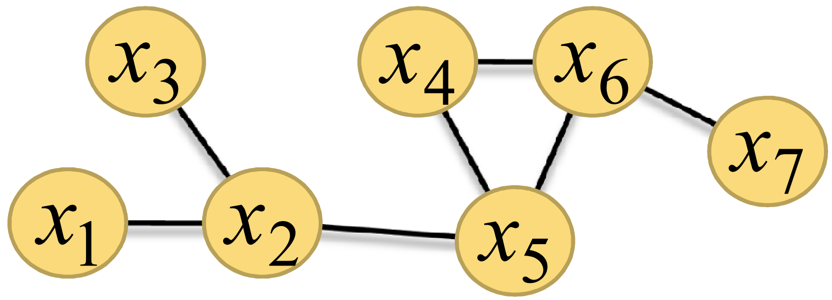
Distributed Function Calculation via Linear Iterations in the Presence of Malicious Agents

Part II: Overcoming Malicious Behavior

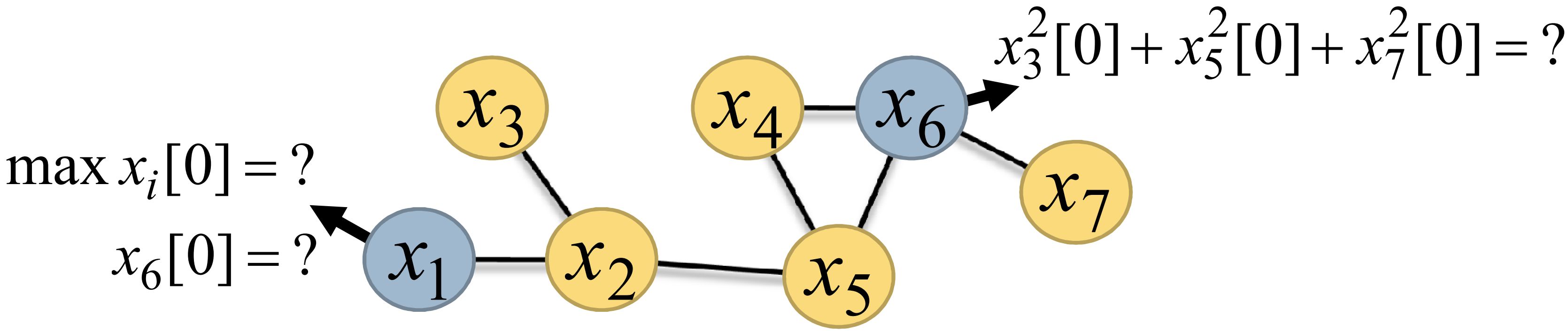
Shreyas Sundaram and Christoforos N. Hadjicostis
Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

Problem Formulation



- Consider a network with nodes $\{x_1, x_2, \ldots, x_N\}$
  - e.g., sensors, robots, unmanned vehicles, computers, etc.
- Each node x_i has some initial value $x_i[0]$
  - e.g., temperature measurement, position, vote, etc.
- **Objective:** Some nodes must calculate certain functions of initial values

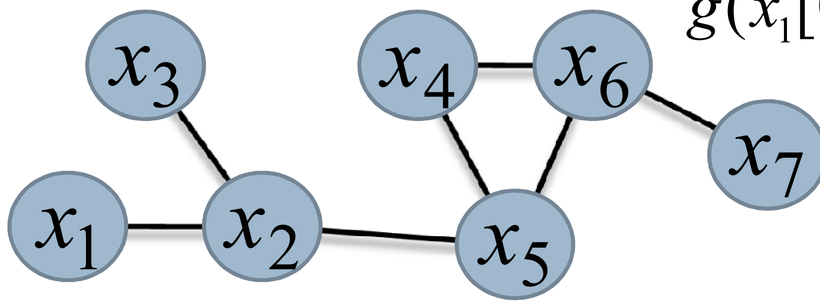
Problem Formulation



- Consider a network with nodes $\{x_1, x_2, \ldots, x_N\}$
  - e.g., sensors, robots, unmanned vehicles, computers, etc.
- Each node x_i has some initial value $x_i[0]$
  - e.g., temperature measurement, position, vote, etc.
- **Objective:** Some nodes must calculate certain functions of initial values

