Automatica 50 (2014) 635–640

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Brief paper

Graph diameter, eigenvalues, and minimum-time consensus[☆]

Julien M. Hendrickx[a], Raphaël M. Jungers[a], Alexander Olshevsky[b], Guillaume Vankeerberghen[a,1]

[a] ICTEAM, Department of Mathematical Engineering, Université catholique de Louvain, 4, Avenue G. Lemaitre, 1348 Louvain-La-Neuve, Belgium

[b] Department of Industrial & Enterprise Systems Engineering, University of Illinois at Urbana–Champaign, 310 Transportation Building, 104 S. Mathews, Urbana, IL 61801, USA

ARTICLE INFO

Article history:
Received 27 November 2012
Received in revised form
29 August 2013
Accepted 8 November 2013
Available online 19 December 2013

Keywords:
Consensus
Distributed computation
Algebraic graph theory
Graph eigenvalues
Distance-regular graphs

ABSTRACT

We consider the problem of achieving average consensus in the minimum number of linear iterations on a fixed, undirected graph. We are motivated by the task of deriving lower bounds for consensus protocols and by the so-called "definitive consensus conjecture", which states that for an undirected connected graph $\mathcal{G}$ with diameter D there exist D matrices whose nonzero-pattern complies with the edges in $\mathcal{G}$ and whose product equals the all-ones matrix. Our first result is a counterexample to the definitive consensus conjecture, which is the first improvement of the diameter lower bound for linear consensus protocols. We then provide some algebraic conditions under which this conjecture holds, which we use to establish that all distance-regular graphs satisfy the definitive consensus conjecture.

1. Introduction

Consensus algorithms are a class of iterative update schemes commonly used as building blocks for distributed estimation and control protocols. The progress made in recent years in analyzing and designing consensus algorithms has led to advances in a number of fields, for example, distributed estimation and inference in sensor networks (Carli, Chiuso, Schenato, & Zampieri, 2008; Xiao, Boyd, & Kim, 2007; Xiao, Boyd, & Lall, 2005), distributed optimization (Nedic & Ozdaglar, 2009), and distributed machine learning (Ang, Gopalkrishnan, Hoi, & Ng, 2008). These are among the many subjects that have benefited from the use of consensus algorithms.

One of the available methods to design an average consensus algorithm is to use constant update weights satisfying some conditions for convergence (as can be found in Blondel, Hendrickx, Olshevsky, & Tsitsiklis, 2005 for instance). However, the associated rate of convergence might be a limiting factor, and this has spurred a literature dedicated to optimizing the speed of consensus algorithms. For example, in the discrete-time case in which we are interested, recent work has studied optimizing the spectral gap of the stochastic update matrix (Boyd, Diaconis, & Xiao, 2004; Xiao & Boyd, 2004) or choosing an optimal network structure (Delvenne, Carli, & Zampieri, 2007).

Other recent work has focused on achieving average consensus in finite time. For instance, Ghosh and Lee (2012) shows that if the interaction network is given by a fixed undirected graph (about which the agents initially only know their neighbors and the number of nodes) and if the agents have total recall of the previously sent and received information, then a control scheme that is optimal in time can be found. Again, on fixed undirected networks, it is shown in Sundaram and Hadjicostis (2008) that any node can compute any function of the initial values after using almost any constant weights for a certain finite number of iterations given that the agents have either the memory of their past values or one dedicated register of memory, and given that some parameters are set at design time based on the network and the weights to be used. Moreover, Sundaram and Hadjicostis (2008) presents a decentralized way of computing the required parameters if the nodes have sufficient memory, the computing power to check rank conditions on matrices, and know a bound on the total number of nodes. These results are further improved and extended in Yuan, Stan, Shi, Barahona, and Goncalves (2013) by providing a decentralized procedure using fewer iterations and not requiring the agents to know a bound on the number of agents. However, this last fact implies that the procedure works only for almost all initial conditions. Finally, Kibangou and Commault

[☆] The material in this paper has not been presented at any conference. This paper was recommended for publication in revised form by Associate Editor Antonis Papachristodoulou under the direction of Editor Frank Allgöwer.

