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

Themistoklis Charalambous[⋆] and Christoforos N. Hadjicostis[♯]

[⋆]Department of Electrical Engineering and Automation, Aalto University
[♯]Department of Electrical and Computer Engineering, University of Cyprus

Aalto-yliopisto

European Control Conference (ECC '18)
Limassol, Cyprus, June 2018

Introduction

1. **Distributed coordination:** the subject of extensive research for a long time
   - parallel and distributed computation, distributed optimization in sensor networks, formation control of robotic networks, dynamics of opinion forming, . . .

2. **Finite-time algorithms** are generally more desirable
   - they converge in finite-time
   - closed-loop systems under finite-time control usually demonstrate better disturbance rejection properties
   - important in applications where, for example, the averaging operation is a first step towards more involved control or coordination tasks

Motivation

Why is a termination mechanism needed?

- If a node computes (or estimates within an acceptable precision) its desired value, it can stop updating and communicating with other nodes
- Such an action might cause the multi-agent system to become disconnected and prevent other nodes in the system from completing their task

Motivation

Why is a termination mechanism needed?

- If a node computes (or estimates within an acceptable precision) its desired value, it can stop updating and communicating with other nodes
- Such an action might cause the multi-agent system to become disconnected and prevent other nodes in the system from completing their task

Motivation

Why is a termination mechanism needed?

- If a node computes (or estimates within an acceptable precision) its desired value, it can stop updating and communicating with other nodes
- Such an action might cause the multi-agent system to become disconnected and prevent other nodes in the system from completing their task

- Distributed coordination algorithm will work only for a subset of the nodes in the multi-agent system

Motivation

Why is a termination mechanism needed?

- If a node computes (or estimates within an acceptable precision) its desired value, it can stop updating and communicating with other nodes
- Such an action might cause the multi-agent system to become disconnected and prevent other nodes in the system from completing their task
- Distributed coordination algorithm will work only for a subset of the nodes in the multi-agent system

Motivation

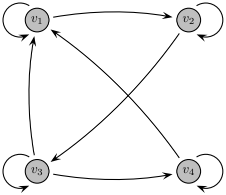
Why is a termination mechanism needed?

- If a node computes (or estimates within an acceptable precision) its desired value, it can stop updating and communicating with other nodes
- Such an action might cause the multi-agent system to become disconnected and prevent other nodes in the system from completing their task
- Distributed coordination algorithm will work only for a subset of the nodes in the multi-agent system
- A mechanism is needed to determine when an agreement between nodes has been reached so that nodes can terminate the distributed process

Existing work

- Typically, finite-time average consensus is handled by **pre-determining the number of steps that all nodes in the network will perform**
  - ✗ requires some prior knowledge about the network and the convergence rate of the iteration process
- Finite-time approximate average consensus there have been two main approaches:
  - synchronous, in which nodes assume knowledge of the diameter of the network and terminate the process all together at a multiple of the diameter
    - ✗ requires global information
  - event-triggered, in which nodes terminate their process provided their value is close enough to the desired one
    - ✗ the magnitude of the error is proportional to the diameter
- Finite-time exact average consensus: nodes do not require any global parameter and compute the exact value
  - ✗ nodes never stop the iterations because it is impossible for them to know whether other nodes have computed the average

- Distributed systems conveniently captured by digraphs
    1. Components represented by vertices (nodes)
    2. Communication and sensing links represented by edges



- Consider a network with nodes $(v_1, v_2, \ldots, v_N)$ (e.g., sensor networks, computers, etc.)
- Nodes can receive information according to (possibly directed) communication links
- Each node v_j has some initial value $x_j[0]$ (e.g., belief, position, velocity)

Graph Notation

- *Digraph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
  - Nodes (system components) $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$
  - Edges (directed communication links) $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ where $(v_j, v_i) \in \mathcal{E}$ iff node v_j can receive information from node v_i
  - In-neighbors $\mathcal{N}_j^- = \{v_i \mid (v_j, v_i) \in \mathcal{E}\}$; in-degree $\mathcal{D}_j^- = |\mathcal{N}_j^-|$
  - Out-neighbors $\mathcal{N}_j^+ = \{v_l \mid (v_l, v_j) \in \mathcal{E}\}$; out-degree $\mathcal{D}_j^+ = |\mathcal{N}_j^+|$
- *Adjacency matrix A:* $A(j,i) = 1$ if $(v_j, v_i) \in \mathcal{E}$; $A(j,i) = 0$ otherwise
- *Undirected graph:* $(v_j, v_i) \in \mathcal{E}$ iff $(v_i, v_j) \in \mathcal{E}$ (bidirectional links)
  In undirected graphs, we have (for each node j)
  $\mathcal{N}_j^+ = \mathcal{N}_j^-$ and $\mathcal{D}_j^+ = \mathcal{D}_j^- = \mathcal{D}_j$; also, $A = A^T$
- *(Strongly) connected (di)graph* if for any $i, j \in \mathcal{V}$, $j \neq i$, there exists a (directed) path connecting them, i.e.,