Problem Formulation

$$g(x_1[0], \ldots, x_N[0]) = ?$$



- Consider a network with nodes $\{x_1, x_2, \ldots, x_N\}$
  - e.g., sensors, robots, unmanned vehicles, computers, etc.
- Each node x_i has some initial value $x_i[0]$
  - e.g., temperature measurement, position, vote, etc.
- **Objective:** Some nodes must calculate certain functions of initial values
  - **Consensus:** All nodes calculate the same function

Linear Iterative Schemes

- Investigate **linear iterative schemes** for distributed function calculation
  - At each time-step k, every node updates its value as

$$x_i[k+1] = w_{ii} x_i[k] + \sum_{j \in nbr(i)} w_{ij} x_j[k]$$

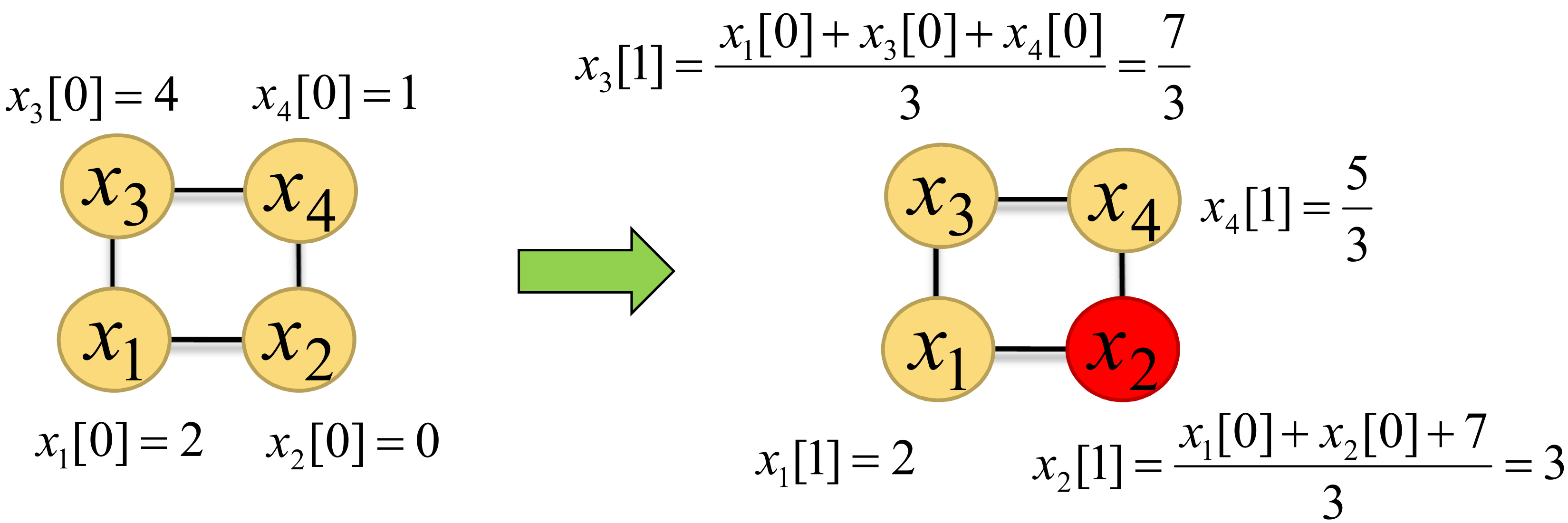
- **Theorem ([1]):** If the network is strongly connected, then for **almost any choice of weights**, each node x_i can calculate **any arbitrary function** of the initial values after running the linear iteration for **at most N-deg(i) time-steps**.
  - "**Almost any**": For all but a set of measure zero

[1] Sundaram & Hadjicostis, Distributed Function Calculation and Consensus Using Linear Iterative Strategies, IEEE Journal on Selected Areas in Communications, May 2008

Potential for Incorrect Behavior

- What if some nodes do not follow the linear iterative strategy?
  - **Faulty nodes:** update their values incorrectly due to hardware faults, or stop working altogether
  - **Malicious nodes:** willfully update their values incorrectly (perhaps in a **coordinated** manner) in an attempt to prevent other nodes from calculating functions

