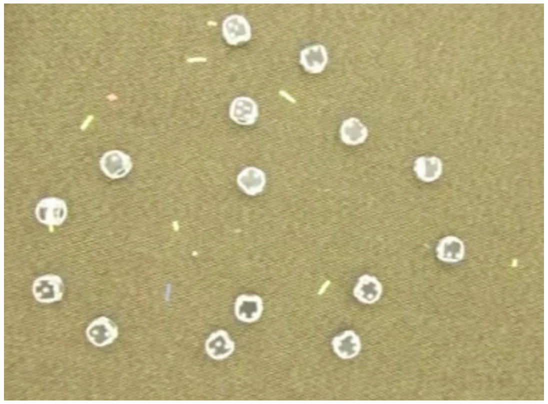
Networked Control
Modeling, Design, and Applications

Magnus Egerstedt

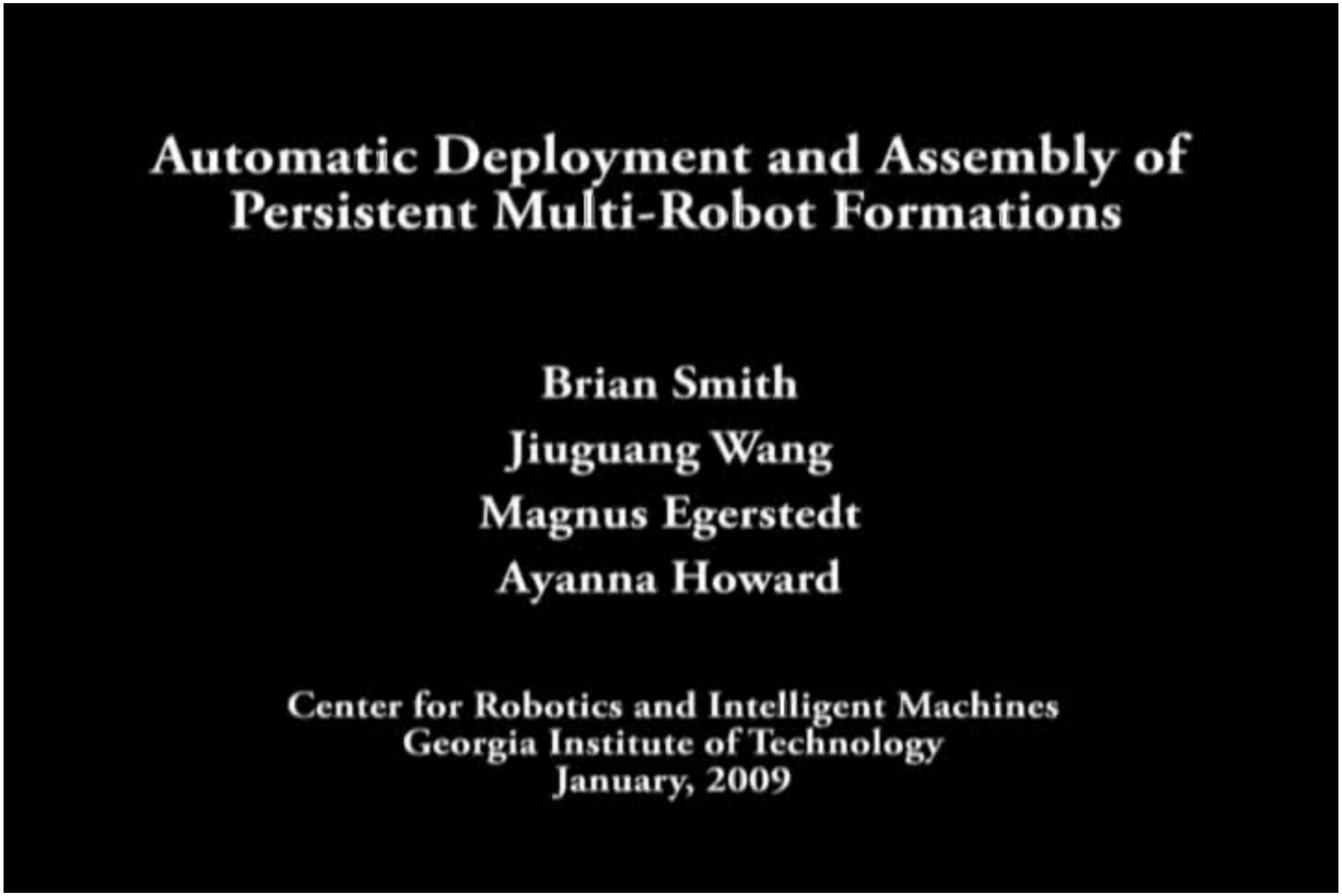
GRITSLab
Electrical and Computer Engineering
Georgia Institute of Technology
www.ece.gatech.edu/~magnus

Outline:

1. Graph-Based Control
2. Multi-Agent Networks
3. Control of Robot Teams
4. Sensor Networks

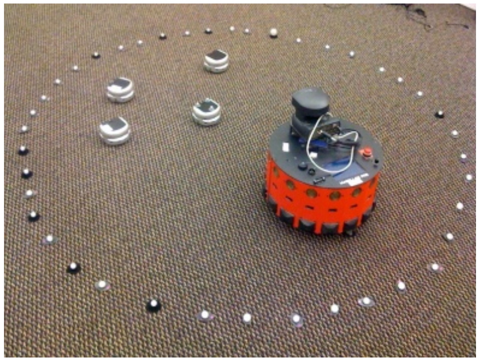


A Mood Picture

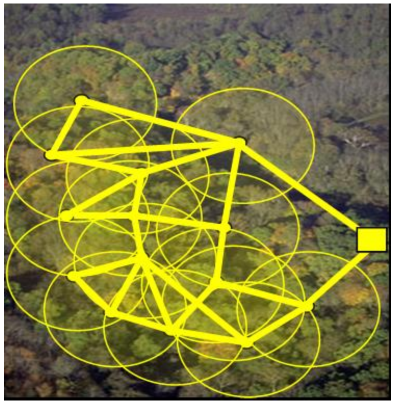


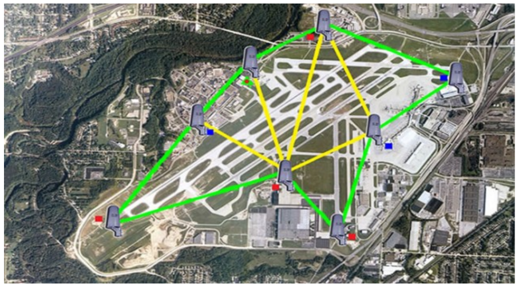
Ruining the Mood…

Application Domains

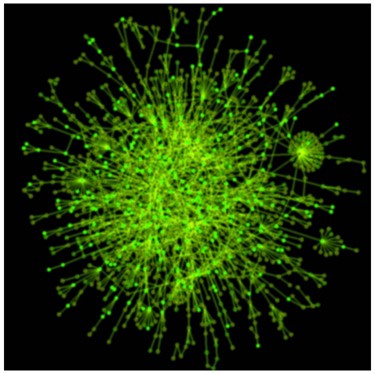


Multi-agent robotics

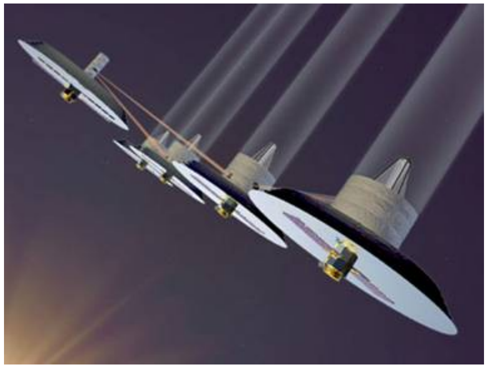




Sensor and communications networks



Biological networks



Coordinated control

The Mandatory Bio-Slide

- As sensor webs, large-scale robot teams, and networked embedded devices emerge, algorithms are needed for inter-connected systems with ***limited communication, computation, and sensing capabilities***

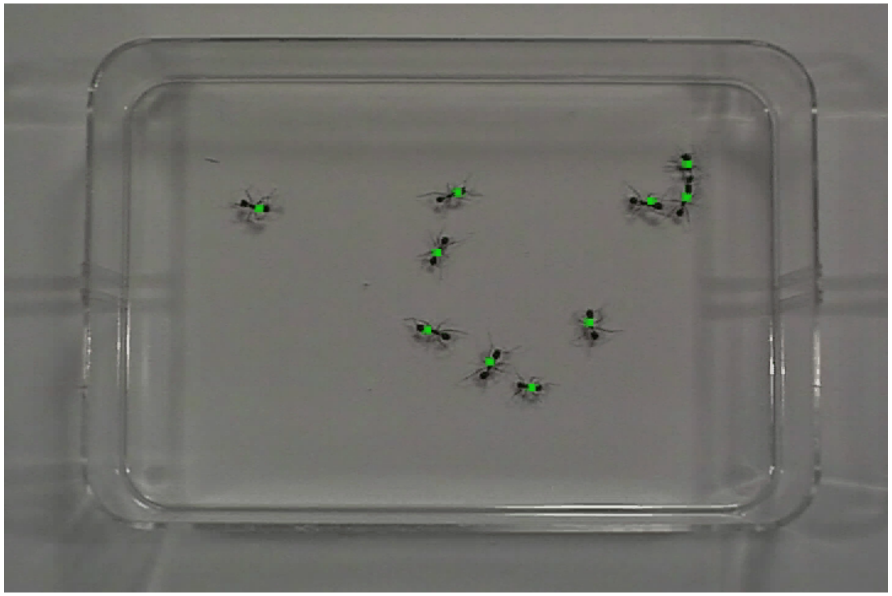
    

- How to effectively control such systems?
  - What is the correct model?
  - What is the correct mode of interaction?
  - Does every individual matter?

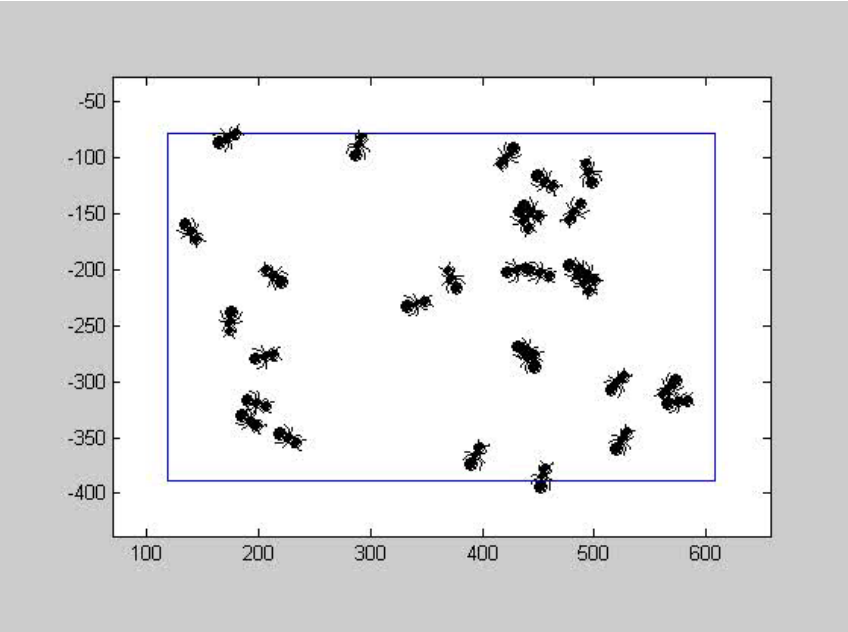
The Starting Point

SESSION 1
GRAPH-BASED CONTROL

Why I Started Caring About Multi-Agent Systems



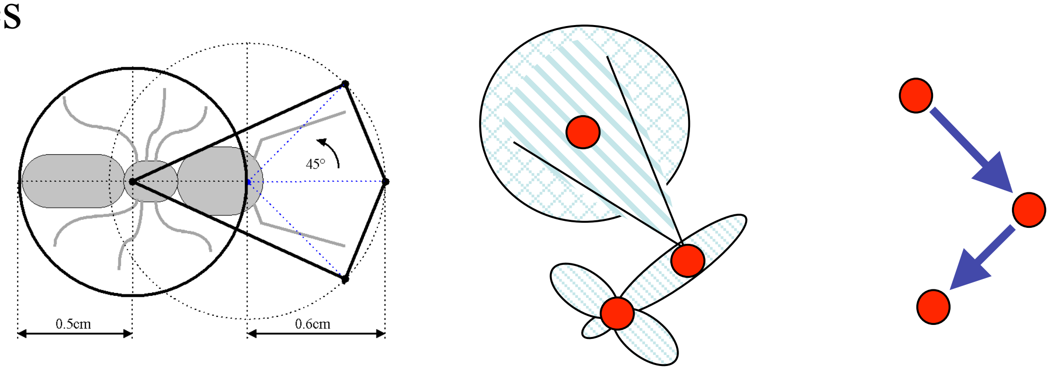
?



"They look like ants."
– Stephen Pratt, Arizona State University

Graphs as Network Abstractions

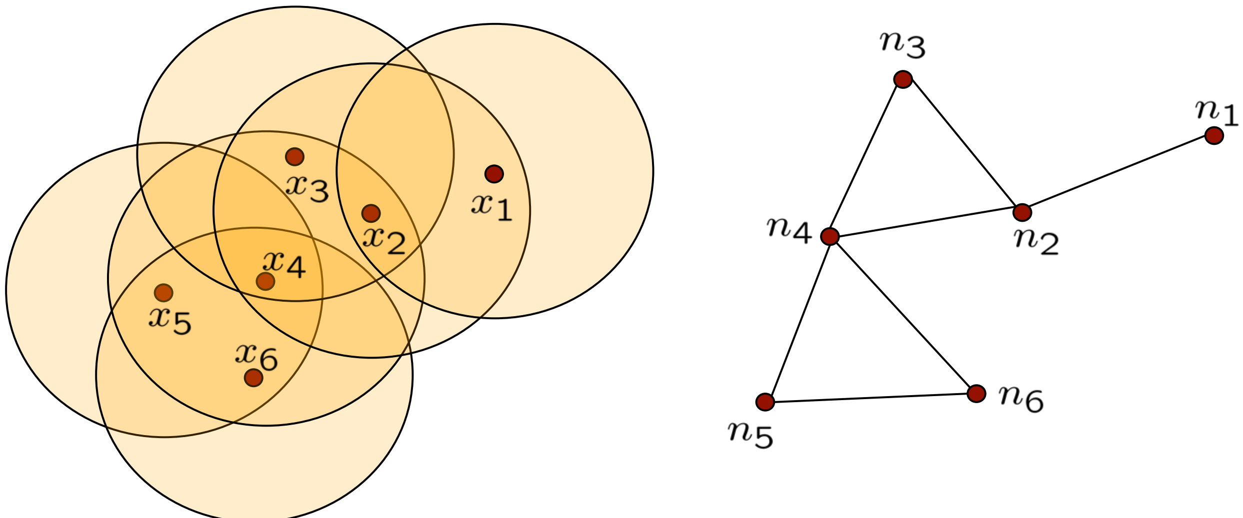
- A networked sensing and actuation system consists of
  - **NODES** - physical entities with limited resources (computation, communication, perception, control)
  - **EDGES** - virtual entities that encode the flow of information between the nodes



- The "right" mathematical object for characterizing such systems at the network-level is a **GRAPH**
  - Purely ***combinatorial*** object (no ***geometry*** or ***dynamics***)
  - The characteristics of the information flow is abstracted away through the (possibly weighted and directed) edges

Graphs as Network Abstractions

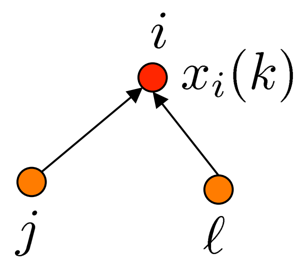
- The connection between the combinatorial graphs and the geometry of the system can for instance be made through geometrically defined edges.
- Examples of such proximity graphs include **disk-graphs**, **Delaunay graphs, visibility graphs**, and **Gabriel graphs** [1].



$$\mathcal{N} = \{n_1, n_2, n_3, n_4, n_5, n_6\}$$
$$\mathcal{E} = \{(n_1, n_2), (n_2, n_3), (n_3, n_4), (n_2, n_4), (n_4, n_5), (n_4, n_6), (n_5, n_6)\}$$

The Basic Setup

- $x_i(k) =$ "state" at node i at time k
- $N_i(k) =$ "neighbors" to agent i



- Information "available to agent i

$$I_i^c(k) = \{x_j(k) \mid j \in N_i(k)\}$$ ⟵ common ref. frame (comms.)

or

$$I_i^r(k) = \{x_i(k) - x_j(k) \mid j \in N_i(k)\}$$ ⟵ relative info. (sensing)