$$v_i = v_{i_0} \to v_{i_1},\ v_{i_1} \to v_{i_2},\ \ldots,\ v_{i_{t-1}} \to v_{i_t} = v_j$$

Preliminary results - I

Average Consensus using Ratio Consensus

- Run two iterations [Benezit *et al*, 2010], [D-G & H, 2010]

$$\begin{array}{rl|rl} y[k+1] &= P_c y[k] & x[k+1] &= P_c x[k] \\ y[0] &= [y_1[0] \dots y_N[0]]^T & x[0] &= \mathbf{1} \end{array}$$

- Matrix P_c st $P_c(l,j) = \frac{1}{1+\mathcal{D}_j^+}$ for $v_l \in \mathcal{N}_j^+$ (zero otherwise)
- Since P_c is primitive column stochastic, we know that as $k \to \infty$, $P_c^k \to \mathbf{v}\mathbf{1}^T$ for a *strictly positive* vector $\mathbf{v}$ such that $\mathbf{v} = P_c\mathbf{v}$ ($\mathbf{v}$ is *normalized* so that its entries sum to unity)
- This implies that

$$\lim_{k\to\infty} y[k] = \mathbf{v}\mathbf{1}^T y[0] = \left(\sum_{\ell=1}^N y_\ell[0]\right)\mathbf{v}$$
$$\lim_{k\to\infty} x[k] = \mathbf{v}\mathbf{1}^T x[0] = N\mathbf{v}$$

- For all nodes $j \in \{1, 2, \dots, N\}$, ratio converges

$$\frac{y_j[k]}{x_j[k]} \to \frac{v_j \sum_{\ell=1}^N y_\ell[0]}{v_j N} = \frac{\sum_{\ell=1}^N y_\ell[0]}{N} \equiv \overline{y}$$

Minimal polynomial of a matrix pair

The minimal polynomial associated with the matrix pair $[P, e_j^\mathsf{T}]$, 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 $e_j^\mathsf{T} q_j(P) = 0$.

Easy to show (e.g., using the techniques in [Y. Yuan *et al.*, 2009]) that

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

where $\alpha_{M_j+1}^{(j)} = 1$. Denote z-transform of $w_j[k]$ as $W_j(z) \triangleq \mathbb{Z}(w_j[k])$. Then,

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

where $q_j(z)$ is the minimal polynomial of $[P, e_j^\mathsf{T}]$

Define the following polynomial:

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

The application of the final value theorem (FVT) yields:

$$\phi_y(j) = \lim_{k\to\infty} y_j[k] = \lim_{z\to 1}(z-1)Y_j(z) = \frac{y_{M_j}^{\mathsf{T}}\boldsymbol{\beta}_j}{\mathbf{1}^{\mathsf{T}}\boldsymbol{\beta}_j}$$

$$\phi_x(j) = \lim_{k\to\infty} x_j[k] = \lim_{z\to 1}(z-1)X_j(z) = \frac{x_{M_j}^{\mathsf{T}}\boldsymbol{\beta}_j}{\mathbf{1}^{\mathsf{T}}\boldsymbol{\beta}_j}$$

where

- $y_{M_j}^{\mathsf{T}} = (y_j[0], y_j[1], \ldots, y_j[M_j])$
- $x_{M_j}^{\mathsf{T}} = (x_j[0], x_j[1], \ldots, x_j[M_j])$
- $\boldsymbol{\beta}_j$ is the vector of coefficients of the polynomial $p_j(z)$

Preliminaries - IV

How can we obtain β_j in the computation of final values?