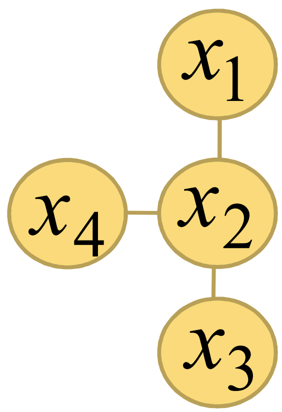
An Example of Malicious Behavior



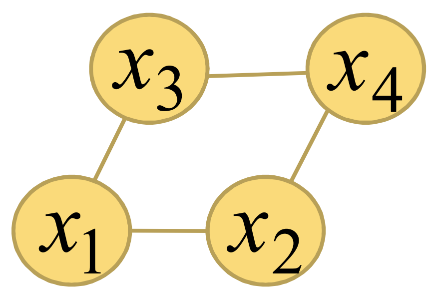
- Node x_2 is malicious and pretends $x_4[0] = 7$ in its update
- Node x_3 behaves correctly and uses $x_4[0] = 1$ in its update
- Node x_1 doesn't know who to believe
  - i.e., is node x_4's value equal to 7 or 1?
- Node x_1 needs another node to act as tie-breaker

Key Concept: Graph Connectivity

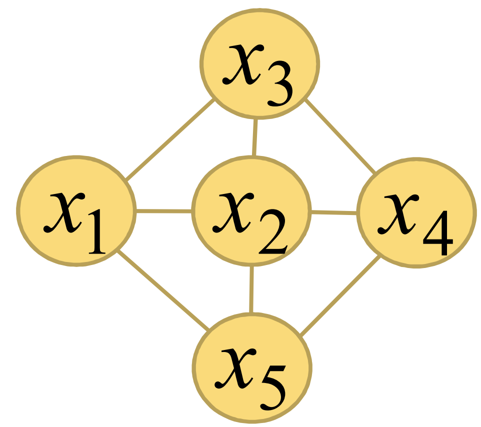
- The **connectivity** of a graph is the maximum number of vertex disjoint paths between any two nodes



Connectivity: 1



Connectivity: 2



Connectivity: 3

Main Result

- We show that if network connectivity is **2f+1** or more, linear iteration is **robust** to **f or fewer malicious (possibly coordinated) nodes**
  - Run linear iteration for **at most N time-steps** with **almost any weights**
  - **Every node** can calculate any **arbitrary** function of all values

- In Part I, we proved the converse result:
  - If network connectivity is **2f or less**, **f malicious nodes** can update their values so that **one or more nodes cannot calculate an arbitrary function** of the initial values (regardless of choice of weights)

Modeling Faulty/Malicious Behavior

- **Correct** update equation for node x_i:

$$x_i[k+1] = w_{ii} x_i[k] + \sum_{j \in nbr(i)} w_{ij} x_j[k]$$

- **Faulty or malicious** update by node x_i:

$$x_i[k+1] = w_{ii} x_i[k] + \sum_{j \in nbr(i)} w_{ij} x_j[k] + u_i[k]$$

  - $u_i[k]$ is an additive error at time-step k
  - Allows x_i to update its value in a **completely arbitrary** manner!

Linear Iteration with Faulty/Malicious Nodes

- Let $S = \{x_{j1}, x_{j2}, \ldots, x_{jf}\}$ be set of nodes that are malicious

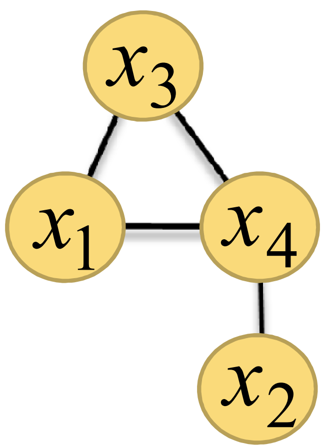
- Update equation for entire system:

