Distributed Consensus and Linear Functional Calculation in Networks: An Observability Perspective

Shreyas Sundaram and Christoforos N. Hadjicostis

Coordinated Science Laboratory
and
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
Consider a network (directed or undirected) with nodes $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$

- Each node $i$ has some initial value $x_i[0]$
  
  - e.g., temperature measurement, position, vote, fault status, etc.

- Each node can only receive information from its neighbors

- Objective: A subset of the nodes must calculate some function of the initial values

  - e.g., average, max, min, mode, etc.

  - **Consensus**: All nodes calculate the same function
Background on Consensus Protocols

Consensus problems studied for several decades (e.g., [Lynch, \textit{Distributed Algorithms}])

- Simple solution: Flooding
  - Each node repeatedly sends all received values to neighbors
  - All nodes will know all initial values after several time-steps

- Another solution: “Convergecast”
  - Nodes organize themselves into a tree
  - Leaves propagate their values towards root
  - Root calculates function and broadcasts result back to leaves

- Many other protocols that deal with node/link faults, etc.
A different approach: Linear iterations

At each time-step, each node updates its value to be a linear combination of itself and its neighbors:

\[
\begin{bmatrix}
  x_1[k+1] \\
  x_2[k+1] \\
  \vdots \\
  x_N[k+1]
\end{bmatrix} =
\begin{bmatrix}
  w_{11} & w_{12} & \cdots & w_{1N} \\
  w_{21} & w_{22} & \cdots & w_{2N} \\
  \vdots & \vdots & \ddots & \vdots \\
  w_{N1} & w_{N2} & \cdots & w_{NN}
\end{bmatrix} \begin{bmatrix}
  x_1[k] \\
  x_2[k] \\
  \vdots \\
  x_N[k]
\end{bmatrix}
\]

- \( w_{ij} = 0 \) if \( x_j \) is not a neighbor of \( x_i \)
- Choose weight matrix \( W \) so that \( \lim_{k \to \infty} x[k] = 1c'x[0] \)
  - \( 1 \) is column vector with all 1’s, and \( c' \) is some row vector
  - Special case: \( c' = \frac{1}{N} 1' \) (distributed averaging)

Substantial analysis of convergence conditions available in the literature (e.g., see survey paper by Olfati-Saber, Proc. IEEE, Jan. 2007)
Convergence in Time-Invariant Graphs

- Linear iteration: \( x[k + 1] = W x[k] \)

- Necessary and sufficient conditions for \( x[k] \to 1c'x[0] \) [Xiao & Boyd, 2004]:
  - \( W1 = 1 \)
  - \( c'W = c' \)
  - All other eigenvalues of \( W \) must have magnitude strictly less than 1

- Rate of convergence given by second largest eigenvalue of \( W \)
  - For faster convergence, minimize second largest eigenvalue by choosing weights appropriately [Xiao & Boyd, 2004]

- However, almost all existing methods only consider asymptotic convergence

**Contribution:** We analyze linear iteration schemes using observability theory

- Allows nodes to reach consensus in finite-time
Modeling the Values Seen by Each Node

- Each node receives the values of its neighbors at each time-step
- Let $y_i[k] = E_i x[k]$ denote values received by node $i$ at time-step $k$
  - Rows of $E_i$ index portions of state-vector $x[k]$ that are available to node $i$
- e.g., for following graph,

$$x[k + 1] = \begin{bmatrix} w_{11} & w_{12} & 0 & w_{14} \\ w_{12} & w_{22} & 0 & w_{24} \\ 0 & 0 & w_{33} & w_{34} \\ w_{14} & w_{24} & w_{34} & w_{44} \end{bmatrix} \begin{bmatrix} x[k], & y_2[k] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x[k] \end{bmatrix}$$
The Observability Matrix

- Since $x[k] = W^k x[0]$, we have $y_i[k] = E_i x[k] = E_i W^k x[0]$
- Set of values seen by node $i$ over $L + 1$ time-steps is

$$
\begin{bmatrix}
  y_i[0] \\
  y_i[1] \\
  y_i[2] \\
  \vdots \\
  y_i[L]
\end{bmatrix}
= \begin{bmatrix}
  E_i \\
  E_i W \\
  E_i W^2 \\
  \vdots \\
  E_i W^L
\end{bmatrix}
\begin{bmatrix}
  x[0]
\end{bmatrix}
\underbrace{\quad O_{i,L}}_{\text{Row-space of } O_{i,L}}
$$

