Graph theory in Systems and Control

Mehran Mesbahi

Department of Aeronautics & Astronautics

University of Washington

CDC 2018
Miami Beach, Florida, December 2018

Networked systems

Why a tutorial on graph theory in systems and control?

- networks are all around us
- this trend will continue, e.g., internet of things, next generation mobility
- networked robotics and aerospace systems will play an ever increasing role in the society at all levels
- system and control theory can play a significant role in this new era of networked systems ...
- however, we need to start blending in combinatorial/discrete mathematics in mainstream control theory even more ...

This tutorial is framed around this objective ...

a Few Immediate Observations

- networked systems are coupled through information exchange
- inter-agent information exchange is through sensing and communication
- the collective dynamics is a function of "agent" dynamics and the information-induced coupling
- we can synthesize collective behavior by making the control action on each agent a function of the information available to the agent (sense, communicated, etc.)

a powerful abstraction for encoding "interactions" in a network
is that of a graph

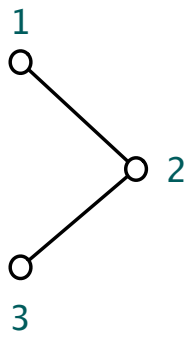
Graph Abstraction

- a finite, undirected, simple graph, or a graph for short, is built upon a finite set of "nodes" or vertex set $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$
- the edge set is a subset of the two-element subsets of $\mathcal{V}$, i.e., $\mathcal{E} \subseteq [\mathcal{V}]^2$
- the graph is then specified by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

for example, we can have $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where

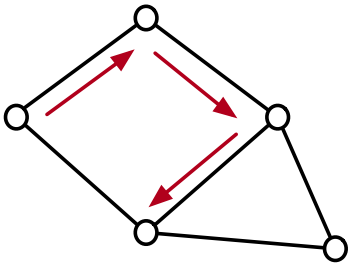
$$\mathcal{V} = \{1,2,3\} \quad \text{and} \quad \mathcal{E} = \{\{1,2\},\{2,3\}\}$$

a simpler representation
however would be

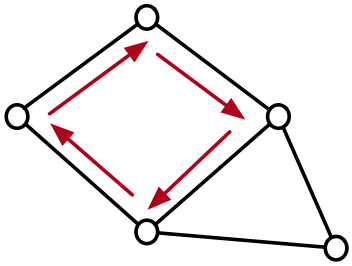


Some natural constructs based on the correspondence between set theoretic and graph-theoretic representation can now be defined– examples: paths, walks, cycles, etc.

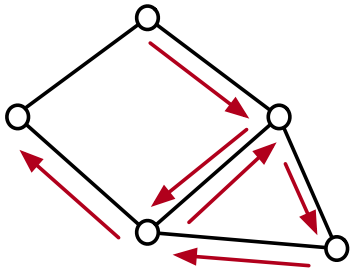
Simple Constructs on Graphs



a path



a cycle

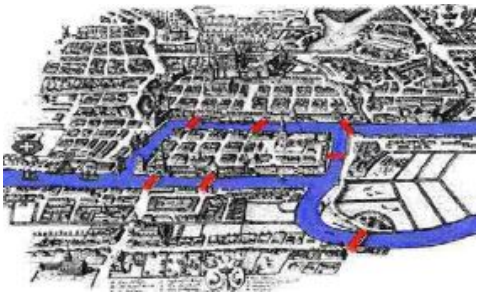


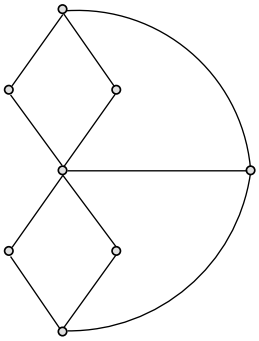
a walk

graphs can be used in general to encode relations between objects, e.g., existence of communication or sensing links, routes, etc.

Birth of Graph Theory

bridges of Konigsberg and Euler's abstraction:





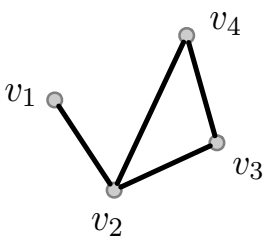
this is an important step, as it stripes away all particular details related to the Konigsberg bridges that are not relevant to the problem at hand! so now we have a graph! what are we looking for now? We want to find out if there is a closed walk traversing all edges of the graph exactly once. If such a walk exists we call the graph Eulerian.

Theorem

A connected graph $\mathcal{G}$ is Eulerian if and if only every vertex has an even degree.

Graphs and Matrices

As we aim to embed graph/networks in dynamic systems, it is natural to work with linear algebraic representation. For example, a graph can be represented as,



$$A(\mathcal{G}) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

the adjacency matrix for the n-node graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is the $n \times n$ matrix:

$$[A(\mathcal{G})]_{ij} = \begin{cases} 1 & \text{if } \ v_i v_j \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Degree Matrix and the Laplacian

note that the adjacency for the graph is symmetric by construction
there are other matrices associated with the graph, for example, let $d(v)$ be the number of neighbors of vertex v (its degree) and define the degree matrix as,

$$\Delta(\mathcal{G}) = \begin{pmatrix} d(v_1) & 0 & \cdots & 0 \\ 0 & d(v_2) & \cdots & 0 \\ \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d(v_n) \end{pmatrix}$$

note that the adjacency and the degree matrices are both square, say, $n \times n$, where n is the number of nodes

Another useful matrix representation is the Laplacian:

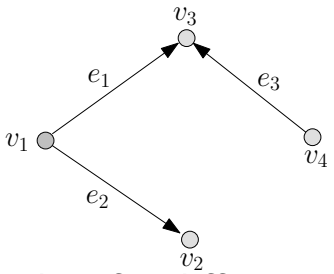
$$L(\mathcal{G}) = \Delta(\mathcal{G}) - A(\mathcal{G})$$

graph Laplacian has been very popular in multiagent networks!

Incidence Matrix

Yet another matrix representation can in fact capture the orientation of the edge as well: suppose the graph has n nodes and m edges: the $n \times m$ *incidence matrix* $E(\mathcal{G})$ is defined as

$$E(\mathcal{G}) = [E_{ij}], \quad \text{where} \quad E_{ij} = \begin{cases} -1 & \text{if } v_i \text{ is the tail of } e_j, \\ 1 & \text{if } v_i \text{ is the head of } e_j, \\ 0 & \text{otherwise.} \end{cases}$$



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

note that for different orientations on the edges we get a different incidence matrix! (same dimension though!)

Let us see what happens when we consider $E(\mathcal{G})E(\mathcal{G})^T$ for some arbitrary orientation. First notice that the resulting matrix will be $n \times n$.

Incidence and Laplacian

A compact formula for matrix multiplication is of course:

$$[AB]_{ij} = \sum_k A_{ik} B_{kj}$$

$$[E(\mathcal{G})E(\mathcal{G})^T]_{ij} = \sum_k E(\mathcal{G})_{ik} E(\mathcal{G})_{jk}$$

which is -1 when i and j are incident on the same edge k, that is if they are neighbors! Moreover,

$$[E(\mathcal{G})E(\mathcal{G})^T]_{ii} = \sum_k E(\mathcal{G})_{ik} E(\mathcal{G})_{ik}$$

counts the number of edges incident on node i, i.e., its degree! so guess what:

$$L(\mathcal{G}) = E(\mathcal{G})E(\mathcal{G})^T$$

independent of the orientation that you have given to the incidence matrix! This also shows that $L(\mathcal{G})$ is positive semi-definite, since for all $x \in \mathbf{R}^n$:

$$x^T L(\mathcal{G}) x = x^T E(\mathcal{G})E(\mathcal{G})^T x = \|E(\mathcal{G})^T x\|^2 \geq 0$$

which means that not only are the eigenvalues of the Laplacian real numbers (as the Laplacian is symmetric) but also non-negative

Spectra of the Graph Laplacian

For Laplacian, we can order the eigenvalues as follows,

$$0 \leq \lambda_1(\mathcal{G}) \leq \lambda_2(\mathcal{G}) \leq \ldots \lambda_n(\mathcal{G});$$

in this case, λ_k refers to the kth smallest eigenvalue of the (graph) Laplacian ...

- By construction, $L(\mathcal{G})\mathbf{1} = 0$ for any graph (why?). So $\lambda_1(\mathcal{G}) = 0$.
- A natural question (with many consequences) is whether $\lambda_2(\mathcal{G}) > 0$?
- In other words, we need to characterize the null space of $L(\mathcal{G})$:

$$\mathcal{N}(L(\mathcal{G})) = \{z \in \mathbf{R}^n \,|\, L(\mathcal{G})z = 0\}$$

What are the vectors in $\mathcal{N}(L(\mathcal{G}))$ except the subspace generated by $\mathbf{1}$, namely,

$$\mathcal{A} = \{x \,|\, x = \alpha\mathbf{1},\ \alpha \in \mathbf{R}\}$$

Null Space of the Laplacian

in order to answer this question, notice that if $z \in \mathcal{N}(L(\mathcal{G}))$, then

$$L(\mathcal{G})z = E(\mathcal{G})E(\mathcal{G})^T z = 0$$

that is,

$$z^T E(\mathcal{G})E(\mathcal{G})^T z = 0$$

or $\|E(\mathcal{G})^T z\|^2 = 0$ or $E(\mathcal{G})^T z = 0$ or $z^T E(\mathcal{G}) = 0$. This means that if $ij \in E$, then $z_i = z_j$; so if the graph is connected,

$$z_1 = z_2 = \ldots = z_n$$

that is $z = \alpha \mathbf{1}$ for some α! And in fact, if we think of z as

$$z : \mathcal{V}(\mathcal{G}) \to \mathbf{R}^n$$

then z is constant on each (connected) component of $\mathcal{G}$. What that means is that for each component we get one extra dimension for the null space of $L(\mathcal{G})$.

Lemma

Let $\mathcal{G}$ have c connected components (when $c = 1$ the graph is connected). Then **rank** $L(\mathcal{G})$ is $n - c$.

Rank, λ_2, and Connectivity

and in fact, **rank** $L(\mathcal{G}) = n-1$ if and only if $\mathcal{G}$ is connected! this is our first encounter with how the "linear algebra" of the Laplacian tells us something about the structure of the graph.
another way to say the same thing is that

$$\mathcal{G} \text{ is connected if and only if } \lambda_2(\mathcal{G}) > 0$$

a natural question now is whether more positive λ_2 captures some qualitative notion of "more" connectivity? For example, we can define the node connectivity of $\mathcal{G}$, denoted by $\kappa_0(\mathcal{G})$ as the minimum number of nodes that needs to be removed from the graph before the graph becomes disconnected.

Courant-Fisher to the rescue:

$$\lambda_2(\mathcal{G}) = \min_{x \perp \mathbf{1}, \|x\|=1} x^\top L(\mathcal{G}) x$$

So this means that

$$\lambda_2(\mathcal{G}) \leq x^\top L(\mathcal{G}) x \quad \text{for all } x \perp \mathbf{1}, \|x\| = 1$$

Structure vs. Spectra

Let us consider removing $S \subset \mathcal{V}$ (subset of nodes) from the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$; we denote the Laplacian of this new graph as $L(\mathcal{G} \backslash S)$.

Let y be the normalized eigenvector corresponding to $\lambda_2(\mathcal{G} \backslash S)$:

$$L(\mathcal{G} \backslash S) y = \lambda_2(\mathcal{G} \backslash S) y; \quad \|y\| = 1, y \perp \mathbf{1}$$

Now define the vector

$$z = \begin{bmatrix} y \\ 0 \end{bmatrix};$$

note that $\|z\| = 1$ and $z \perp 1$; as such $\lambda_2(\mathcal{G}) \leq z^\top L(\mathcal{G}) z$. That is,

$$\lambda_2(\mathcal{G}) \leq \sum_{uv \in E(\mathcal{G} \backslash S)} (y_u - y_v)^2 + \underbrace{\sum_{uv \in E(S)} (z_u - z_v)^2}_{0} + \sum_{u \in S} \sum_{v \in \mathcal{G} \backslash S} (\underbrace{z_u}_{0} - z_v)^2$$

so,

$$\lambda_2(\mathcal{G}) \leq \lambda_2(\mathcal{G} \backslash S) + \sum_{u \in S} 1 = \lambda_2(\mathcal{G} \backslash S) + |S|$$

Spectra vs. Structure

Okay! Now suppose that S is chosen as the cutset corresponding to $\kappa_0(\mathcal{G})$. Then $\lambda_2(\mathcal{G}\backslash S) = 0$ and

$$\lambda_2(\mathcal{G}) \leq \kappa_0(\mathcal{G})$$

Upshot: $\lambda_2(\mathcal{G})$ is a lower bound for node connectivity!

The bound is actually tight, for example $\lambda_2(C_4) = \kappa_0(C_4) = 2$

summary so far:

- $L(\mathcal{G}) = E(\mathcal{G})E(\mathcal{G})^\top = \Delta(\mathcal{G}) - A(\mathcal{G})$
- $L(\mathcal{G})$ is positive semidefinite
- $\lambda_2(\mathcal{G}) > 0$ iff $\mathcal{G}$ is connected
- $\lambda_2(\mathcal{G})$ is a measure of connectivity

Oh ... one last thing: trace of any matrix is the sum of its eigenvalues, so

$$\textbf{trace}\, L(\mathcal{G}) = \sum_i d(v_i) = 2\,|\mathcal{E}(\mathcal{G})|$$

Spectra of Some Classes of Graphs

Complete Graph

It would be good to develop some intuition for spectra of graphs, and in particular their dependencies on n, if any. Of course we have to start with the complete graph on n nodes, denoted by K_n:

$$L(K_n) = \begin{bmatrix} n-1 & -1 & \cdots & -1 & -1 \\ -1 & n-1 & \cdots & -1 & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ -1 & -1 & -1 & -1 & n-1 \end{bmatrix} = nI - \mathbf{1}\mathbf{1}^T$$

as always, $\lambda_1(K_n) = 0$ and $u_1 = \mathbf{1}/\sqrt{n}$. The other eigenvectors, generically denoted by x for now, can chosen to be orthogonal to $\mathbf{1}$. So

$$L(K_n)x = (nI - \mathbf{1}\mathbf{1}^T)x = \lambda x$$

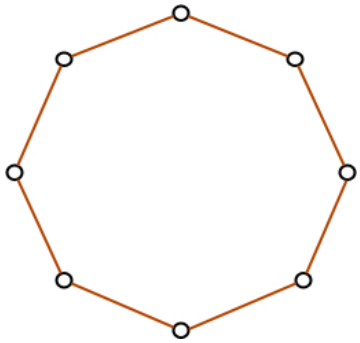
Hence for all these eigenvectors

$$nx = \lambda x!$$

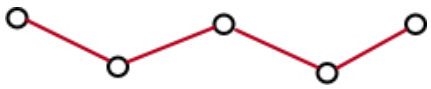
The spectrum of $L(K_n)$ is thus

$$0, n, n, \ldots n; \quad \text{check that} \quad \textbf{trace}\{L(K_n)\} = n(n-1)$$

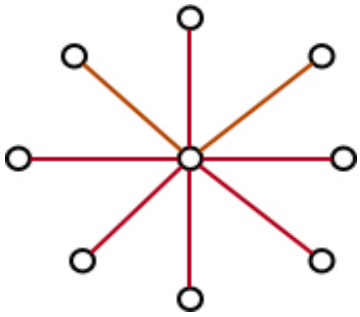
Spectra of some other classes of graphs



$$2\,(1-\cos 2k\pi/n), \quad k=0,1,\ldots n-1$$



$$2(1-\cos k\pi/n), \quad k=0,1,\ldots n-1$$



$n-2$ eigenvalues of 1, one eigenvalue of zero (as always) and last one is $2(n-1)-(n-2)=n$

dynamics on graphs

so far, graphs and some linear algebra, spectra vs. structure, and examples on how to find the spectra in closed form for certain classes of graphs. We now what to see how this machinery actually helps us understand dynamics on networks

Our Action Plan is as follows:

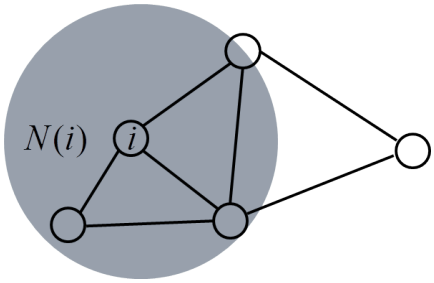
1. we start with a baseline dynamics/distributed algorithm called consensus
2. we relate consensus behavior to structure of the graph
3. this setup can then be extended to directed graphs

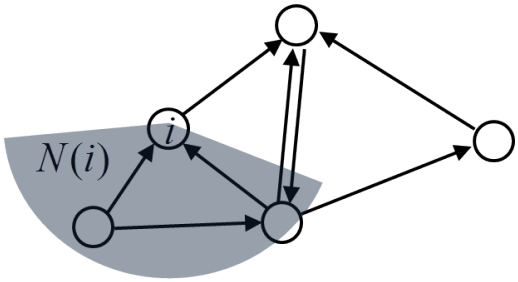
We then move on to show that this distributed algorithm can be used in many different context to do very useful distributed tasks for us

However, it is important to note that the same line of research could have been pursued with a different baseline/distributed protocol or view completely from the perspective of patterned matrices independent of particular protocol!