$$\underbrace{\begin{bmatrix} x_1[k+1] \\ \vdots \\ x_N[k+1] \end{bmatrix}}_{\mathbf{x}[k+1]} = \underbrace{\begin{bmatrix} w_{11} & \cdots & w_{1N} \\ \vdots & \ddots & \vdots \\ w_{N1} & \cdots & w_{NN} \end{bmatrix}}_{\mathbf{W}} \underbrace{\begin{bmatrix} x_1[k] \\ \vdots \\ x_N[k] \end{bmatrix}}_{\mathbf{x}[k]} + \underbrace{\begin{bmatrix} \mathbf{e}_{j_1} & \mathbf{e}_{j_2} & \cdots & \mathbf{e}_{j_f} \end{bmatrix}}_{\mathbf{B}_S} \underbrace{\begin{bmatrix} u_{j_1}[k] \\ u_{j_2}[k] \\ \vdots \\ u_{j_f}[k] \end{bmatrix}}_{\mathbf{u}_S[k]}$$

  - Weight $w_{ij} = 0$ if node x_j is not a neighbor of node x_i
  - $\mathbf{e}_j$ is vector with 1 in j-th position and 0's elsewhere

Modeling the Values Seen by Each Node

- At each time-step, each node has access to values of its neighbors (and its own value)
- Let $\mathbf{y}_i[k] = \mathbf{C}_i \mathbf{x}[k]$ denote values seen by node x_i at time-step k
  - Rows of $\mathbf{C}_i$ index portions of $\mathbf{x}[k]$ available to x_i



For node x_3:

$$\mathbf{y}_3[k] = \begin{bmatrix} x_1[k] \\ x_3[k] \\ x_4[k] \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{C}_3} \mathbf{x}[k]$$

Modeling the Values Seen by Each Node

- Set of all values seen by node x_i over L+1 time-steps:

$$\underbrace{\begin{bmatrix} \mathbf{y}_i[0] \\ \mathbf{y}_i[1] \\ \mathbf{y}_i[2] \\ \vdots \\ \mathbf{y}_i[L] \end{bmatrix}}_{\mathbf{y}_i[0:L]} = \underbrace{\begin{bmatrix} \mathbf{C}_i \\ \mathbf{C}_i\mathbf{W} \\ \mathbf{C}_i\mathbf{W}^2 \\ \vdots \\ \mathbf{C}_i\mathbf{W}^L \end{bmatrix}}_{O_{i,L}} \mathbf{x}[0] + \underbrace{\begin{bmatrix} 0 & 0 & \cdots & 0 \\ \mathbf{C}_i\mathbf{B}_S & 0 & \cdots & 0 \\ \mathbf{C}_i\mathbf{W}\mathbf{B}_S & \mathbf{C}_i\mathbf{B}_S & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{C}_i\mathbf{W}^{L-1}\mathbf{B}_S & \mathbf{C}_i\mathbf{W}^{L-2}\mathbf{B}_S & \cdots & \mathbf{C}_i\mathbf{B}_S \end{bmatrix}}_{M_{i,L}^S} \underbrace{\begin{bmatrix} \mathbf{u}_S[0] \\ \mathbf{u}_S[1] \\ \vdots \\ \mathbf{u}_S[L-1] \end{bmatrix}}_{\mathbf{u}_S[0:L-1]}$$

- Matrices $O_{i,L}$ and $M^S_{i,L}$ characterize ability of node x_i to calculate functions of **x**[0] in the presence of faulty or malicious nodes
  - $O_{i,N-1}$ is the **observability matrix** for the pair (**W**,$\mathbf{C}_i$)
  - $M^S_{i,L}$ is the **fault matrix** for the set S

Decoding Procedure

- Each node x_i has access to
  - Weight matrix **W**
  - A finite nonnegative integer L_i (described later)
  - $L_{Max} = Max_i \; L_i$

Decoding Procedure

- Each node x_i has access to
  - Weight matrix **W**
  - A finite nonnegative integer L_i (described later)
  - $L_{Max} = Max_i \, L_i$

All nodes run linear iteration for L_{Max} + 1 time-steps. Let S be the set of malicious nodes.

