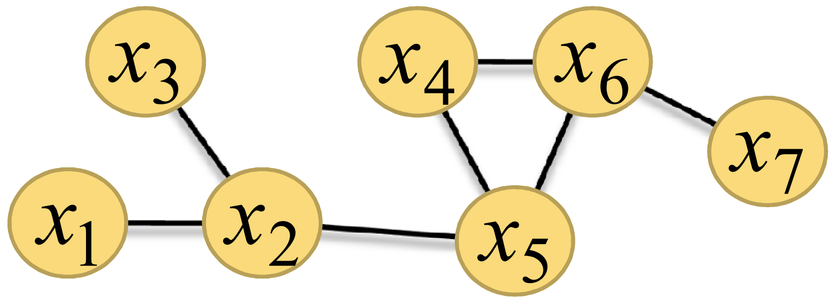
Distributed Function Calculation via Linear Iterations in the Presence of Malicious Agents

Part I: Attacking the Network

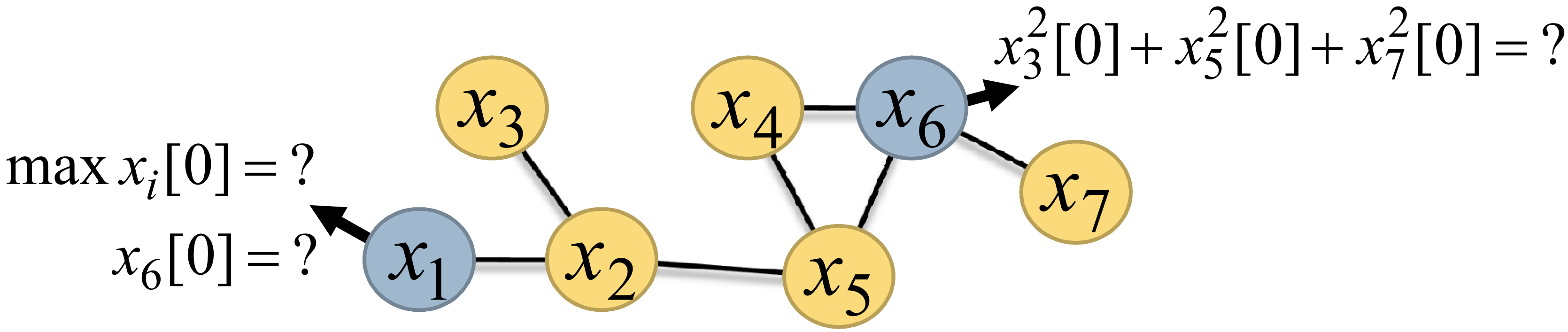
Shreyas Sundaram and Christoforos N. Hadjicostis
Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

Problem Formulation



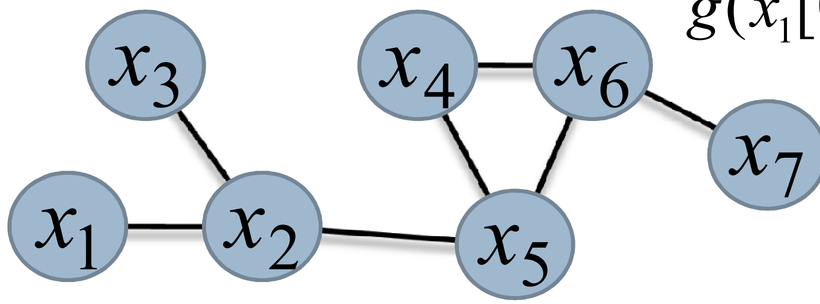
- Consider a network with nodes $\{x_1, x_2, \ldots, x_N\}$
  - e.g., sensors, robots, unmanned vehicles, computers, etc.
- Each node x_i has some initial value $x_i[0]$
  - e.g., temperature measurement, position, vote, etc.
- **Objective:** Some nodes must calculate certain functions of initial values

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

Previous Work

- Distributed function calculation schemes have been well studied over past few decades
  - Issues of communication complexity, computational complexity, time complexity, fault tolerance, …