Network in the Dynamics- general setup

- Graph $\mathcal{G}$ is composed of physical nodes $\mathcal{V}$ and coupling edges $\mathcal{E}$
- Node i acquires information from the set of its neighbors $\mathcal{N}(i)$





- Node i has a state $x_i(t)$ and neighbor information $I_i(t) = \{x_j(t) | j \in \mathcal{N}(i)\}$
- Provides a naturally distributed dynamics over $\mathcal{G}$

$$\dot{x}_i(t) = f_i(x_i(t), I_i(t))$$

- some of the earlier works in distributed decision-making include: DeGroot ('74), Borkar and Varaiya ('82), Tsitsiklis ('84) ...

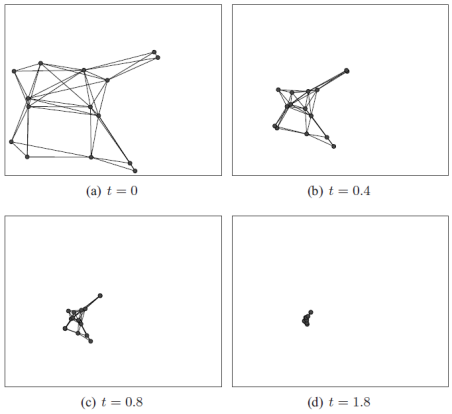
Agreement/Consensus Protocol

Consensus Model

$$\dot{x}_i(t) = -\sum_{j \in N(i)} w_{ij}\,(x_i(t) - x_j(t))$$

$$\rightsquigarrow \dot{x}(t) = -L(\mathcal{G})\,x(t)$$

where $L(\mathcal{G})$ is the (weighted) Laplacian matrix.



- appears in: flocking, formation control, opinion dynamics, energy systems, synchronization, distributed estimation, distributed optimization, among many others!

Let us examine the convergence of the algorithm a bit more ... in terms of the graph structure. We will assume that $w_{ij} = 1$ for this purpose, although our observations generalize seamlessly to weighted graphs

Consensus and λ_2

Let us consider consensus on undirected networks ... spectral factorization of the Laplacian is of the form

$$L(\mathcal{G}) = U\Lambda U^\top$$

where

$$U = \begin{bmatrix} u_1 & u_2 & \cdots & u_n \end{bmatrix} \quad \text{and} \quad \Lambda = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix}$$

as such,

$$\begin{aligned} x(t) &= e^{-L(\mathcal{G})t}x(0) = Ue^{-t\Lambda}U^T x(0) \\ &= u_1^\top x(0)\, u_1 + e^{-\lambda_2 t} u_2^\top x(0)\, u_2 + \ldots + e^{-\lambda_n t} u_n^\top x(0)\, u_n \end{aligned}$$

so if the graph is connected (noting that $u_1 = \mathbf{1}/\sqrt{n}$)

$$x(t) \to \frac{\mathbf{1}^T x(0)}{n}\,\mathbf{1} \quad \text{at a rate proportional to } \lambda_2(\mathcal{G})!$$

more on consensus and λ_2

in fact,

$$\begin{aligned} \|x(t) - \frac{\mathbf{1}^T x(0)}{n}\| &= \|\sum_{i=2}^{n} e^{-\lambda_i t} \underbrace{u_i^\top x(0)}_{\alpha_i} u_i\| \\ &= \sum_{i=2}^{n} e^{-\lambda_i t}|\alpha_i| \leq (n-1) \underbrace{\beta}_{\max_i |\alpha_i|} e^{-\lambda_2 t} \end{aligned}$$

so if we want $\|x(t) - \frac{\mathbf{1}^T x(0)}{n}\| \leq \varepsilon$ for some $\varepsilon > 0$, then we need

$$t \geq \{\ln \frac{\beta(n-1)}{\varepsilon}\}/\lambda_2(\mathcal{G}) \propto \frac{1}{\lambda_2(\mathcal{G})}$$

higher algebraic connectivity directly translates to faster convergence (in a linear way)!

what insights graph theory provides for consensus

some observations:

- Recall that $\lambda_2(P_n) = 2(1-\cos k\pi/n)$, $\lambda_2(C_n) = 2(1-\cos 2k\pi/n)$, $\lambda_2(S_n) = 1$, and $\lambda_2(K_n) = n$
- what this means is that as $n \to \infty$, the rate of convergence for P_n and C_n goes to zero!
- in the meantime, the rate of convergence for K_n grows linearly with n
- however, the number of edges for P_n, C_n grow linearly with n but for K_n the number of edges is $O(n^2)$!

this thread of thought leads to the area of graph synthesis

how baseline consensus can be used for more elaborate distributed algorithms

- as a distributed subroutine for mixing
- including the right inputs to consensus (not just driven by initial conditions)
- consensus with nonlinear and/or state-dependent weights (used in preserving connectivity in distributed robotics)
- consensus with negative, complex-valued, and matrix weights
- consensus across scales
- consensus with security and privacy considerations

Structural Stability of Linear Time-Invariant Systems

Graph Theory in Systems and Controls: part 2

M.-A. Belabbas

[1] University of Illinois at Urbana-Champaign
Electrical and Computer Engineering
Coordinated Science Laboratory

Conference on Decision and Control, 2018
Miami Beach, FL, USA

Which structured LTI systems can sustain stable dynamics?

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} a_{11} & 0 & 0 & a_{14} \\ 0 & 0 & a_{23} & a_{24} \\ a_{31} & 0 & a_{32} & 0 \\ 0 & a_{42}0 & 0 & a_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b_1 \\ 0 \\ b_3 \\ 0 \end{bmatrix} u$$

- Does there exist values of the a_{ij}'s that yield asymptotically stable dynamics? If so, we call the system structurally stable.
- Does there exist values of the a_{ij}'s and b_i's that yield controllable dynamics? If so, we call the system structurally controllable.
- Recall: Linear time-invariant dynamics is asymptotically stable iff the eigevalues of the system matrix have strictly negative real parts.
- Graph theory is the natural framework to study structural stability.

Reformulating the structural stability problem

$$A = \begin{bmatrix} 0 & * & * & 0 & * \\ * & * & 0 & * & * \\ 0 & * & 0 & * & 0 \\ 0 & * & 0 & * & * \\ * & 0 & 0 & * & 0 \end{bmatrix}$$

$*$ entries are arbitrary real

0 entries are fixed to zero

Definition (Zero-pattern (ZP))

Set E_{ij} to be the $n \times n$ matrix with all entries 0 except for the ijth one, which is 1. We call a *zero pattern* a vector space $\mathcal{Z}$ of matrices

$$A = \sum_{(i,j)\in\mathcal{N}} a_{ij} E_{ij}.$$

- Does the ZP contain stable (Hurwitz) matrices?
- We call a ZP that contains Hurwitz matrices stable

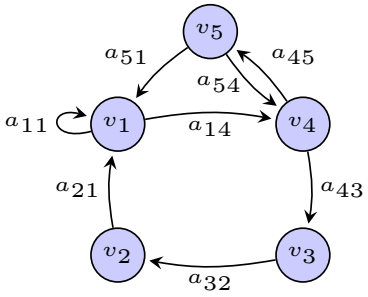
Hurwitz Digraphs and Zero-Patterns

- Think of a ZP as an adjacency matrix with

$$\begin{aligned} 0 &\longrightarrow 0 \\ * &\longrightarrow 1 \end{aligned}$$

- There is a bijection between zero patterns $\mathcal{Z}$ and digraphs $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$ and $E = \mathcal{N}$.

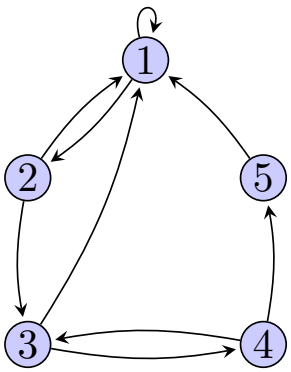
$$\begin{bmatrix} * & 0 & 0 & * & 0 \\ * & 0 & 0 & 0 & 0 \\ 0 & * & 0 & 0 & 0 \\ 0 & 0 & * & 0 & * \\ * & 0 & 0 & * & 0 \end{bmatrix}$$

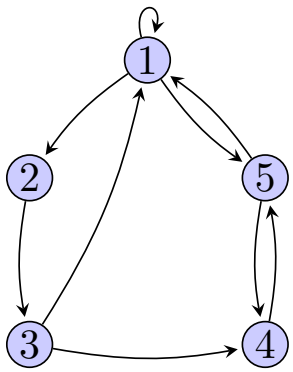


- We call a graph Hurwitz or stable if the corresponding ZP is stable.

How to determine if a graph is Hurwitz? How to create Hurwitz graphs?

Which graph is stable?





$$\begin{bmatrix} * & * & 0 & 0 & 0 \\ * & 0 & * & 0 & 0 \\ * & 0 & 0 & * & 0 \\ 0 & 0 & * & 0 & * \\ * & 0 & 0 & 0 & 0 \end{bmatrix}$$

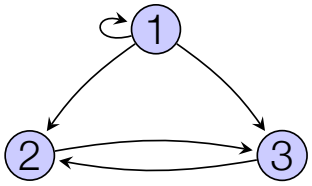
$$\begin{bmatrix} * & * & 0 & 0 & * \\ 0 & 0 & * & 0 & 0 \\ * & 0 & 0 & * & 0 \\ 0 & 0 & 0 & 0 & * \\ * & 0 & 0 & * & 0 \end{bmatrix}$$

Which graph is stable?

Key idea: need enough mixing of information

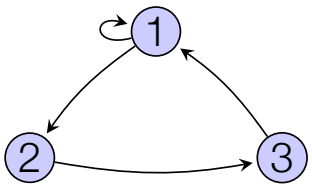
Lemma

A digraph G is stable only if every strongly connected component has a node with a self-loop



Not stable: the strongly connected component $\{2, 3\}$ has no nodes with a self-loop.

This is not the end of the story...

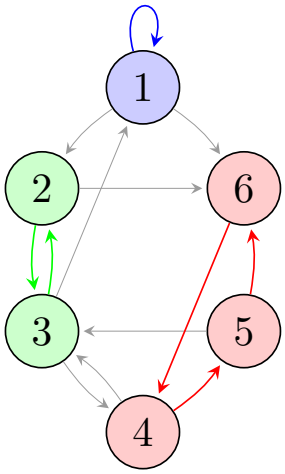


The graph is strongly connected and has a self-loop, yet not stable.

$\longrightarrow$ need to find the graphical structure that enables stability

k-decompositions

- k-cycle in G: a sequence of k distinct nodes connected by edges.
- Two cycles are disjoint if they have no nodes in common.
- k-decomposition in G: union of *disjoint cycles* covering k nodes.
  A k-decomposition is given by cycles $S_1, \ldots, S_l$ if the S_i are disjoint and $|S_1| + \cdots + |S_l| = k$.
- Hamiltonian cycle (resp. decomposition): n-cycle (resp. decomposition).

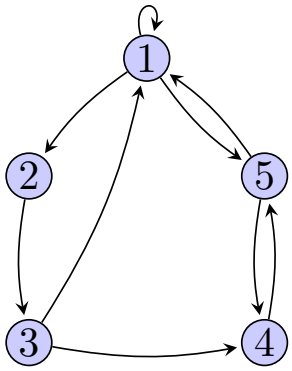


1-cycle = (1)
2-cycle: (23)
3-cycle: (456)
3-decomp.: $(1)(23)$ or (456)
4-decomp.: $(1)(456)$
5-decomp.: $(23)(456)$

A necessary condition for stability

Theorem[1]

A digraph G is stable only if it contains a k-decomposition for each $k = 1, 2, \ldots, n$



$$\begin{bmatrix} * & * & 0 & 0 & * \\ 0 & 0 & * & 0 & 0 \\ * & 0 & 0 & * & 0 \\ 0 & 0 & 0 & 0 & * \\ * & 0 & 0 & * & 0 \end{bmatrix}$$

1-decomp.: (1), 2-decomp.: (15), 3-decomp.:$(1)(45)$ but no 4-decomp. $\longrightarrow$ not stable.

[1] B. "Sparse Stable Systems", Systems and Control Letters, 2013

A necessary condition for stability: sketch of proof

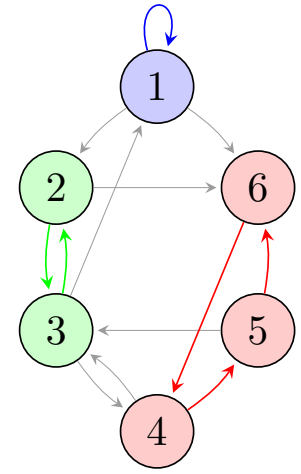
- S_k: symmetric group on k characters.
- For $\sigma \in S_k$, let $\sigma(i)$ be the position of the ith in the permutation.

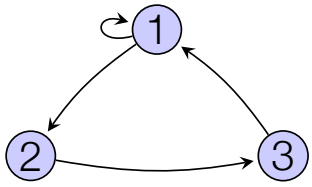
  e.g. $\sigma : \{1,2,3,4\} \to \{2,1,4,3\}$ then $\sigma(1) = 2$ and $\sigma(3) = 4$.
- It is known that A is Hurwitz only if all coefficients of its characteristic polynomial are non-zero.
- Characteristic polynomial of A is given by

$$\det(I\lambda - A) = \sum_{k=0}^{n-1} (-1)^k \lambda^k \sum_{\sigma \in S_{n-k}} (-1)^\sigma \prod_{i=1}^{n-k} a_{i,\sigma(i)}$$

A necessary condition for stability: sketch of proof (II)

- Each term $\prod_{i=1}^{k} a_{i,\sigma(i)}$ corresponds to a k-decomposition.
- Said otherwise: each permutation in S_k corresponds to a k-decomposition:
  e.g. permutation in S_3 that sends $\{4,5,6\}$ to $\{5,6,4\}$ is depicted in red.
  permutation in S_3 that sends $\{1,2,3\}$ to $\{1,3,2\}$ is depicted in blue+green.
- Conclusion: no k-decompositions $\Longrightarrow$ degree $n-k$ term in characteristic polynomial of any matrix in $\mathcal{Z}$ is zero $\Longrightarrow$ graph and ZP are not stable





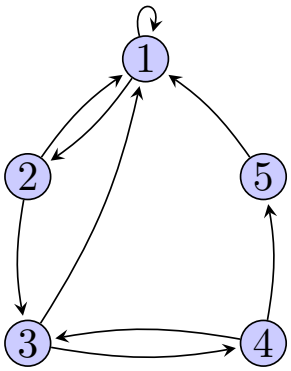
$$p(s) = s^3 - a_{11}s^2 + 0s - a_{12}a_{23}a_{31}.$$

A sufficient condition for stability

Theorem[2]

A digraph G is stable if it contains a sequence of *nested* k-decomposition for each $k = 1, 2, \ldots, n$.