↓

Values seen by node x_i over first L_i + 1 time-steps:

$$\mathbf{y}_i[0:L_i] = O_{i,L_i}\mathbf{x}[0] + M_{i,L_i}^{S}\mathbf{u}_S[0:L_i-1]$$

↓

Node x_i finds **candidate set** S_c of f nodes such that

$$\mathbf{y}_i[0:L_i] = O_{i,L_i}\mathbf{z} + M_{i,L_i}^{S_C}\mathbf{v}$$

(for some vectors **z** and **v**)

Decoding Procedure

- Each node x_i has access to
  - Weight matrix **W**
  - A finite nonnegative integer L_i (described later)
  - $L_{Max} = Max_i\ L_i$

We show: if graph has connectivity 2f+1 or higher, then for almost any choice of **W**, **z** = **x**[0] (i.e., node x_i can obtain **x**[0] from $\mathbf{y}_i[0:L_i]$)

All nodes run linear iteration for L_{Max} + 1 time-steps. Let S be the set of malicious nodes.

Values seen by node x_i over first L_i + 1 time-steps:

$$\mathbf{y}_i[0:L_i] = O_{i,L_i}\mathbf{x}[0] + M^S_{i,L_i}\mathbf{u}_S[0:L_i-1]$$

Node x_i finds **candidate set** S_c of f nodes such that

$$\mathbf{y}_i[0:L_i] = O_{i,L_i}\mathbf{z} + M^{S_C}_{i,L_i}\mathbf{v}$$

(for some vectors **z** and **v**)

Sketch of Proof

- Values seen by node x_i over L_i+1 time-steps:

$$\mathbf{y}_i[0:L_i] = O_{i,L_i}\mathbf{x}[0] + M_{i,L_i}^{S}\mathbf{u}_S[0:L_i-1]$$

- Node x_i finds a **candidate set** S_C of f nodes such that

$$O_{i,L_i}\mathbf{z} + M_{i,L_i}^{S_C}\mathbf{v} = \mathbf{y}_i[0:L_i] \quad \text{(for some vectors } \mathbf{z} \text{ and } \mathbf{v}\text{)}$$

$$= O_{i,L_i}\mathbf{x}[0] + M_{i,L_i}^{S}\mathbf{u}_S[0:L_i-1]$$

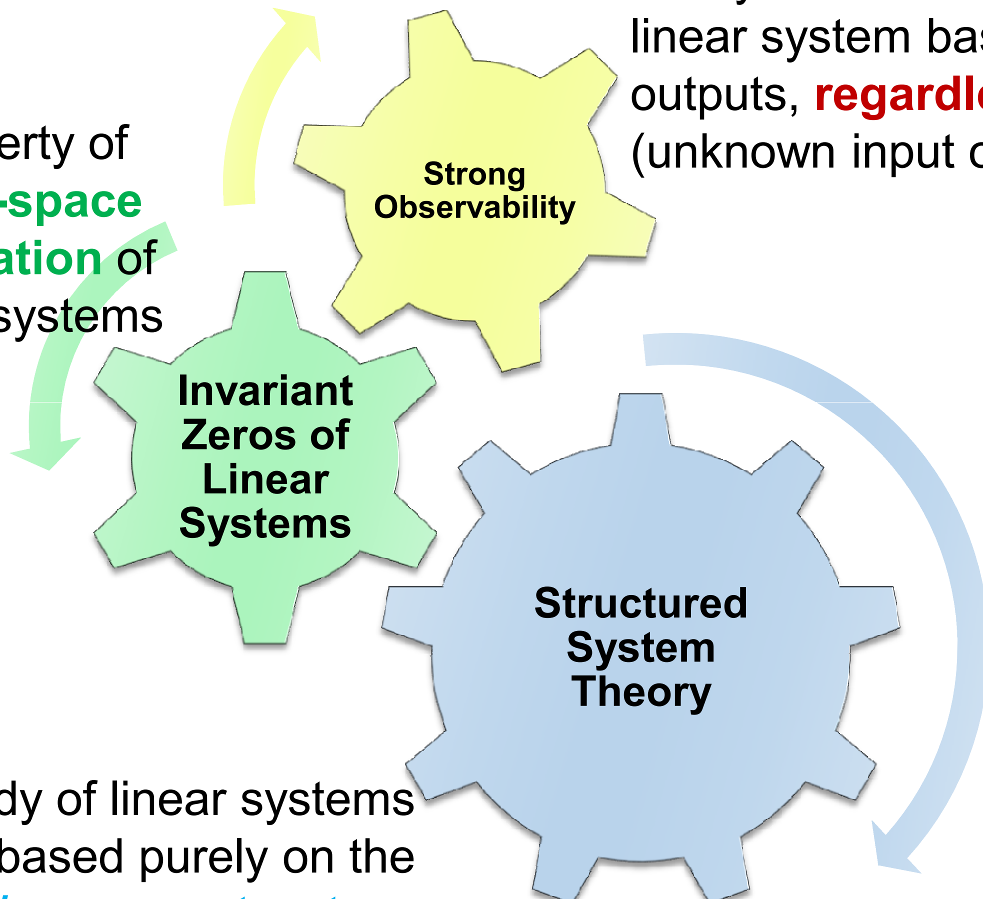
$$\Leftrightarrow O_{i,L_i}(\mathbf{z}-\mathbf{x}[0]) + M_{i,L_i}^{S_C}\mathbf{v} - M_{i,L_i}^{S}\mathbf{u}_S[0:L_i-1] = 0$$