E-mail addresses: julien.hendrickx@uclouvain.be (J.M. Hendrickx), raphael.jungers@uclouvain.be (R.M. Jungers), aolshev2@illinois.edu (A. Olshevsky), guillaume.vankeerberghen@uclouvain.be (G. Vankeerberghen).

[1] Tel.: +32 10478039; fax: +32 10472180.

0005-1098/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.automatica.2013.11.034

(2012) also treats the problem of computing the consensus value after a finite number of constant-weight steps when the agents have memory and computing power. It treats this problem in the context of decentralized Laplacian eigenvalues estimation and topology identification.

In this paper, we study finite-time consensus in the setting where nodes memorize only their current state and, as before, update it as a linear function of their state and the states of their neighbors, but the arbitrary time-varying weights can be chosen at the time of design. Then, the problem is to pick update matrices at design time, so that average consensus is achieved in finite time. This setting has been investigated in Ko (2010) with the added restriction to use left-stochastic matrices corresponding to different types of consensus algorithms, e.g., various forms of pairwise exchanges such as gossip matrices. Algorithms and asymptotic results, mainly based on using spanning trees, are proposed there. The same question is treated in Georgopoulos (2011) when the matrices do not have to be left-stochastic nor have to correspond to a known consensus algorithm. It introduces a Gather and Distribute scheme for trees (Georgopoulos, 2011, Section 4.2), which is also similar to Ko (2010, Algorithm 3.3) or the ConvergeCast algorithm, that shows how to reach finite-time consensus on any tree or on any graph based on a spanning tree. Finally, Georgopoulos (2011) conjectures that, on any connected undirected graph, it is possible to design consensus schemes which take only as long as the diameter of the underlying graph to achieve average consensus.

The conjecture just cited is the starting point of our developments. That is, we are concerned with picking update matrices, at design time, so that average consensus is achieved in the fastest possible time. Furthermore, we are concerned with deriving lower bounds on such consensus schemes.

After posing the problem and presenting previous results in Section 2, we answer the conjecture with a counterexample in Section 3. This proves that, on some graphs, consensus may not be achieved by linear consensus protocols as fast as the diameter of the graph. We note that this is the first time the diameter lower bound on linear consensus protocols has been improved for any graph.

In Section 4, we provide some algebraic conditions under which the definitive consensus conjecture holds and show that these conditions are easily met on all distance-regular graphs. In particular, this implies that graphs which possess a lot of symmetry automatically satisfy the conjecture; we will make this statement precise in that section.

Finally, in Section 5, we show how our conditions can be used to find a number (possibly larger than the diameter) of matrices compliant with the graph that lead to average consensus and we introduce the notion of consensus number of a graph. Our results in Section 5 are related to those obtained independently in Kibangou (2011, 2012). In these references the author does not try to minimize the running time of the consensus protocols, but provides a general method for obtaining consensus in finite time. This method can be seen as a particular case of Corollary 9 in the present paper and is further discussed at that point.

2. Problem setting and previous results

2.1. Problem setting

For ease of exposition, we consider only undirected graphs; when some results can be easily extended to digraphs, we explicitly mention it.

Given a connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we define the *distance*, $\delta(i,j)$, separating two nodes $i, j \in \mathcal{V}$ as the number of edges in a shortest-path between i and j. Following this definition, the *diameter*, $D(\mathcal{G})$, of a graph corresponds to the maximum distance between two nodes, i.e., $D(\mathcal{G}) = \max_{i,j\in\mathcal{V}} \delta(i,j)$. Another linked parameter is the *eccentricity*, $\epsilon(i)$, of a node i. This is the maximum distance separating i from another node, i.e., $\epsilon(i) = \max_{j\in\mathcal{V}} \delta(i,j)$. The *radius* $R(\mathcal{G})$ of a graph is the minimum eccentricity over all nodes, i.e., $R(\mathcal{G}) = \min_{i\in\mathcal{V}} \epsilon(i) = \min_{i\in\mathcal{V}} \left(\max_{j\in\mathcal{V}} \delta(i,j)\right)$. Finally, a node is called *central* if its eccentricity equals the radius of the graph. More details on these notions can be found in West (2000).