We say that a k-decomposition K_1 is nested in K_2 if the node set of K_1 is included in the one of K_2



$$\begin{bmatrix} * & * & 0 & 0 & 0 \\ * & 0 & * & 0 & 0 \\ * & 0 & 0 & * & 0 \\ 0 & 0 & * & 0 & * \\ * & 0 & 0 & 0 & 0 \end{bmatrix}$$

1-decomp.: (1), 2-decomp.: (12), 3-decomp.:(123), 4-decomp.:$(12(34)$, 5-decomp.:(12345).

[2] B. "Sparse Stable Systems", Systems and Control Letters, 2013

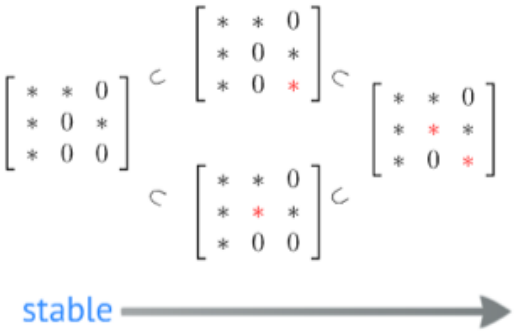
Are the necessary and sufficient conditions close?

- There are many graphs that are stable, but do not pass the sufficient condition.
- From our simulations, we observe that the necessary condition is close to being sufficient: the number of graphs that pass the necessary condition and are *not* stable is relatively small.
- Stability is not generic. The proportion of stable matrices in a ZP can be *very small*.
- Hence simulations studies are "hard": one needs to sample many matrices in a SMS to conclude non-stability. Very unlike structural controllability: almost all systems in a zero-pattern are controllable. Sample one system: with probability one, it is controllable if the zero pattern is.

Minimal stable graphs and notions of robustness

Observation: adding an edge to a stable graph yields another stable graph.

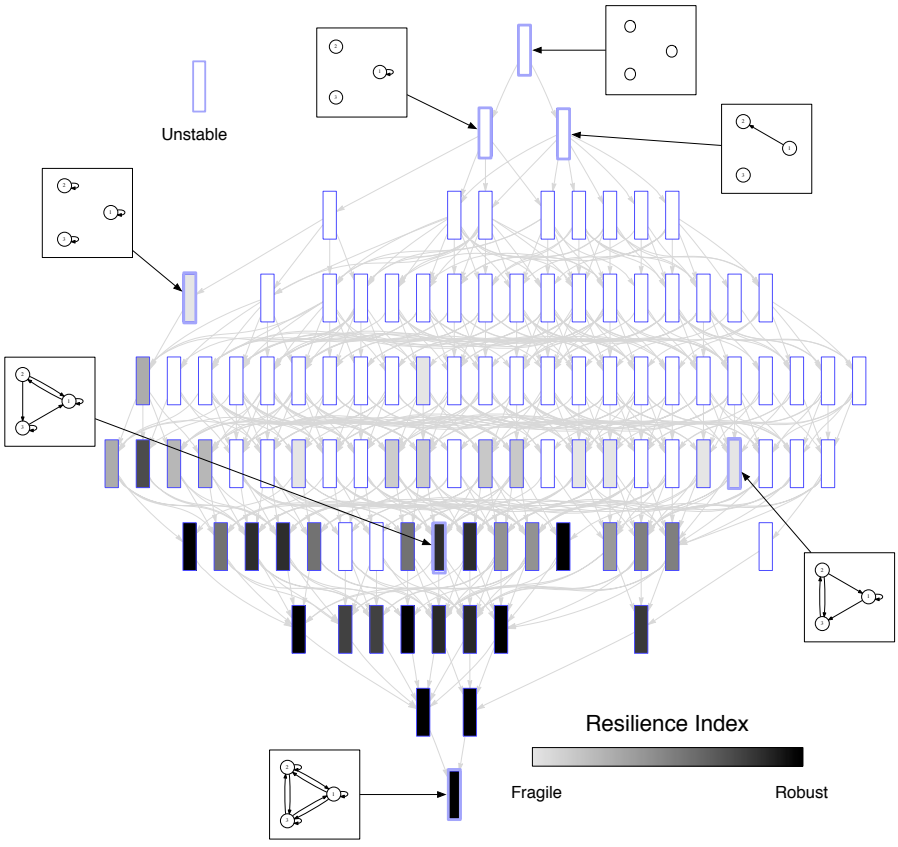
We say that graph stability is monotone with respect to edge addition.



This simple observation yields two interesting definitions:

- Minimal stable graphs: stable graphs for which removing *any* edge yields an *unstable* graph.
  All stable graphs are "descendants" of minimal stable graphs. We can think of them as "prime" graphs.
- Robustly stable graphs: stable graphs for which removing *any* edge yields a *stable* graph.

The Tree of Three-Graphs



Box $\rightarrow$ graph on three nodes

Same # edges $\rightarrow$ same row

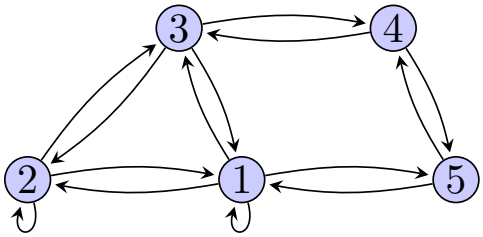
Edge between box denotes inclusion

Shade: $\frac{\#\text{ stable ancestors}}{\#\text{ ancestors}}$

Minimal stable: lightest shade. There are 7.

Reciprocal or Symmetric graphs

- It is often the case that information exchange is *bilateral*: $i \leftrightarrow j$.
- We call a graph reciprocal or symmetric of to every edge $(i,j) \in E$ there is an edge $(j,i) \in E$.
- The corresponding ZP is symmetric:



$$A = \begin{bmatrix} * & * & * & 0 & * \\ * & * & * & 0 & 0 \\ * & * & 0 & * & 0 \\ 0 & 0 & * & 0 & * \\ * & 0 & 0 & * & 0 \end{bmatrix}$$

- Two cases: either the matrices in the ZP are symmetric (*strongly symmetric ZP*) or not necessarily symmetric (*weakly symmetric ZP*).

Stability of Symmetric Graphs

Definition[3]

A ZP is *(weakly symmetric* if to a free variable in position ij corresponds a free variable in position ji. A ZP is *strongly symmetric* if it only contains symmetric matrices.

Theorem[3]

A strongly symmetric ZP is stable *if and only if* all its diagonal elements are free.

Theorem[3]

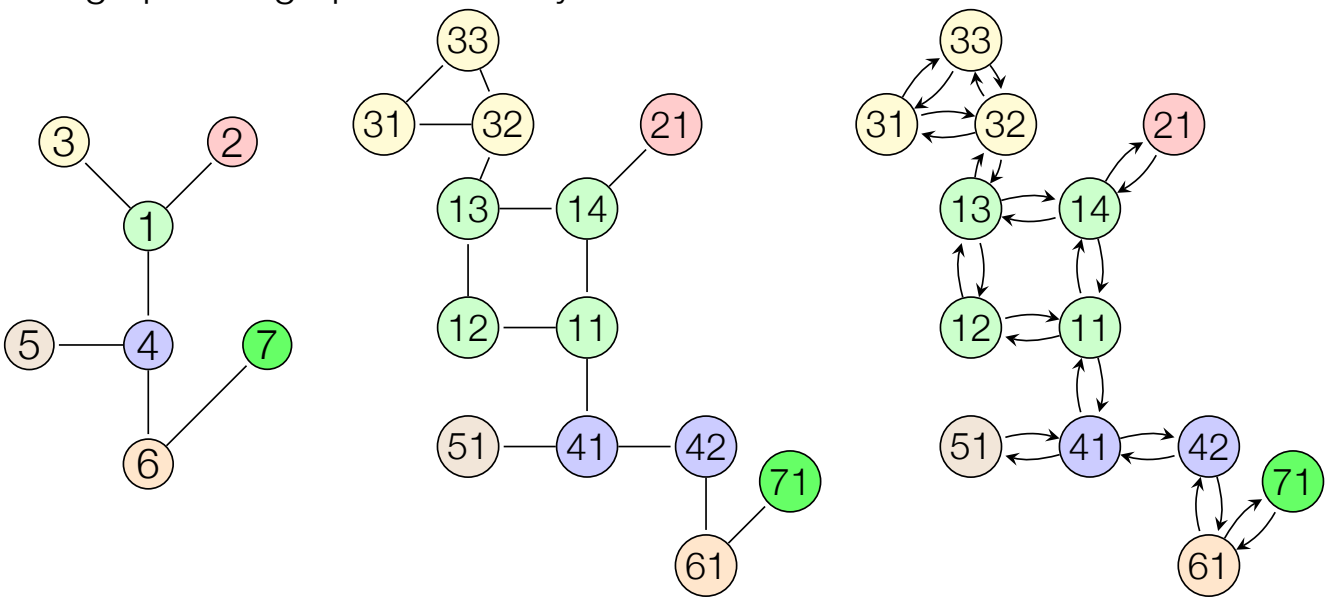
A weakly symmetric ZP is stable *if and only if* its graph is so that

1. Every node is strongly connected to a self-loop
2. The graph contains a Hamiltonian decomposition.

[3] A. Kirkoryan and B. "Symmetric Sparse Systems", CDC 2014.

Key notion: fat trees

The proof of the last theorem is graphical in nature. We sketch is here. A tree graph is a graph without cycles.



- Tree graph → Nodes can be cycles → Edges are symmetric → fat tree

Stability of symmetric graphs

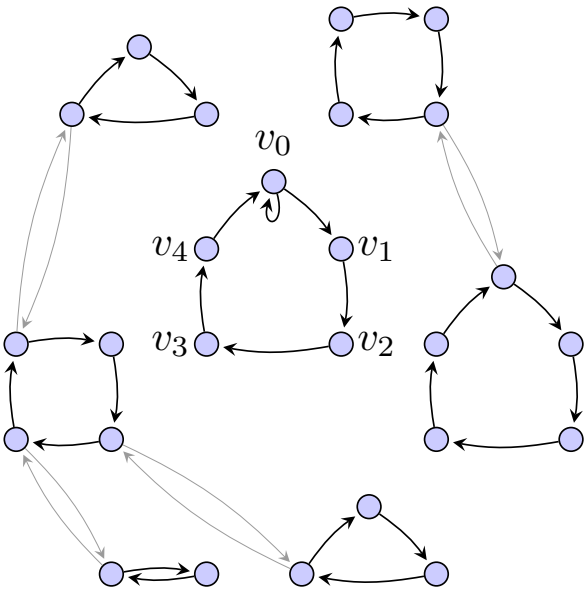
- Proof idea: Given a symmetric graph G, show that if
  1. Every node in G is connected to a self-loop
  2. G contains a Hamiltonian decomposition

  $\rightarrow$ then there exists a sequence of *nested* k-decompositions, $k = 1, \ldots, n$.

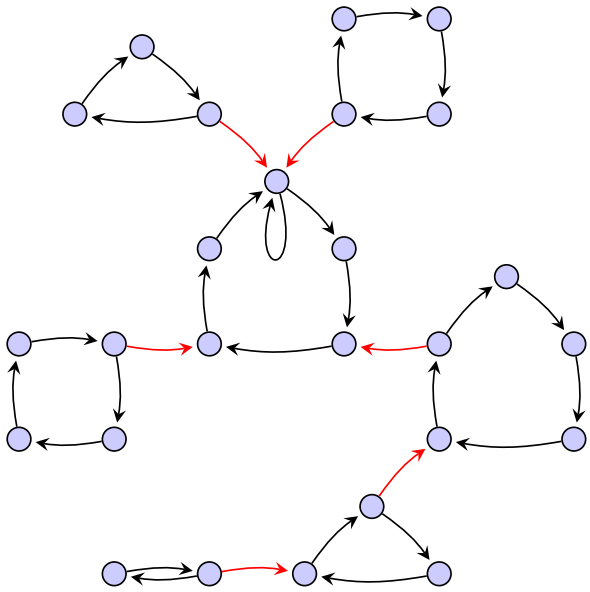
- The conclusion above says that we satisfy the sufficient condition presented earlier.

- Proof technique: find a fat tree in G. Fat trees provide a natural ordering of nodes. Use the ordering to exhibit nested k-decompositions:
  We label (order) the nodes so that $\{1\}, \{1, 2\}, \{1, 2, 3\}, \ldots, \{1, \ldots, n\}$ all have k-decompositions. By construction, they are nested.

Stability of symmetric graphs (II)

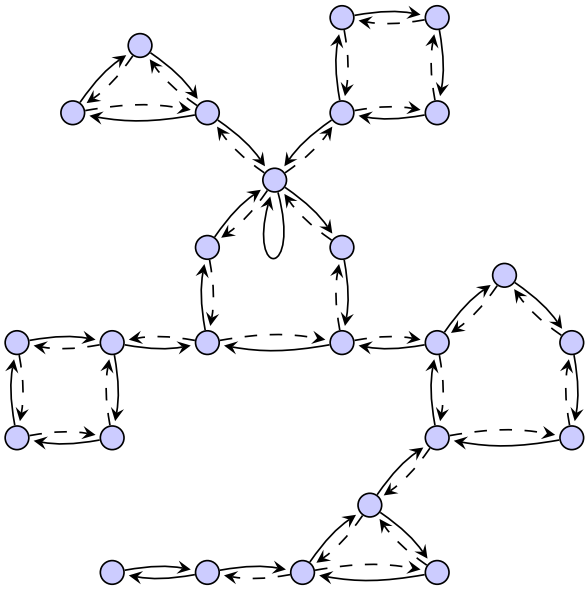


Draw the cycles of a Hamiltonian decomposition of G. This is a subgraph of G.

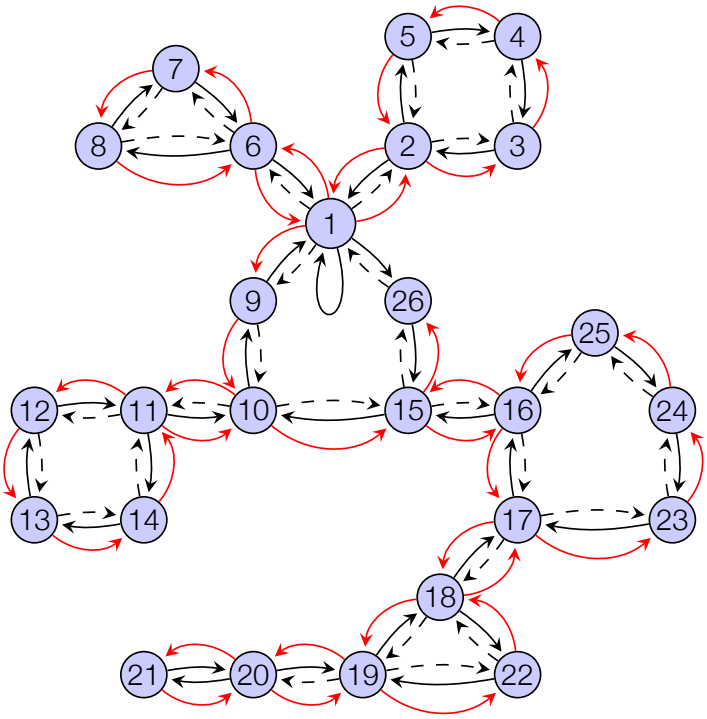


Connect every cycle to the cycle with the self-loop. We can do so by assumption 1.