- $O_{i,L}$ is the **observability matrix** for the pair $(W, E_i)$
- Row-space of $O_{i,L}$ characterizes all linear functionals that can be calculated by node $i$ after $L + 1$ time-steps
Calculating the Consensus Value

- Assume $W$ is designed so that $c'$ is in the row-space of the observability matrix for every node.
- For each node $i$, find smallest $L_i$ such that $\text{rank}\left[O_{i,L_i} c'\right] = \text{rank}\left[O_{i,L_i}\right]$.
- Find a vector $\Gamma'_i$ for each $i$ satisfying $\Gamma'_i O_{i,L_i} = c'$.

Protocol:
- Nodes run linear iteration for $\max_i L_i + 1$ time-steps.
- Each node $i$ calculates $c' x[0]$ after $L_i + 1$ time-steps as

$$
\begin{bmatrix}
    y_i[0] \\
    y_i[2] \\
    \vdots \\
    y_i[L_i]
\end{bmatrix}
\Gamma'_i = \Gamma'_i O_{i,L_i} x[0] = c' x[0]
$$
The Observability Index

From observability theory:

- There exists a positive integer $\nu_i$ such that
  \[
  \text{rank}(O_{i,0}) < \text{rank}(O_{i,1}) < \cdots < \text{rank}(O_{i,\nu_i-1}) = \text{rank}(O_{i,\nu_i}) = \text{rank}(O_{i,\nu_i+1}) = \cdots
  \]
- Rank of $O_{i,L}$ increases until $L = \nu_i - 1$, and then stops increasing
- $\nu_i$ is known as the **observability index**

Implication for linear iteration model:

- Values seen by node $i$ after time-step $\nu_i - 1$ are linear combinations of $y_i[0], y_i[1], \ldots, y_i[\nu_i - 1]$
- If node $i$ can calculate $c'x[0]$, it will require at most $\nu_i$ time-steps
- We show $\nu_i$ is upper-bounded by $\nu_i \leq N - \text{degree}(i)$
Row-Space of the Observability Matrix

- How do we choose the weight matrix so that $c'$ is in the row-space of $O_{i,\nu_i-1}$?
- For the linear iteration model, we show:
  - If $\mu$ is a simple eigenvalue of $W$, with right eigenvector $d$ that has all entries nonzero, and left-eigenvector $c'$, then $c'$ is in the row-space of $O_{i,\nu_i-1}$ for all $i$
- Recall: need $W1 = 1$ for asymptotic consensus
  - Any matrix that provides asymptotic consensus also allows finite-time consensus
- Do not need to restrict attention to matrices that provide asymptotic consensus
  - **Perron-Frobenius Theorem:**
    If $W$ is nonnegative and the graph is strongly connected, then $W$ has a simple eigenvalue $\mu$ with left and right-eigenvectors that have all positive entries.
Consider example from [Xiao & Boyd, 2004]

Weights on edges and nodes chosen to maximize asymptotic rate of convergence

$W$ has simple eigenvalue 1, with $W \mathbf{1} = \mathbf{1}$, $\frac{1}{8} \mathbf{1}' W = \frac{1}{8} \mathbf{1}'$

Vector $c' = \frac{1}{8} \mathbf{1}'$ will be in the row-space of observability matrix for every node
Example: Finding the Coefficient Vector

Consider node 2 in the network

\[
y_2[k] = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix} x[k]
\]

Find smallest \( L_2 \) such that \( \text{rank} \left[ O_{2,L_2}^{1'1} \right] = \text{rank} [O_{2,L_2}] \)

- \( L_2 = 0 \): \( O_{2,0} = E_2 \)
- \( L_2 = 1 \): \( O_{2,1} = \begin{bmatrix} E_2 \\ E_2 W \end{bmatrix} \)

Node 2 can calculate \( c'x[0] \) after two time-steps

Calculate \( \Gamma_2' = \)

\[
\begin{bmatrix}
-0.2679 & -0.4464 & -0.3214 & -0.1964 & 0.6250 & -0.2098 & 2.6965 & -0.8795 \\
\end{bmatrix}
\]
Example: Running the Linear Iteration