- Many excellent books on this topic
  - **Dissemination of Information in Communication Networks**, Hromkovic et. al., 2005
  - **Communication Complexity**, Kushilevitz and Nisan, 1997
  - **Distributed Algorithms**, Lynch, 1997
  - **Elements of Distributed Computing**, Garg, 2002
  - **Parallel and Distributed Computation**, Bertsekas and Tsitsiklis, 1997
  - …

Linear Iterative Schemes

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

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

- Linear iterative schemes extensively studied in control literature in order to obtain **asymptotic consensus**
  - For all i, $\lim_{k\to\infty} x_i[k] = g(x_1[0], \cdots, x_N[0])$

- Results derived using eigenvalue/eigenvector analysis

- Survey papers:
  - **Olfati-Saber, Fax & Murray**, *Proc. IEEE*, 2007
  - **Ren, Beard & Atkins**, *Proc. ACC*, 2005

Finite-Time Distributed Function Calculation via Linear Iterations

- Linear iterative strategy allows distributed calculation of **arbitrary functions** in **finite-time**

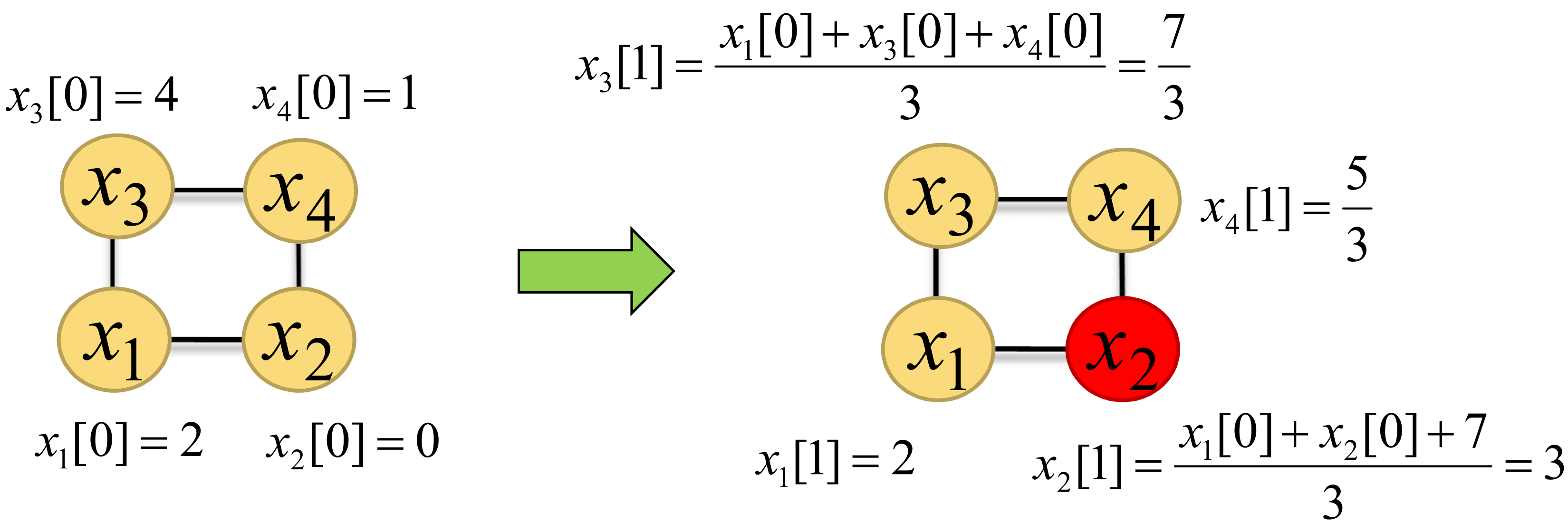
- **Theorem ([1]):** If the network is strongly connected, then for **almost any choice of weights**, each node x_i can calculate **any arbitrary function** of the initial values after running the linear iteration for **at most N-deg(i) time-steps**.
  - "**Almost any**": For all but a set of measure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