Stability of symmetric graphs (III)



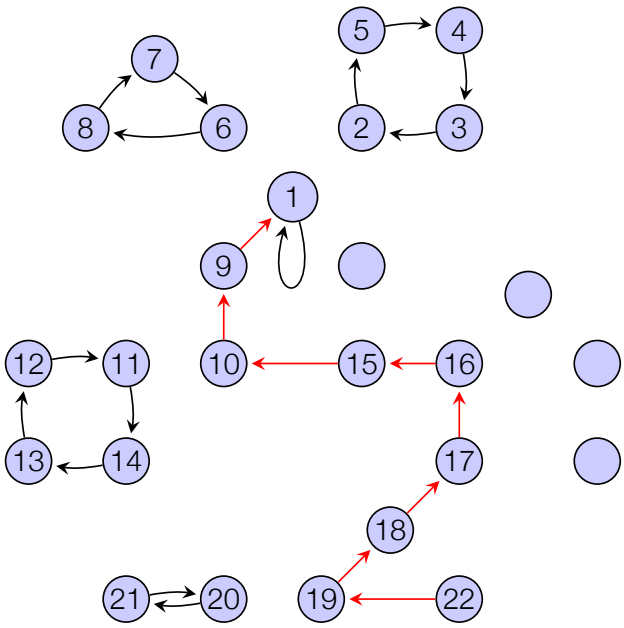
Add reciprocal edges. The resulting graph is a planar subgraph of G by construction.



Ordering: Set v_0 at 1. Order nodes counter-clockwise. Skip already numbered nodes. By construction, no node lies inside $\rightarrow$ complete ordering. Call this graph P.

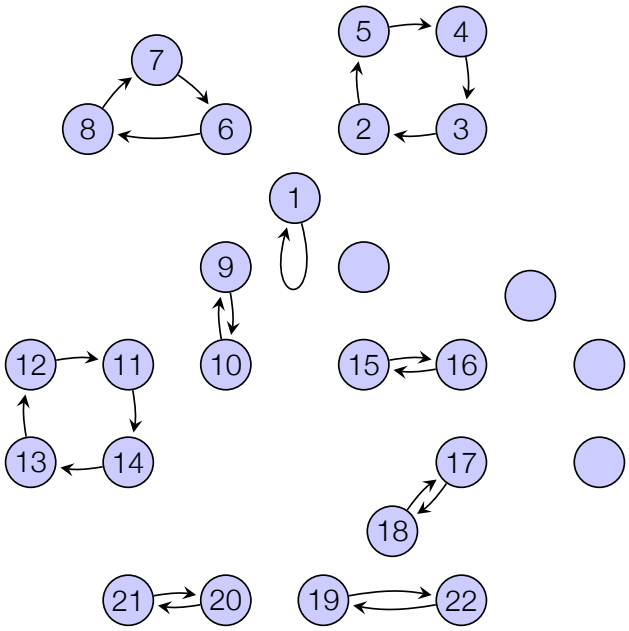
Stability of symmetric graphs: (IV)

The last graph shown is a subgraph of G. We show that is satisfies the hypothesis of Theorem 2.



- There is a unique path from any node k to 1 using the plain edges of P only.
- Key observation: by construction, the subgraph induced by the node set $\{1, 2, \ldots, k\}$ is the union of the path joining 1 to k and l-cycles.

Stability of symmetric graphs: (V)



A $n = 22$-decomposition

- The subgraph induced by nodes $\{1, \ldots, k\}$ admits a Hamiltonian decomposition, which is thus a k-decomposition of G.
- Depending on whether the path joining 1 to k has an even or odd number of nodes, the decomposition is in 2-cycles (even) or self=loop+2 cycles (odd).
- Repeating the procedure for each node $k = 1, \ldots, n$, we obtain nested k-decompositions.

Structural Stability of Random Graphs

- Random graph theory provides a different lens to look at what may otherwise be hard problems.
- We look for conditions under which a sample graph form a given distribution is structurally stable with overwhelming probability.
- The results are asymptotic in the number of nodes.
- Allows us to overlook finer structural details and obtain answers when the graph is very large.
- Recall: Bernoulli distribution with parameter p: $P(\omega = 1) = p$ $P(\omega = 0) = 1 - p$, $\omega \in \Omega = \{0, 1\}$.

Random graphs models

- We look at two random graph models for symmetric ZP
- Model 1: variable number of edges $\mathcal{G}^n_{p,q}$
  1. Graph on n nodes
  2. Existence of an edge between nodes i and j, $i \neq j$ are independent Bernoulli random variables with parameter p
  3. Existence of a self-loop are independent Bernoulli r.v. with parameter q.
- Model 2: fixed number of edges $\mathcal{F}^n_{M,K}$
  1. Graph on n nodes
  2. Exactly M edges (i,j), chosen uniformly at random amongst all possible edges (i,j), $i \neq j$.
  3. Exactly K self-loops chosen uniformly at random.

Definition

We say that **almost every** random graph G^n has a property X, if $\mathbb{P}(G^n \text{ has } X) \to 1$ as $n \to \infty$

Problem Statement

- We consider probabilities that depend on n. We need $p(n), q(n) \to 0$ $n \to \infty$, otherwise random graphs a very dense.

Problem

For what magnitudes of $p = p(n)$ and $q = q(n)$, is almost every random graph $\mathcal{G}^n_{p,q}$) stable? For what magnitudes of $M = M(n)$ and $K = K(n)$, is almost every random graph $\mathcal{F}^n_{M,K}$) stable?

- Define ω_1, ω_2, such that:

$$p = p(n) = \frac{\ln(n) + \omega_1}{n}, \quad q = q(n) = \frac{\omega_2}{n}.$$

This particular form for $p(n)$, $q(n)$ makes statements easier.

Results for Model 1

Theorem[4]

Assume that $q(n) < 1 - \varepsilon$ for some $\varepsilon > 0$

1. Almost every graph in $\mathcal{G}^n_{0,q}$ contains a self-loop *if and only if* $\omega_2 \to \infty$.
2. Almost every graph in $\mathcal{G}^n_{p,0}$ contains a Hamiltonian decomposition *if and only if* $\omega_1 \to \infty$.
3. $$\mathbb{P}(\mathcal{G}^n_{p,q} \text{ is stable}) \to 1 \quad \Longleftrightarrow \quad \omega_1, \omega_2 \to \infty.$$

$$p = p(n) = \frac{\ln(n) + \omega_1}{n}, \quad q = q(n) = \frac{\omega_2}{n}.$$

p is probability of an edge, q is probability of a self loop

[4] B., A. Kirkoryan, preprint; A. Kirkoryan PhD thesis

Results for Model 2

Define ω_1, ω_2 such that:

$$M = M(n) = \frac{n(\ln(n) + \omega_1)}{2}, \quad K = K(n) = \omega_2.$$

Theorem[5]

Assume that $M < \frac{n^2(1-\varepsilon)}{2}$ for some $\varepsilon > 0$, then

$$\mathbb{P}(\mathcal{G}^n_{M,K} \text{ is stable}) \to 1 \iff \omega_1 \to \infty, \ \ \omega_2 \geq 1.$$

[5] B., A. Kirkoryan, preprint; A. Kirkoryan PhD thesis

Thank you for your attention!

Graph Theory in Systems and Control

Graphs and Performance in Network Systems

Mehran Mesbahi[1], Daniel Zelazo[2]

[1] University of Washington
[2] Technion-Israel Institute of Technology

CDC
Miami Beach, Florida, December 19, 2018

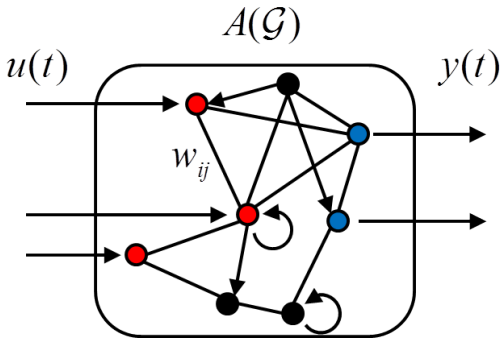
Table of Contents

Symmetry and Controllability

Control of Networks

- Model

$$\dot{x}_i(t) = -w_{ii}x_i(t) + \sum_{i\sim P} w_{iP}x_P(t) + u_i(t)$$



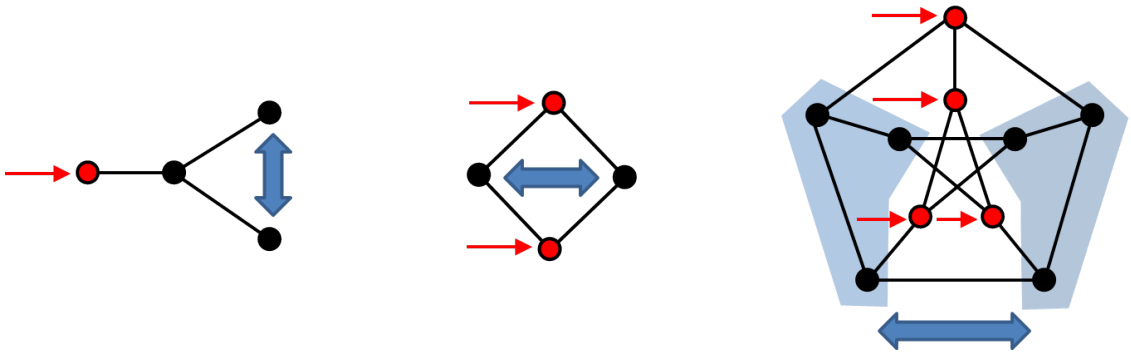
that in general assumes the form:

$$\dot{x}(t) = A(\mathcal{G})x(t) + B(S)u(t)$$

Controllability/observability: stabilization via feedback, observer design, disturbance/noise rejection, optimal control, and pole placement

Network Controllability

For the LTI plant $(A(\mathcal{G}, S), B(S))$ what are the structural conditions for controllability? One approach is to link uncontrollability to symmetry



For today, we will use the edge leader follower dynamics

$$\dot{x} = A(\mathcal{G}, S)x + B(S)u = -(L(\mathcal{G}) + B(S)B(S)^T)x + B(S)u.$$

(These results can be extended to the leader follower dynamics $\dot{x} = A(\mathcal{G}, \mathcal{R})x + B(\mathcal{R})u$ and controlled consensus dynamics $\dot{x} = -L(\mathcal{G})x + B(S)u$)

Symmetry

First, what do we mean by symmetry...

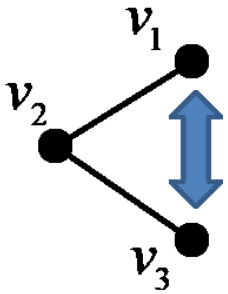
Definition

An automorphism of the graph is a mapping $\pi : \mathcal{V}(\mathcal{G}) \to \mathcal{V}(\mathcal{G})$ such that if $\{i, p\} \in \mathcal{E}(\mathcal{G}) \iff \{\pi(i), \pi(p)\} \in \mathcal{E}(\mathcal{G})$

Represented as $\pi : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$, $\pi(i) = p$

$$\begin{array}{ccccccc} 1 & & 2 & & 3 & \cdots & n \\ \downarrow & & \downarrow & & \downarrow & & \downarrow \\ \pi(1) & & \pi(2) & & \pi(3) & & \pi(n) \end{array}$$

Example



$1 \to 3$, $2 \to 2$, $3 \to 1$
Mapping $\pi : \mathcal{V}(\mathcal{G}) \to \mathcal{V}(\mathcal{G})$
$\pi(1) = 3$, $\pi(2) = 2$, $\pi(3) = 1$
The edges $\{\pi(i), \pi(P)\}$
$\{1, 2\} \to \{3, 2\} \in \mathcal{E}$, $\{2, 3\} \to \{2, 1\} \in \mathcal{E} \implies \pi$ is an automorphism

Symmetry

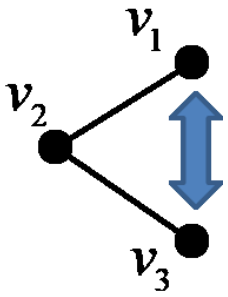
We need an algebraic representation of the automorphism π.

Definition

A permutation matrix is a $\{0,1\}$ square matrix with one "1" and one "zero" in each row and column.

$\pi \rightarrow$permutation matrix P such that $PA(\mathcal{G}) = A(\mathcal{G})P$

Example



$$PA(\mathcal{G}) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$A(\mathcal{G})P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Symmetry

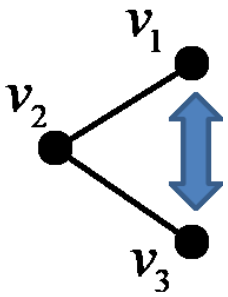
We also need a link between the automorphism and the inputs.

Definition

A system is input symmetric with respect to the input nodes if there exists a nonidentity automorphism with input nodes invariant under its action.

Input symmetry (permutation P) w.r.t. to the input nodes $\iff$ $P \neq I$, $A(\mathcal{G})P = PA(\mathcal{G})$ and $PB(S) = B(S)$.

Example



$$PB(\{v_2\}) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = B(\{v_2\})$$

$\rightarrow$ Input symmetric

$$PB(\{v_3\}) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \neq B(\{v_3\})$$

$\rightarrow$ Input asymmetric

Symmetry

Some more preliminary work before showing our controllability conditions
For an automorphism π of $\mathcal{G}$ with permutation matrix P

$$A(\mathcal{G})P = PA(\mathcal{G}) \implies \deg(v) = \deg(\pi(v)) \implies \Delta(\mathcal{G})P = P\Delta(\mathcal{G})$$

then as $L(\mathcal{G}) = A(\mathcal{G}) - \Delta(\mathcal{G})$ we have

$$L(\mathcal{G})P = PL(\mathcal{G}).$$

For input symmetry $PB(S) = B(S)$ then

$$PB(S) = B(S) \implies \pi(\{s\}) = \{s\} \text{ for all } s \in S$$

Finally,

$$\begin{aligned} A(\mathcal{G}, S)P &= -(L(\mathcal{G}) + B(S)B(S)^T)P \\ &= -P(L(\mathcal{G}) + B(S)B(S)^T) \\ &= PA(\mathcal{G}, S). \end{aligned}$$

Symmetry

Theorem

Input symmetry implies uncontrollability.

Proof.

For $P \neq I$, $A(\mathcal{G})P = PA(\mathcal{G})$ and $PB(S) = B(S) \implies A(\mathcal{G}, S)P = PA(\mathcal{G}, S)$
Let v be an eigenvector of $A(\mathcal{G}, S) := A$ then

$$APv = PAv \quad = \quad P(\lambda v) = \lambda Pv$$

So Pv is also an eigenvector.
As $A(\mathcal{G}, S)$ is symmetric with a spanning set of eigenvectors then for some v, $Pv \neq v$.
Then $v - Pv$ is an eigenvector and $(v - Pv)^T B(S) = v^T B(S) - v^T P^T B(S)$; hence

$$(v - Pv)^T B(S) = v^T B(S) - v^T B(S) = 0$$

and the pair $(A(\mathcal{G}, S), B(S))$ is uncontrollable by PBH □

more generally ...