- Consider the vectors of differences between $2k+1$ successive discrete-time values of $y_j[k]$ and $x_j[k]$ at node v_j:

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

- Let us define their associated Hankel matrix:

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

- β_j can be computed as **the kernel of the first defective Hankel matrix** for each of $\Gamma\{\overline{y}_{2k}^{\mathsf{T}}\}$ and $\Gamma\{\overline{x}_{2k}^{\mathsf{T}}\}$
- Works for arbitrary initial conditions y_0 and x_0, except a set of initial conditions with Lebesgue measure zero

Preliminaries - V

Computation of the average in digraphs in a finite number of steps

Theorem [TC *et al.*, 2015])

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 for ratio consensus. Then, the solution to the average consensus can be distributively obtained in minimum number of steps at each node v_j, by computing

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

- A distributed algorithm with which the exact average is computed in $2(M_j + 1)$ time steps for each node v_j

Theorem [TC *et al.*, 2015])

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 for ratio consensus. Then, the solution to the average consensus can be distributively obtained in minimum number of steps at each node v_j, by computing

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

- A distributed algorithm with which the exact average is computed in $2(M_j + 1)$ time steps for each node v_j

Since nodes do not know whether other nodes have computed the average yet, the iterations continue indefinitely

Main results

- [Y. Yuan *et al.*, 2013] showed that after $2(M_j+1)$ the kernel of the Hankel matrices $\Gamma\{\overline{y}_{2k}^{\mathsf{T}}\}$ and $\Gamma\{\overline{x}_{2k}^{\mathsf{T}}\}$ for arbitrary initial conditions y_0 and x_0 (except a set of initial conditions with *Lebesgue measure zero*) become defective
- **Main idea:**
  - $2(M_j+1)$ 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 (Theorem 1), node v_j can use it to decide when everybody is done
  - 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 not, then all other nodes have also computed their final value and the node can terminate the iterations
    - ✗ Otherwise, at least one other node v_i has not finished computing its average and its value participating in the $\max$-consensus algorithm keeps increasing

Termination algorithm

- Once ratio consensus iterations 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
- A $\max$-consensus algorith is initiated as well, given by

$$\theta_j[k+1] = \max_{v_i \in \mathcal{N}_j \cup \{v_j\}} \big\{ \max\{\theta_i[k], c_i[k]\} \big\}, \quad \theta_j[0] = 0$$

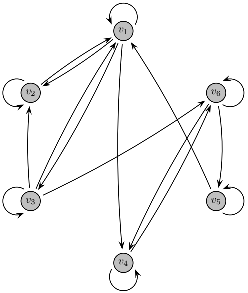
- Every time step k for which $\theta_j[k+1] = \theta_j[k]$, counter r_j increments by one; otherwise, r_j is set to zero, i.e.,

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

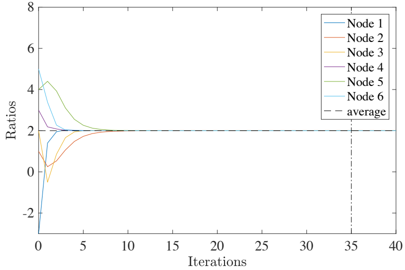
- Once the square Hankel matrices $\Gamma\{\overline{y}_{M_j}^{\mathsf{T}}\}$ and $\Gamma\{\overline{x}_{M_j}^{\mathsf{T}}\}$ for node v_j lose rank, node v_j saves the count of the counter c_j at that time step; note that c_j stops at $2(M_j + 1)$
- Node v_j terminates iterations when $r_j = 2(M_j + 1)$

Examples

Example 1: Simple 6-node digraph



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



- The number of steps, T_j, needed for node v_j to terminate a process is

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

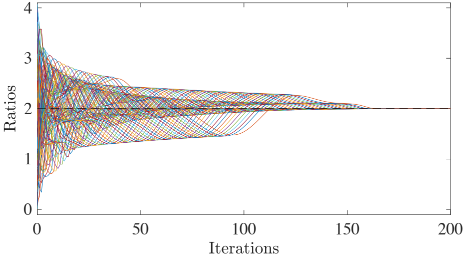
As a result, $T_{\max}$, is given by

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

Examples

Example 2: Leslie network (100 nodes)

- A discrete, age-structured model of population growth
- Due to the structure of the network, the asymptotic convergence time is large:



- (Asymptotic) ratio-consensus algorithm seems to need more than 160 steps for the error to be small enough
- **By running our distributed termination algorithm, all nodes terminate their process in 75 steps**

Thank You!

Questions?

For more information:

themistoklis.charalambous@aalto.fi
chadjic@ucy.ac.cy

PhD Position on Wireless Networked Control Systems

- A fully-funded PhD student position is available at the Distributed Systems Control Group, Department of Electrical Engineering and Automation, School of Electrical Engineering, Aalto University
- Apply by 29th June 2018 (http://www.aalto.fi/en/about/careers/jobs/view/1877/)
- Available from August 2018