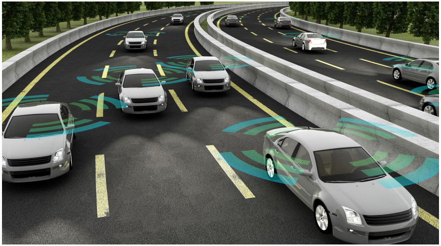
Detecting Network Faults in Multi-Agent Systems Using Graph Connectivity and Passivity

Miel Sharf Daniel Zelazo

Faculty of Aerospace Engineering
Technion - Israel Institute of Technology

“Resilience and Controllability of Large Scale Systems: A Network-Theoretic Approach Workshop”,
58$^{\mathrm{th}}$ IEEE Conference on Decision and Control, December 2019

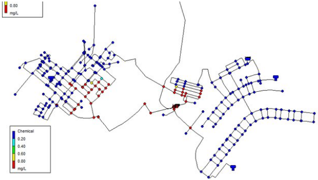
Multi-Agent Systems



Network Faults in Multi-Agent Systems

Network faults can be detrimental to multi-agent systems:


Power networks - Blackouts



Water networks - Lack of clean water, firefighting not operational

Software

Hacker jailed for revenge sewage attacks

Job rejection caused a bit of a stink

By Tony Smith 31 Oct 2001 at 15:55 SHARE ▼

An Australian man was today sent to prison for two years after he was found guilty of hacking into the Maroochy Shire, Queensland computerised waste management system and caused millions of litres of raw sewage to spill out into local parks, rivers and even the grounds of a Hyatt Regency hotel.

Sewage, oil and gas networks - Pollution and health concerns

Network Faults for Multi-Agent Systems

- Network faults switch from $\mathcal{G}$ to a subgraph of $\mathcal{G}$ by removing edges.

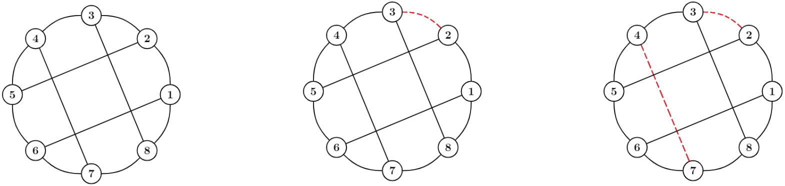
Problem Formulation

Fix a graph $\mathcal{G}$, agents Σ_i, and a desired output $y^\star$. Find networked controllers and a decision algorithm such that:

i) The output of the faultless system converges to $y^\star$, and the algorithm never declares a fault.

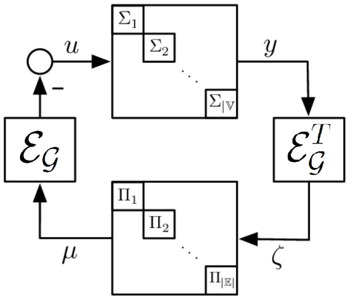
ii) The algorithm declares a fault if at least one edge disappears.

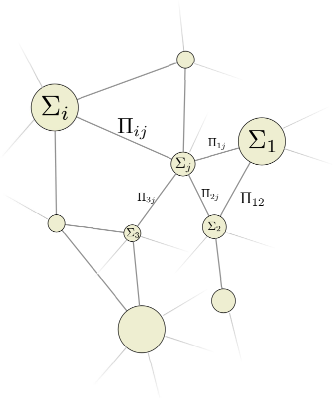
The algorithm is allowed to sample the agents' inputs and/or outputs.



$t = 0$ $\quad$ $t = t_1^\star$ $\quad$ $t = t_2^\star$

Diffusively Coupled Networks





- Σ_i are nonlinear dynamical systems representing the agents.
- Π_e are nonlinear dynamical system representing the edge controllers.
- Can be used to model neural networks, vehicle networks, and networks of oscillators, among others.