Theorem

Suppose that the network dynamics assumes the form

$$\dot{x} = \mathbf{A}(\mathcal{G})x + \mathbf{B}(\mathcal{G})u$$

is such at there exists some $P \in \mathbf{AUT}(\mathcal{G})$ that commutes with the dynamics and leaves the input invariant under its action, i.e.,

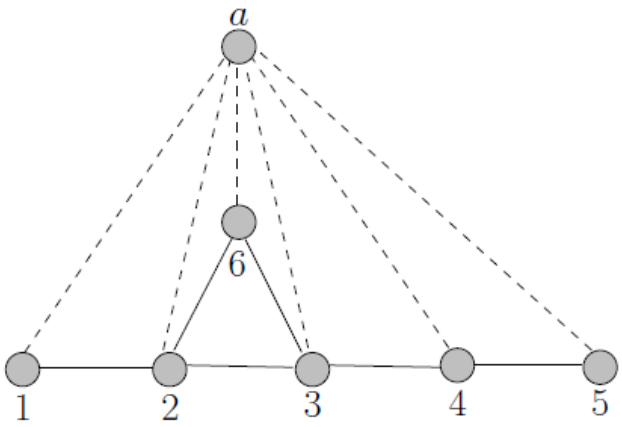
$$P\mathbf{A}(\mathcal{G}) = \mathbf{A}(\mathcal{G})P \qquad P\mathbf{B}(\mathcal{G}) = \mathbf{B}(\mathcal{G});$$

if $\mathbf{A}(\mathcal{G})$ is non-defective, then $(A(\mathcal{G}), B(\mathcal{G}))$ is not controllable.

Does Input Asymmetry $\implies$ controllability?

No!

Consider the smallest asymmetric graph $\mathcal{G}$ controlled through a



Then $A(\mathcal{G},\mathcal{R}) = L(\mathcal{G}) + I$ and $B(\mathcal{R}) = -\mathbf{1}$; $A(\mathcal{G},\mathcal{R})$ has $\mathbf{1}$ as an eigenvector:

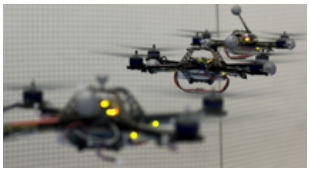
$$A(\mathcal{G},\mathcal{R})\mathbf{1} = L(\mathcal{G})\mathbf{1} + \mathbf{1} = \mathbf{1}$$

All other eigenvectors of $A(\mathcal{G},\mathcal{R})$ are orthogonal to $\mathbf{1}$; now invoke PBH!

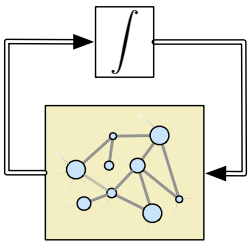
Table of Contents

Consensus-Seeking Networks

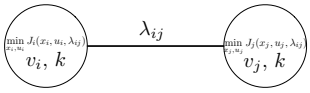
The consensus protocol is a canonical model
for studying complex networked systems



formation
control

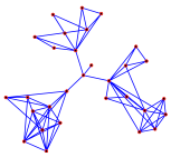


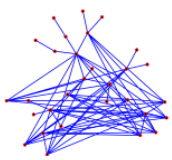
system theory
over graphs



distributed
optimization

Are certain information structures more favorable than others?





Can system performance be characterized using properties of the graph?

$\mathcal{H}_2$
$\mathcal{H}_\infty$ $\propto$ cycle lengths, node degree
$\vdots$ $\vdots$

How do we synthesize good information structures?

$$\min_{\mathcal{G}\in\mathbb{G}} \|\Sigma(\mathcal{G})\|$$

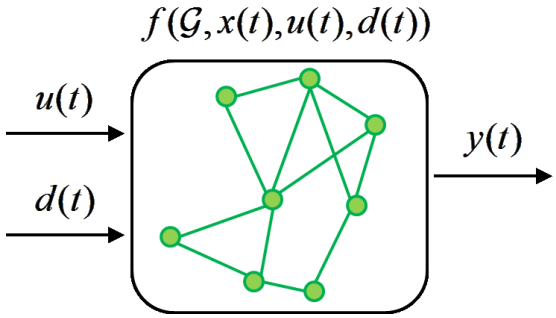
Influenced Networked Dynamics

Networks may be influenced by

- selected leaders
- exogenous inputs (disturbances or noises)
- malicious agents

General Dynamics

$$\dot{x}(t) = f(\mathcal{G}, x(t), u(t), d(t))$$
$$y(t) = g(\mathcal{G}, x(t), u(t), d(t))$$



Analysis draws upon:

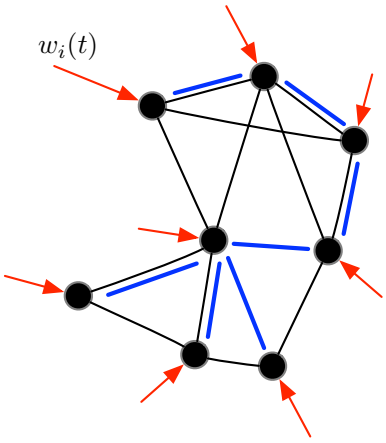
- Control theory:
  Input-output dynamics
- Graph theory:
  Design and reasoning on $\mathcal{G}$
- Large-scale Optimization:
  For large # nodes n
- Machine-learning:
  For uncertain dynamics and inputs

The Noisy Consensus Protocol

Dynamics

$$\dot{x}(t) = -L(\mathcal{G})x(t) + w(t)$$
$$y(t) = E(\mathcal{H})^T x(t)$$

- Each node corrupted by zero-mean white Gaussian noise.
- $\mathcal{H}$ models the performance network (i.e., $\mathcal{H} \subseteq \mathcal{G}$ or $\mathcal{H} = \mathcal{K}_n$)



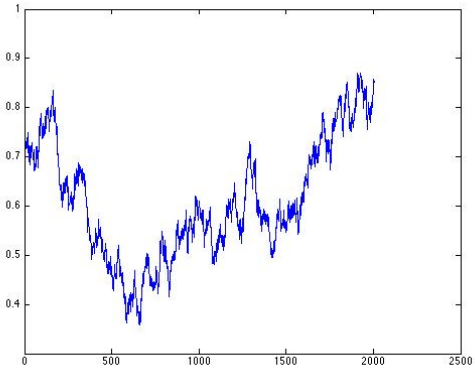
---

consensus state (average) is driven by noise

$$\frac{d}{dt}\text{avg}(x(t)) = \frac{1}{n}\mathbf{1}^\top w(t)$$

covariance exhibits a random walk

$$\mathcal{E}(\text{avg}(x(t)^2)) = \frac{\sigma_w}{n}t$$

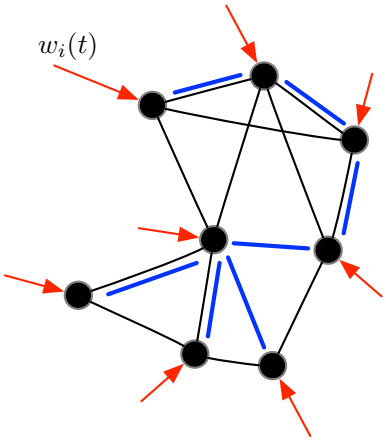


The Noisy Consensus Protocol

Dynamics

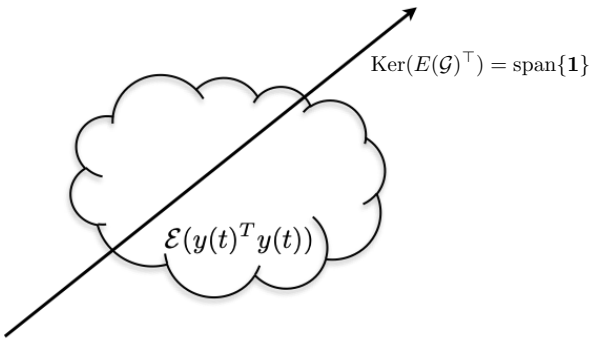
$$\dot{x}(t) = -L(\mathcal{G})x(t) + w(t)$$
$$y(t) = E(\mathcal{H})^T x(t)$$

- Each node corrupted by zero-mean white Gaussian noise.
- $\mathcal{H}$ models the performance network (i.e., $\mathcal{H} \subseteq \mathcal{G}$ or $\mathcal{H} = \mathcal{K}_n$)



When driven by noise, it is meaningful to examine how noises effect the stead-state covariance of the relative states

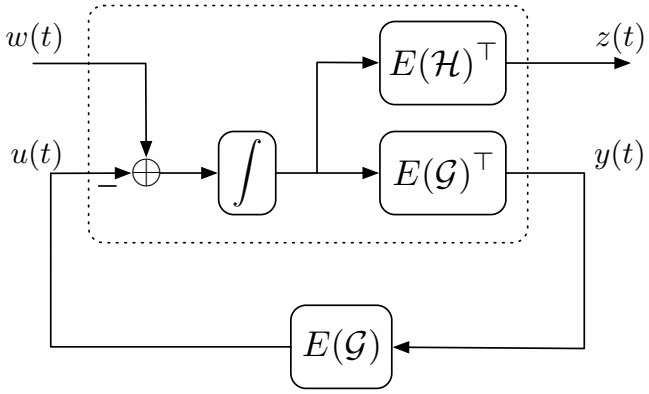
Characterized by the $\mathcal{H}_2$ performance



Minimal Realizations and the Edge Laplacian

A two-port model

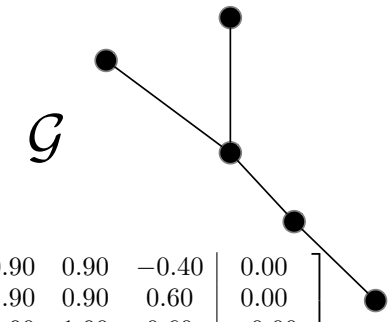
$$\dot{x}(t) = -L(\mathcal{G})x(t) + w(t)$$
$$z(t) = E(\mathcal{H})^T x(t)$$



Note the system is *not* minimal (unobservable) and also has unbounded $\mathcal{H}_2$ norm (eigenvalue at 0)

$\Rightarrow$ Find a stable minimal realization!

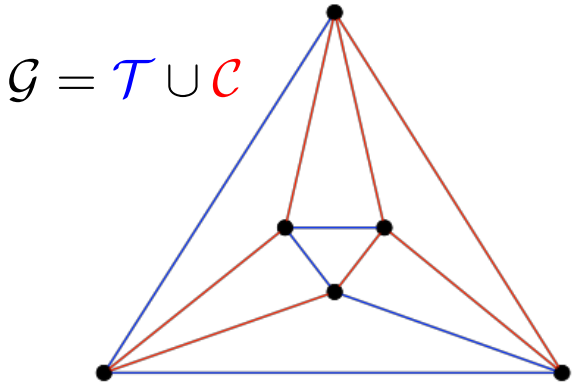
$$S = \begin{bmatrix} P & \frac{1}{\sqrt{n}}\mathbf{1} \end{bmatrix} \qquad \mathbf{1}^\top P = 0$$
$$\tilde{x}(t) = S^{-1}x(t)$$



$$S^{-1}L(\mathcal{G})S = \left[\begin{array}{cccc|c} 2.90 & 0.90 & 0.90 & -0.40 & 0.00 \\ 0.90 & 1.90 & 0.90 & 0.60 & 0.00 \\ 0.90 & 0.90 & 1.90 & 0.60 & -0.00 \\ -0.40 & 0.60 & 0.60 & 1.29 & -0.00 \\ \hline 0.00 & 0.00 & 0.00 & 0.00 & -0.00 \end{array}\right]$$

Spanning Trees and Co-Trees

A connected graph can be decomposed into a spanning tree and the edges that complete cycles (co-tree)



Cycles can be expressed as a "linear combination" of edges in the tree

$$E(\mathcal{C}) = E(\mathcal{T})R$$
$$E(\mathcal{G}) = E(\mathcal{T}) \begin{bmatrix} I & R \end{bmatrix}$$

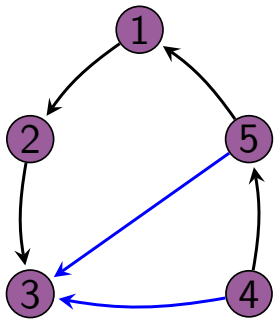
R is referred to as the *Tucker representation* of $\mathcal{G}$ with spanning tree $\mathcal{T}$

Theorem [Godsil and Royle, 2001]

The cycle space of $\mathcal{G}$ is spanned by the fundamental cycles of $\mathcal{G}$.

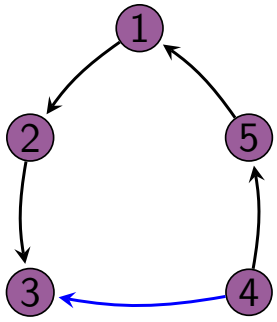
$$\text{Ker}[E(\mathcal{G})] = \text{Im} \begin{bmatrix} -R \\ I \end{bmatrix}$$
$$R = \underbrace{(E(\mathcal{T})^\top E(\mathcal{T}))^{-1} E(\mathcal{T})^\top}_{E_{\mathcal{T}}^L} E(\mathcal{C})$$

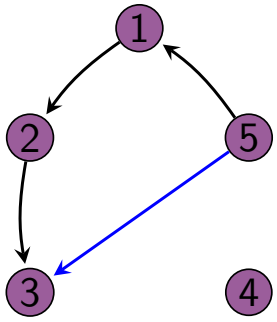
Spanning Trees and Co-Trees



$$E(\mathcal{T}) = \begin{bmatrix} 1 & 0 & 0 & -1 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

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

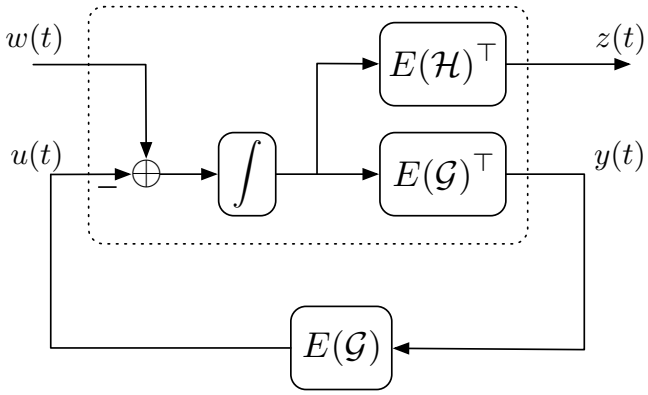




Minimal Realizations and the Edge Laplacian

A two-port model

$$\dot{x}(t) = -L(\mathcal{G})x(t) + w(t)$$
$$z(t) = E(\mathcal{H})^T x(t)$$



$\Rightarrow$ Find a stable minimal realization!

$$S^{-1} = \begin{bmatrix} E(\mathcal{T})^\top \\ \frac{1}{\sqrt{n}}\mathbf{1}^\top \end{bmatrix}$$

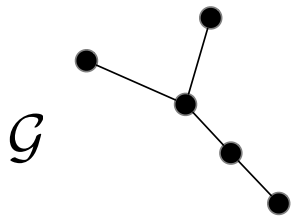
$$\tilde{x}(t) = \begin{bmatrix} x_\tau(t) \\ \text{avg}(x(t)) \end{bmatrix} = S^{-1}x(t)$$

$$S^{-1}L(\mathcal{G})S = \begin{bmatrix} L_{ess}(\mathcal{G}) & \mathbf{0}^\top \\ \mathbf{0} & 0 \end{bmatrix}$$