[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

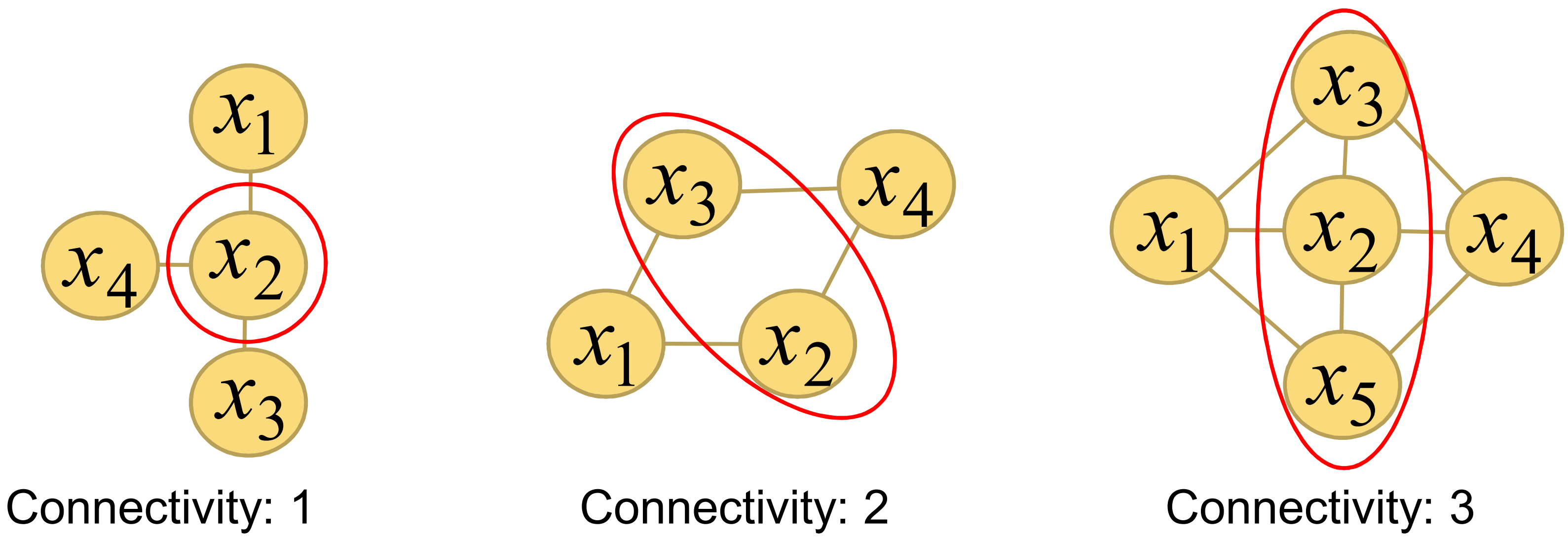
An Example of Malicious Behavior



- 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



- **Menger's Theorem:** If a graph has connectivity κ, there is a set of κ nodes that disconnects the graph
  - This set of nodes is called a **vertex cut**

Main Result

- We show:
  - 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
- In Part II, we prove the converse result:
  - If network connectivity is **2f+1 or more**, linear iteration is **robust to f or fewer malicious nodes**
  - **Any node** can calculate **any function** via linear iteration

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 = $\{x_{i_1}, x_{i_2}, \ldots, x_{i_f}\}$ be set of faulty/malicious nodes

- Update equation for entire system:

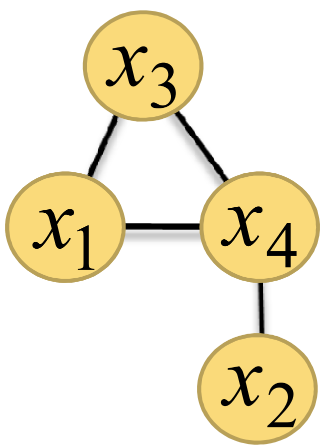
$$\underbrace{\begin{bmatrix} x_1[k+1] \\ \vdots \\ x_N[k+1] \end{bmatrix}}_{\mathbf{x}[k+1]} = \underbrace{\begin{bmatrix} w_{11} & \cdots & w_{1N} \\ \vdots & \ddots & \vdots \\ w_{N1} & \cdots & w_{NN} \end{bmatrix}}_{\mathbf{W}} \underbrace{\begin{bmatrix} x_1[k] \\ \vdots \\ x_N[k] \end{bmatrix}}_{\mathbf{x}[k]} + \underbrace{\begin{bmatrix} \mathbf{e}_{i_1} & \mathbf{e}_{i_2} & \cdots & \mathbf{e}_{i_f} \end{bmatrix}}_{\mathbf{B}_S} \underbrace{\begin{bmatrix} u_{i_1}[k] \\ u_{i_2}[k] \\ \vdots \\ u_{i_f}[k] \end{bmatrix}}_{\mathbf{u}_S[k]}$$

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

- Note: the nodes in S can **conspire** to update their values in a **coordinated** manner!

