Automatica 93 (2018) 454–461

Contents lists available at ScienceDirect

Automatica

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

Brief paper

Request-based gossiping without deadlocks☆

Ji Liu [a,*], Shaoshuai Mou [b], A. Stephen Morse [c], Brian D.O. Anderson [d,e], Changbin (Brad) Yu [d,e,f]

[a] Coordinated Science Laboratory, University of Illinois at Urbana–Champaign, USA
[b] Purdue University, USA
[c] Yale University, USA
[d] Australian National University, Australia
[e] National ICT Australia Ltd., Australia
[f] Shandong Computer Science Center, Jinan, China

ARTICLE INFO

Article history:
Received 20 March 2013
Received in revised form 22 August 2017
Accepted 1 January 2018

ABSTRACT

By the distributed averaging problem is meant the problem of computing the average value of a set of numbers possessed by the agents in a distributed network using only communication between neighboring agents. Gossiping is a well-known approach to the problem which seeks to iteratively arrive at a solution by allowing each agent to interchange information with at most one neighbor at each iterative step. Crafting a gossiping protocol which accomplishes this is challenging because gossiping is an inherently collaborative process which can lead to deadlocks unless careful precautions are taken to ensure that it does not. Many gossiping protocols are request-based which means simply that a gossip between two agents will occur whenever one of the two agents accepts a request to gossip placed by the other. In this paper, we present three deterministic request-based protocols. We show by example that the first can deadlock. The second is guaranteed to avoid deadlocks by exploiting the idea of local ordering together with the notion of an agent's neighbor queue; the protocol requires the simplest queue updates, which provides an in-depth understanding of how local ordering and queue updates avoid deadlocks. It is shown that a third protocol which uses a slightly more complicated queue update rule can lead to significantly faster convergence; a worst case bound on convergence rate is provided.

1. Introduction

Over the past decade, there has been considerable interest in developing algorithms for distributed computation and decision making among the members of a group of sensors or mobile autonomous agents via local interactions. Probably the most notable among these are those algorithms intended to cause such a group to reach a consensus in a distributed manner (Jadbabaie, Lin, & Morse, 2003; Liu, Morse, Nedić, & Başar, 2014; Olfati-Saber & Murray, 2004).

We are interested in distributed averaging, a particular type of consensus process which has received much attention recently (Xiao & Boyd, 2004). A typical distributed averaging process deals with a network of $n > 1$ *agents* and the constraint that each agent i is able to communicate only with certain other agents called agent i's *neighbors*. Neighbor relationships are conveniently characterized by a simple, undirected, connected graph $\mathbb{A}$ in which vertices correspond to agents and edges indicate neighbor relationships. Thus the neighbors of an agent i have the same labels as the vertices in $\mathbb{A}$ which are adjacent to vertex i. Initially, each agent i has or acquires a real number y_i which might be a measured temperature or something similar. The *distributed averaging problem* is to devise an algorithm which will enable each agent to compute the average $y_{\text{avg}} = \frac{1}{n}\sum_{i=1}^{n} y_i$ using information received only from its neighbors.

There are three important approaches to the distributed averaging problem: linear iterations (Xiao & Boyd, 2004), gossiping (Boyd, Ghosh, Prabhakar, & Shah, 2006), and double linear iterations (Liu & Morse, 2012a) (which are also known as push-sum algorithms (Kempe, Dobra, & Gehrke, 2003), weighted gossip (Bénézit, Blondel, Thiran, Tsitsiklis, & Vetterli, 2010), and ratio consensus (Domínguez-García, Cady, & Hadjicostis, 2012)). Double linear iterations are specifically tailored to the case in which

☆ Proofs of some results in this paper are not included due to space limitations and can be found in Liu et al. (2016). The material in this paper was presented at the 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC), December 12–15, 2011, Orlando, Florida, USA. This paper was recommended for publication in revised form by Associate Editor Claudio De Persis under the direction of Editor Christos G. Cassandras.

* Corresponding author.
E-mail addresses: jiliu@illinois.edu (J. Liu), mous@purdue.edu (S. Mou), as.morse@yale.edu (A.S. Morse), brian.anderson@anu.edu.au (B.D.O. Anderson), brad.yu@anu.edu.au (C. Yu).

https://doi.org/10.1016/j.automatica.2018.03.001
0005-1098/

unidirectional communications exist; they can solve the problem when $\mathbb{A}$ is directed, strongly connected, but under the assumption that each agent is aware of the number of its out-going neighbors. Both linear iterations and gossiping work for the case in which all communications between neighbors are bidirectional; in this case, double linear iterations have the disadvantage that they require updating and transmission of an additional variable for each agent.

Linear iterations are a well studied approach to the problem in which each agent communicates with all of its neighbors on each iteration, and thus are sometimes called *broadcast* algorithms. It is clear that broadcast algorithms typically require a lot of transmissions between neighbors per unit time, which may not be possible to secure in some applications, particularly when communication cost is an important issue on each iteration. For example, fewer transmissions per iteration can increase the time interval between any two successive recharges of a sensor, and improve the security of the network by reducing the opportunities of being hacked or eavesdropped.

Gossiping is an alternative approach to the distributed averaging problem which does not involve broadcasting. An important rule of gossiping is that each agent is allowed to gossip with at most one neighbor at one time. This is the reason why gossiping algorithms do not involve broadcasting. Thus gossiping algorithms have the potential to require less transmissions per iteration than broadcast algorithms. Moreover, the peer-to-peer nature of gossiping simplifies the implementation of algorithms and reduces computation complexity on each agent. As a trade-off, one would not expect gossiping algorithms to converge as fast as broadcast algorithms.