The Essential Edge Laplacian

$$L_{ess}(\mathcal{G}) := (E(\mathcal{T})^\top E(\mathcal{T}))(I + RR^\top)$$

The Edge Laplacian



$$S^{-1}L(\mathcal{G})S = \begin{bmatrix} E(\mathcal{T})^\top E(\mathcal{T})(I + RR^\top) & \mathbf{0}^\top \\ \mathbf{0} & 0 \end{bmatrix} = \begin{bmatrix} 2 & 1 & -1 & 0 \\ 1 & 2 & -1 & 0 \\ -1 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} = L_e(\mathcal{T})$$

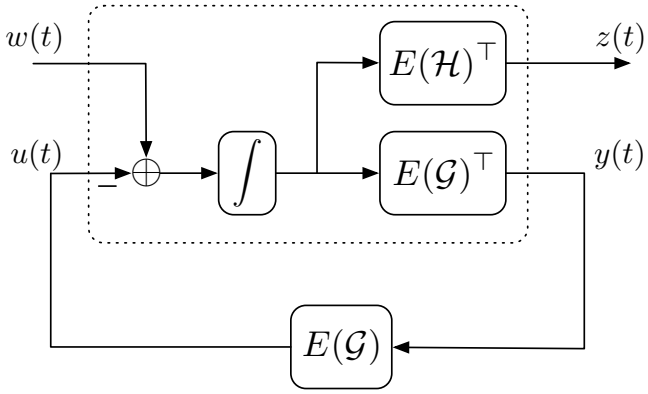
Edge Laplacian

$$L_e(\mathcal{G}) = E(\mathcal{G})^\top E(\mathcal{G}) \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$$

- shares the same non-zero eigenvalues of $L(\mathcal{G})$
- $L_e(\mathcal{T})$ is positive definite
- indexed by the edges in the graph
- $[L_e(\mathcal{G})]_{ij} = \pm 1$ when *edge* i is adjacent to *edge* j
- Ker$[L_e(\mathcal{G})]$ is spanned by fundamental cycles in $\mathcal{G}$

$\mathcal{H}_2$ Performance of Consensus

$$\dot{x}(t) = -L(\mathcal{G})x(t) + w(t)$$
$$z(t) = E(\mathcal{H})^T x(t)$$



Theorem [Zelazo and Mesbahi, TAC2011]

The $\mathcal{H}_2$ performance of the consensus protocol is

$$\|\Sigma(\mathcal{G})\|_2^2 = \mathsf{Tr}[E(\mathcal{H})^\top E_{\mathcal{T}}^{L^\top} X E_{\mathcal{T}}^{L} E(\mathcal{H})],$$

where

$$X = \frac{1}{2}\left(I + RR^\top\right)^{-1}$$

is the positive definite solution to the Lyapunov equation

$$\mathcal{L}(X) = -L_{ess}(\mathcal{G})X - XL_{ess}(\mathcal{G})^\top + E(\mathcal{T})^\top E(\mathcal{T}) = 0.$$

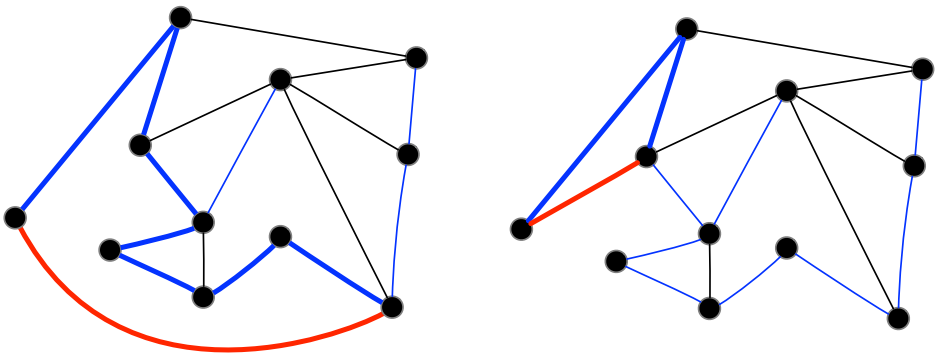
$\mathcal{H}_2$ Performance of Consensus

Theorem [Zelazo et al., Systems & Controls Letters, 2013]

Consider the consensus protocol with $\mathcal{G} = \mathcal{H} = \mathcal{T}$ and an edge $e \notin \mathcal{G}$. Then

$$\|\Sigma(\mathcal{T} \cup e)\|_2^2 = \|\Sigma_e(\mathcal{T})\|_2^2 - \frac{\ell(c) - 1}{2\ell(c)},$$

where $\ell(c)$ is the length of the fundamental cycle created by adding the edge e.



- *long* cycles are better than *short ones*

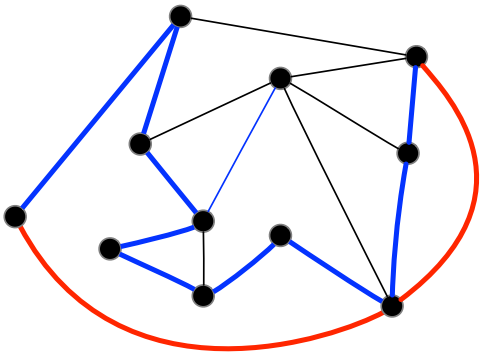
$\mathcal{H}_2$ Performance of Consensus

Corollary [Zelazo et al., Systems & Controls Letters, 2013]

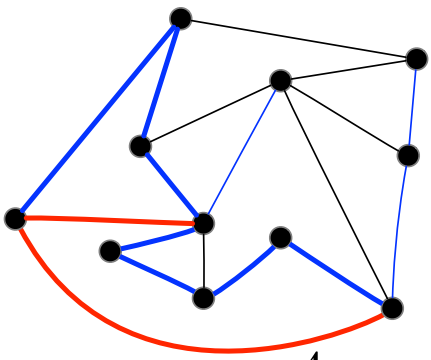
Consider the consensus protocol with $\mathcal{G} = \mathcal{H} = \mathcal{T}$ and an edges $e_1, e_2 \notin \mathcal{G}$. Then

$$\|\Sigma(\mathcal{T} \cup \{e_1, e_2\})\|_2^2 = \|\Sigma_e(\mathcal{T})\|_2^2 - \left(1 - \frac{\ell(c_1) + \ell(c_2)}{2(\ell(c_1)\ell(c_2) - s_{12}^2)}\right),$$

where s_{ij} is the edge correlation number for cycles c_i and c_j.



$s_{12} = 0$



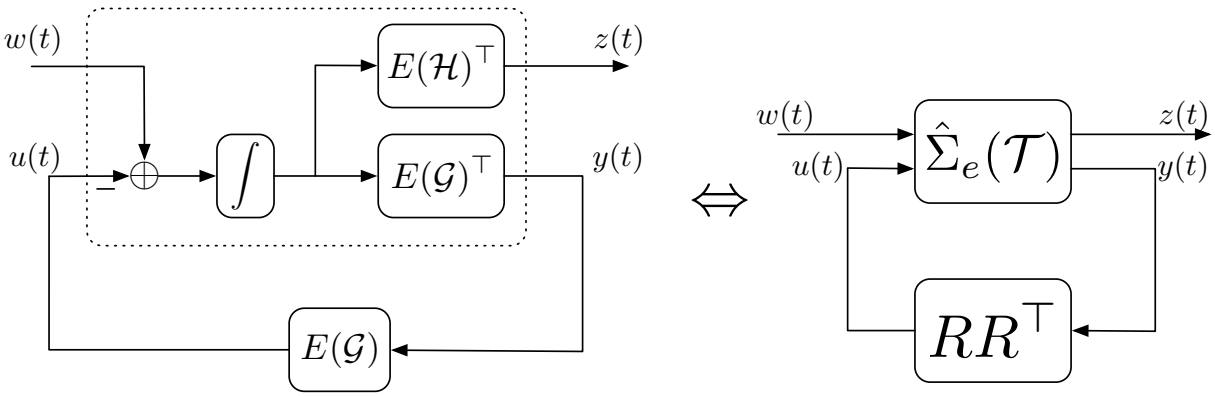
$s_{12} = 4$

- *edge disjoint* cycles are better

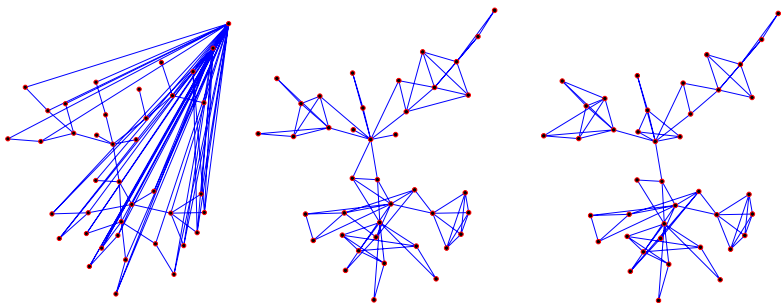
Design of Cycles

A network design problem

Given a graph $\mathcal{G}$ with spanning tree $\mathcal{T}$, add k edges that optimizes $\|\Sigma(\mathcal{G})\|_2^2$.

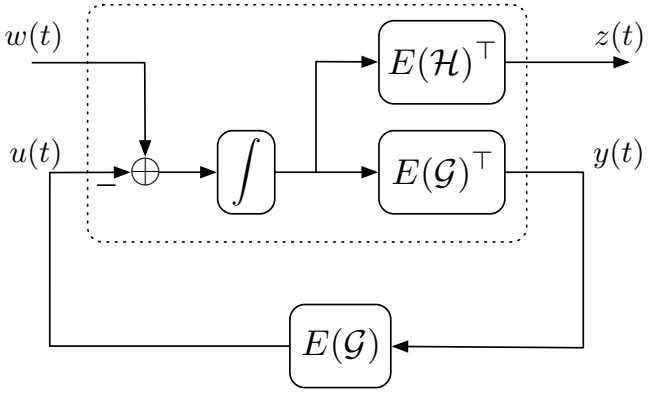


- Cycles interpreted as a *feedback* system
- Can be formulated as a *mixed-integer SDP*
- re-weighted ℓ_1 optimization; ADMM



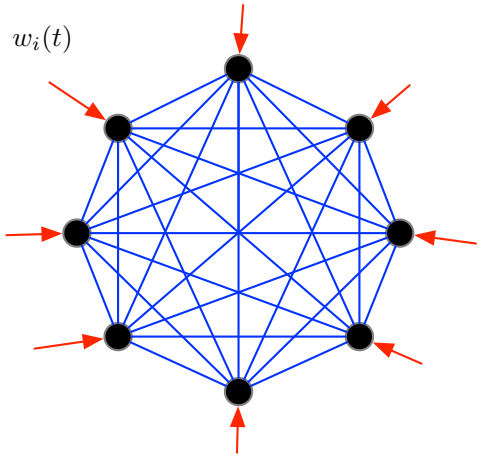
$\mathcal{H}_2$ Performance of Consensus

$$\dot{x}(t) = -L(\mathcal{G})x(t) + w(t)$$
$$z(t) = E(\mathcal{H})^T x(t)$$



What is the performance when monitoring all relative state pairs?

$$\mathcal{H} = \mathcal{K}_n$$



Circuit Interpretations

Linear Consensus as an RC-Circuit

$$\dot{x}(t) = -L(\mathcal{G})x(t) + w(t)$$
$$y(t) = E(\mathcal{H})^T x(t)$$

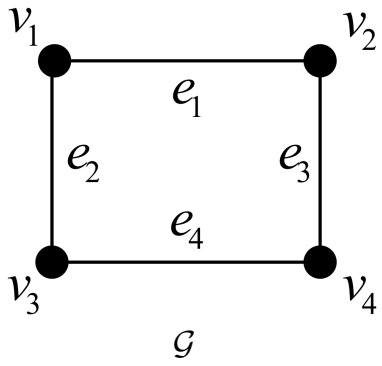
Capacitors $\Leftrightarrow$ Node Dynamics (integrators)

Resistors $\Leftrightarrow$ Edge Dynamics (linear gain)

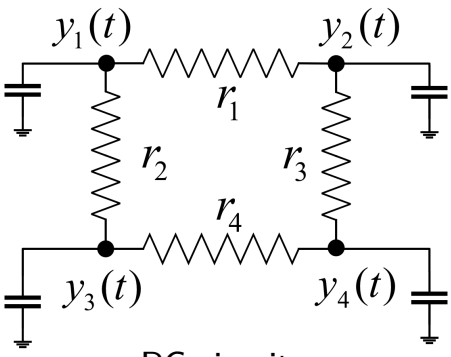
- edge weights model the *admittance* of the resistor

$$r_i = \frac{1}{w_i}$$

- in steady-state, network corresponds to a *resistive circuit*



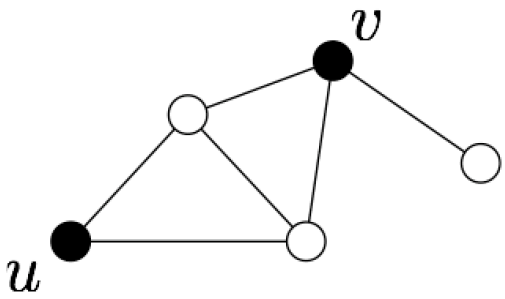
$\mathcal{G}$

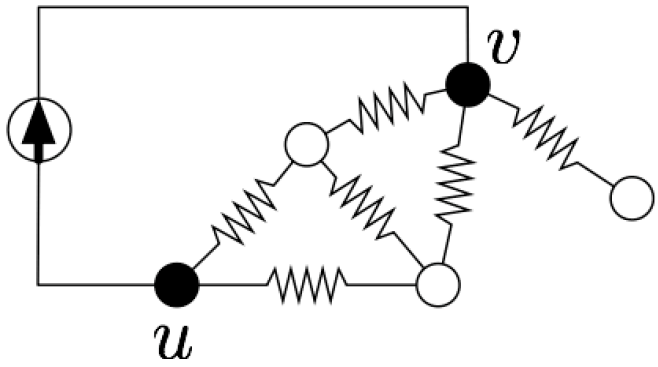


RC circuit

Effective Resistance

The effective resistance between two nodes u and v is the electrical resistance measured across the nodes when the graph represents a resistive circuit.