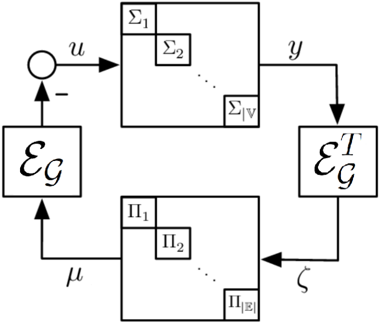
Model Formulation and Analysis

We restrict to a multi-agent systems evolving over some graph $\mathcal{G} = (\mathbb{V}, \mathbb{E})$ of the form:

$$\boxed{\dot{x}_i = f_i(x_i) + q_i(x_i) \sum_{\{i,j\} \in \mathbb{E}} g_{ij}(h_j(x_j) - h_i(x_i))}$$

$$\Sigma_i : \begin{cases} \dot{x}_i = f_i(x_i) + q_i(x_i) u_i \\ y_i = h_i(x_i) \end{cases}$$

$$\Pi_{ij} : \mu_{ij} = g_{ij}(\zeta_{ij})$$



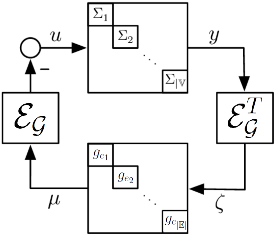
Steady States of Diffusively-Coupled Networks

- Under some passivity assumption on the agents and controllers, the output of the network converges to some steady-state.
- For the closed loop to reach a steady-state, each agent and controller must reach steady-state.

Definition

The collection of all steady-state input-output pairs of a system is called a *steady-state input-output relation*.

- Let k_i be the relations for Σ_i, and let k be the stacked relation.



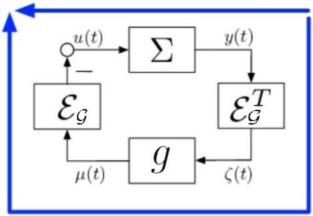
The Steady-State Equation

- If k is the steady-state relation, u is a steady-state input, and y a steady-state output, then:

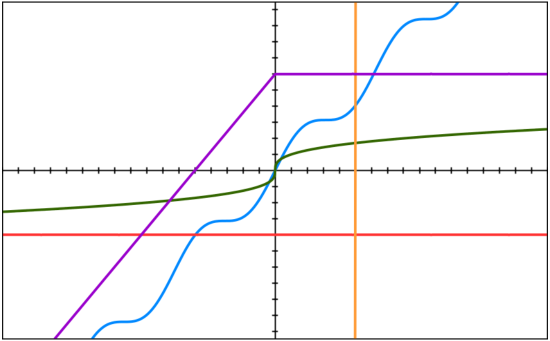
$$k(\mathrm{u}) = \{\mathrm{y} : (\mathrm{u}, \mathrm{y}) \in k\}$$
$$k^{-1}(\mathrm{y}) = \{\mathrm{u} : (\mathrm{u}, \mathrm{y}) \in k\}$$

- For this talk - we assume that k^{-1} is a function.
- Let $\mathrm{u}, \mathrm{y}, \zeta, \mu$ be the steady-state of the closed-loop system. The consistency of the steady-states yields the following equation:

$$0 = k^{-1}(\mathrm{y}) + \mathcal{E}_{\mathcal{G}} g(\mathcal{E}_{\mathcal{G}}^T \mathrm{y})$$



The Role of Maximal Monotonicity



Theorem

Suppose all the relations k_i are maximally monotone, and all g_{ij} are monotone. Then there is a vector y *such that* $0 = k^{-1}(\mathrm{y}) + \mathcal{E}_{\mathcal{G}}\, g(\mathcal{E}_{\mathcal{G}}^T \mathrm{y})$.

- We demand that k_i are maximally monotone and g_{ij} are monotone.

MEIP

The discussion above motivates the following refinement of passivity[1]

Definition (MEIP)

A SISO system is called *(output-strictly) maximal monotone equilibrium-independent passive* (MEIP) if:

i) The system is (output-strictly) passive with respect to any steady-state input-output pair.

ii) The steady-state input-output relation is maximally-monotone.

Many SISO systems are MEIP:

- Port-Hamiltonian systems;
- Reaction-diffusion systems;
- Gradient-descent systems;
- Single integrators.

[1] M. Bürger, D.Zelazo and F. Allgöwer, "Duality and network theory in passivity-based cooperative control", Automatica, vol. 50, no. 8, pp, 2051–2061, 2014.