- Update rule:

$$x_i(k+1) = F_i(x_i(k), I_i(k))$$ ⟵ discrete time

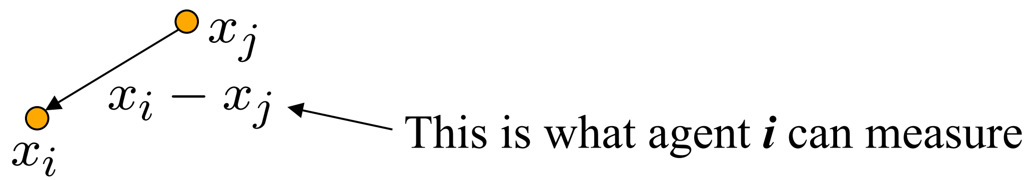
or

$$\dot{x}_i(t) = F_i(x_i(t), I_i(t))$$ ⟵ continuous time

- ***How pick the update rule?***

Rendezvous – A Canonical Problem

- Given a collection of mobile agents who can only measure the relative displacement of their neighbors (no global coordinates)



- Problem: Have all the agents meet at the same (unspecified) position

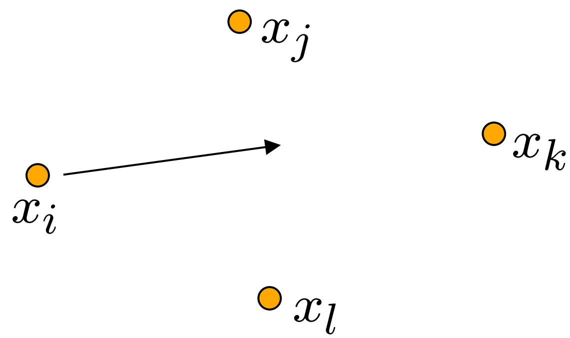
- If there are only two agents, it makes sense to have them drive towards each other, i.e.

$$\dot{x}_1 = -\gamma_1(x_1 - x_2)$$
$$\dot{x}_2 = -\gamma_2(x_2 - x_1)$$

- If $\gamma_1 = \gamma_2$ they should meet halfway

Rendezvous – A Canonical Problem

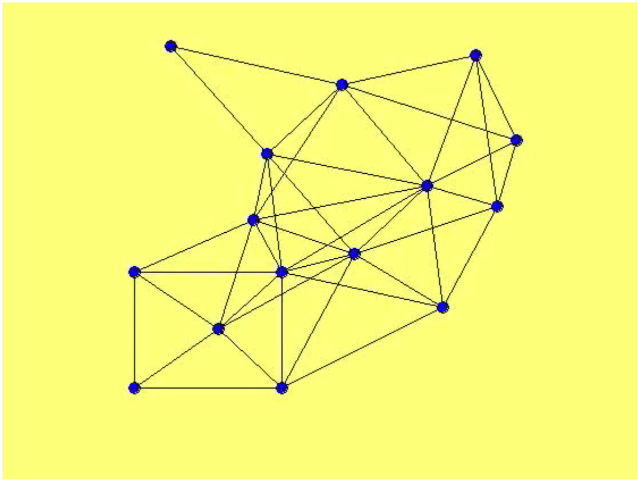
- If there are more than two agents, they should probably aim towards the centroid of their neighbors (or something similar)



$$\dot{x}_i = -\gamma \sum_{j \in \mathcal{N}_i} (x_i - x_j)$$

Fact [2-4]: As long as the graph is connected (iff), the ***consensus equation*** drives all agents to the same state value

$$\lim_{t \to \infty} x_i(t) = \bar{x} = \frac{1}{N} \sum_{j=1}^{N} x_j(0)$$



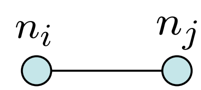
Algebraic Graph Theory

- To show this, we need some tools…
- Algebraic graph theory provides a bridge between the combinatorial graph objects and their matrix representations
  - **Degree matrix**:

$$D = \text{diag}(\text{deg}(n_1), \ldots, \text{deg}(n_N))$$

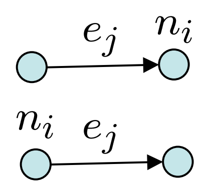
  - **Adjacency matrix**:

$$A = [a_{ij}], \;\; a_{ij} = \begin{cases} 1 & \text{if} \\ 0 & \text{o.w.} \end{cases}$$



  - **Incidence matrix** (directed graphs):

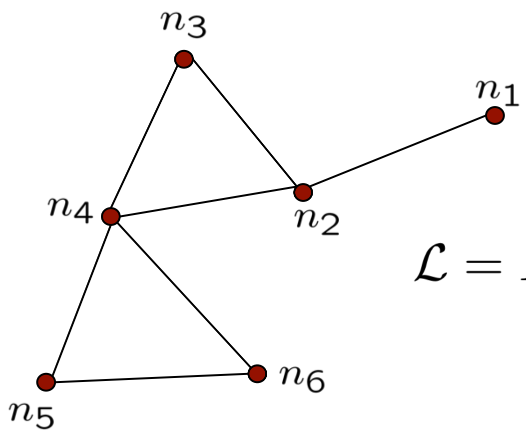
$$\mathcal{I} = [\iota_{ij}], \;\; \iota_{ij} = \begin{cases} 1 & \text{if} \\ -1 & \text{if} \\ 0 & \text{o.w.} \end{cases}$$



  - **Graph Laplacian**:

$$\mathcal{L} = D - A = \mathcal{I}\mathcal{I}^T$$

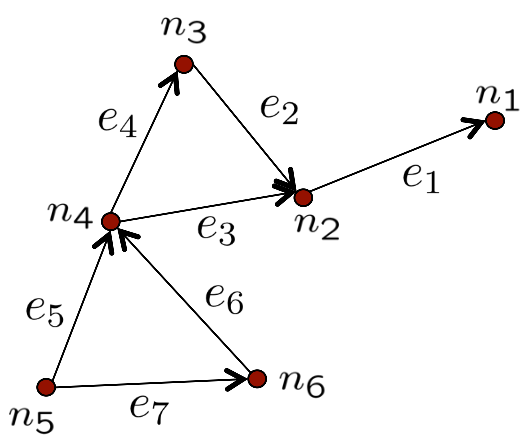
Algebraic Graph Theory - Example



$$\mathcal{L} = D - A = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 4 & -1 & -1 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & -1 & 2 \end{bmatrix}$$

$$D = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix} \qquad A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Algebraic Graph Theory - Example



$$\mathcal{I} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

$$\mathcal{L} = \mathcal{I}\mathcal{I}^T = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 4 & -1 & -1 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & -1 & 2 \end{bmatrix}$$

The Consensus Equation

- The reason why the graph Laplacian is so important is through the already seen "consensus equation"

$$\dot{x}_i = -\sum_{j \in \mathcal{N}_i} (x_i - x_j), \;\; i = 1, \ldots, N$$

or equivalently (W.L.O.G. scalar agents)

$$\left.\begin{array}{l} \dot{x}_i = -\mathrm{deg}(n_i) x_i + \sum_{j=1}^{N} a_{ij} x_j \\ x = \left[\begin{array}{cccc} x_1 & x_2 & \cdots & x_N \end{array} \right]^T \end{array} \right\} \Rightarrow \;\; \dot{x} = -\mathcal{L} x$$

- This is an autonomous LTI system whose convergence properties depend purely on the spectral properties of the Laplacian.

Graph Laplacians: Useful Properties

- It is orientation independent
- It is symmetric and positive semi-definite
- If the graph is ***connected*** then

$$\text{eig}(\mathcal{L}) = \{\lambda_1, \ldots, \lambda_N\}, \text{ with } 0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N$$

$$\text{eigv}(\mathcal{L}) = \{\nu_1, \ldots, \nu_N\}, \text{ with } null(\mathcal{L}) = \text{span}\{\nu_1\} = \text{span}\{\mathbf{1}\}$$

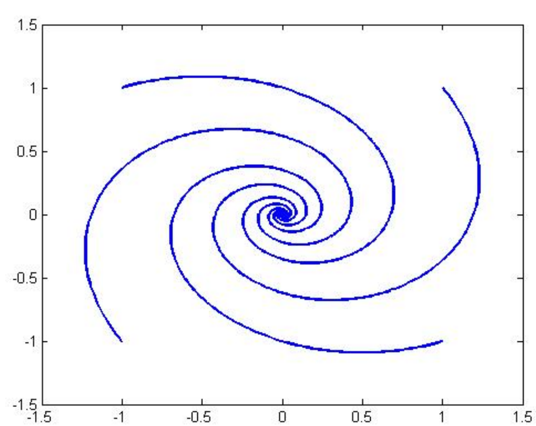
Stability - Basics

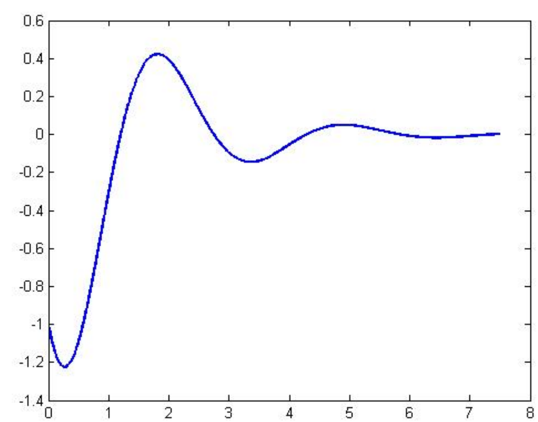
- The stability properties (what happens as time goes to infinity?) of a linear, time-invariant system is completely determined by the eigenvalues of the system matrix

$$\dot{x} = Ax \quad (\dot{x} = -Lx)$$

- Eigenvalues $\lambda(A) = \lambda_1, \ldots, \lambda_n$

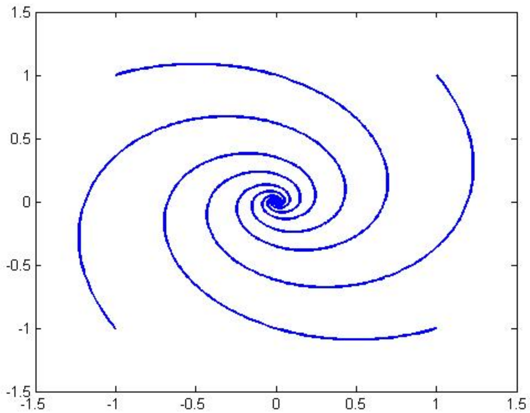
- Asymptotic stability: $\text{Re}(\lambda_i) < 0, \; i = 1, \ldots, n \; \Rightarrow \; \lim_{t \to \infty} x(t) = 0$

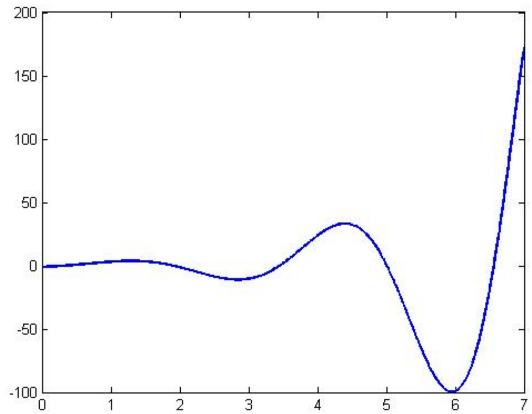




Stability - Basics

- Unstable: $\exists i$ s.t. $\text{Re}(\lambda_i) > 0, \quad \Rightarrow \quad \lim_{t\to\infty} x(t) = \infty$

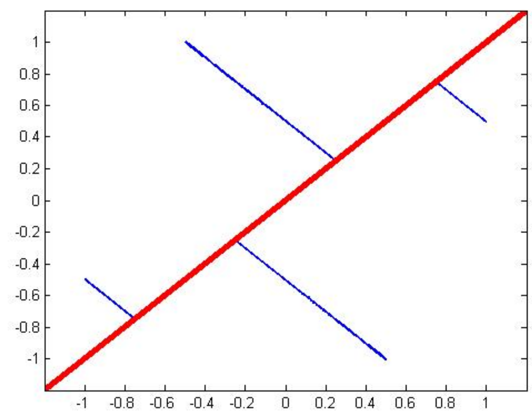




- Critically stable:

$0 = \lambda_1 > \lambda_2 \geq \ldots \geq \lambda_n, \quad \Rightarrow$
$\lim_{t\to\infty} x(t) \in \text{ null}(A)$

This is the case for the consensus equation



Static Consensus

- If the graph is static and connected, under the consensus equation, the states will reach null(L)
- Fact (again):

$$\text{null}(L) = \text{span}\{\mathbf{1}\}, \ x \in \text{null}(L) \ \Leftrightarrow \ x = \begin{bmatrix} \alpha \\ \alpha \\ \vdots \\ \alpha \end{bmatrix}, \ \alpha \in \Re$$

- So all the agents state values will end up at the same value, i.e. the consensus/rendezvous problem is solved!

$$\dot{x}_i = -\sum_{j \in N_i} (x_i - x_j) \ \Rightarrow \ \lim_{t \to \infty} x_i(t) = \frac{1}{n} \sum_{j=1}^{n} x_j(0) = \frac{1}{n} \mathbf{1}^T x(0)$$

Formation Control

- Being able to reach consensus goes beyond solving the rendezvous problem.
- Formation control:

$$\underset{\text{agent positions}}{x_1, \ldots, x_N} \longrightarrow \underset{\text{target positions}}{y_1, \ldots, y_N}$$

- But, formation achieved if the agents are in any translated version of the targets, i.e.,

$$x_i = y_i + \tau, \ \forall i, \ \text{for some } \tau$$

- Enter the consensus equation [5]:

$$e_i = x_i - y_i$$

$$\dot{e}_i = -\sum_{j \in N_i} (e_i - e_j)$$

$$e_i(\infty) = e_j(\infty) = \tau$$

$$\dot{x}_i = \sum_{j \in N_i} [(x_i - x_j) - (y_i - y_j)]$$

$$x_i(\infty) = y_i + \tau, \ \forall i$$

Formation Control

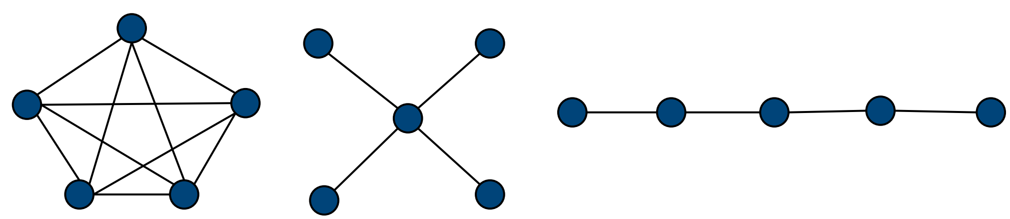
Convergence Rates

- The second smallest eigenvalue of the graph Laplacian is really important!
- Algebraic Connectivity (= 0 if and only if graph is disconnected)
- Fiedler Value (robustness measure)
- ***Convergence Rate:***

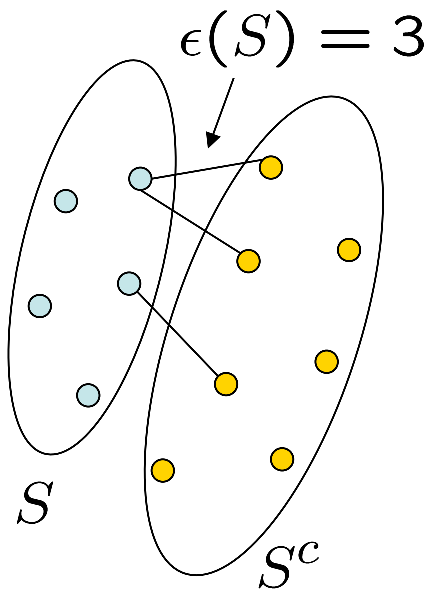
$$\|x(t) - \frac{1}{n}\mathbf{1}^T x(0)\| \le Ce^{-\lambda_2 t}$$

- **Punch-line:** The more connected the network is, the faster it converges (and the more information needs to be shuffled through the network)

- Complete graph: $\lambda_2 = n$
- Star graph: $\lambda_2 = 1$
- Path graph: $\lambda_2 < 1$



Cheeger's Inequality



$$\phi(S) = \frac{\epsilon(S)}{\min\{|S|, |S^c|\}}$$

(measures how many edges need to be cut to make the two subsets disconnected as compared to the number of nodes that are lost)

isoperimetric number:

$$\phi(G) = \min_S \phi(S)$$

(measures the robustness of the graph)

$$\phi(G) \geq \lambda_2 \geq \frac{\phi(G)^2}{2\Delta(G)}$$

Beyond Static Consensus

- So far, the consensus equation will drive the node states to the same value if the graph is static and connected.
- But, this is clearly not the case in a number of situations:
  - **Edges = communication links**
    - Random failures
    - Dependence on the position (shadowing,…)
    - Interference
    - Bandwidth issues
  - **Edges = sensing**
    - Range-limited sensors
    - Occlusions
    - Weirdly shaped sensing regions

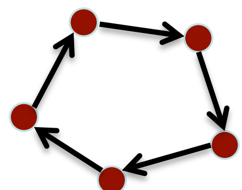
Summary I

- Graphs are natural abstractions (combinatorics instead of geometry)
- Consensus problem (and equation)
- Static Graphs:
    - Undirected: Average consensus iff G is connected
- Need richer network models!