Effective Resistance Calculation [Klein and Randić 1993]

$$\mathcal{R}_{uv}(\mathcal{G}) = [L^{\dagger}(\mathcal{G})]_{uu} + 2[L^{\dagger}(\mathcal{G})]_{uv} + [L^{\dagger}(\mathcal{G})]_{vv}$$

The *total effective resistance* of a graph is the sum over all pairs of nodes of $\mathcal{R}_{uv}(\mathcal{G})$,

$$\mathcal{R}_{tot}(\mathcal{G}) = \sum_{\{u,v\} \in \mathcal{E}} \mathcal{R}_{uv}(\mathcal{G}).$$

Effective Resistance and the Edge Laplacian

Proposition

Consider a graph $\mathcal{G}$ with spanning tree $\mathcal{T}$ and Tucker matrix R. Let R_{uv} satisfy $(\mathbf{e}_u - \mathbf{e}_v) = E(\mathcal{T})R_{uv}$. Then the effective resistance between nodes u and v can be computed as

$$\mathcal{R}_{uv}(\mathcal{G}) = R_{uv}^{\top}(I + RR^{\top})^{-1}R_{uv}.$$

This can be extended to derive an expression for the total effective resistance. Let $R_{\mathcal{K}_n}$ satisfy $E(\mathcal{K}_n) = E(\mathcal{T})R_{\mathcal{K}_n}$, representing the Tucker matrix for all possible edges, then

$$\mathcal{R}_{tot}(\mathcal{G}) = \mathsf{Tr}[R_{\mathcal{K}_n}^{\top}(I + RR^{\top})^{-1}R_{\mathcal{K}_n}].$$

Performance when $\mathcal{H} = \mathcal{K}_n$

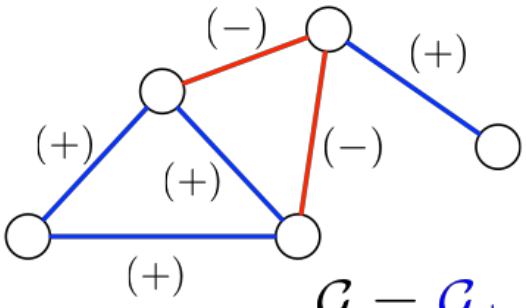
$$\|\Sigma(\mathcal{G})\|_2^2 = \frac{1}{2}\mathcal{R}_{tot}(\mathcal{G})$$

Effective Resistance and Signed Networks

a **signed graph** is a graph with positive and negative edge weights

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$$

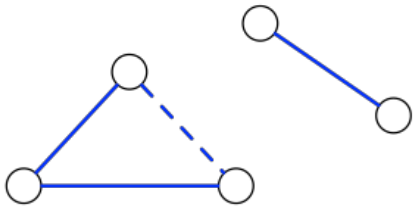
$$\mathcal{W} : \mathcal{E} \to \mathbb{R}$$



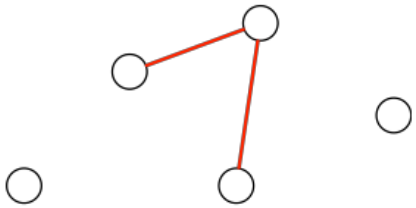
$$\mathcal{G} = \mathcal{G}_+ \cup \mathcal{G}_-$$

$$\mathcal{E}_+ = \{e \in \mathcal{E} : \mathcal{W}(e) > 0\}$$

$$\mathcal{E}_- = \{e \in \mathcal{E} : \mathcal{W}(e) < 0\}$$



$$E(\mathcal{G}_+) = E_+$$



$$E(\mathcal{G}_-) = E_-$$

$$L(\mathcal{G}) = E(\mathcal{G}_+)W_+E(\mathcal{G}_+)^T - E(\mathcal{G}_-)|W_-|E(\mathcal{G}_-)^T$$

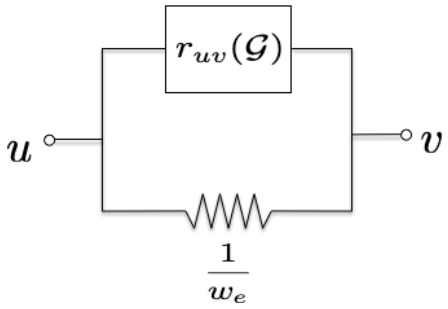
Effective Resistance and Signed Networks

Theorem [Zelazo and Bürger, TCNS2017]

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}_>)$ be a strictly positive network with edge functions $\mu_k = w_k \zeta_k$ (i.e., $w_k > 0$ for all $k \in \mathcal{E}$) and let $\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E}_> \cup e)$ where $e = (u, v)$ is a negative edge with weight $w_e < 0$. Then the signed consensus network reaches agreement if and only if

$$|w_e| \leq r_{uv}^{-1},$$

where r_{uv} is the effective resistance in $\mathcal{G}$ between nodes u and v.

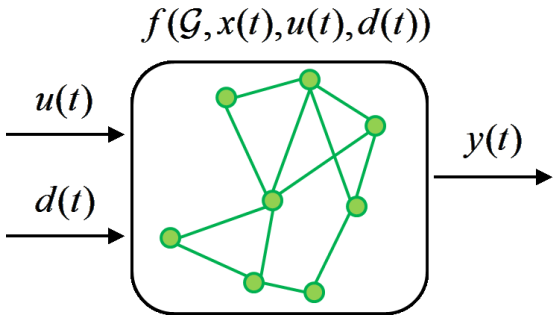


The negative edge weights effectively creates an open circuit

Summary and Outlooks

General Dynamics

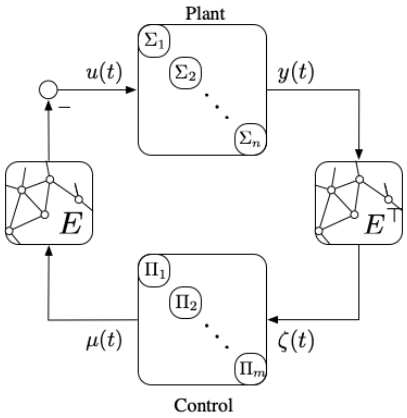
$$\dot{x}(t) = f(\mathcal{G}, x(t), u(t), d(t))$$
$$y(t) = g(\mathcal{G}, x(t), u(t), d(t))$$



- network structure influences the performance of network systems
- in linear consensus, $\mathcal{H}_2$ performance can be understood in terms of fundamental structural properties of the graph: trees and co-trees
- effective resistance is a powerful concept for analyzing performance and robustness of linear consensus
- design of networks leverages combinatorial understanding of performance with modern optimization methods

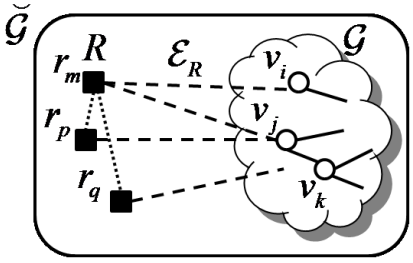
Summary and Outlooks

Explore graph-theoretic interpretations for more general networked systems structures



Leader-follower networks

$$\dot{x}(t) = A(\mathcal{G}, \mathcal{R})x(t) + B(\mathcal{R})u(t)$$



- leader selection and $\mathcal{H}_2$ performance
- effective resistance interpretations
- network design using online optimization

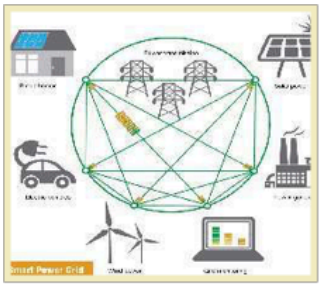
Graph Theory in Systems and Control

Graphs: Unexplored Opportunities

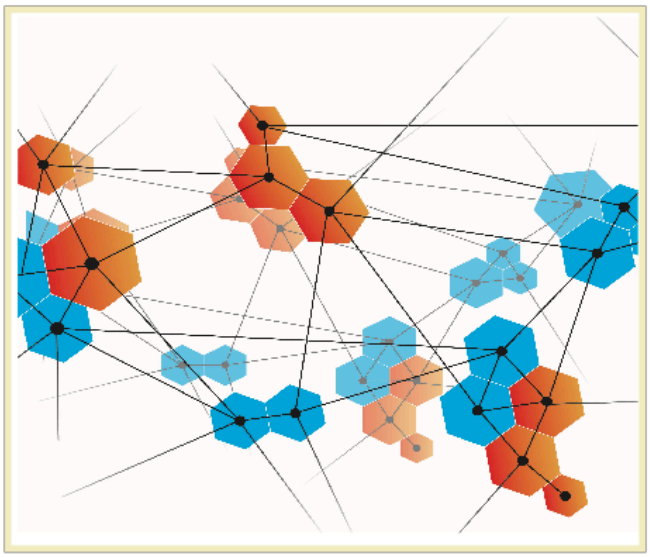
Daniel Zelazo, Mehran Mesbahi, M. Ali Belabbas

CDC
Miami Beach, Florida, December 19, 2018

Networked Dynamic Systems



NETWORKS OF DYNAMICAL SYSTEMS ARE ONE OF THE ENABLING TECHNOLOGIES OF THE FUTURE



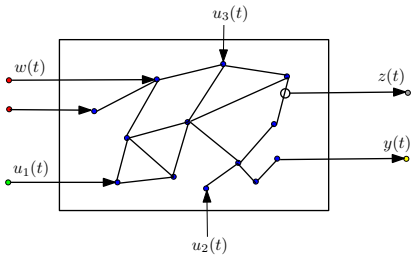
Graphs at CDC

Why do we need this tutorial?

Network analysis and control	MoA03.5, MoA05.6, MoA07.2, MoA09.6, MoA12.2, MoA12.6, MoB03.3, MoB05.3, MoB10.6, MoB12.1, MoC03.3, MoC03.4, MoC06.1, MoC06.2, MoC13.6, MoC14.3, TuA03.6, TuA04.3, TuB09.1, TuB09.2, TuB12.1, TuB12.2, TuB12.3, TuB12.4, TuB12.5, TuB12.6, TuB16.3, TuB16.4, TuC04.3, TuC04.6, TuC05.1, TuC09.1, TuC09.3, TuC10.5, TuC12.1, TuC12.2, TuC12.3, TuC12.4, TuC12.5, TuC12.6, TuC18.5, TuC18.6, WeA09.2, WeA09.3, WeA09.4, WeA09.5, WeA09.6, WeA12.1, WeA12.2, WeA12.3, WeA12.4, WeA12.5, WeA12.6, WeB03.6, WeB05.1, WeB05.4, WeB12.5, WeB12.6, WeB13.1, WeB13.2, WeB13.3, WeB13.4, WeB14.1, WeB14.6, WeB18.6, WeC01.6, WeC05.1, WeC05.5, WeC12.1, WeC12.3
Networked control systems	MoA01.3, MoA03.4, MoA03.5, MoA04.2, MoA04.3, MoA04.4, MoA04.5, MoA04.6, MoA05.1, MoA05.3, MoA10.5, MoA12.1, MoA12.2, MoA12.3, MoA12.4, MoA12.5, MoA12.6, MoB03.2, MoB04.3, MoB04.4, MoB04.6, MoB09.4, MoB12.1, MoB12.2, MoB12.3, MoB12.4, MoB12.5, MoB12.6, MoB14.4, MoC03.4, MoC04.2, MoC04.3, MoC04.5, MoC09.4, MoC10.5, MoC12.1, MoC12.2, MoC12.3, MoC12.4, MoC12.5, MoC12.6, MoC13.1, MoC13.2, MoC13.4, MoC13.5, MoC18.4, MoC19.4, TuA01.6, TuA05.2, TuA10.1, TuA12.1, TuA12.2, TuA12.3, TuA12.4, TuA12.5, TuA12.6, TuA15.5, TuA21.3, TuB02.4, TuB04.4, TuB07.2, TuB12.4, TuB12.5, TuB14.3, TuB14.4, TuB19.3, TuC02.4, TuC03.2, TuC05.3, TuC05.5, TuC09.1, TuC11.6, TuC13.1, TuC16.4, WeA03.1, WeA03.4, WeA03.5, WeA05.6, WeA09.3, WeA09.5, WeA12.4, WeA12.5, WeA12.6, WeA14.4, WeA16.1, WeB01.3, WeB04.3, WeB04.4, WeB04.5, WeB08.1, WeB09.5, WeB10.3, WeB10.4, WeB12.3, WeB12.4, WeB13.1, WeB18.1, WeB19.3, WeB19.4, WeC07.2, WeC15.6, WeC17.3, WeC20.4, WeC21.1, WeC21.4
Control system architecture	MoA17.2, MoC07.6, TuA04.5, TuB06.3, TuB12.6, WeA06.2, WeB14.3, WeC05.4 See also Large-scale Systems
Cooperative control	MoA03.3, MoA03.4, MoA03.6, MoA11.6, MoA14.1, MoA14.2, MoA14.3, MoA14.4, MoA14.5, MoB03.1, MoB03.4, MoB05.5, MoB12.6, MoB14.1, MoB14.2, MoB14.3, MoB14.4, MoB14.5, MoB16.5, MoB17.2, MoB17.4, MoC03.6, MoC12.2, MoC14.1, MoC14.2, MoC14.3, MoC14.4, MoC14.5, MoC14.6, MoC17.2, MoSP1.1, TuA03.2, TuA03.3, TuA05.2, TuA09.6, TuA10.6, TuA11.1, TuA12.1, TuA14.4, TuA16.5, TuB04.1, TuB14.5, TuB17.6, TuC05.1, TuC09.2, TuC11.5, TuC11.6, TuC14.2, TuC14.3, WeA03.5, WeA05.2, WeA05.4, WeA05.5, WeA05.6, WeA14.2, WeA14.3, WeA14.5, WeB04.4, WeB12.3, WeB13.1, WeB13.2, WeB14.2, WeB14.3, WeB14.4, WeB14.5, WeC14.1, WeC20.4

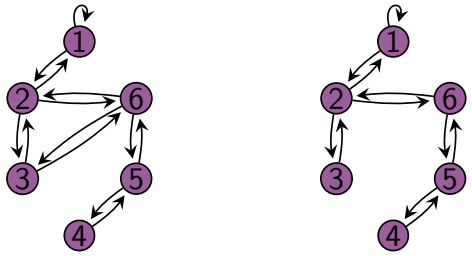
The network approach to systems is here to stay. This tutorial aims to bring to the forefront the role of graphs in these systems.

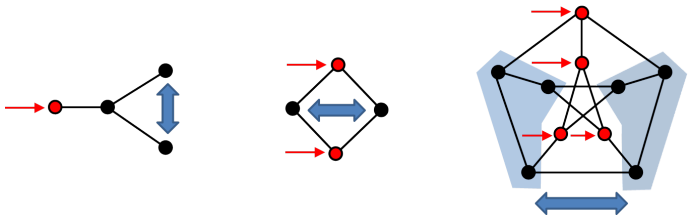
Networked Dynamic Systems



So far in this tutorial...

- graphs and modelling of network systems
- stability of network systems
- input-output properties of network systems





A Graph Structure $\Leftrightarrow$ System Behavior Morphism

We are interested in morphisms between

$$\text{(networks/operations)} \quad \Longleftrightarrow \quad \text{(systems/properties)}$$

Our thesis is that for control theoretic methods to have an impact in the growing field of networks, our techniques should be modular, scalable, and offer flexibility in their use.
Some areas that have been explored in this direction include:

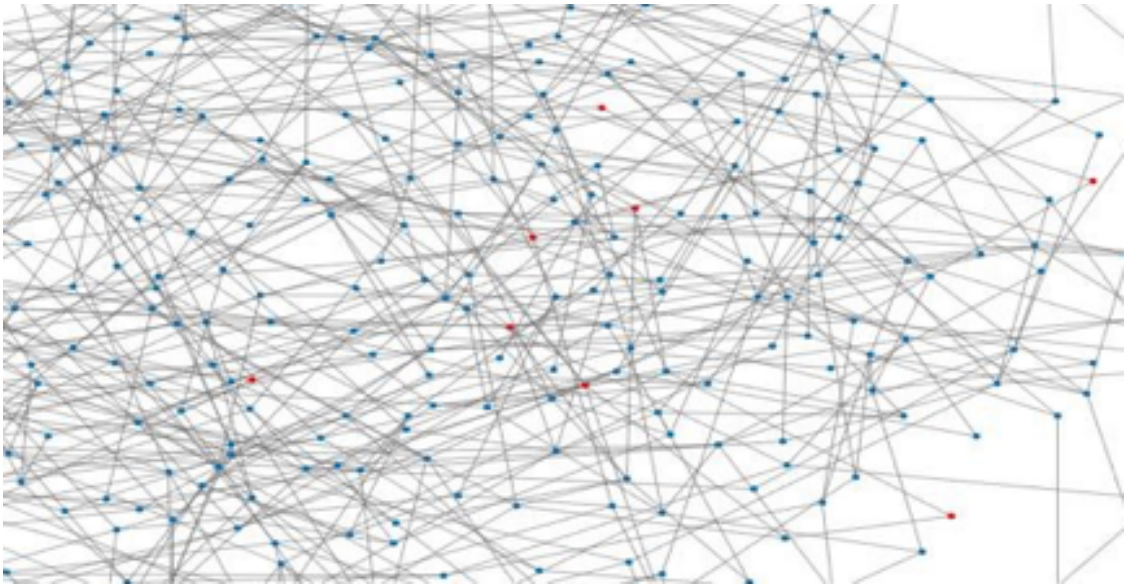
- structural considerations
- compositional perspective/motifs
- approximations
- randomness

We believe this area is highly unexplored!