Analysis Theorem of MEIP Multi-Agent Systems

Recall we are interested in the multi-agent system governed by the following equation:

$$\dot{x}_i = f_i(x_i) + q_i(x_i) \sum_{\{i,j\}\in\mathbb{E}} g_{ij}(h_j(x_j) - h_i(x_i))$$

Theorem (Bürger, Zelazo and Allgöwer, 2014)

Consider the closed loop system, and suppose all agents Σ_i are output-strictly MEIP and g_{ij} are all monotone functions.
Then the signals $u(t), y(t), \zeta(t)$ and $\mu(t)$ converge to constants $\hat{u}, \hat{y}, \hat{\zeta}$ and $\hat{\mu}$, and $\hat{y}$ satisfies $0 = k^{-1}(\hat{y}) + \mathcal{E}_\mathcal{G} g(\mathcal{E}_\mathcal{G}^T \hat{y})$.

Network Faults for Multi-Agent Systems

Back to the synthesis problem.

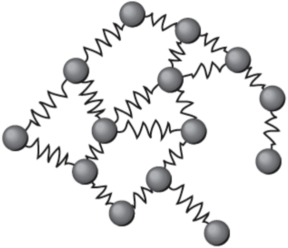
Theorem [S. and Zelazo, 2019, TAC]

Given a desired output $y^\star$, there exists affine monotone functions g_{ij} such that $(\mathcal{G}, \Sigma, g)$ converges to $y^\star$.

- How to assure that $(\mathcal{G}, \Sigma, g)$ can be distinguished from $(\mathcal{H}, \Sigma, g)$ using data, where $\mathcal{H}$ is a subgraph of $\mathcal{G}$?
- Idea: "Asymptotic" fault detection.
    - If the output of the systems converge to different outputs, we can check the steady-state limit of the output of the network, and declare a fault if it is different from $\mathrm{y}^\star$.

Forced equilibria and Asymptotic Differentiation - Intuition

- Consider a network of nodes joined by damped springs.
- Suppose the network is at an equilibrium, so that the total force exerted on each node is zero, but each spring is not in its resting position.
- If we cut any spring, the associated end nodes will have a non-zero force exerted on them. Thus, the new network is no longer at an equilibrium, and will thus converge to some different steady-state.



Forced equilibria and Asymptotic Differentiation

- We try to find a controller forcing the system to converge to $y^\star$, so that the controllers exert a non-zero force at equilibrium.
- Suppose that g_{ij} are affine monotone functions such that $(\mathcal{G}, \Sigma, g)$ converges to $y^\star$. Consider $\tilde{g}_{ij}(x) = g_{ij}(x) + \mathrm{w}_{ij}$ for some constant vector w defined on the edges of $\mathcal{G}$.
- Let $\mathcal{H}$ be a subgraph of $\mathcal{G}$. The steady-state equation for the system $(\mathcal{H}, \Sigma, \tilde{g})$ reads as:

$$\boxed{0 = k^{-1}(\mathrm{y}) + \mathcal{E}_\mathcal{H} g(\mathcal{E}_\mathcal{H}^T \mathrm{y}) + \mathcal{E}_\mathcal{H} P_\mathcal{H} \mathrm{w}.}$$

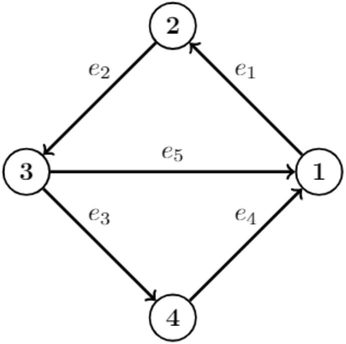
  where $P_\mathcal{H}$ is a projection operator, nullifying all entries corresponding to edges in $\mathcal{G} \setminus \mathcal{H}$.
- Note that if $\mathrm{w} \in \ker(\mathcal{E}_\mathcal{G})$, then $(\mathcal{G}, \Sigma, \tilde{g})$ converges to $\mathrm{y}^\star$.