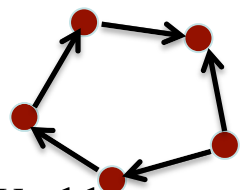
SESSION 2
MULTI-AGENT NETWORKS

Variations on the Theme: Directed Graphs

- Instead of connectivity, we need directed notions:
  - ***Strong connectivity*** = there exists a directed path between any two nodes
  - ***Weak connectivity*** = the disoriented graph is connected



Strongly connected



Weakly connected

- Directed consensus:

$$\dot{x}_i = -\sum_{j \in N_i^{in}} (x_i - x_j)$$

Directed Consensus

- Undirected case: Graph is connected = sufficient information is flowing through the network
- Clearly, in the directed case, if the graph is strongly connected, we have the same result

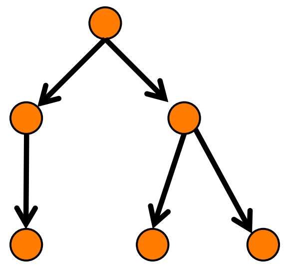
- **Theorem**: If G is strongly connected, the consensus equation achieves

$$\lim_{t \to \infty} (x_i - x_j) = 0, \ \forall i, j$$

- This is an unnecessarily strong condition! Unfortunately, weak connectivity is too weak.

Rooted Outbranching Trees

- Consider the following structure



- Seems like all agents should end up at the root node

- **Theorem [2]**: Consensus in a directed network is achieved if and only if G contains a spanning rooted outbranching tree (ROT).

Where Do the Agents End Up?

- Recall: Undirected case

$$\lim_{t\to\infty} x_i(t) = \bar{x}(0) = \frac{1}{N}\sum_{j=1}^{N} x_j(0),\ \forall i$$

- How show that?
- The centroid is invariant under the consensus equation

$$\dot{\bar{x}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\in N_i}(x_j - x_i) = 0$$

- And since the agents end up at the same location, they must end up at the static centroid (average consensus).

Where Do the Agents End Up?

- When is the centroid invariant in the directed case?

$$q^T L = 0,\ w = q^T x \ \Rightarrow \ \dot{w} = q^T \dot{x} = -q^T L x = 0$$

- w is invariant under the consensus equation
- The centroid is given by

$$\bar{x} = \frac{1}{N}\mathbf{1}^T x$$

which thus is invariant if

$$\mathbf{1}^T L = 0$$

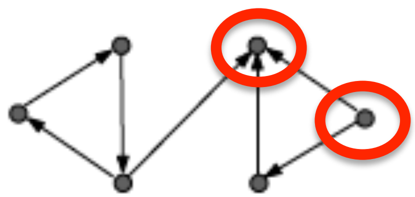
- **Def**: G is balanced if

$$deg^{in}(i) = deg^{out}(i),\ \forall i \in V \ \Leftrightarrow \ \mathbf{1}^T L = 0$$

- **Theorem [2]:** If G is balanced and consensus is achieved then average consensus is achieved!

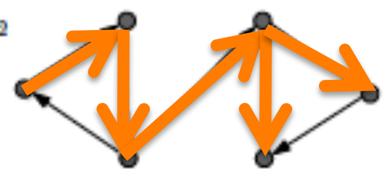
Example

G_1



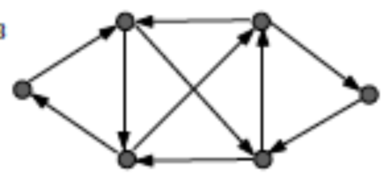
No ROT – Consensus is not achieved

G_2



ROT but not balanced – Consensus but not average consensus is achieved

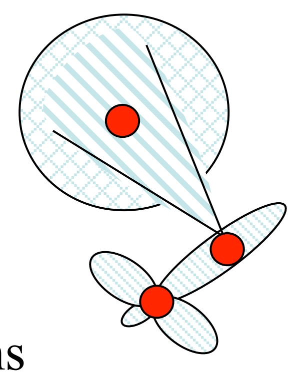
G_3

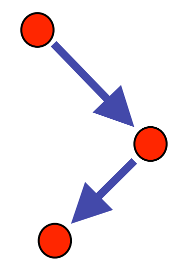


ROT and balanced – Average consensus is achieved

Beyond Static Consensus

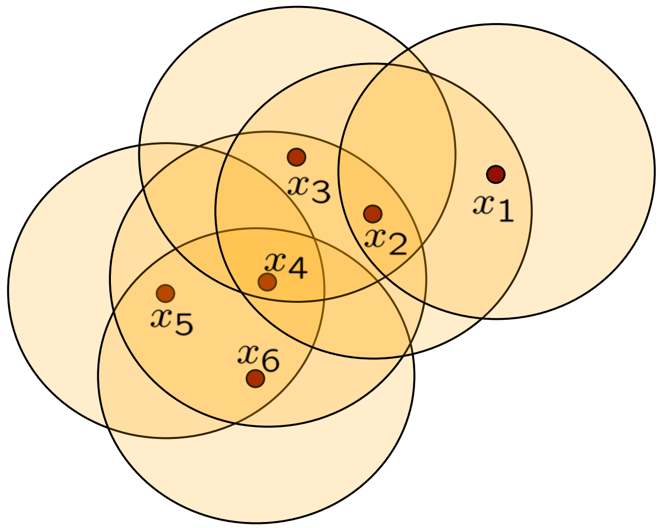
- So far, the consensus equation will drive the node states to the same value if the graph is static and connected.
- But, this is clearly not the case in a number of situations:
  - **Edges = communication links**
    - Random failures
    - Dependence on the position (shadowing,…)
    - Interference
    - Bandwidth issues
  - **Edges = sensing**
    - Range-limited sensors
    - Occlusions
    - Weirdly shaped sensing regions

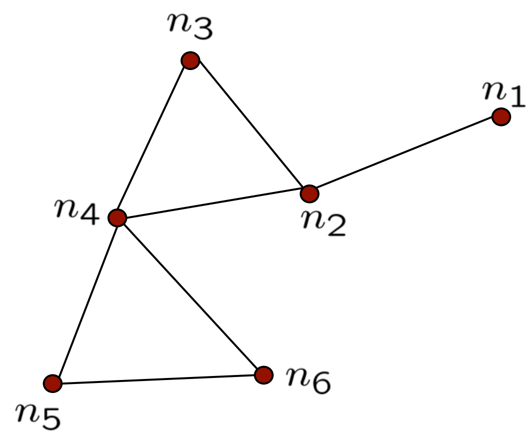




Dynamic Graphs

- In most cases, edges correspond to available sensor or communication data, i.e., the edge set is time varying





- We now have a switched system and spectral properties do not help for establishing stability
- Need to use Lyapunov functions

Lyapunov Functions

- Given a nonlinear system

$$\dot{x} = f(x)$$

- V is a (weak) Lyapunov function if

$$\begin{aligned} (i) \quad & V(x) > 0, \ \forall x \neq 0 \\ (ii) \quad & \dot{V}(x) = \frac{\partial V}{\partial x} f(x) < 0, \ \forall x \neq 0 \quad (\leq 0) \end{aligned}$$

- The system is asymptotically stable if and only if there exists a Lyapunov function
- [LaSalle's Invariance Principle] If it has a weak Lyapunov function the system converges asymptotically to the largest invariant set (f=0) s.t. the derivative is 0

Switched Systems

- Similarly, consider a switched system

$$\dot{x} = f_\sigma(x), \quad \sigma(t) \in \{1, \ldots, q\}$$

- The system is ***universally asymptotically stable*** if it is asymptotically stable for all switch sequences

- A function V is a common Lyapunov function if it is a Lyapunov function to all subsystems

$$V > 0, \ \frac{\partial V}{\partial x} f_i < 0, \ i = 1, \ldots, q$$

- **Theorem [9]**: Universal stability if and only if there exists a common Lyapunov function. (Similar for LaSalle.)

Switched Networked Systems

- Switched consensus equation

$$\dot{x} = -L_\sigma x$$

- Consider the following candidate Lyapunov function

$$V(x) = \frac{1}{2}x^T x, \quad \dot{V}(x) = x^T \dot{x} = -x^T L_\sigma x$$

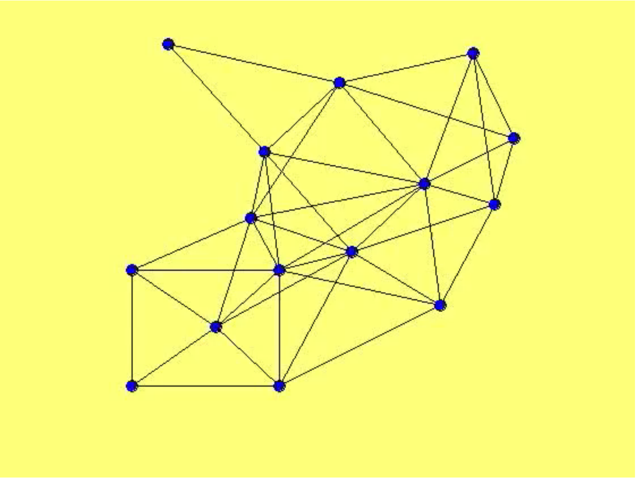
- This is a common (weak) Lyapunov function as long as *G* is connected for all times

- Using LaSalle's theorem, we know that in this case, it ends up in the null-space of the Laplacians

Switched Consensus

Theorem [2-4]: As long as the graph stays connected, the ***consensus equation*** drives all agents to the same state value

$$\lim_{t\to\infty} x_i(t) = \bar{x} = \frac{1}{N}\sum_{j=1}^{N} x_j(0)$$

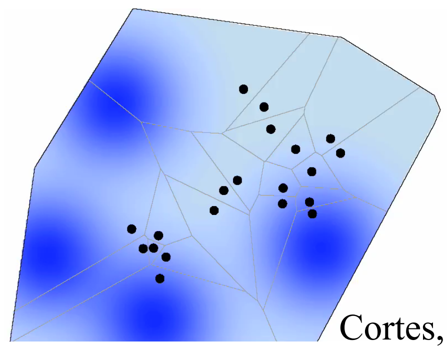


Adding Weights

- Sometimes it makes sense to add weights

$$\dot{x}_i = -\sum_{j \in N_i} w(\|x_i - x_j\|)(x_i - x_j)$$

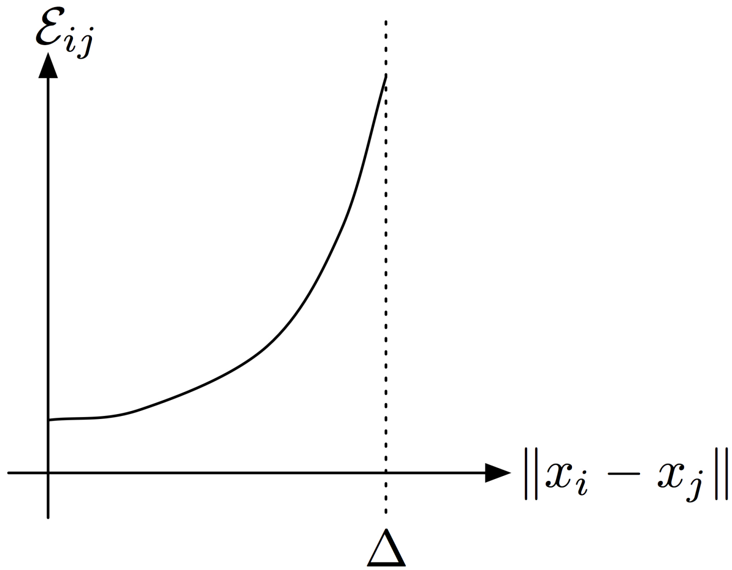
- Collision avoidance
- Coverage
- Connectivity maintenance



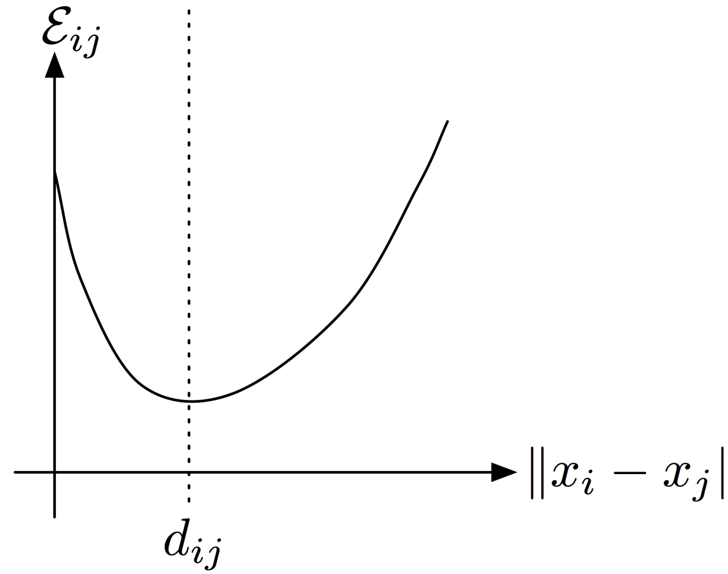
Cortes, Martinez, Bullo

Weights Through Edge Tensions

- How select appropriate weights?
- Let an edge tension be given by $\mathcal{E} = \sum_{i=1}^{N} \sum_{j=1}^{N} a_{i,j} \mathcal{E}_{i,j}(\|x_i - x_j\|)$



Connectivity Maintenance



Formation Control

Weights Through Edge Tensions

- How select appropriate weights?
- Let an edge tension be given by $\mathcal{E} = \sum_{i=1}^{N} \sum_{j=1}^{N} a_{i,j} \mathcal{E}_{i,j}(\|x_i - x_j\|)$
- We get

$$\frac{\partial \mathcal{E}_{i,j}}{\partial x_i} = w_{i,j}(\|x_i - x_j\|)(x_i - x_j)$$

- Gradient descent

$$\dot{x}_i = -\frac{\partial \mathcal{E}}{\partial x_i} = -\sum_{j \in N_i} w_{i,j}(\|x_i - x_j\|)(x_i - x_j)$$

$$\frac{d\mathcal{E}}{dt} = \frac{\partial \mathcal{E}}{\partial x}\dot{x} = -\left\| \frac{\partial \mathcal{E}}{\partial x} \right\|^2$$

Energy is non-increasing!
(weak Lyapunov function)

Examples

$$\mathcal{E}_{ij} = \frac{1}{2}\|x_i - x_j\|^2 \;\Rightarrow\; w_{ij} = 1$$

$$\dot{x}_i = -\sum_{j \in N_i} (x_i - x_j)$$

Standard, linear consensus!

$$\mathcal{E}_{ij} = \|x_i - x_j\| \;\Rightarrow\; w_{ij} = \frac{1}{\|x_i - x_j\|}$$

$$\dot{x}_i = -\sum_{j \in N_i} \frac{x_i - x_j}{\|x_i - x_j\|}$$

Unit vector (biology)

Examples

$$\mathcal{E}_{ij} = \frac{1}{2}(\|x_i - x_j\| - d_{ij})^2 \;\Rightarrow\; w_{ij} = \frac{\|x_i - x_j\| - d_{ij}}{\|x_i - x_j\|}$$

$$\dot{x}_i = -\sum_{j \in N_i} \frac{(\|x_i - x_j\| - d_{ij})(x_i - x_j)}{\|x_i - x_j\|}$$

Formation control

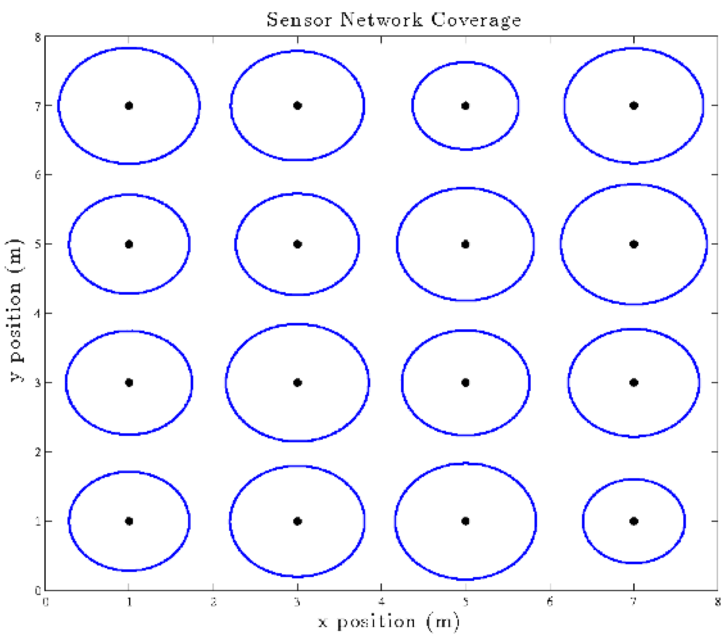
$$\mathcal{E}_{ij} = \frac{\|x_i - x_j\|^2}{\Delta - \|x_i - x_j\|} \;\Rightarrow\; w_{ij} = \frac{2\Delta - \|x_i - x_j\|}{(\Delta - \|x_i - x_j\|)^2}$$