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

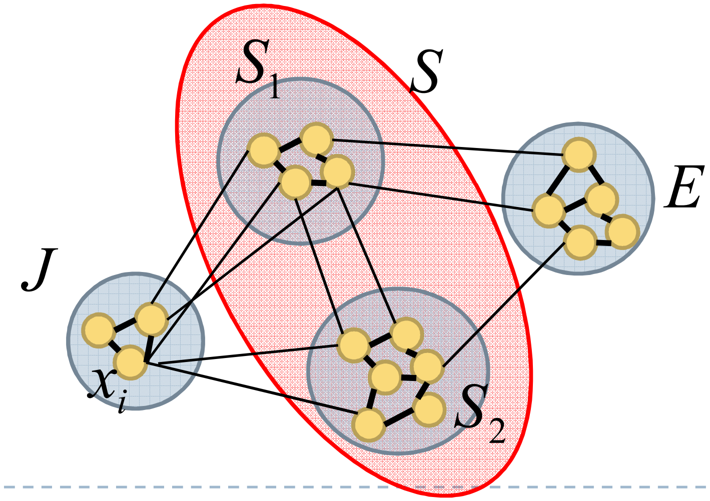


For node x_3:

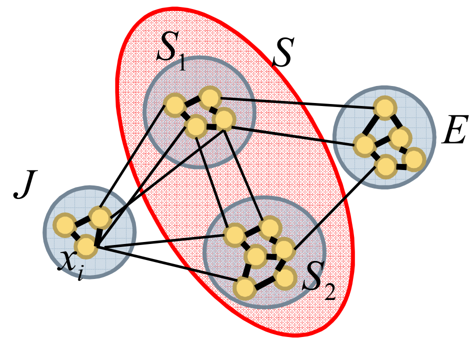
$$\mathbf{y}_3[k] = \begin{bmatrix} x_1[k] \\ x_3[k] \\ x_4[k] \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{C}_3} \mathbf{x}[k]$$

Partitioning the Distributed System

- Let S_1 and S_2 be disjoint sets of nodes, such that $S = S_1 \bigcup S_2$ is a vertex cut
  - E: set of nodes that do not have a path to node x_i when the nodes in S are removed
  - J: set of nodes that have a path to node x_i when the nodes in S are removed (note: $x_i \in J$)
- **Note:** All information about nodes in E must go through either S_1 or S_2 in order to reach x_i



Partitioning the Linear Iterative Model



- Assume (without loss of generality) that nodes are ordered as $\mathbf{x}[k] = \begin{bmatrix} \mathbf{x}_J^T[k] & \mathbf{x}_{S_1}^T[k] & \mathbf{x}_{S_2}^T[k] & \mathbf{x}_E^T[k] \end{bmatrix}^T$
- Since no node in J has an edge from a node in E, weight matrix for linear iteration has the form

$$\mathbf{W} = \begin{bmatrix} W_{11} & W_{12} & W_{13} & 0 \\ W_{21} & W_{22} & W_{23} & W_{24} \\ W_{31} & W_{32} & W_{33} & W_{34} \\ W_{41} & W_{42} & W_{43} & W_{44} \end{bmatrix}$$

Disrupting the System

- Let **a** and **b** be two different vectors

Scenario 1:
$\mathbf{x}_E[0]$ = **a** and nodes in S_1 maliciously update their values with additive error

$$\mathbf{u}_{S_1}[k] = W_{24} W_{44}^{k} (\mathbf{b} - \mathbf{a})$$

Scenario 2:
$\mathbf{x}_E[0]$ = **b** and nodes in S_2 maliciously update their values with additive error

$$\mathbf{u}_{S_2}[k] = W_{34} W_{44}^{k} (\mathbf{a} - \mathbf{b})$$

- We show values seen by node x_i at each time-step under **either** scenario are **exactly the same**
- Node x_i **cannot distinguish** malicious behavior by nodes in S_1 from malicious behavior by nodes in S_2