The Cycle Space $\ker(\mathcal{E}_\mathcal{G})$

- Any element of $\ker(\mathcal{E}_\mathcal{G})$ is a linear combination of vectors, each corresponding to a cycle in $\mathcal{G}$.

$$\mathcal{E}_\mathcal{G} = \begin{bmatrix} 1 & 0 & 0 & -1 & -1 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & -1 & 1 & 0 \end{bmatrix}$$

$$\ker(\mathcal{E}_\mathcal{G}) = \operatorname{span}\left\{ \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ -1 \end{bmatrix} \right\}$$



Indication Vectors

$$0 = k^{-1}(\mathrm{y}) + \mathcal{E}_{\mathcal{H}} g(\mathcal{E}_{\mathcal{H}}^T \mathrm{y}) + \mathcal{E}_{\mathcal{H}} P_{\mathcal{H}} \mathrm{w}.$$

- Let $y_{\mathcal{H}} = y_{\mathcal{H}}(\mathrm{w})$ be the solution to the equation for the subgraph $\mathcal{H}$.

Definition (Indication Vectors)

The vector w is a *$\mathcal{G}$-indication vector* if $y_{\mathcal{H}} \neq y_{\mathcal{G}}$ for all $\mathcal{H} \subset \mathcal{G}$.

Theorem (Constructing Indication Vectors)

Suppose that $\mathcal{G}$ is "connected enough". The collection of vectors $\mathrm{w} \in \ker(\mathcal{E}_{\mathcal{G}})$ which are not $\mathcal{G}$-indication vectors is a zero-measure set.

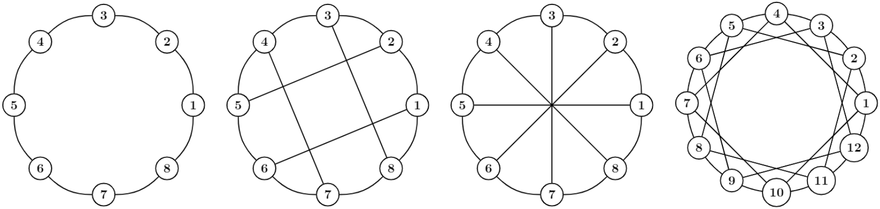
- For a spring network, if the resting positions are chosen randomly, the chance that a spring will be at its resting position in $y^\star$ is zero.

What is "connected enough"?

r-Connected Graphs

Definition (r-Connected Graph)

Let $r \geq 1$. A graph $\mathcal{G}$ is *r-connected* if $\mathcal{G}$ is connected, and at least r vertices must be removed from $\mathcal{G}$ before it becomes disconnected.

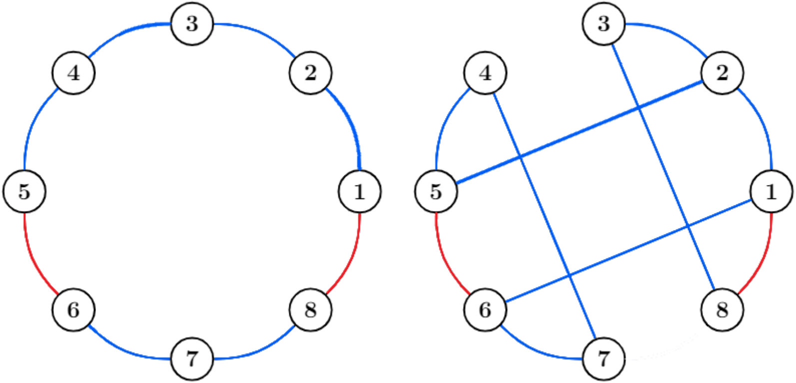


- Cycles are 2-connected graphs on n nodes with n edges.
- In general, there are r-connected graphs on n nodes with $O(rn)$ edges.

2-Connected Graphs

Key Lemma

If the edges of a 2-connected graph $\mathcal{G}$ are colored in red and blue, then there exists a simple cycle with edges of both colors.