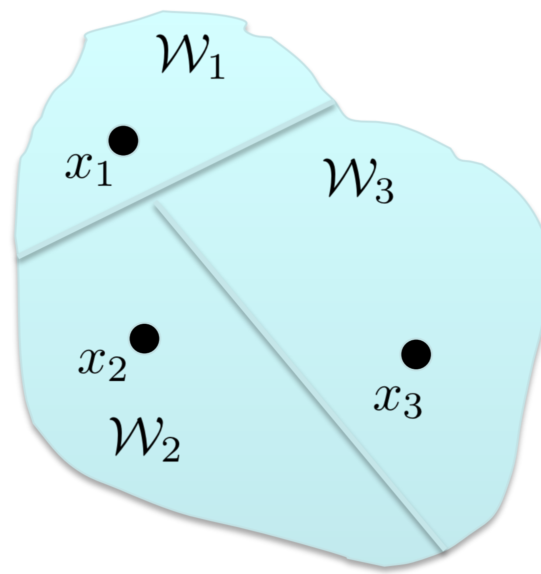
$$\dot{x}_i = -\sum_{j \in N_i} \frac{(2\Delta - \|x_i - x_j\|)(x_i - x_j)}{(\Delta - \|x_i - x_j\|)^2}$$

Connectivity maintenance

Coverage Control

- Objective: Deploy sensor nodes in a distributed manner such that an area of interest is covered





- Idea: Divide the responsibility between nodes into regions

Coverage Control

- The coverage cost:

$$J(x,\mathcal{W}) = \frac{1}{2}\sum_{i=1}^{N}\int_{\mathcal{W}_i} \|x_i - q\|^2 dq$$

- Simplify (not optimal):

$$\hat{J}(x) = \frac{1}{2}\sum_{i=1}^{N}\int_{\mathcal{V}_i(x)} \|x_i - q\|^2 dq$$

where the Voronoi regions are given by

$$\mathcal{V}_i(x) = \{q \in \mathcal{D} \mid \|x_i - q\| \leq \|x_j - q\|\}$$

Deployment

- Using a gradient descent (cost = weak Lyapunov function)

$$\dot{x}_i = -\frac{\partial \hat{J}}{\partial x_i} \quad \Rightarrow \quad \frac{d}{dt}\hat{J} - \left\| \frac{\partial \hat{J}}{\partial x} \right\|^2$$

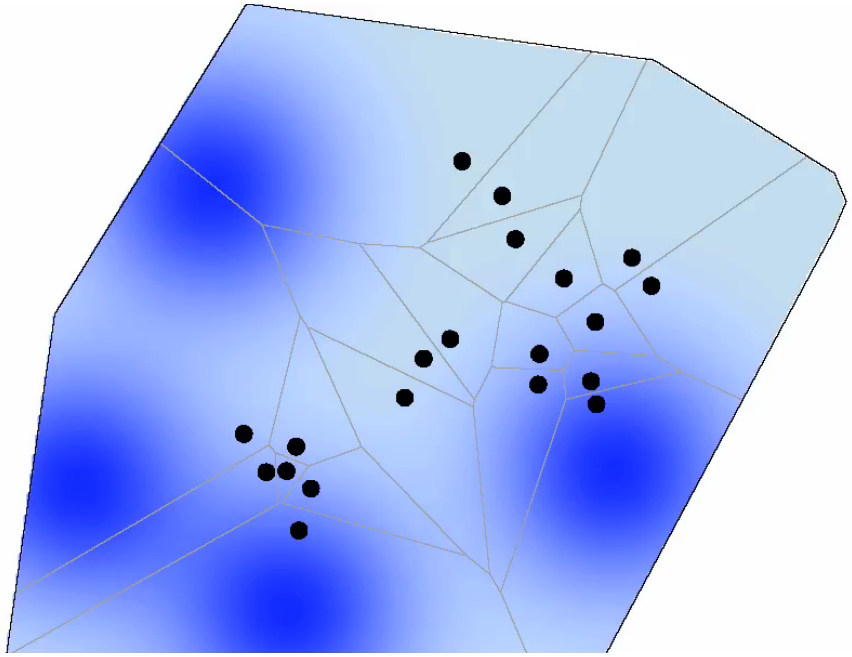
$$\dot{x}_i = -\int_{\mathcal{V}_i(x)} (x_i - q)dq$$

- We only care about directions so this can be re-written as Lloyd's Algorithm [1]

$$\dot{x}_i = \rho_i(x) - x_i$$

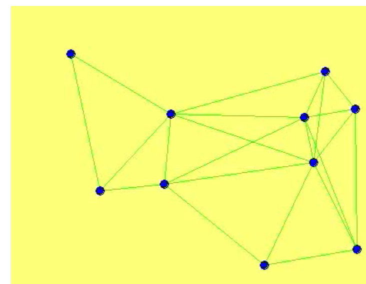
Deployment

- Lloyd's Algorithm:
  - Converges to a local minimum to the simplified cost
  - Converges to a Central Voronoi Tessellation
  - It is decentralized

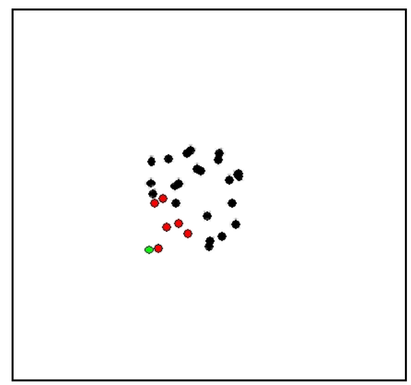


Graph-Based Control

- In fact, based on variations of the consensus equation, a number of different multi-agent problems have been "solved", e.g.
  - **Formation control** (How drive the collection to a predetermined configuration? [2,5])
  - **Coverage control** (How produce triangulations or other regular structures? [1,6])



- ***OK – fine. Now what?***
- Need to be able to **reprogram and redeploy** multi-agent systems (**HSI = Human-Swarm Interactions**)
- This can be achieved through active control of so-called leader-nodes



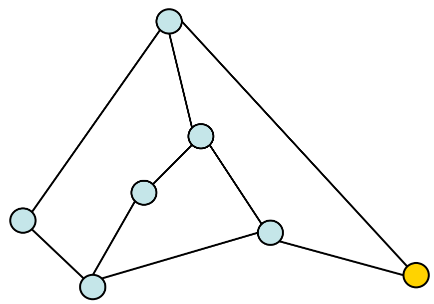
Summary II

- Static Graphs:
  - Undirected: Average consensus iff G is connected
  - Directed: Consensus iff G contains a spanning, outbranching tree
  - Directed: Average consensus if consensus and G is balanced
- Switching Graphs:
  - Undirected: Average consensus if G is connected for all times
  - Directed: Consensus if G contains a spanning, outbranching tree for all times
  - Directed: Average consensus if consensus and G is balanced for all times
- Additional objectives is achieved by adding weights (edge-tension energies as weak Lyapunov functions)

SESSION 3
CONTROL OF ROBOT TEAMS

Leader (Anchor) Nodes

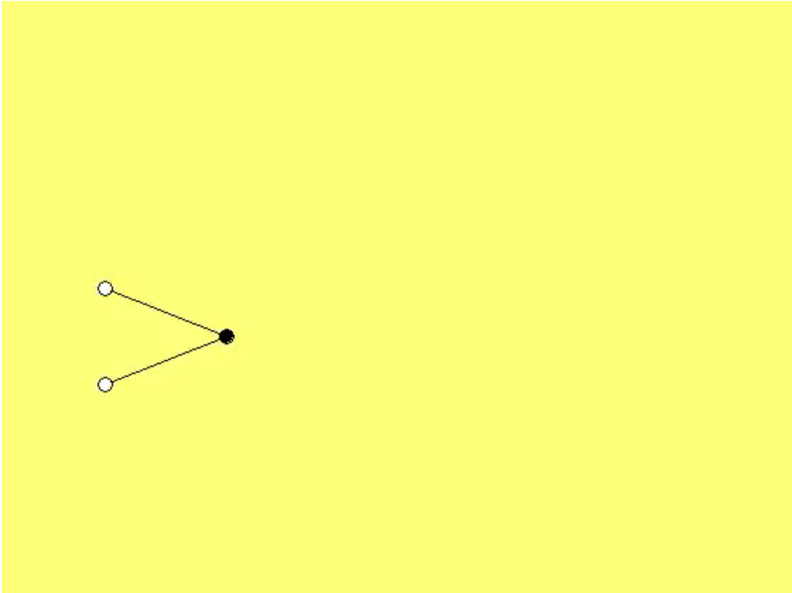
- **Key idea**: Let some subset of the agents act as control inputs and let the rest run some cohesion ensuring control protocol



A Mood-Picture

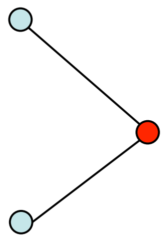
Graph-Based Controllability?

- We would like to be able to determine controllability properties of these systems directly from the graph topology



- For this we need to tap into the world of algebraic graph-theory.
- But first, some illustrative examples

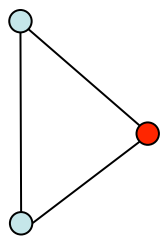
Some Examples



Not controllable!

Why?

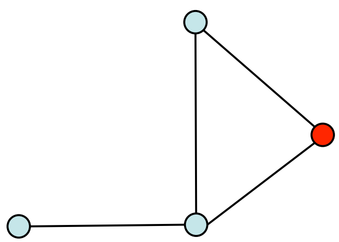
$$\dot{x}_1 = -(x_1 - u), \ \dot{x}_2 = -(x_2 - u)$$
$$x_1(0) = x_2(0) \ \Rightarrow \ x_1(t) = x_2(t) \ \forall u, t \geq 0$$



Not controllable!

Why? - Same reason!

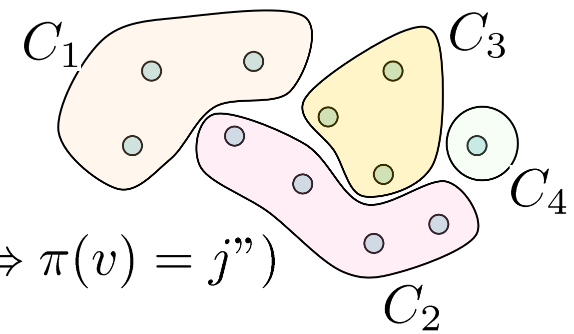
$$\dot{x}_1 = -(x_1 - u) - (x_1 - x_2)$$
$$\dot{x}_2 = -(x_2 - u) - (x_2 - x_1)$$
$$x_1(0) = x_2(0) \ \Rightarrow \ x_1(t) = x_2(t) \ \forall u, t \geq 0$$



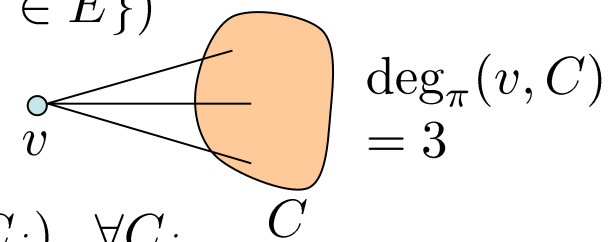
Controllable!

Why? - Somehow it seems like some kind of "symmetry" has been broken.

Symmetry? - External Equitable Partitions

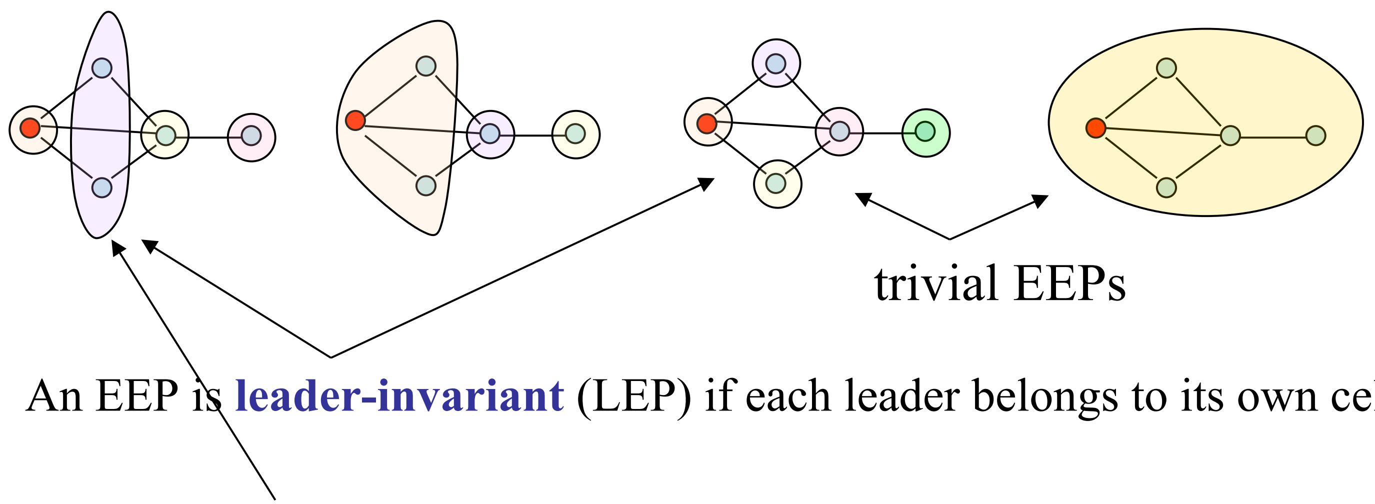


- Given a graph
$$G = (V, E)$$
- Define a **partition** of the node set into cells
$$\pi : V \to \{1, \ldots, K\}, \quad (\text{"}v \in_\pi C_j \Leftrightarrow \pi(v) = j\text{"})$$
- Let the **node-to-cell degree** be given by
$$\deg_\pi(v, C) = card(\{v' \in_\pi C \mid (v, v') \in E\})$$



- The partition is an **equitable partition** if
$$\pi(v) = \pi(v') \;\Leftrightarrow\; \deg_\pi(v, C_j) = \deg_\pi(v', C_j),\ \forall C_j$$
- The partition is an **external equitable partition** if
$$\pi(v) = \pi(v') \;\Leftrightarrow\; \deg_\pi(v, C_j) = \deg_\pi(v', C_j),\ \forall C_j,\ j \neq \pi(v)$$
(it does not matter what edges are inside a cell)

External Equitable Partitions



- An EEP is **leader-invariant** (LEP) if each leader belongs to its own cell
- A LEP is **maximal** if no other LEP with fewer cells exists

Controllability?

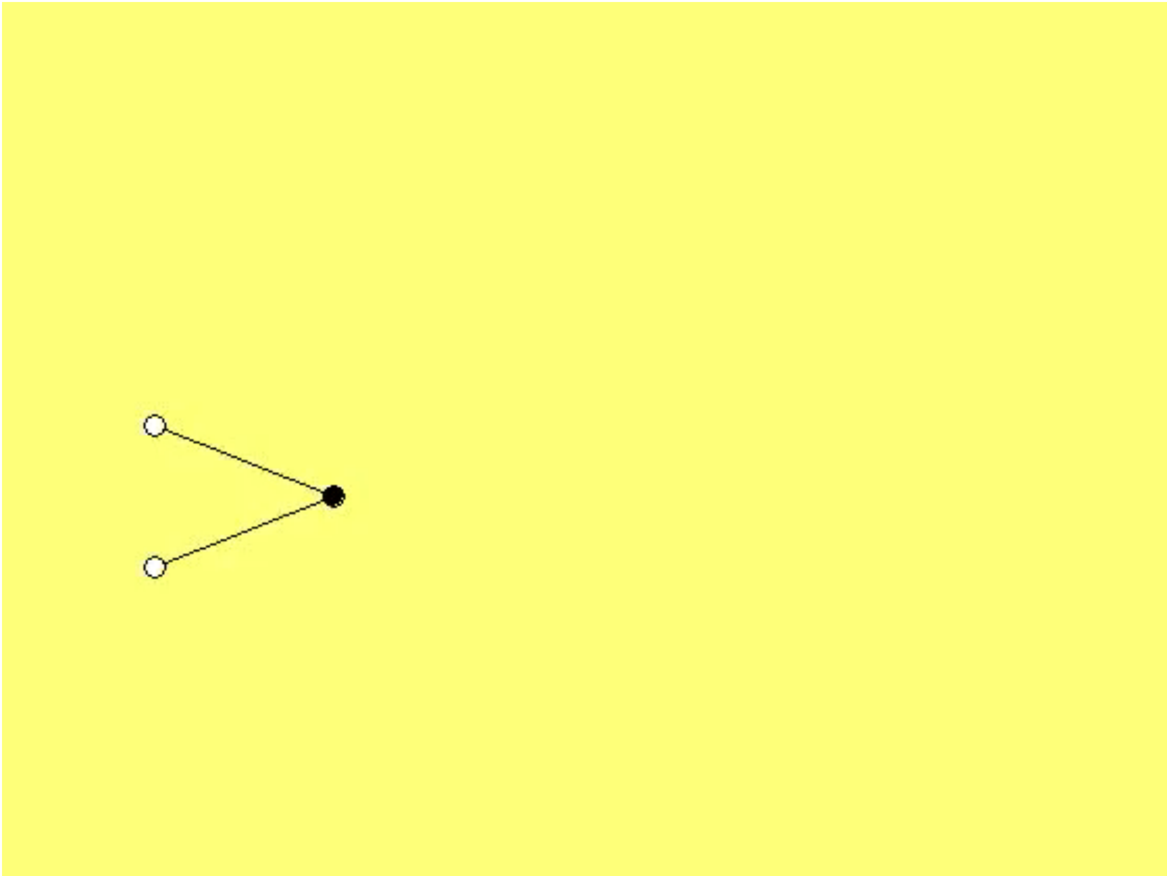
- From the leaders' vantage-point, nodes in the same cell "look" the same
- Let

$$\dot{x}_i = -\sum_{j \in N_i} (x_i - x_j), \;\; v_i \in V_F$$
$$\dot{x}_i = u_i, \;\; v_i \in V_L$$

- **Theorem** [7,8]: The uncontrollable part is asymptotically stable (if the graph is connected). It is moreover given (in part) by the difference between agents inside the same cell in the maximal LEP.