Our setting is that of a fixed interaction network encoded by an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Agents are allowed to store only their current state and to synchronously update it to a linear combination of their state and the states of their neighbors. Hence, if $x_i^{(t)}$ denotes the state of agent i at time t and $\mathcal{N}_i \subset \mathcal{V}$ denotes the set of agents with which i can communicate, then the updated state of agent i is $x_i^{(t+1)} = a_{ii}^{(t+1)} x_i^{(t)} + \sum_{j\in\mathcal{N}_i} a_{ij}^{(t+1)} x_j^{(t)}$ for some choice of weights $a_{ij}^{(t)}$. More compactly, a synchronous linear update can be written as $\mathbf{x}^{(t+1)} = A^{(t+1)}\mathbf{x}^{(t)}$, where the states of the agents at time t are in the column vector $\mathbf{x}^{(t)}$ and the weights $a_{ij}^{(t+1)}$ are the entries of $A^{(t+1)}$. The matrices $A^{(t)}$ are required to comply with the underlying graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ in the following sense.

Definition 1. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ on N nodes, we define the set of matrices that comply with $\mathcal{G}$ as $\mathcal{M}(\mathcal{G}) = \left\{A = [a_{ij}] \in \mathbb{R}^{N\times N} \mid i \neq j \text{ and } (i,j) \notin \mathcal{E} \Rightarrow a_{ij} = 0\right\}$.

We say that a network of agents has reached *average consensus* at time t^* if the state of each agent is equal to the average of the initial states, i.e., if $\mathbf{x}^{(t^*)} = (\frac{1}{N}\mathbf{1}^T\mathbf{x}^{(0)})\mathbf{1} = \overline{\mathbf{x}^{(0)}}\mathbf{1}$ with $\mathbf{1}$ the column vector of ones. This is the case for any initial vector $\mathbf{x}^{(0)}$ if and only if $A^{(t^*)}\cdots A^{(1)} = \frac{1}{N}\mathbf{1}\mathbf{1}^T$.

In Sections 3 and 4 we answer and analyze the following conjecture.

Conjecture 1 (*Definitive Consensus Conjecture (Georgopoulos, 2011)*). *For any connected graph $\mathcal{G}$ on N vertices, there exist $D(\mathcal{G})$ matrices $A^{(t)}$ that comply with the graph and are such that*

$$A^{(D(\mathcal{G}))}A^{(D(\mathcal{G})-1)}\cdots A^{(1)} = \frac{1}{N}\mathbf{1}\mathbf{1}^T, \tag{1}$$

so that $A^{(D(\mathcal{G}))}A^{(D(\mathcal{G})-1)}\cdots A^{(1)}\mathbf{x}^{(0)} = \overline{\mathbf{x}^{(0)}}\mathbf{1}$.

In other words, the conjecture states that it is always possible to reach average consensus with linear updates in only $D(\mathcal{G})$ steps.

We remark that the definitive consensus conjecture may be viewed as a statement on the feasibility of certain polynomial equalities. Indeed, developing Eq. (1) leads to a system of N^2 polynomial equations in the $D(\mathcal{G})(N+2|\mathcal{E}|)$ weights. For instance, let us consider a simple graph of diameter 2, such as P_3 pictured on Fig. 1. In P_3 there is no link between nodes 1 and 3, hence $a_{13}^{(t)}$ and $a_{31}^{(t)}$ must be zero. Thus, we have the liberty to choose the other 14 nonzero entries in order to fulfil

$$\begin{pmatrix} a_{11}^{(2)} & a_{12}^{(2)} & 0 \\ a_{21}^{(2)} & a_{22}^{(2)} & a_{23}^{(2)} \\ 0 & a_{32}^{(2)} & a_{33}^{(2)} \end{pmatrix}\begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & 0 \\ a_{21}^{(1)} & a_{22}^{(1)} & a_{23}^{(1)} \\ 0 & a_{32}^{(1)} & a_{33}^{(1)} \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

Developing the product on the left-hand side in this last equation yields a system of 9 polynomial equations in the 14 weights. In this system, the monomials in the polynomial equation of an entry ij correspond to all the walks of length $D(\mathcal{G})$ from j to i.