consider integer-valued weights.

Remark 1. *Priolo* et al. *[13] propose a distributed real-weight balancing method, which relies on the decentralized estimation of the left eigenvector associated to the zero structural eigenvalue of the Laplacian matrix. For this computation, however, a predefined maximum value of the iterations of the algorithm is necessary. This value is chosen large enough such that the error from the actual value is small enough; this value is dependent on the network size and structure (parameters that affect the predefined maximum value). Our proposed approach can be seamlessly applied on their algorithm and hence lift the need of the predefined maximum value, while also guaranteeing the computations are carried in minimum time for each node.*

VI. Conclusions

In this work, we have considered the weight balancing problem for a distributed system that forms a digraph whose nodes can exchange information iteratively in a distributed fashion. We proposed a distributed algorithm for solving the weight balancing problem when the weights are nonnegative real numbers. To the best of the authors' knowledge, this is the first time weight balancing is reached in finite time for real-valued weights. The proposed algorithm is shown to converge after a minimum number of iterations that it is explicitly bounded.

References

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

[2] 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.

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

[4] Q. Hui, M. Haddad, and S. Bhat, "Finite-time semistability and consensus for nonlinear dynamical networks," *IEEE Transactions on Automatic Control*, vol. 53, no. 8, pp. 1887–1900, 2008.

[5] 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.

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

[7] 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*, Dec. 2009, pp. 2664–2669.

[8] B. Gharesifard and J. Cortés, "Distributed strategies for generating weight-balanced and doubly stochastic digraphs," *European Journal of Control*, vol. 18, no. 6, pp. 539–557, 2012.

[9] A. I. Rikos and C. N. Hadjicostis, "Distributed balancing of a digraph with integer weights," in *Proceedings of the IEEE Conference on Decision and Control*, Florence, Italy, December 2013, pp. 1983–1988.

[10] T. Charalambous and C. N. Hadjicostis, "Distributed formation of balanced and bistochastic weighted adjacency matrices in digraphs," in *Proceedings of the European Control Conference*, July 2013, pp. 1752–1757.

[11] R. Olfati-Saber and R. 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, September 2004.

[12] B. Gharesifard and J. Cortés, "When does a digraph admit a doubly stochastic adjacency matrix?" in *American Control Conference*, July 2010, pp. 2440–2445.

[13] A. Priolo, A. Gasparri, E. Montijano, and C. Sagues, "A decentralized algorithm for balancing a strongly connected weighted digraph," in *Proceedings of the American Control Conference*, June 2013, pp. 6547–6552.