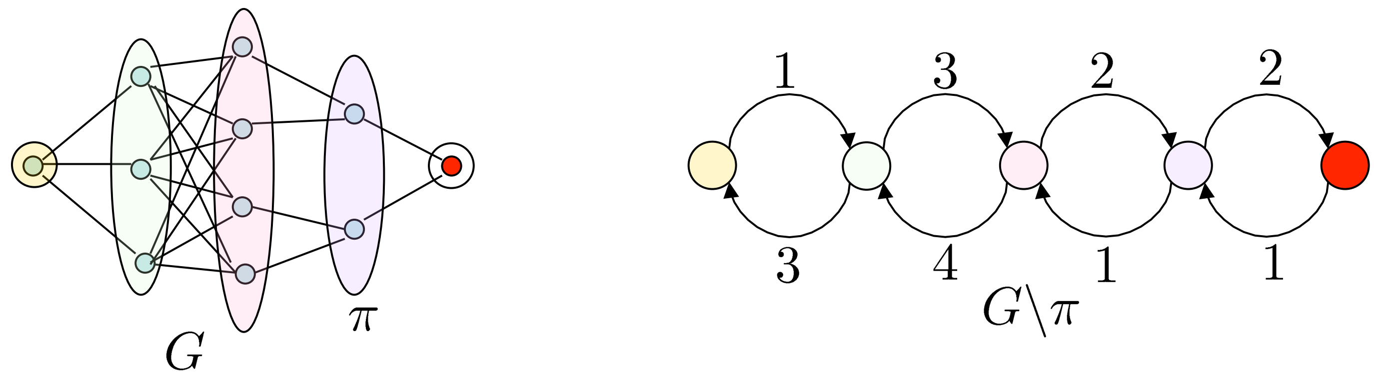
- **Corollary:** The system is completely controllable only if the only LEP is the trivial EEP

Uncontrollable Part



Quotient Graphs

- To understand the controllable subspace, we need the notion of a **quotient graph:**
  - Identify the vertices with the cells in the partition (maximal LEP)
  - Let the edges be weighted and directed in-between cells



- What is the dynamics over the quotient graph?

Quotient Graphs = Controllable Subspace

- Original system:

$$\Sigma_1 : \begin{cases} \dot{x}_i = -\sum_{j \in N_i} (x_i - x_j), \;\; v_i \in V_F \\ \dot{x}_i = u_i, \;\; v_i \in V_L \end{cases}$$

- Quotient graph dynamics:

$$\Sigma_2 : \begin{cases} \dot{\xi}_i = -\sum_{C_j \in N_{i,\pi}} \deg_\pi(C_j, C_i)(\xi_i - \xi_j), \;\; \pi(v) = i, \;\; v \in V_F \\ \dot{\xi}_i = u_i, \;\; \pi(v) = i, \;\; v \in V_L \end{cases}$$

- **Theorem** [8]**:**

$$\xi_i(0) = \frac{1}{|C_i|} \sum_{j \;|\; \pi(v_j)=i} x_j(0) \;\; \Rightarrow \;\; \xi_i(t) = \frac{1}{|C_i|} \sum_{j \;|\; \pi(v_j)=i} x_j(t)$$

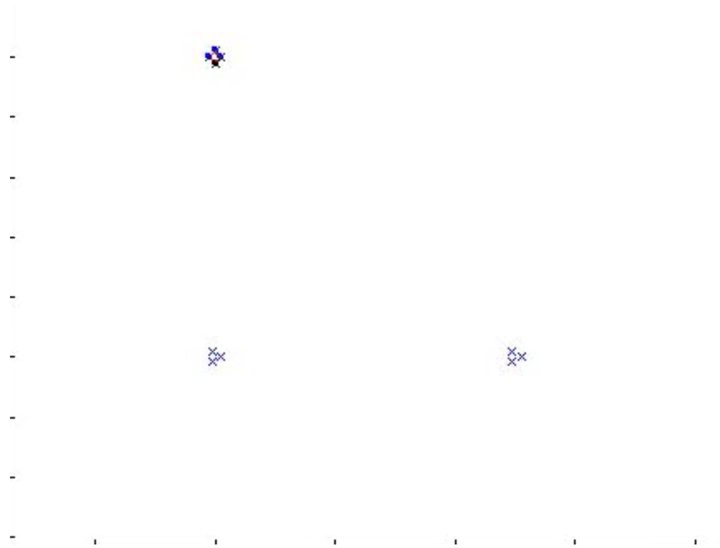
Graph-Based Controllability

- So what have we found?
  1. The system is completely controllable only if the only LEP is the trivial LEP
  2. The controllable subspace has a graph-theoretic interpretation in terms of the quotient graph of the maximal LEP
  3. The uncontrollable part decays asymptotically (all states become the same inside cells)
  4. Why bother with the full graph when all we have control over is the quotient graph? (= smaller system!)

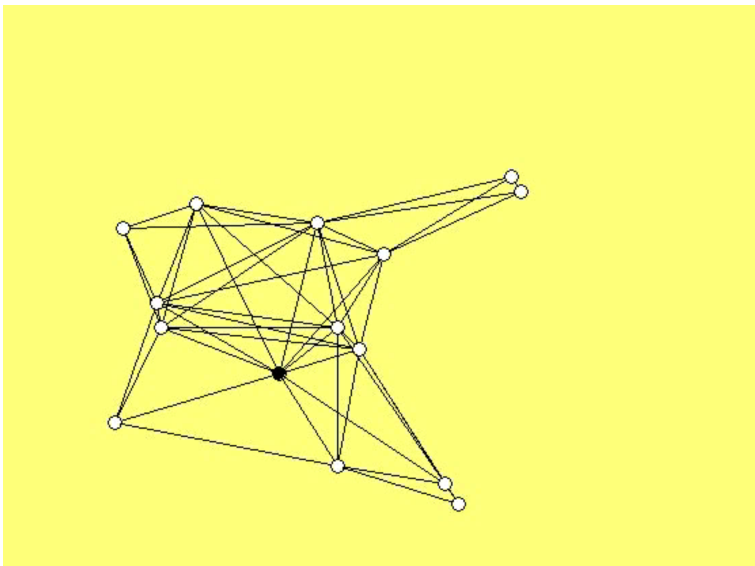
- **Now, let's put it to use!**

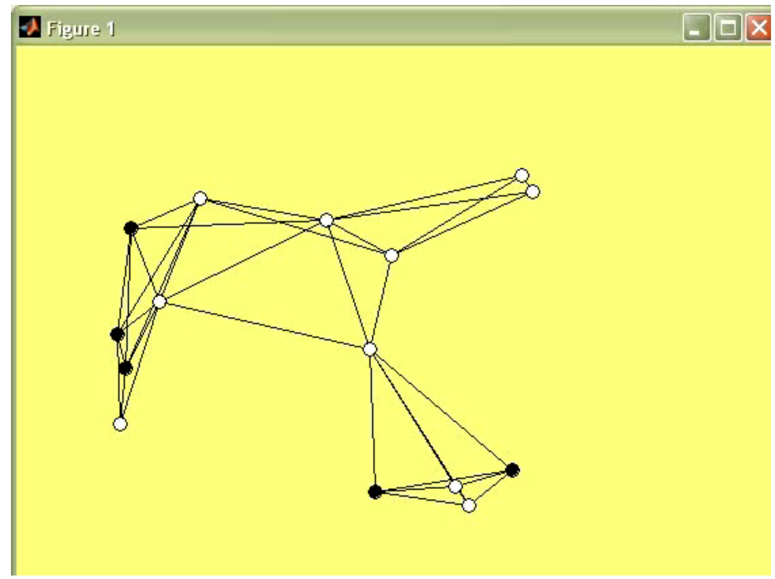
General Control Problems

- Controllability = We can solve general control problems for leader-based robot networks

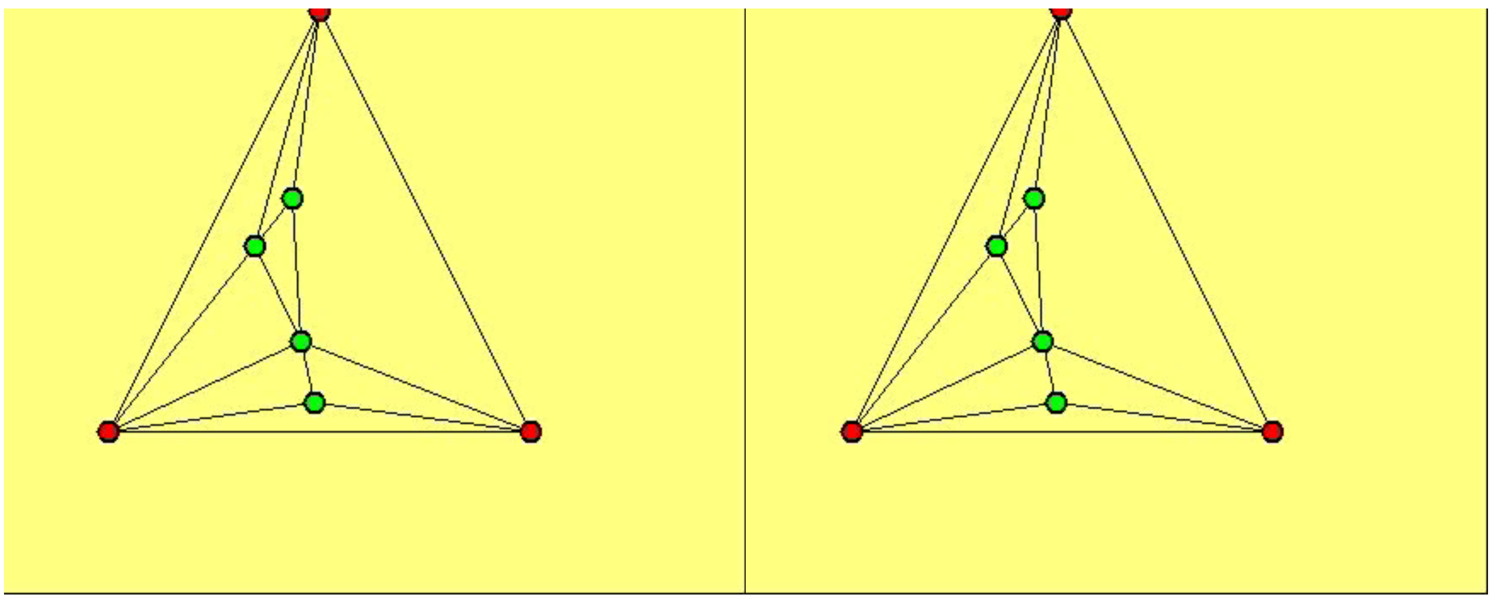


Stationary Leaders as Anchors



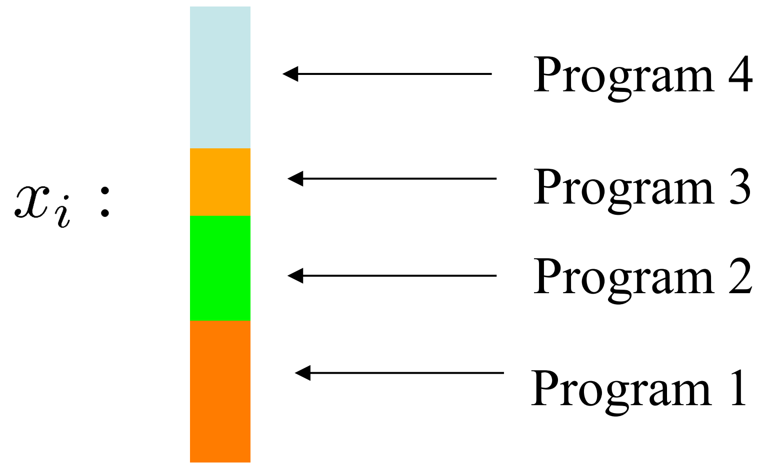


Containment Control



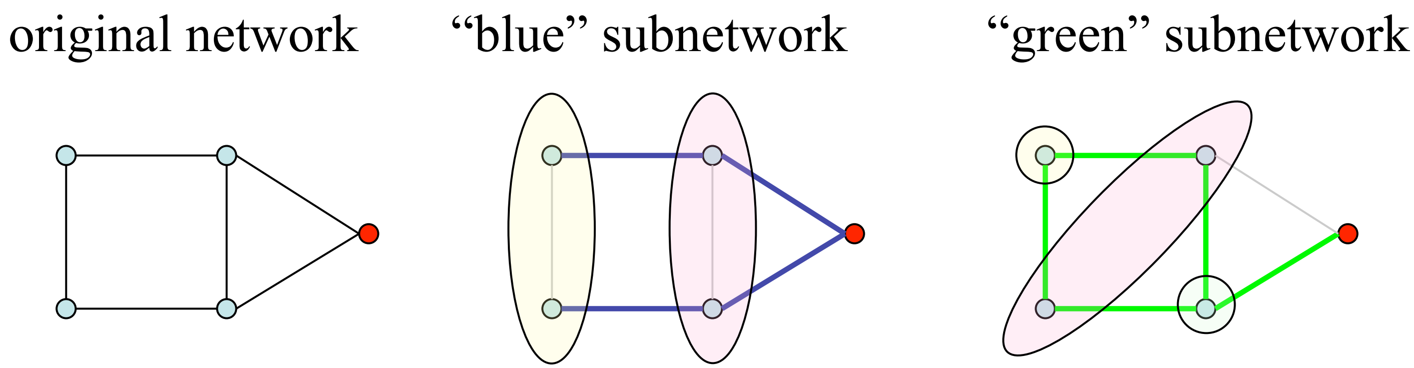
Epidemic Programming

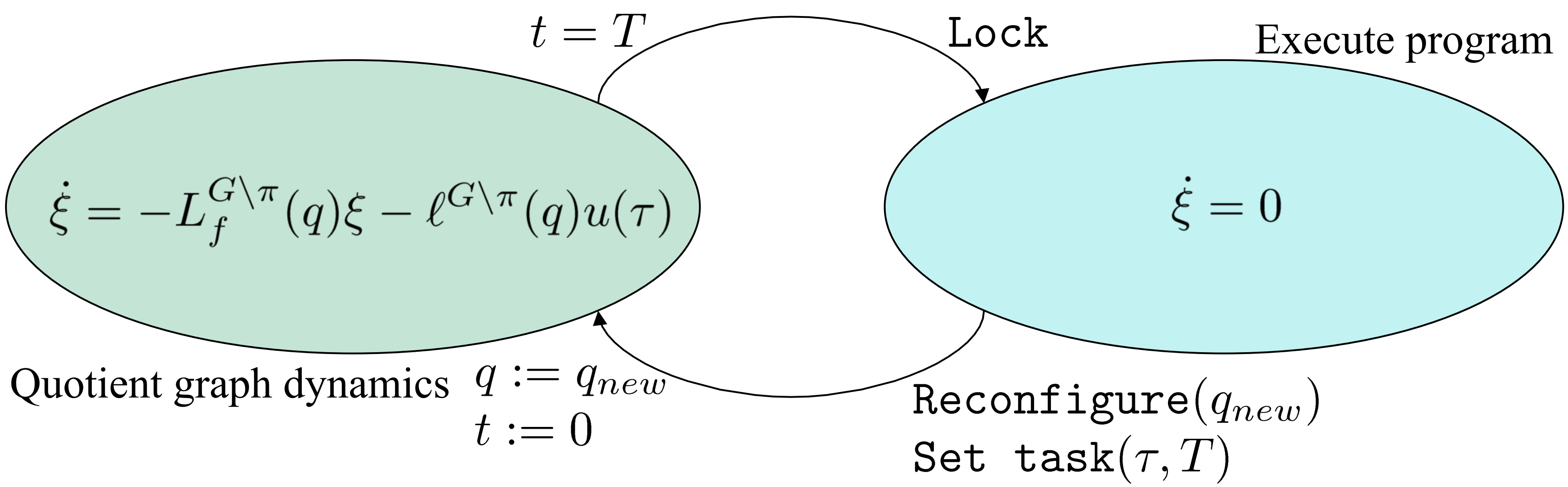
- Given a scalar state of each agent whose value determines what "program" the node should be running