- Choose **W** and L_i so that, for **any** sets S_C and S of f nodes each,

$$\text{rank}\left(\begin{bmatrix} O_{i,L_i} & M_{i,L_i}^{S} & M_{i,L_i}^{S_C} \end{bmatrix}\right) = N + \text{rank}\left(\begin{bmatrix} M_{i,L_i}^{S} & M_{i,L_i}^{S_C} \end{bmatrix}\right)$$

- Then $\mathbf{z} = \mathbf{x}[0]$

Using Linear System Theory to Design the Weight Matrix



Ability to determine state of a linear system based on outputs, **regardless of inputs** (unknown input observers)

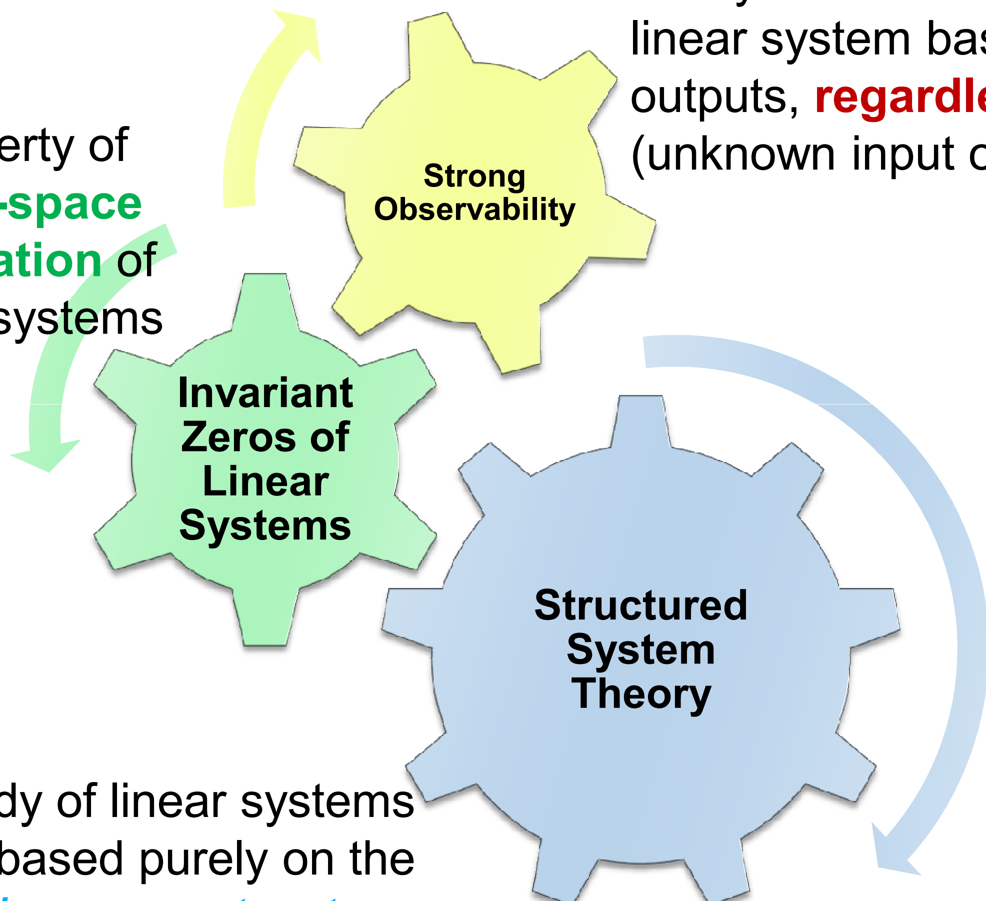
Property of **state-space realization** of linear systems