Asymptotic Fault Detection

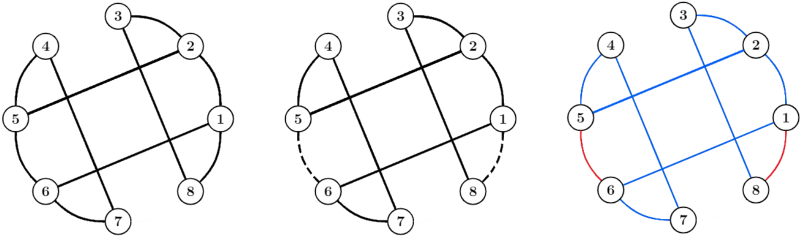
Theorem (Asymptotic Fault Detection)

Let $\mathcal{G}$ *be a 2-connected graph,* $\{\Sigma_i\}$ *be output-strictly MEIP agents, let* $y^\star$ *be any desired output. Suppose that* g *is an affine monotone (or any other) non-linearity such that* $(\mathcal{G}, \Sigma, g)$ *converges to* $y^\star$*.*
Let $\mathbb{P}$ *be any absolutely continuous probability measure (e.g., Gaussian) on* $\ker(\mathcal{E}_\mathcal{G})$*. Suppose that* w *is sampled according to* $\mathbb{P}$*, and let* $\tilde{g}_{ij}(x) = g_{ij}(x) + \mathrm{w}_{ij}$*.*

- i) *If no network faults occur, the network* $(\mathcal{G}, \Sigma, \tilde{g})$ *converges to* $\mathrm{y}^\star$*.*
- ii) *With probability* 1*, if faults do occur, the network with* $\tilde{g}$ *converges to a limit different than* $\mathrm{y}^\star$*.*

Proof Sketch

- Any subgraph $\mathcal{H} \subset \mathcal{G}$ induces a coloring of $\mathcal{G}$:



$\mathcal{G}$ $\mathcal{H} \subset \mathcal{G}$ Coloring induced by $\mathcal{H}$.

- By lemma, we have a simple cycle with edges from both $\mathcal{H}$ and $\mathcal{G} \setminus \mathcal{H}$.
- There's a vector $v \in \ker(\mathcal{E}_{\mathcal{G}})$ corresponding to the simple cycle.
- Consider the set M of vectors w such that $y_{\mathcal{H}}(\mathrm{w}) = y^{\star}$.
- If $\mathrm{w}^T v \neq 0$, then M is a submanifold of $\ker(\mathcal{E}_{\mathcal{G}})$ of smaller dimension near w (implicit function theorem). Thus $\mathbb{P}(M) = 0$.

Choosing Indication Vectors

How to choose a random vector in $\ker(\mathcal{E}_{\mathcal{G}})$?

- Take a basis $v_1, \cdots, v_\ell$ to $\ker(\mathcal{E}_{\mathcal{G}})$.
- Take ℓ i.i.d. Gaussian random variables $\alpha_1, \cdots, \alpha_\ell$.
- Define $\mathrm{w} = \alpha_1 v_1 + \cdots + \alpha_\ell v_\ell$.

How to choose a random vector in $\ker(\mathcal{E}_{\mathcal{G}})$ distributedly?

- Assume all nodes know the graph $\mathcal{G}$.
- Each node randomly chooses some vector $\mathrm{w}^{(i)} \in \ker(\mathcal{E}_{\mathcal{G}})$. Run a finite-time consensus protocol with these vectors as initial conditions.
- The resulting consensus value is again a random vector in $\ker(\mathcal{E}_{\mathcal{G}})$, and thus an indication vector.

Toward Real-Time Fault Detection

- We built an asymptotic fault detection scheme!
  - We take a solution to the synthesis problem for $\mathcal{G}$, and add a random vector in $\ker(\mathcal{E}_\mathcal{G})$.
  - Assures that for each subgraph $\mathcal{H} \subseteq \mathcal{G}$, $(\mathcal{G}, \Sigma, g)$ has a different limit than $(\mathcal{H}, \Sigma, g)$.
- How to adapt this asymptotic fault detection framework to a real-time fault detection algorithm?
  - *Idea* - use passivity of agents and monotonicity of g_{ij} to get convergence rate estimates.