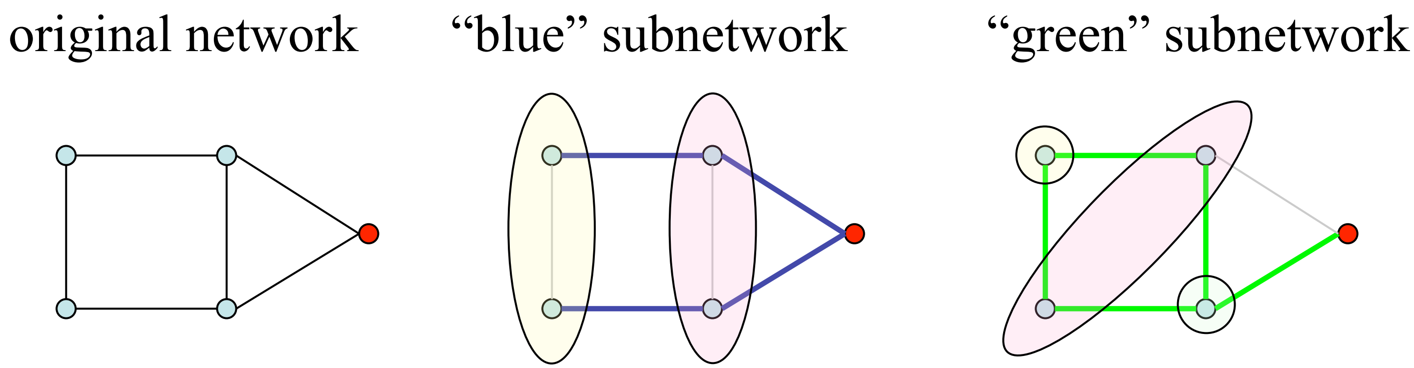
- By controlling this state, new tasks can be spread through the network
- But, we do not want to control individual nodes – rather we want to specify what each node "type" should be doing
- Idea: Produce sub-networks that give the desired LEPs and then control the system that way

Epidemic Programming



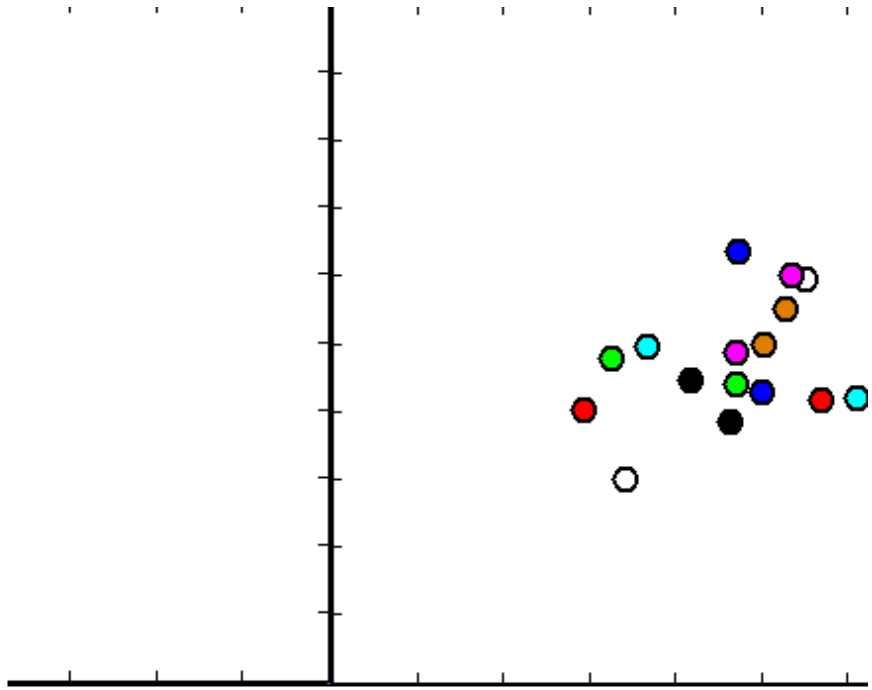


Epidemic Programming

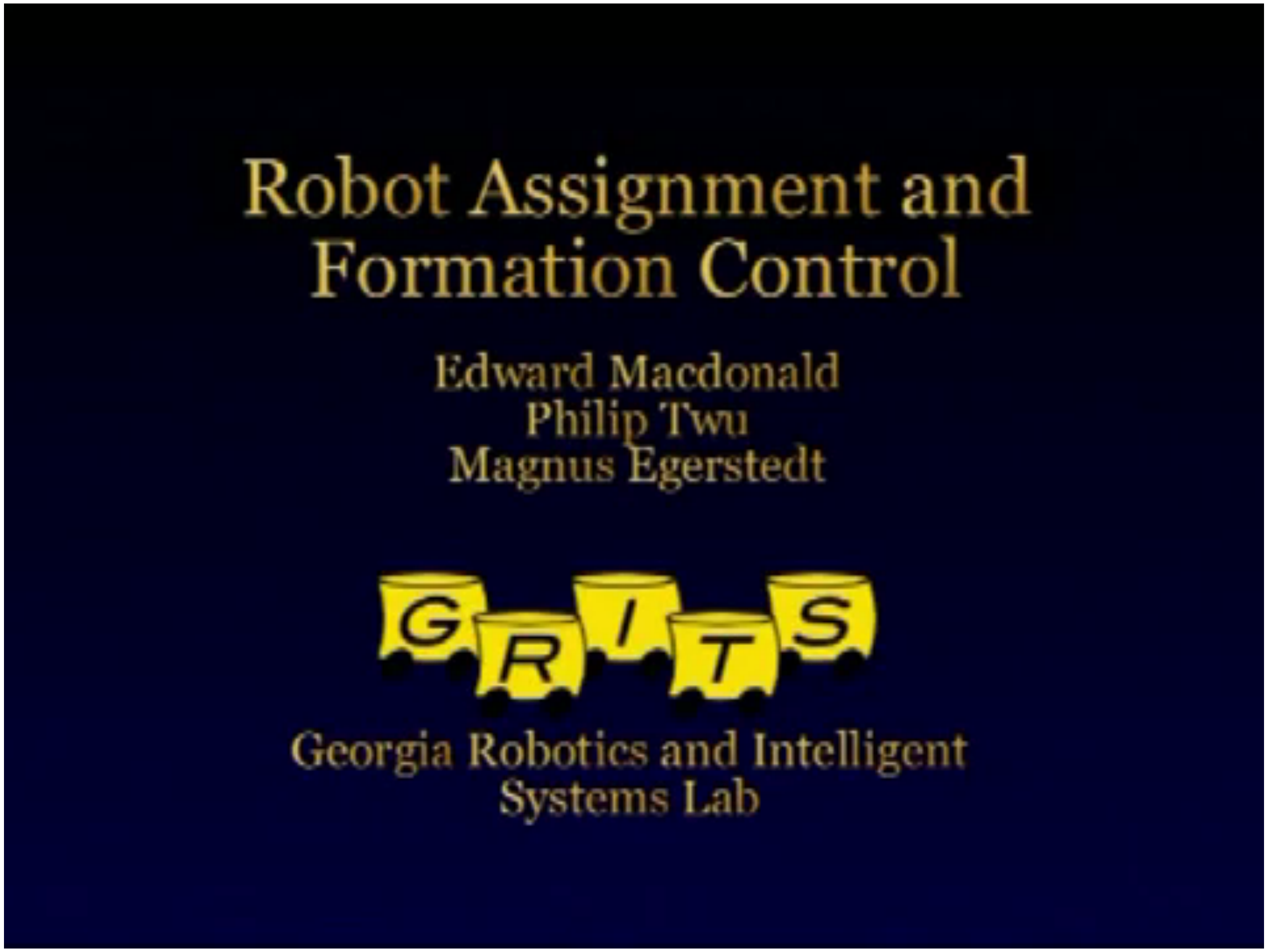


- Given a complete graph and a desired grouping of nodes into cells, produce a maximal LEP for exactly those cells using the fewest possible edges. (Answer is surprisingly enough not a combinatorial explosion…)

Epidemic Programming



Heterogeneous Networks



Summary III

- By introducing leader-nodes, the network can be “reprogrammed” to perform multiple tasks such as move between different spatial domains
- Controllability based on graph-theoretic properties was introduced through external equitable partitions

SESSION 4
SENSOR NETWORKS

Introduction

- Sensor networks are becoming an important component in cyber-physical systems:
  - smart buildings
  - unmanned reconnaissance




- Limited power capacity requires algorithms that can maintain area coverage and limit power consumption.

Node Models

- Consider a network of N sensors, with the following characteristics:

$p_i \in \Re^2$ ← position

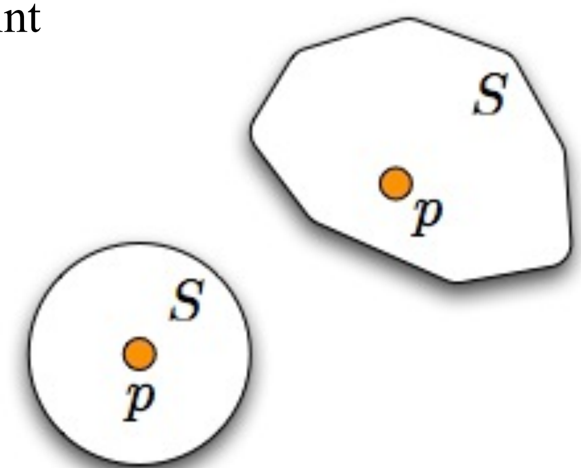
$\eta_i \in \Re_+$ ← power level

$S_i \subset \Re^2$ ← sensor footprint

- For example – standard disk model

$$S_i = \{x \in \Re^2 \mid \|x - p_i\| \leq \Delta\}$$

- Question: What is the connection between power level and performance?



Node Models

- A sensor can either be awake or asleep

$$\sigma = \begin{cases} 1 & \leftarrow \text{sensor on} \\ 0 & \leftarrow \text{sensor off} \end{cases}$$

- Power usage

$$\dot{\eta} = f_{pow}(\eta, \sigma), \quad \sigma = 0 \;\Rightarrow\; \dot{\eta} = 0$$

- Sensor footprint

$$S = S(p, \eta, \sigma), \quad \sigma = 0 \;\Rightarrow\; S = 0$$

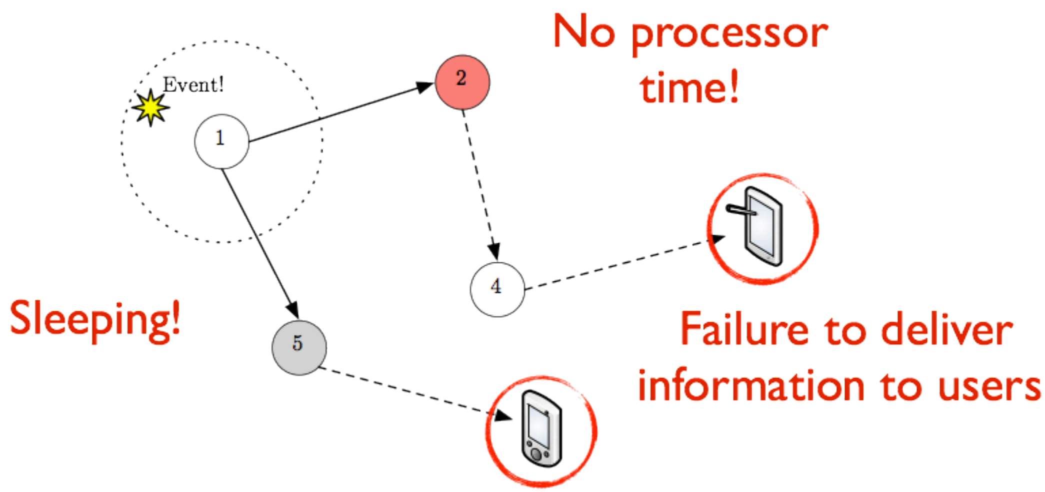
- Mobility

$$\dot{p} = f_{mob}(p, \eta, u)$$

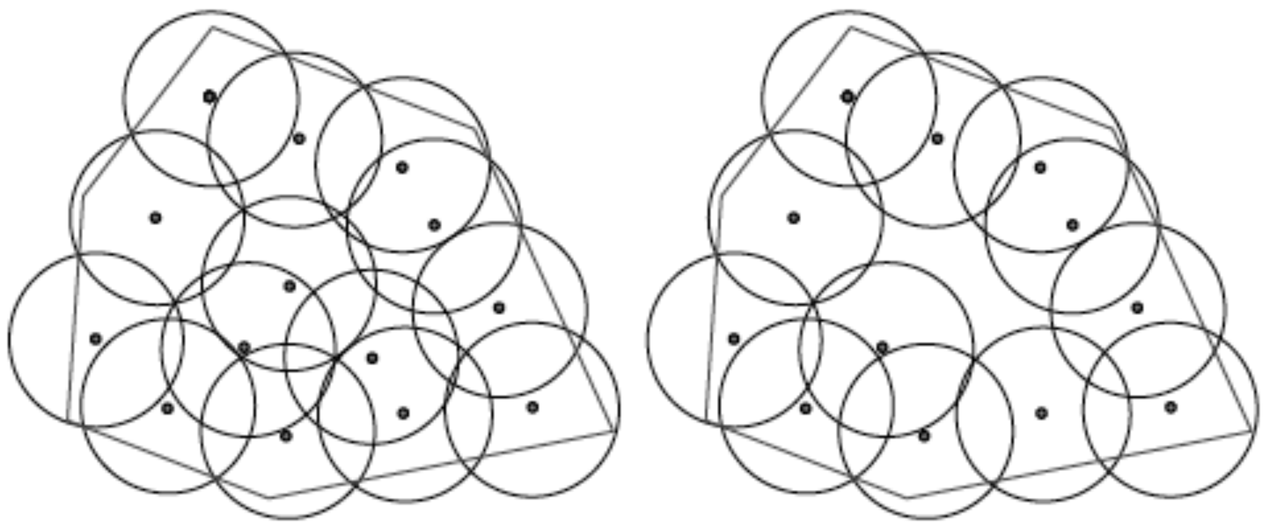
Node-level control variables (σ, u)

Node Models

- The available power levels affect the performance of the sensor nodes
- Sensor footprint – RF or radar-based sensors
  - Decreasing power levels leads to shrinking footprints
- Frame rates – vision based sensors
- Latency issues across the communications network



Coverage Problems



Coverage Problems

- Given a domain M. ***Complete coverage*** is achieved if

$$M \subseteq \bigcup_{i=1}^{N} S_i$$

- Areas are easier to manipulate than sets, and ***effective area coverage*** is achieved if

$$m \leq \left| \bigcup_{i=1}^{N} S_i \right|$$

$$G_{cov}(S) \geq 0$$

- Instead one can see whether or not events are detected with ***sufficient even detection probability***

$$\mu \leq \text{prob} \left(\text{event} \in \bigcup_{i=1}^{N} S_i \right)$$

Coverage/Life-Time Problems

- Now we can formulate the general life-time problem as

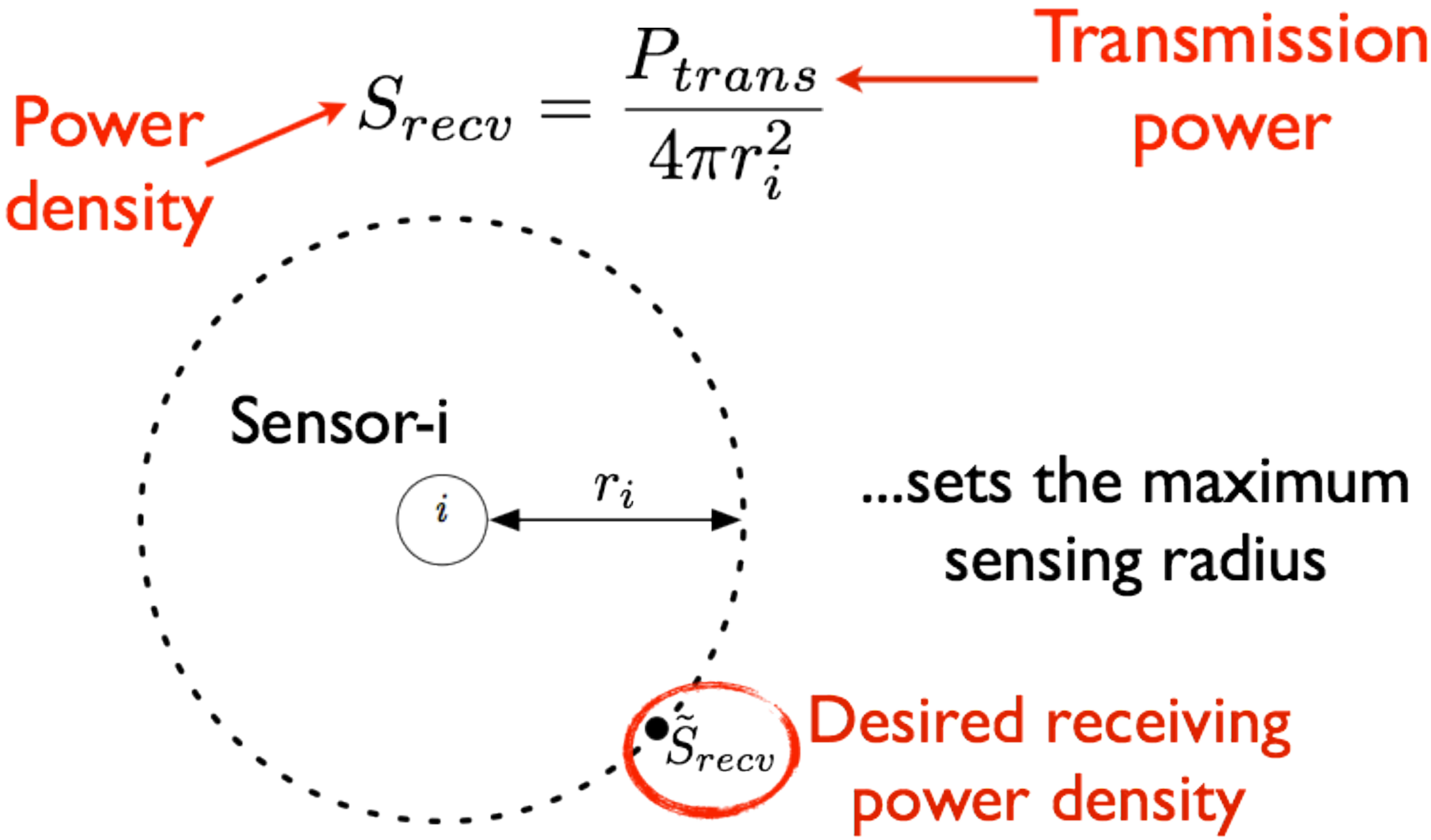
$$\max \; T \text{ such that } G_{cov}(S(T)) \geq 0, \; \forall t \leq T$$