Table of Contents

Large Scale Networks

How do we approach the analysis of networks that are too large to model?

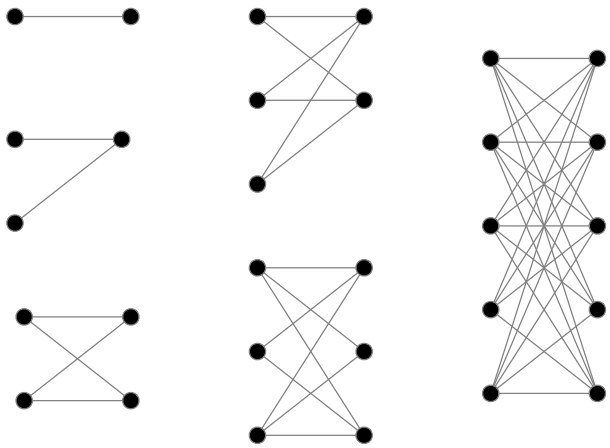


- fault detection and isolation
- power distribution networks
- transportation networks
- internet-of-things
- cyber-pysical systems
- social networks

Extremal Graph Theory

Mantel's Theorem (1907)

If a graph $\mathcal{G}$ on n vertices contains no triangles, then it contains at most $\frac{n^2}{4}$ edges.



The complete bipartite graphs are extremal

Extremal graph theory studies how global properties of a graph (i.e., number of edges) relate to local substructures (i.e., a triangle subgraph)

Forbidden Graphs

Forbidden Subgraph Problem

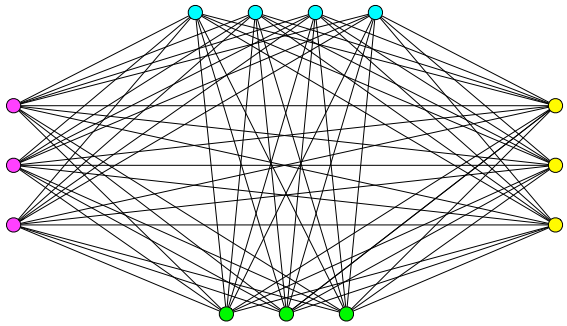
Given a set $\mathbb{H}$ of *forbidden graphs*, what is the maximum number of edges in a graph $\mathcal{G}$ on n nodes (denoted $e(\mathcal{G})$) such that $\mathcal{H} \not\subseteq \mathcal{G}$ for any $\mathcal{H} \in \mathbb{H}$?

$$\text{Extremal Number} \quad ex(n, \mathcal{G}) = \max_{\mathcal{H} \not\subseteq \mathcal{G}} e(\mathcal{G})$$

Generalize Mantel's Theorem for $\mathcal{K}_r$

Túran Graphs $T(n, r)$ - complete r-partite graphs with n vertices

$$e(n, \mathcal{K}_r) \leq \frac{n^2}{2}\left(1 - \frac{1}{r-1}\right)$$

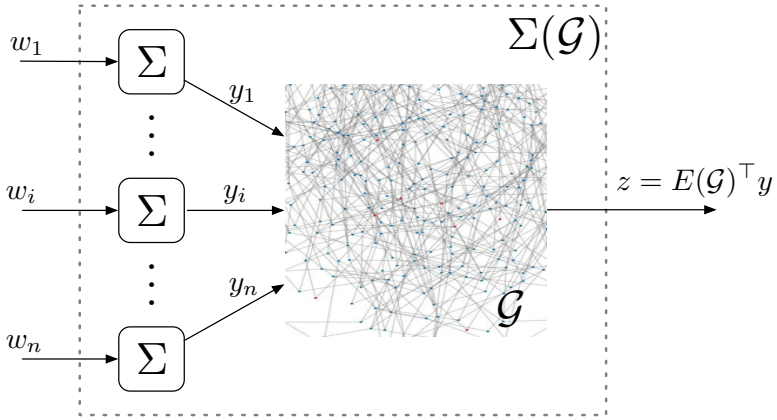


$T(13, 4)$

- avoiding paths of length k
- avoiding Hamiltonian cycles
- avoiding even length cycles
- avoiding edge disjoint cycles

Extremal Networked Systems

A simple example...



A relative sensing network

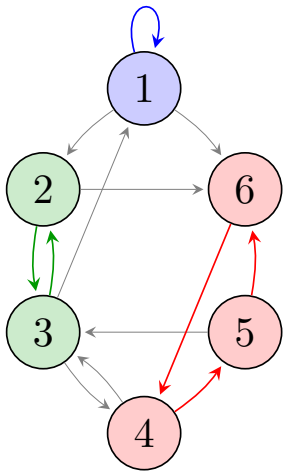
$$\|\Sigma(\mathcal{G})\|_2^2 = 2|\mathcal{E}|\|\Sigma\|_2^2$$

Proposition

Let $\Sigma(\mathcal{G})$ be a relative sensing network with n agents such that $\mathcal{G}$ is K_{r+1}-free. Then the $\mathcal{H}_2$ performance of $\Sigma(\mathcal{G})$ is at most $n^2 \frac{r-1}{r} \|\Sigma\|_2^2$.

recall: k-decompositions

- k-cycle in $\mathcal{G}$: a sequence of k distinct nodes connected by edges.
- Two cycles are disjoint if they have no nodes in common.
- k-decomposition in $\mathcal{G}$: union of *disjoint cycles* covering k nodes.
  A k-decomposition is given by cycles $S_1, \ldots, S_l$ if the S_i are disjoint and $|S_1| + \cdots + |S_l| = k$.
- Hamiltonian cycle (resp. decomposition): n-cycle (resp. decomposition).

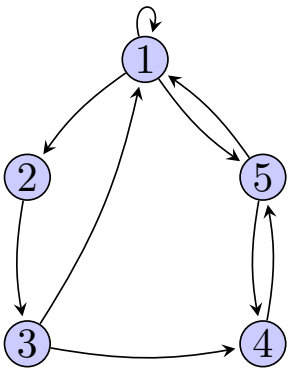


1-cycle $= (1)$
2-cycle: (23)
3-cycle: (456)
3-decomp.: $(1)(23)$ or (456)
4-decomp.: $(1)(456)$
5-decomp.: $(23)(456)$

A necessary condition for stability

Theorem[1]

A digraph $\mathcal{G}$ is stable only if it contains a k-decomposition for each $k = 1, 2, \ldots, n$



$$\begin{bmatrix} * & * & 0 & 0 & * \\ 0 & 0 & * & 0 & 0 \\ * & 0 & 0 & * & 0 \\ 0 & 0 & 0 & 0 & * \\ * & 0 & 0 & * & 0 \end{bmatrix}$$

An extremal question

What is the maximum number of edges in a graph $\mathcal{G}$ on n nodes before a k-decomposition appears?

[1] B. "Sparse Stable Systems", Systems and Control Letters, 2013

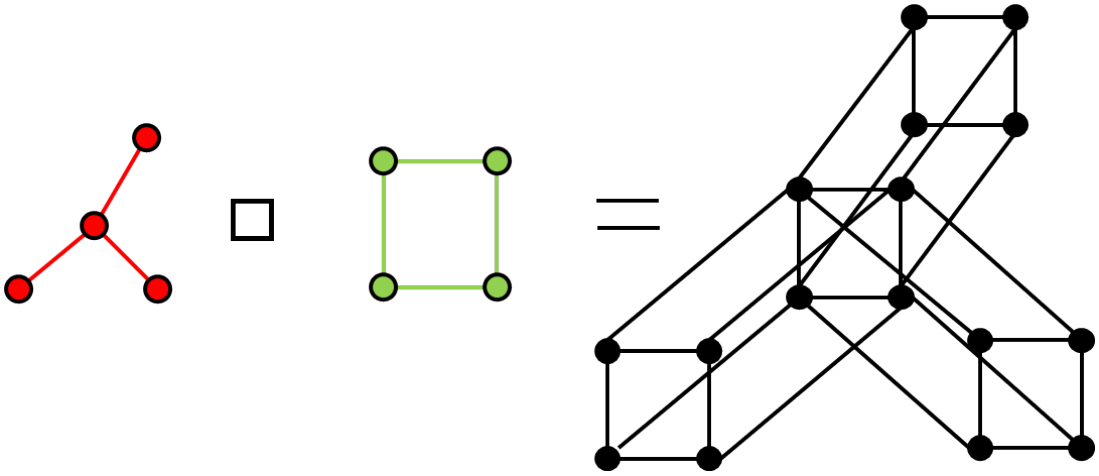
Table of Contents

Compositional approaches: A general setup

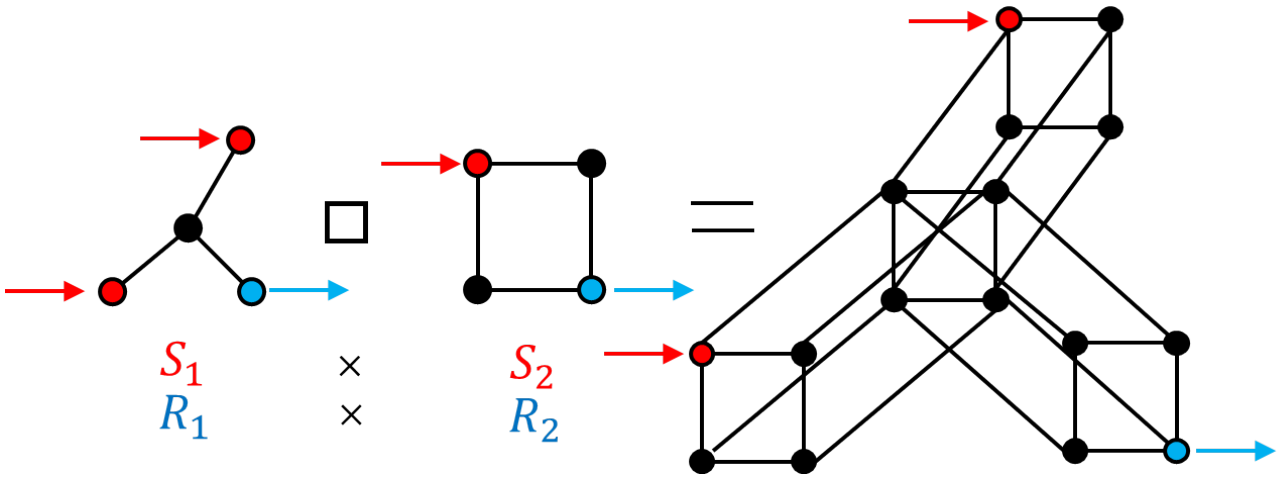
- let $\mathcal{P}$ be a system theoretic property, $\mathbf{G}$ be a class of graphs, and consider $\mathcal{P}(\mathbf{G})$
- consider a subset of $\mathbf{G}$ and examine how $\mathcal{P}$ varies over this subset
- impose algebraic operations on $\mathbf{G}$ and examine how $\mathcal{P}$ behaves with respect to this algebra
- make $\mathbf{G}$ a semi-lattice and examine how the ordering on $\mathbf{G}$ is reflected on $\mathcal{P}$

Case in point: Composite networks

Controllability of the product networks?



Input and Output Set Product



Controllability Factorization - Product Control

Theorem 1: Product Controllability

The dynamics

$$\dot{x}(t) = -A(\prod_{\square} \mathcal{G}_i)x(t) + B(\prod_{\times} S_i)u(t)$$
$$y(t) = C(\prod_{\times} R_i)x(t)$$

where $A(\prod_{\square} \mathcal{G}_i)$ has simple eigenvalues is controllable/observable if and only if

$$\dot{x}_i(t) = -A(\mathcal{G}_i)x_i(t) + B(S_i)u_i(t)$$
$$y_i(t) = C(R_i)x_i(t)$$

is controllable/observable for all i.

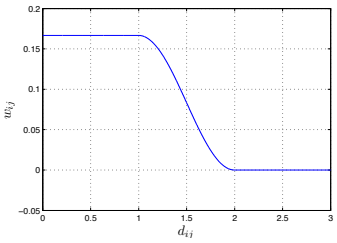
Network Learning

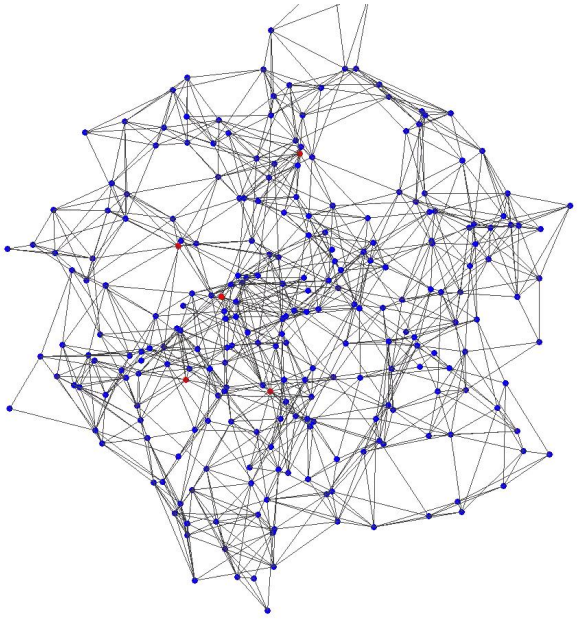
- Sensing accuracy/ confidence is coupled to an edge state, i.e., $w_{ij}(x) = g(\|x_i - x_j\|)$
- Online performance with respect to edge state control

edge states: $x_i(t)$;
coordinated state $y_i(t)$

$$\dot{y}_i(t) = \sum_{j \in N(i)} w_{ij}(x)(y_j(t) - y_i(t))$$

$$\dot{x}_i(t) = f(x_i)$$





Questions: time-scale analysis, learning, gradient flow on space of graphs

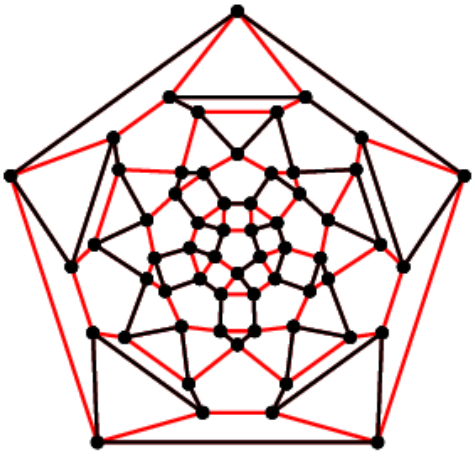
Conclusions

Graph Theory

- Algebraic graph theory
- Geometric graph theory
- Extremal graph theory
- Probabilistic graph theory
- Topological graph theory

Systems Theory

- Stability
- Performance
- Input-Output Properties
- Control Synthesis
- Control Architectures



Thank you!