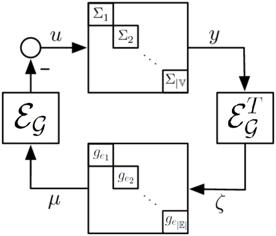
Toward Real-Time Fault Detection

- Let $(\mathrm{u}^\star, \mathrm{y}^\star, \zeta^\star, \mu^\star)$ be the closed-loop steady-state of $(\mathcal{G}, \Sigma, g)$.
- By output-strict passivity, each agent has a storage function S_i and some positive number ρ_i such that:

$$\frac{d}{dt} S_i(x_i) \leq (u_i - \mathrm{u}_i^\star)(y_i - \mathrm{y}_i^\star) - \rho_i (y_i - \mathrm{y}_i^\star)^2.$$

- By monotonicity of g_{ij}:

$$0 \leq (\zeta_{ij} - \zeta_{ij}^\star)(\mu_{ij} - \mu_{ij}^\star).$$

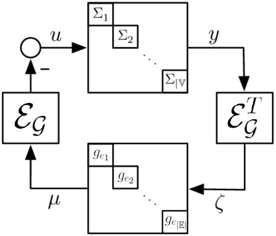


Toward Real-Time Fault Detection

- Recall that $u = -\mathcal{E}_{\mathcal{G}}\mu, \zeta = \mathcal{E}_{\mathcal{G}}^T y$, and the same holds for equilibria.
- By summing over all nodes and edges, we get:

$$\boxed{\frac{d}{dt}S(x) \triangleq \frac{d}{dt}\left(\sum_{i=1}^{n} S_i(x_i)\right) \leq -\sum_{i=1}^{n} \rho_i (y_i - \mathrm{y}_i^\star)^2}$$

- If the inequality always holds, we must have $y(t) \to \mathrm{y}^\star$, as $S_i \geq 0$.
- **We need to discretize the inequality in order to verify it from samples**.



Discretizing - Option 1

- By integrating from time T_0 to $T_1 = T_0 + \Delta T$, we get:

$$\boxed{S(x(T_1)) - S(x(T_0)) \leq -\sum_{i=1}^{n} \rho_i (y_i(T_0) - \mathrm{y}_i^\star)^2 \Delta T + M \Delta T^2.}$$

  where $M\Delta T^2$ is a term added to compensate for the error in $\int_{T_0}^{T_1} \sum_i \rho_i (y_i - \mathrm{y}_i^\star)^2 dt \approx \sum_i \rho_i (y_i(T_0) - \mathrm{y}_i^\star)^2 \Delta T$. It must be added to avoid declaring a nonexistent fault.
- If ΔT is small enough, the quadratic term is very small, and faults can be identified easily.
  - If the limit of the output of the system is not $\mathrm{y}^\star$, the inequality will be violated relatively quickly.
- Verification can be distributed - enough to check that similar inequalities hold at each node and on each edge separately.

Discretizing - Option 2

- We recall that y_i is a function of the state x_i. One can find some monotone function Ω such that:

$$\boxed{\frac{d}{dt}S(x) \leq -\sum_{i=1}^{n} \rho_i (y_i - \mathrm{y}_i^\star)^2 \leq -\Omega(S(x))}$$

- By integrating from time T_0 to T_1 and using monotonicity, we get:

$$S(x(T_1)) - S(x(T_0)) \leq -\Omega(S(x(T_0)))(T_1 - T_0)$$

- Does not require a high sampling frequency.
- Verification cannot be distributed easily - Ω can be nonlinear.

Discretizing - Option 2 (Example)

- Consider the LTI agents $\dot{x}_i = -x_i + u_i;\ y_i = x_i$ with transfer function $G_i(s) = \frac{1}{s+1}$. We focus on the equilibrium $\mathrm{u} = 0, \mathrm{y} = 0$.
- These are output-strict passive with $\rho_i = 1$ and storage function $S_i(x_i) = \frac{1}{2}x_i^2$. Thus $S(x) = \frac{1}{2}x^T x$.
- We want to find a positive monotone function Ω such that:

$$-\sum_{i=1}^{n} x_i^2 = -\sum_{i=1}^{n} \rho_i (y_i - \mathrm{y}_i^\star)^2 \leq -\Omega(S(x)) = -\Omega\left(\frac{1}{2}x^T x\right)$$

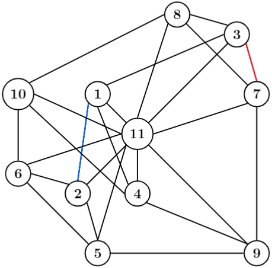
- We can choose $\Omega(\theta) = 2\theta$.
- In general, if the agents are LTI, then Ω is linear.