Study of linear systems based purely on the **zero/nonzero structure** of the system matrices

Using Linear System Theory to Design the Weight Matrix

Ability to determine state of a linear system based on outputs, **regardless of inputs** (unknown input observers)

Property of **state-space realization** of linear systems



Design weight matrix **W** for linear iterative model

$$\mathbf{x}[k+1] = \mathbf{W}\mathbf{x}[k] + \mathbf{B}_S\mathbf{u}_S[k]$$

$$\mathbf{y}_i[k] = \mathbf{C}_i\mathbf{x}[k]$$

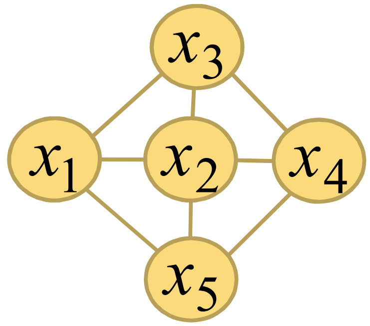
Study of linear systems based purely on the **zero/nonzero structure** of the system matrices

Robustness of the Linear Iterative Scheme

- Using previously mentioned tools, we prove:
  - **Theorem**: If the network has **connectivity 2f+1** or higher, then for **almost any choice of weight matrix W**, and **any node x_i**, the columns of $O_{i,N-1}$ will be **linearly independent** of the columns of $M^{S}_{i,N-1}$ and $M^{S_C}_{i,N-1}$ for any sets S and S_C of f nodes each.

- Any node x_i can obtain the **entire** initial value vector **x**[0] after running the linear iteration for at most N time-steps (i.e., $L_i < N$), **despite** the actions of f malicious nodes

Example



$$W = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix}$$

- **Objective:** Each node has to calculate $\mathbf{x}^T[0]\mathbf{x}[0]$ even when there is up to f = 1 malicious node
  - Connectivity of the network is 3, so this is possible
- Consider node x_1:

$$\mathbf{y}_1[k] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}[k] = \mathbf{C}_1 \mathbf{x}[k]$$

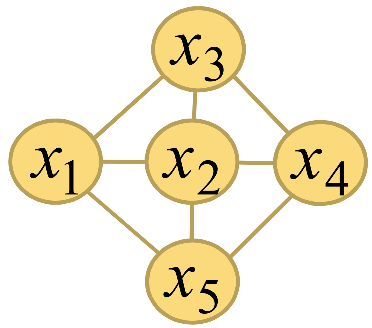
Example (cont.)

- Find smallest delay L_1 for node x_1 to be able to calculate its function
  - For L_1 = 1, and any sets S and S_C of f = 1 node each:

$$O_{1,1} = \begin{bmatrix} \mathbf{C}_1 \\ \mathbf{C}_1\mathbf{W} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix} \quad M_{1,1}^{S} = \begin{bmatrix} 0 \\ \mathbf{C}_1\mathbf{B}_S \end{bmatrix} \quad M_{1,1}^{S_C} = \begin{bmatrix} 0 \\ \mathbf{C}_1\mathbf{B}_{S_C} \end{bmatrix}$$