Most existing gossiping algorithms are probabilistic in the sense that the actual sequence of gossip pairs which occurs during a specific gossip process is determined probabilistically (Boyd et al., 2006). Recently, deterministic gossiping has received some attention (Liu, Mou, Morse, Anderson, & Yu, 2011a). Probabilistic gossiping algorithms aim at achieving consensus asymptotically with probability one, whereas deterministic gossiping algorithms are intended to guarantee that under all conditions, a consensus will be achieved asymptotically. Both approaches have merit. The probabilistic approach is easier both in terms of algorithm development and convergence analysis. The deterministic approach forces one to consider worst case scenarios and has the potential of yielding algorithms which may outperform those obtained using the probabilistic approach. For example, the deterministic approach rules out the possibility of deadlocks which may occur in probabilistic gossiping algorithms.

Crafting a deterministic protocol is challenging because gossiping is an inherently collaborative process which can lead to deadlocks unless careful precautions are taken to ensure that it does not. The global ordering (Mehyar, Spanos, Pongsajapan, Low, & Murray, 2007), centralized scheduling (Liu et al., 2011a), and broadcasting (Olshevsky, 2010) are the existing ways to avoid deadlocks. Both global ordering and centralized scheduling require a degree of network-wide coordination and broadcasting requires each agent to obtain the values of *all* of its neighbors' "gossip variables" at each clock time, which may not be possible to secure in some applications.

The contribution of this paper is to present deterministic gossiping protocols which do not utilize global ordering, centralized scheduling, or broadcasting and are guaranteed to solve the distributed averaging problem. Three gossiping protocols are considered in the paper. We show by example that the first can deadlock. After minor modifications, a second protocol is obtained. The second protocol is guaranteed to avoid deadlocks, which requires the simplest queue updates and thus provides an in-depth understanding of how local ordering and queue updates avoid deadlocks. It is shown both by analysis and computer studies that a third protocol which uses a slightly more complicated queue update rule can lead to significantly faster convergence.

The material in this paper was partially presented in Liu and Morse (2012b) and Liu, Mou, Morse, Anderson, and Yu (2011b), but this paper presents a more comprehensive treatment of the work. Specifically, the paper provides proofs for Theorems 2, 4, Proposition 3, Lemmas 1, 2, and establishes an additional result Proposition 1, which were not included in Liu and Morse (2012b) and Liu et al. (2011b). Note that Protocol III in the paper was briefly outlined in Liu et al. (2011a), but without a proof of correctness.

2. Gossiping

Consider a group of $n > 1$ agents labeled 1 to n.[1] Each agent i has control over a real-valued scalar quantity x_i called agent i's *gossip variable* whose value $x_i(t)$ at time t represents agent i's estimate of y_{avg} at that time. A *gossip* between agents i and j, written (i, j), occurs at time t if the values of both agents' variables at time $t + 1$ equal the average of their values at time t. In other words, $x_i(t + 1) = x_j(t + 1) = \frac{1}{2}(x_i(t) + x_j(t))$. If agent i does not gossip at time t, its gossip variable does not change; thus in this case $x_i(t + 1) = x_i(t)$. Generally not every pair of agents is allowed to gossip. The edges of a simple, undirected, connected graph $\mathbb{A}$ specify which pairs of agents are allowed to gossip. In other words, a gossip between agents i and j is *allowable* if (i, j) is an edge in $\mathbb{A}$. We sometimes call $\mathbb{A}$ an *allowable gossip graph*.

An important rule of gossiping is that in a gossiping process, each agent is allowed to gossip with at most one of its neighbors at one time. This rule does not preclude the possibility of two or more pairs of agents gossiping at the same time, provided that the pairs have no agent in common. To be more precise, two gossip pairs (i, j) and (k, m) are *noninteracting* if neither i nor j equals either k or m. When multiple noninteracting pairs of allowable gossips occur simultaneously, the simultaneous occurrence of all such gossips is called a *multi-gossip*.

Gossiping processes can be modeled by a discrete-time linear system of the form

$$x(t + 1) = M(t)x(t), \qquad t = 0, 1, 2, \ldots \tag{1}$$

where $x \in \mathbb{R}^n$ is a state vector of gossiping variables and $M(t)$ is a matrix characterizing how x changes as the result of the gossips which take place at time t. If a single pair of agents i and j gossip at time $t \geq 0$, then $M(t) = P_{ij}$ where P_{ij} is the $n \times n$ matrix for which $p_{ii} = p_{ij} = p_{ji} = p_{jj} = \frac{1}{2}$, $p_{kk} = 1$, $k \notin \{i, j\}$, and all remaining entries equal 0. We call such P_{ij} a *single-gossip primitive gossip matrix*. For convenience, we include in the set of primitive gossip matrices, the $n \times n$ identity matrix I; the identity matrix can be thought of as the update matrix to model the case in which no gossips occur at time t. If a multi-gossip occurs at time t, then as a consequence of non-interaction, $M(t)$ is simply the product of the single-gossip primitive gossip matrices corresponding to the individual gossips comprising the multi-gossip; moreover, the primitive gossip matrices in the product commute with each other and thus any given permutation of the single-gossip primitive matrices in the product determines the same matrix P. We call P the *primitive gossip matrix* determined by the multi-gossip under consideration.

We will see that for any gossiping process determined by the protocols presented in this paper, the update matrix $M(t)$ in (1) also depends on the state $x(t)$ and thus

$$x(t + 1) = M(x(t), t)x(t), \qquad t = 0, 1, 2, \ldots$$

[1] The purpose of labeling of the agents is only for convenience. We do not require a global labeling of the agents in the network. We only assume that each agent can identify and differentiate between its neighbors.