Example - Fault Detection in Vehicle Networks

- Consider a network of 11 vehicles trying to coordinate their velocity.
- The dynamics of the velocity x_i of the i-th agent is given by

$$\dot{x}_i = \kappa_i(-x_i + V_0^i + V_1^i u_i)$$

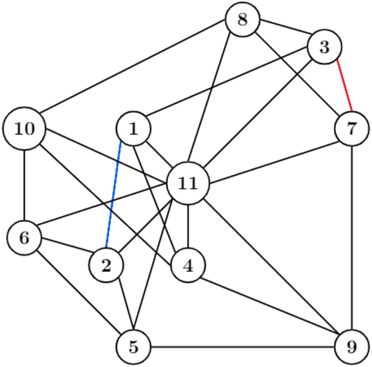
  where $\kappa_i > 0$ is an internal gain, V_0^i is its preferred velocity, and V_1^i is the "sensitivity" to other vehicles.
- We take $g(\zeta_e) = \tanh(\zeta_e)$, and add an indication vector w.



Agents	$y^\star$
$1, 4, 7, 10$	$60_{\mathrm{km/h}}$
$2, 5, 8, 11$	$70_{\mathrm{km/h}}$
$3, 6, 9$	$50_{\mathrm{km/h}}$

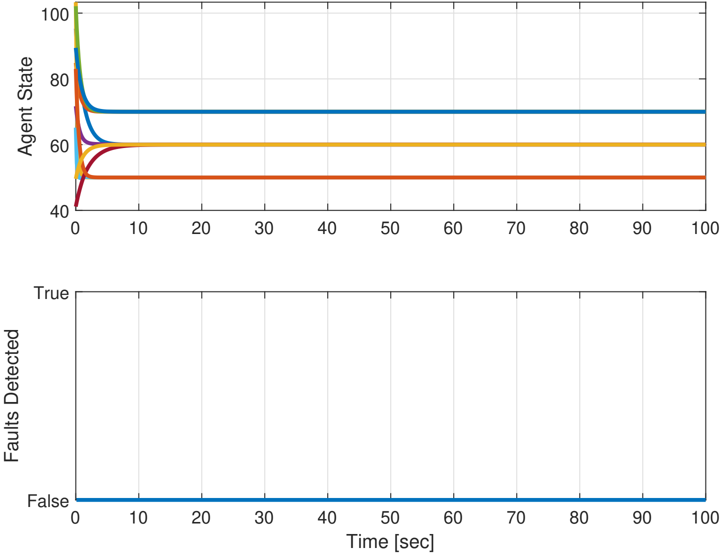
Example - Fault Detection in Vehicle Networks

- In each run, we choose the parameters κ_i, V_0^i, V_1^i log-uniformly within appropriate ranges.
- We choose $\mathrm{w} \sim \mathcal{N}(0,1)$ and $x(0) \sim \mathcal{N}(70_{\mathrm{km/h}}, 20_{\mathrm{km/h}})$.
- The agents' output is sampled at 10_{Hz}, and use the second option for the validation algorithm (using Ω).



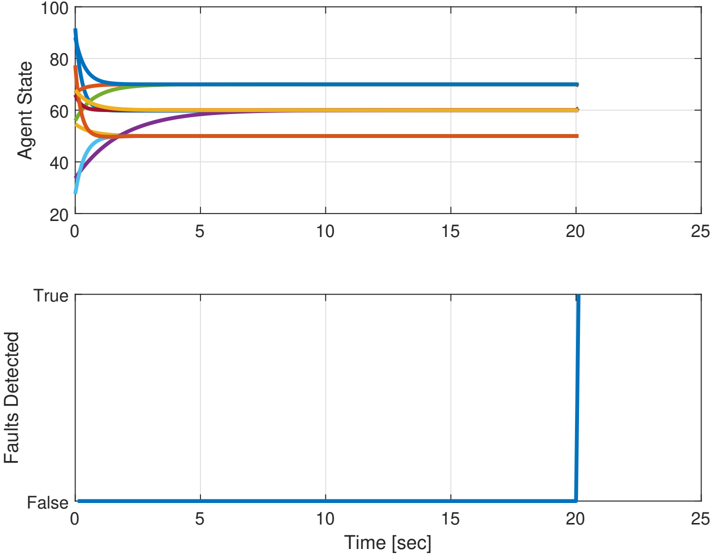
Example - Fault Detection in Vehicle Networks