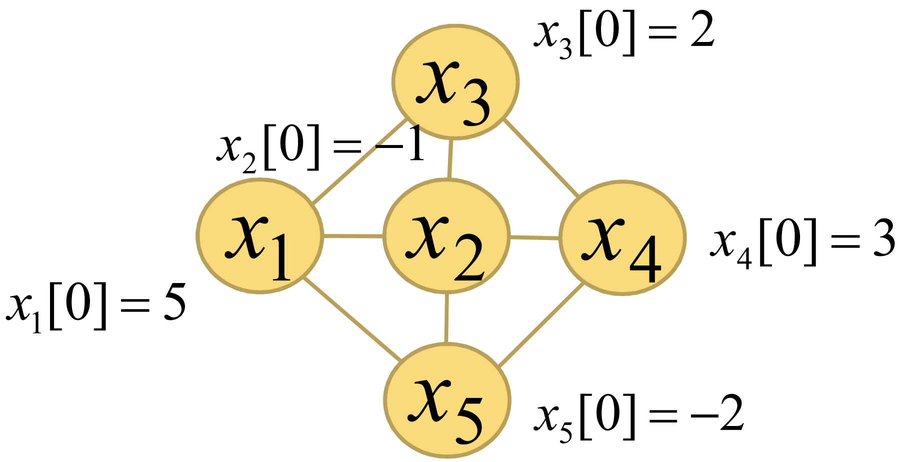
  - For any sets S and S_C, columns of $O_{1,1}$ are **linearly independent** of the columns of $M^S_{1,1}$ and $M^{S_C}_{1,1}$
    - Node x_1 can calculate its function after L_1+1 = 2 time-steps!

Example (cont.)



- For this example, we find nodes x_1, x_3, x_4, x_5 can calculate their functions in 2 time-steps, and node x_2 can calculate its function in 1 time-step
  - Linear iterative strategy is **time-optimal** for this network!

Example (cont.)



$$W = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Performing Function Calculation

- Nodes run linear iteration:
  - Node x_2 is malicious, updates as $x_2[1] = \sum_{j=1}^{5} x_j[0] + 3$ ← Additive error!
  - All other nodes behave correctly:

$$\mathbf{x}[1] = \begin{bmatrix} 4 & 10 & 9 & 2 & 5 \end{bmatrix}^T$$

- Node x_1 sees $\mathbf{y}_1[0] = [5\ -1\ 2\ -2]^T$ and $\mathbf{y}_1[1] = [4\ 10\ 9\ 5]^T$

Example (cont.)

- Node x_1 tries to find a set S_C of f = 1 node so that

$$\begin{bmatrix} \mathbf{y}_1[0] \\ \mathbf{y}_1[1] \end{bmatrix} = O_{1,1}\mathbf{z} + M_{1,1}^{S_C}\mathbf{v}$$

- For $S_C = \{x_2\}$, node x_1 finds: $\begin{bmatrix} \mathbf{y}_1[0] \\ \mathbf{y}_2[0] \end{bmatrix} = O_{1,1}\begin{bmatrix} 5 \\ -1 \\ 2 \\ 3 \\ -2 \end{bmatrix} + M_{1,1}^{S_C}(-3)$
  - Node x_1 thus obtains $\mathbf{x}[0] = [5\ -1\ 2\ 3\ -2]^T$, and calculates $\mathbf{x}[0]^T\mathbf{x}[0] = 43$
- All other nodes calculate their functions by following a similar strategy

Summary

- **Connectivity** of the network characterizes the robustness of linear iterative schemes to malicious behavior

  - If the **connectivity is 2f+1** or more, each node has a **checking/correction scheme** that allows it to **eliminate the effects** of up to f malicious nodes, and calculate any **arbitrary function** of all node values

  - Linear iteration can be run with **almost any choice of weights**, for **at most N time-steps**

Future Work and Open Problems

1. Extend our framework to handle **private** or **secure** function calculation
   - Can the weight matrix be chosen so that some nodes in the network **cannot** calculate certain functions?
2. Extension to **finite fields**
   - What if nodes can only perform finite field arithmetic, and transmit elements from a finite set?
   - Connections to **network coding**
3. Dealing with general **time-varying networks**
   - Small changes can be handled by treating them as faults, but how to deal with large/frequent changes?
4. **Complexity** (communication/computation/time/…) of linear strategy vs. other distributed function calculation schemes