- In this example, \( L_i = 1 \) for all nodes and consensus can be reached in 2 time-steps
- Suppose initial values of the nodes are

\[
x[0] = \begin{bmatrix} -1.3256 & -13.6558 & 4.2533 & 5.8768 - 8.4647 & 14.9092 & 14.8916 & 2.6237 \end{bmatrix}
\]

with mean 2.3885
- Run iteration \( x[k + 1] = W x[k] \) for 2 time-steps:

\[
\begin{bmatrix} x[0] & x[1] \end{bmatrix} = \begin{bmatrix} -1.3256 & -3.5040 \\ -13.6558 & -1.8860 \\ 4.2533 & 1.3574 \\ 5.8768 & 5.1774 \\ -8.4647 & 5.1774 \\ 14.9092 & 5.1774 \\ 14.8916 & 3.0078 \\ 2.6237 & 4.6009 \end{bmatrix}
\]
Example: Calculating the Consensus Value

- Values seen by node 2:

\[
y_2[0] = \begin{bmatrix} -1.3256 \\ -13.6558 \\ 4.2533 \\ 2.6237 \end{bmatrix}, \quad y_2[1] = \begin{bmatrix} -3.5040 \\ -1.8860 \\ 1.3574 \\ 4.6009 \end{bmatrix}
\]

- Node 2 calculates \( c'x[0] \) as \( \Gamma'_2 \begin{bmatrix} y_2[0] \\ y_2[1] \end{bmatrix} = 2.3885 \)

- All other nodes calculate \( c'x[0] \) in the same manner

- Note: diameter and radius of network are both 2, so this scheme is time-optimal for consensus in this example
Finding the Coefficients: Decentralized Method

- Nodes require coefficient vectors $\Gamma'_i$ to calculate $c'x[0]$.
- If $W$ is not known to any node, can the nodes calculate their coefficients in a decentralized manner?
- Strategy: reconstruct observability matrix by running linear iterations with appropriate choices of initial conditions.

Algorithm:
- Nodes run $N$ different linear iterations, each for $N - 1$ time-steps.
  - For the $j$'th run, node $j$ sets initial value to be 1 and others set initial value to be 0.
  - Denote values seen by node $i$ during time-step $k$ of $j$'th run as $y_{i,j}[k]$.
Finding the Coefficients: Decentralized Method

- Values seen by node $i$ during $j$'th run:

$$
\begin{bmatrix}
    y_{i,j}[0] \\
    y_{i,j}[1] \\
    \vdots \\
    y_{i,j}[L_i]
\end{bmatrix}
= \mathcal{O}_{i,L_i}
\begin{bmatrix}
    0 \\
    \vdots \\
    1 \\
    0
\end{bmatrix}
$$

- After all runs are completed, node $i$ has access to matrix

$$
\begin{bmatrix}
    y_{i,1}[0] & y_{i,2}[0] & \cdots & y_{i,N}[0] \\
    y_{i,1}[1] & y_{i,2}[1] & \cdots & y_{i,N}[1] \\
    \vdots & \vdots & \ddots & \vdots \\
    y_{i,1}[L_i] & y_{i,2}[L_i] & \cdots & y_{i,N}[L_i]
\end{bmatrix}
= \mathcal{O}_{i,L_i}
\begin{bmatrix}
    1 & 0 & \cdots & 0 \\
    0 & 1 & \cdots & 0 \\
    \vdots & \vdots & \ddots & \vdots \\
    0 & 0 & \cdots & 1
\end{bmatrix}
= \mathcal{O}_{i,L_i}
$$

- Node $i$ can now calculate $\Gamma_i'$ as before
Summary and Future Work

Summary:
- Used observability theory to study problem of distributed consensus and linear functional calculation
- Can obtain distributed consensus after running linear iteration for a finite number of time-steps
- Method does not depend on magnitudes of eigenvalues of weight matrix
- Nodes can learn their coefficients after running $N$ linear iterations

Future Work:
- Choose weights to minimize number of time-steps for consensus
- Incorporate robustness to link/node faults
- Find more efficient methods to perform decentralized computation of coefficients