- We will address this for some versions of the problem
  - Node-based, deterministic
  - Ensemble-based, stochastic

Radial Sensor Model

- Assume an isotropic RF transmission model for each sensor:

$$S_{recv} = \frac{P_{trans}}{4\pi r_i^2}$$



Radial Sensor Model

- Area covered by sensor is given by:

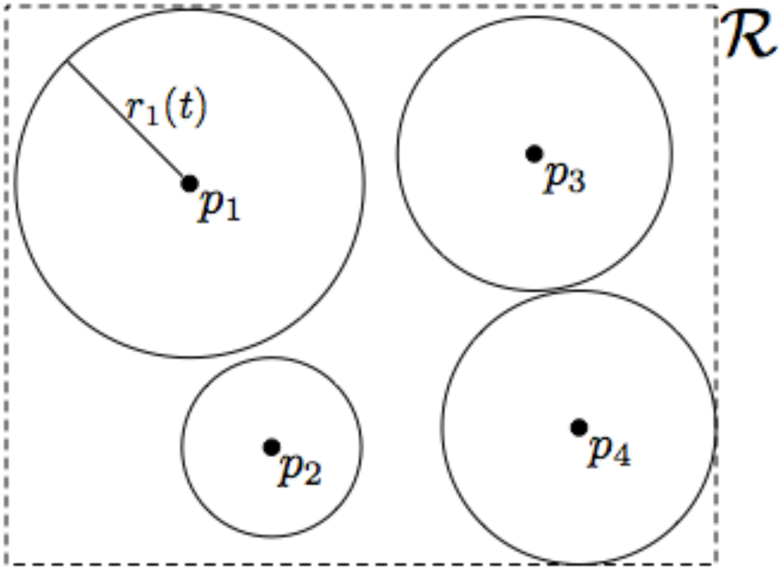
$$\pi r_i(t)^2 = \frac{P_{trans}}{4\tilde{S}_{recv}}$$

- But, sensor-i's transmitted power depends on its current power level:

$$P_{trans} = \sigma_i \eta_i$$

- Footprint:

$$|S(\eta_i, \sigma_i)| = \pi r_i^2(t) = \frac{\sigma_i \eta_i}{4\tilde{S}_{recv}}$$



Problem Formulation

- Our goal is effective area coverage, i.e.,

$$m \leq \left| \bigcup_{i=1}^{N} S_i \right|$$

- Assume sensor footprints do not intersect, then:

$$\left| \bigcup_{i=1}^{N} S_i \right| = \sum_{i=1}^{N} |S_i| \overset{\text{(almost)}}{=} \sum_{i=1}^{n} \sigma_i \eta_i$$

- Coverage constraint:

$$G_{cov}(S(t)) = \sum_{i=1}^{N} \sigma_i(t)\eta_i(t) - m \geq 0$$

Optimal Control

- Let

$$x = [\eta_1, \ldots, \eta_N]^T, \quad u = \text{diag}(\sigma_1, \ldots, \sigma_N)$$

- Aggregate dynamics

$$\dot{x}(t) = -\gamma u(t) x(t)$$

- Problem: Find gain signals that solve

$$\min_u J(u, x, t) = \int_{t_0}^{T} \frac{1}{2} \left(\left(u^T(t) x(t) - M \right)^2 + u^T(t) R u(t) \right) dt$$

Optimal Control

Hamiltonian:

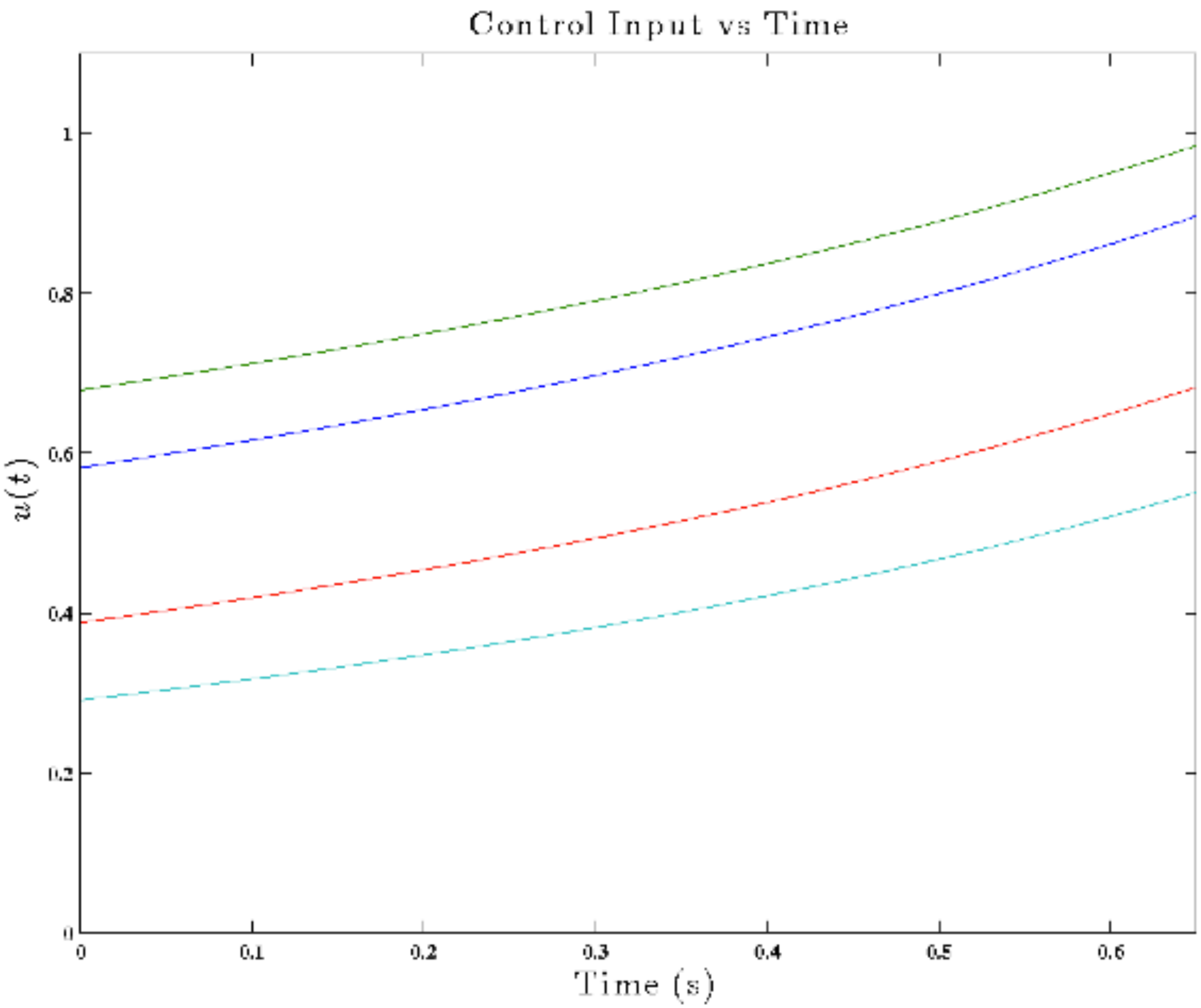
$$H(u,x,t) = -u^T(t)\Lambda(t)x(t) + \frac{1}{2}\left(u(t)^T x(t) - M\right)^2 + \frac{1}{2}u(t)^T R u(t)$$

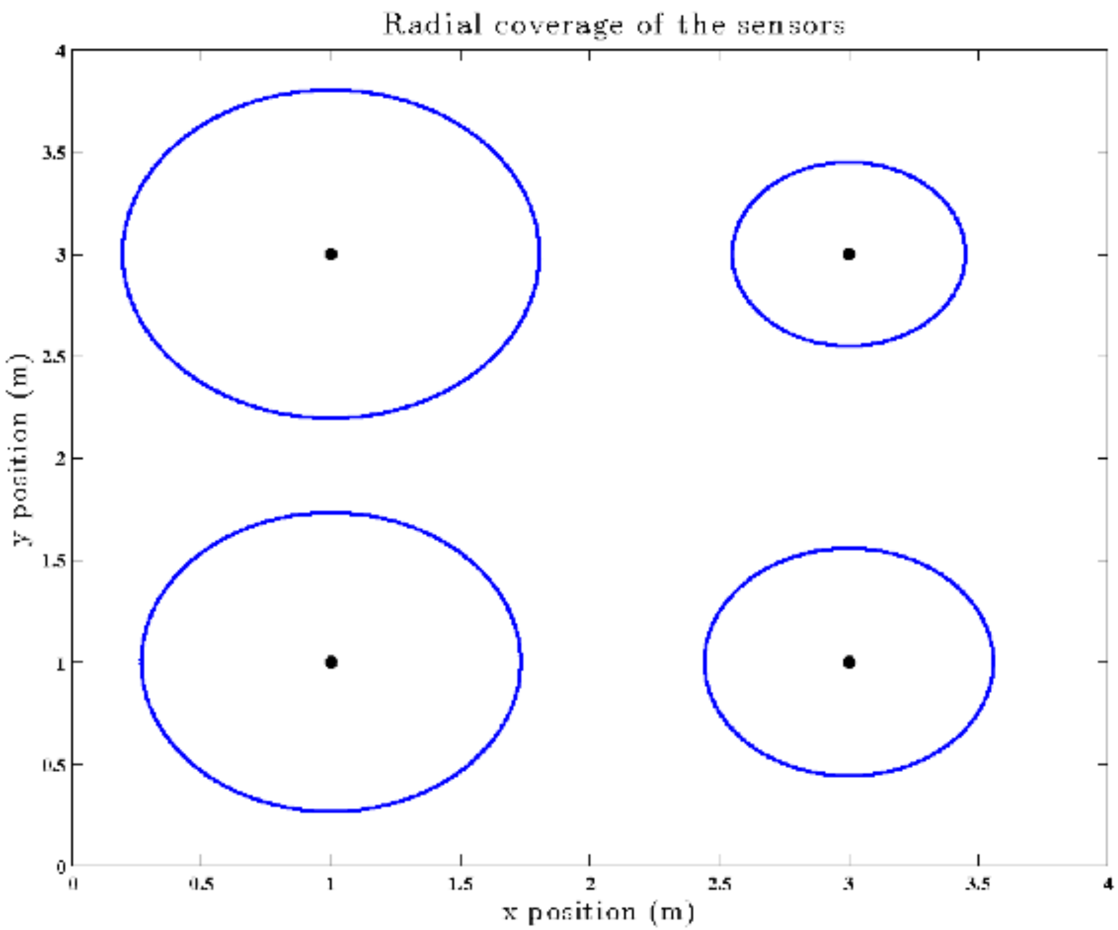
Where $\Lambda(t) = diag(\lambda_i(t))$ represents the co-states satisfying the backward differential equation:

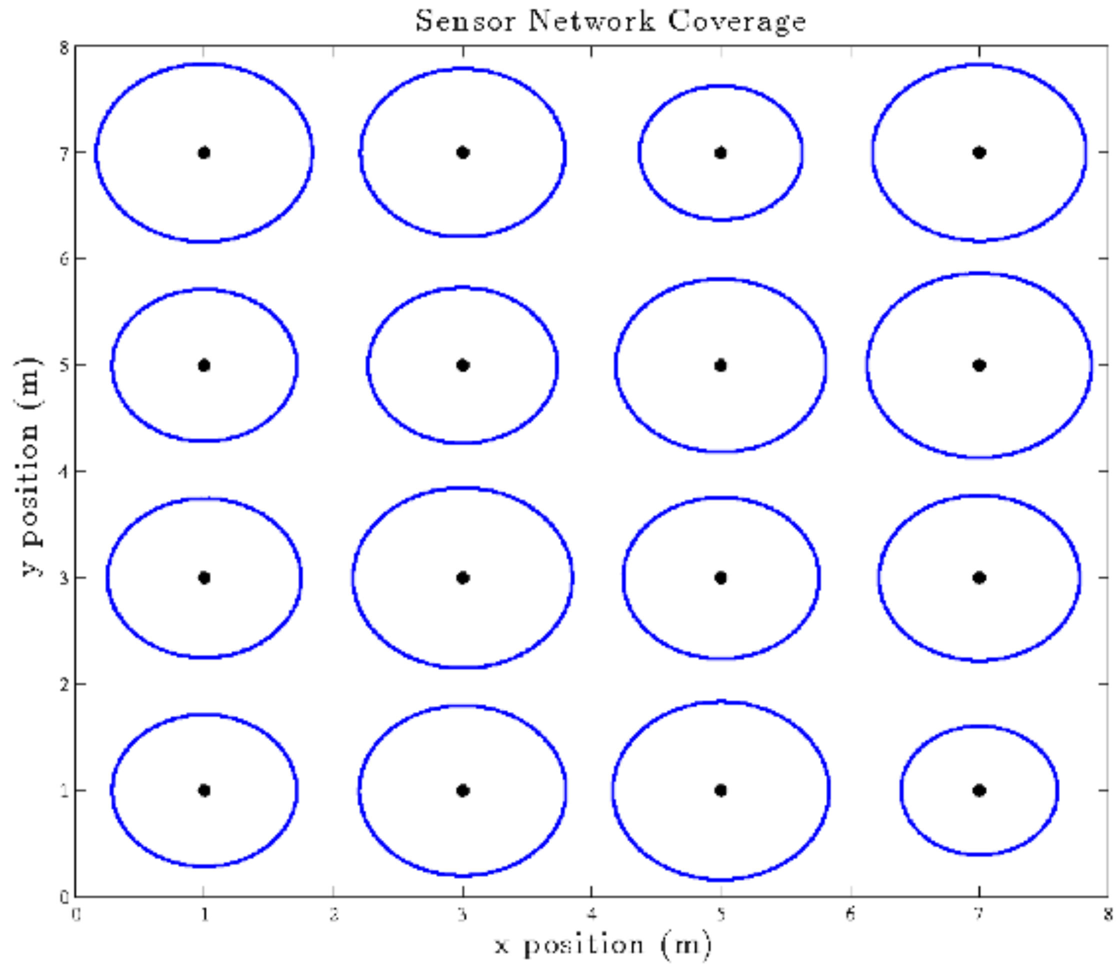
$$\dot{\lambda}(t) = \Lambda(t)u(t) - \left(u(t)^T x(t) - M\right)u(t), \lambda(T) = 0$$

Optimal gain signals:

$$u(t) = \left(x(t)x^T(t) + R\right)^{-1}(\Lambda(t) + MI)\,x(t)$$


Control Input vs Time
$u(t)$
Time (s)
0
0.2
0.4
0.6
0.8
1
0.1
0.3
0.5


Radial coverage of the sensors
y position (m)
x position (m)
0
0.5
1
1.5
2
2.5
3
3.5
4


Sensor Network Coverage
y position (m)
x position (m)
0
1
2
3
4
5
6
7
8

Issues

- Maybe not the right problem:
  - No on/off (relaxation)
  - No life-time maximization
- What we do know about the "right" problem
  - Only switch exactly when the minimum level is reached
  - Knapsack++
- Maybe we can do better if we allow for randomness in the model?