No Faults:



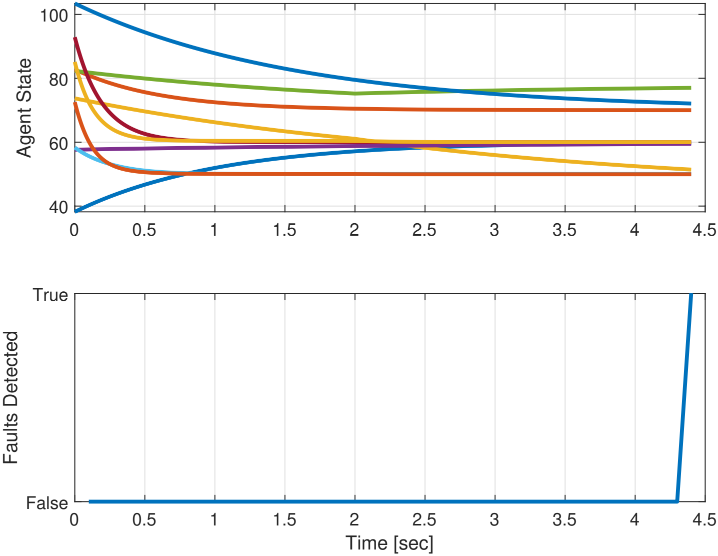
Example - Fault Detection in Vehicle Networks

The edge $\{1, 2\}$ becomes faulty at time 20:



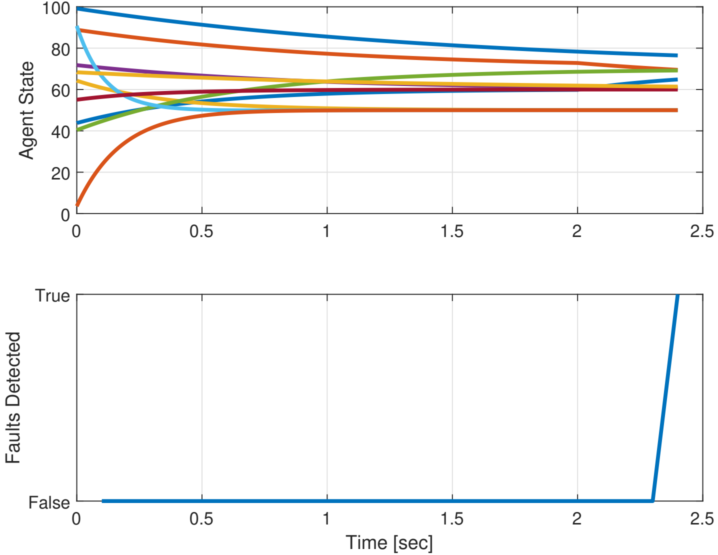
Example - Fault Detection in Vehicle Networks

The edge $\{1, 2\}$ becomes faulty at time 2:



Example - Fault Detection in Vehicle Networks

The edge $\{3, 7\}$ becomes faulty at time 2:



Summary

- Passivity theory is a powerful tool for studying nonlinear multi-agent systems.
- Indication vectors are easy to construct and allow one to asymptotically differentiate between functioning and malfunctioning networks.
- Passivity allows to turn asymptotic differentiation to real-time fault detection.

Possible Extensions

- **Fault Isolation**.
  - Can use a similar framework - if a graph is r-connected for $r > 2$, then one can isolate up to $r - 2$ faults. Appears in preprint.
- **Data-Driven FDI algorithms**.
  - Only step that requires a model is building g so that the output of the faultless system converges to $y^\star$.
- **Framework for passive-short agents**.
  - Use local feedback and/or network feedback to passivize the agents.
- **More delicate graph-theoretical properties**.
  - Can we detect/isolate a bounded number of faults for non-2-connected graphs?

Acknowledgments

German-Israeli Foundation for Scientific Research and Development

IAAC Israeli Association for Automatic Control

The talk is based of the preprint:
Sharf, M., & Zelazo, D. (2019). **A Data-Driven and Model-Based Approach to Fault Detection and Isolation in Networked Systems**. arXiv preprint arXiv:1908.03588.