Sketch of Proof

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

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

- **Theorem:** Columns of $O_{i,L}$ corresponding to nodes in E can be written as a linear combination of the columns in $M^{S_1}_{i,L}$ and $M^{S_2}_{i,L}$:

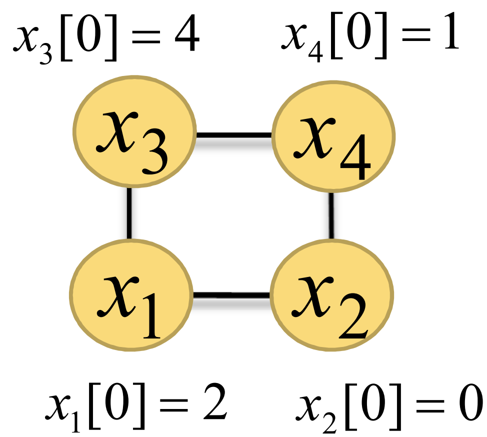
$$O_{i,L}\begin{bmatrix} 0 \\ 0 \\ 0 \\ I_{|E|} \end{bmatrix} = M_{i,L}^{S_1}\begin{bmatrix} W_{24} \\ W_{24}W_{44} \\ \vdots \\ W_{24}W_{44}^{L-1} \end{bmatrix} + M_{i,L}^{S_2}\begin{bmatrix} W_{34} \\ W_{34}W_{44} \\ \vdots \\ W_{34}W_{44}^{L-1} \end{bmatrix}$$

- x_i can be confused if nodes in S_1 or S_2 choose their updates properly

Disruption with f Nodes in Networks with Connectivity 2f or Less

- Only requirement for node x_i to be confused was that S_1 and S_2 together form a **vertex cut**
- If graph has connectivity 2f or less, can find sets S_1 and S_2 so that each set has **at most f nodes**
- Thus, if graph has **connectivity 2f or less**, f malicious nodes can update their values so that some nodes **cannot** calculate a function of other values in the system

Example



$$W = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

- Connectivity of above network is 2
  - Linear iteration can be disrupted by **one** malicious node
- Consider vertex cut $\{x_2, x_3\}$
  - Malicious behavior by x_2 can be **confused** with malicious behavior by x_3

Example (cont.)

- **Partition** the weight matrix:

$$W = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & W_{13} & 0 \\ W_{21} & W_{22} & W_{23} & W_{24} \\ W_{31} & W_{32} & W_{33} & W_{34} \\ W_{41} & W_{42} & W_{43} & W_{44} \end{bmatrix}$$

- Node x_2 wants to pretend that $x_4[0] = 7$ (actual value is $x_4[0] = 1$)
- At each time-step, node x_2 commits an **additive error** of $u_2[k] = W_{24}(W_{44})^k(7-1) = 2(3)^{-k}$:

$$x_2[k+1] = \frac{1}{3}x_1[k] + \frac{1}{3}x_2[k] + \frac{1}{3}x_4[k] + 2(3)^{-k}$$

Example (cont.)

- Values seen by node x_1 during linear iteration:

$$\mathbf{y}_1[0] = \begin{bmatrix} 2 \\ 0 \\ 4 \end{bmatrix}, \quad \mathbf{y}_1[1] = \begin{bmatrix} 2 \\ 3 \\ 2.333 \end{bmatrix}, \quad \mathbf{y}_1[2] = \begin{bmatrix} 2.444 \\ 2.889 \\ 2 \end{bmatrix}, \quad \cdots$$

- These are **same values** seen by node x_1 if $x_4[0] = 7$, and node x_3 maliciously updates values as

$$x_3[k+1] = \frac{1}{3}x_1[k] + \frac{1}{3}x_3[k] + \frac{1}{3}x_4[k] - 2(3)^{-k}$$

- Node x_1 **cannot determine** if $x_4[0] = 1$ or $x_4[0] = 7$
  - Independent of the number of iterations!

Summary

- The **connectivity** of the network characterizes the robustness of linear iterative schemes to malicious behavior by subsets of nodes
  - If the connectivity is **2f or less**, f malicious nodes can coordinate to update their values so that some other nodes **cannot** calculate certain functions

- **In Part II**: Show that linear iteration is robust to f malicious nodes if network connectivity is 2f+1 or more (for almost any choice of weights)