The Setup

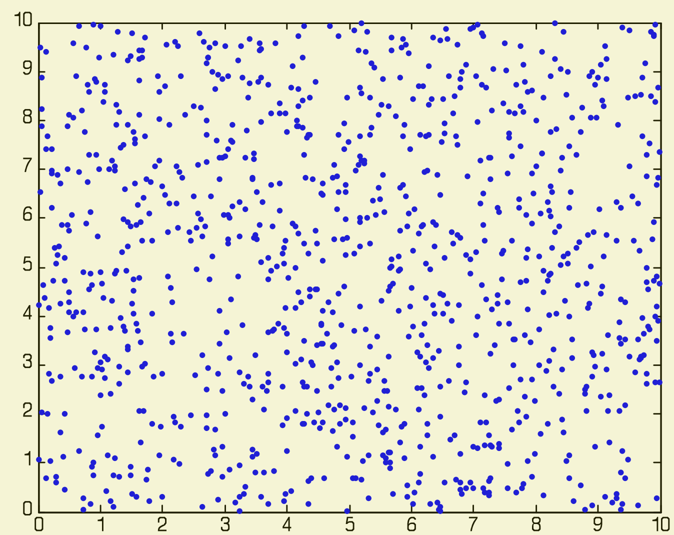
- Given a decaying sensor network we want to find a scheduling scheme that maintains a desired network performance throughout the lifetime of the network.
- The desired network performance is the minimum satisfactory probability of an event being detected.
- Lifetime of the sensor network is the maximal time beyond which the desired network performance cannot be achieved.
- We assume that the sensor nodes are “dropped” over an area.

Spatial Poisson Processes

- We assume that the sensor nodes are dropped according to a spatial Poisson point process:

i. The number of points in any subset X of *D*, n(X), are Poisson distributed with intensity $\lambda||X||$, where λ is the intensity per unit area.

ii. The number of points in any finite number of disjoint subsets of *D* are independent random variables.



$$P(n \text{ sensors in area } A) = \frac{(\lambda A)^n e^{-\lambda A}}{n!}$$

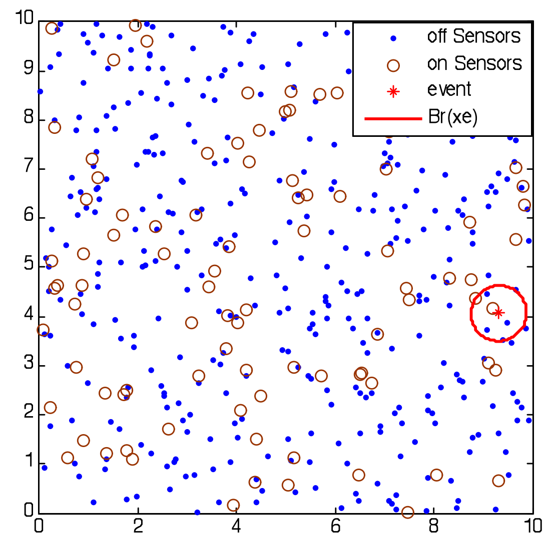
System Model

- All sensors are identical i.e., they have same
  - Initial power and power decay rate
  - Sensing capabilities
- All sensors have circular footprint

$$S_i = B_r(p_i)$$

  - An event at location x_e is detected if

$$x_e \in B_r(p_i)$$



- To conserve power, sensors are switched between on state and off state
  - Power is consumed only when a sensor is on: $\dot{\eta}_i = -\gamma q_i \eta_i$

Prob that sensor is on at time t

Event Detection Probability

- Consider a non-persistent event
  - An event is non-persistent if it does not leave a mark in the environment and can only be detected when it occurs.
- Theorems:
  - Probability of an event going undetected by a non-decaying sensor network is

$$P_u = e^{-\lambda \pi r^2 q}$$

  - Probability of an event going undetected by a decaying sensor network is

$$P_u = e^{-\lambda c e^{-\gamma \int_0^t q(s)ds} q(t)}$$

$$r^2(t) \propto \eta(t)$$

$$A(t) = \pi r(t)^2 = c e^{-\gamma \int_0^t q(s)ds}$$

Controlling Duty Cycles

- We need a controller of the form

$$\dot{q}(t) = u(t)$$

  to maintain a constant P_d (as long as possible)
- Controller:

$$q(0) = \frac{\ln\left(\frac{1}{1-P_d}\right)}{\lambda c}$$

$$u(t) = \gamma q(t)^2$$

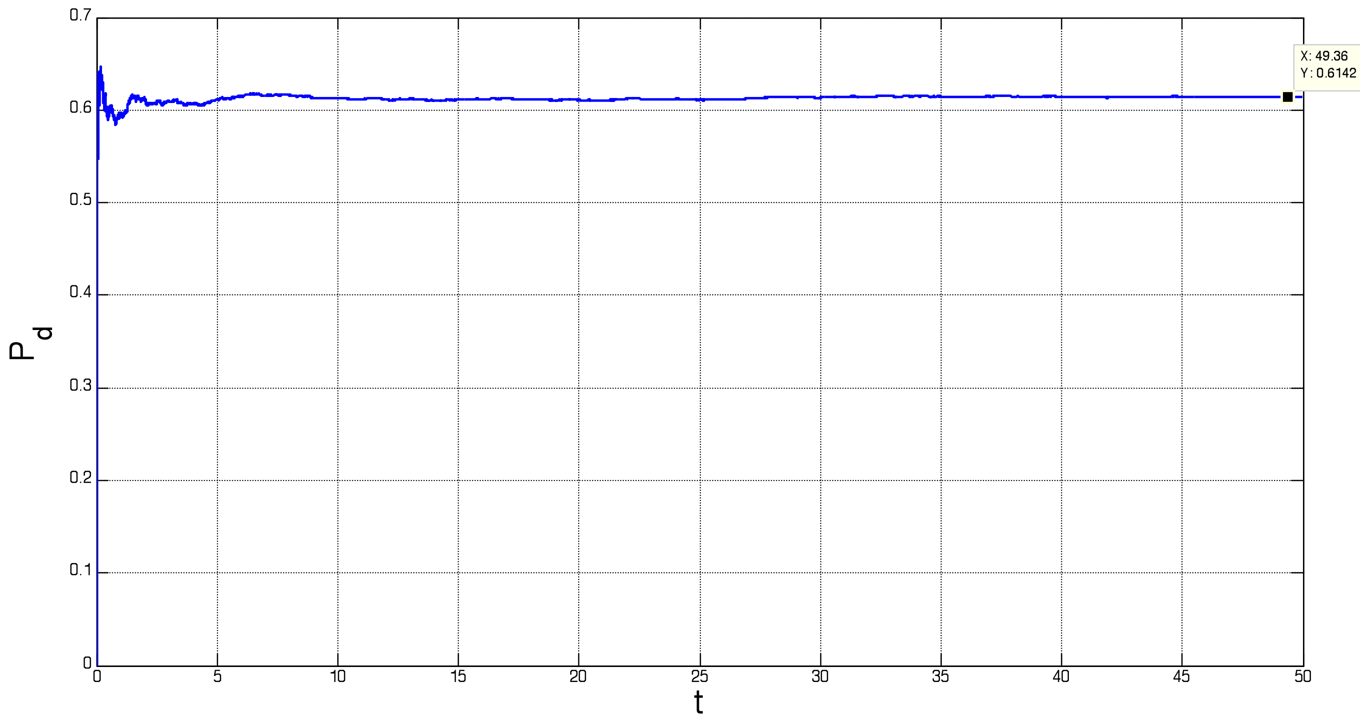
- Life time:

$$T = \frac{1}{\gamma}\left(\frac{\lambda c}{\ln\left(\frac{1}{1-P_d}\right)} - 1\right)$$

Simulation Results

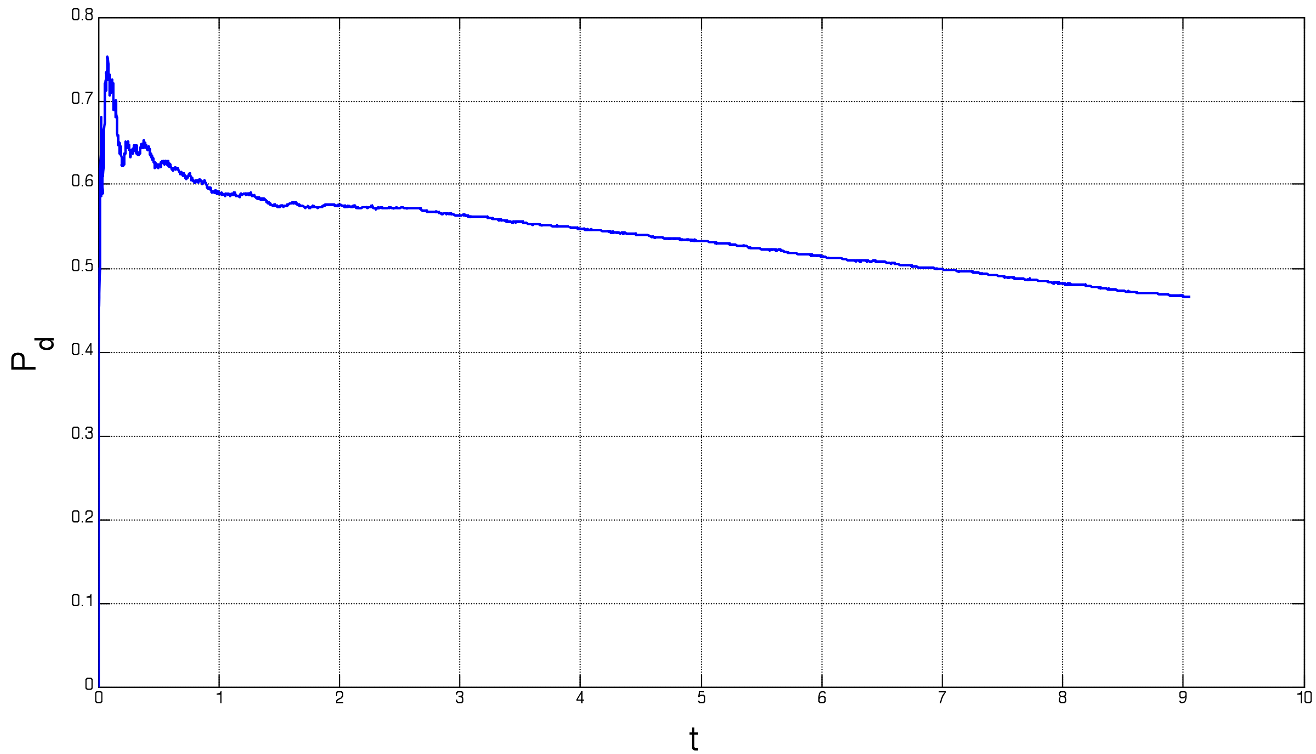
- A Monte – Carlo simulation of the network is performed
- In a (10 x 10) unit rectangular region sensors are deployed according to a spatial stationary Poisson point process with intensity $\lambda = 10$.
- Different scenarios (non – decaying network, decaying network, decaying network with scheduling scheme) are simulated with the following parameters
  - λ (intensity per unit area) = 10
  - γ (power decay rate) = 1
  - P_d (desired probability of event detection) = 0.63

Non-Decaying Footprints



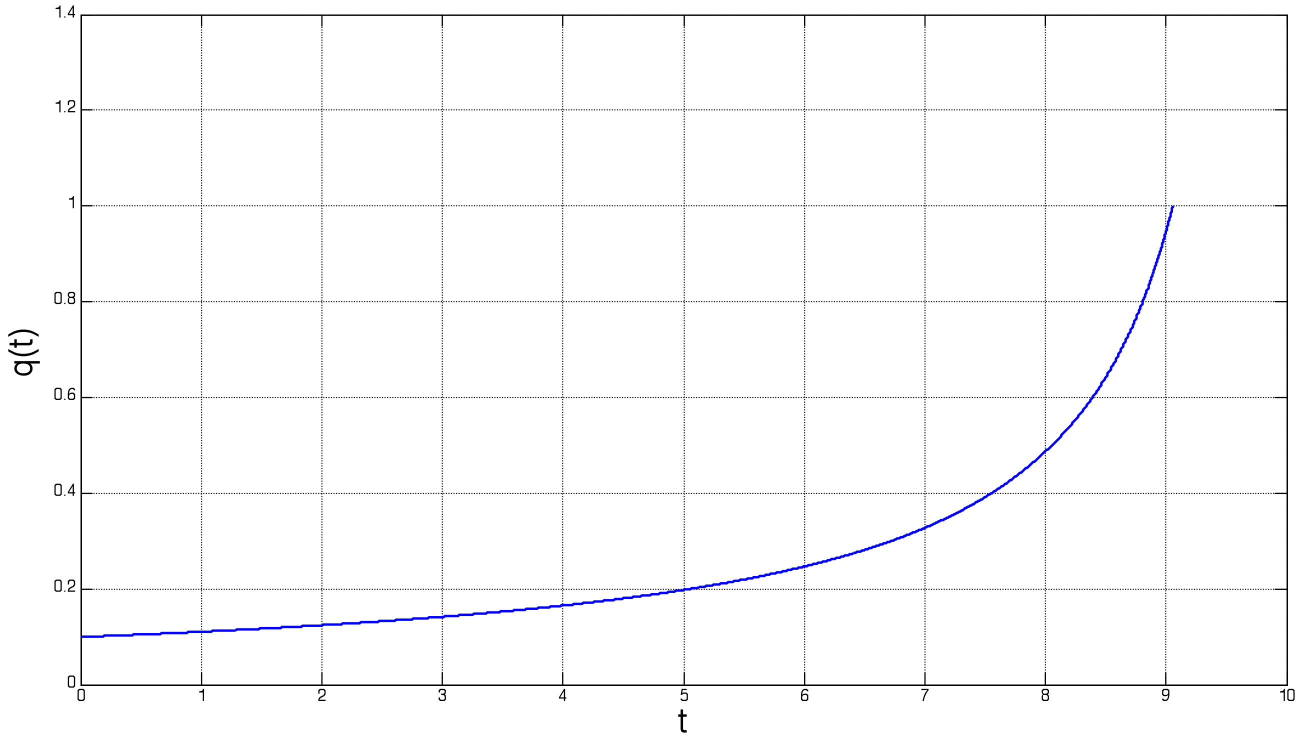
Event detection probability *Pd* vs time *t* for non-decaying networks with q = 0.1.

Decaying Footprints Without Feedback



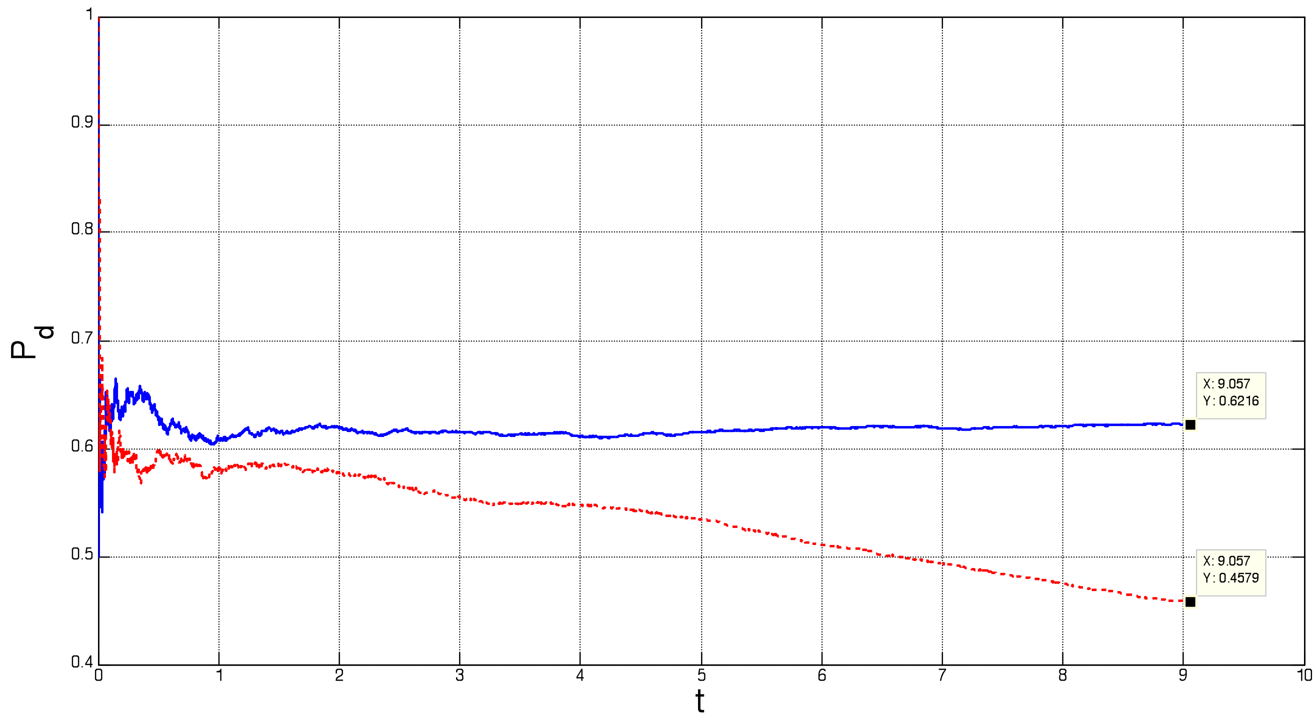
Event detection probability P_d vs time t for decaying networks with q = 0.1 and decay rate $\gamma = 1$

Decaying Footprints With Feedback

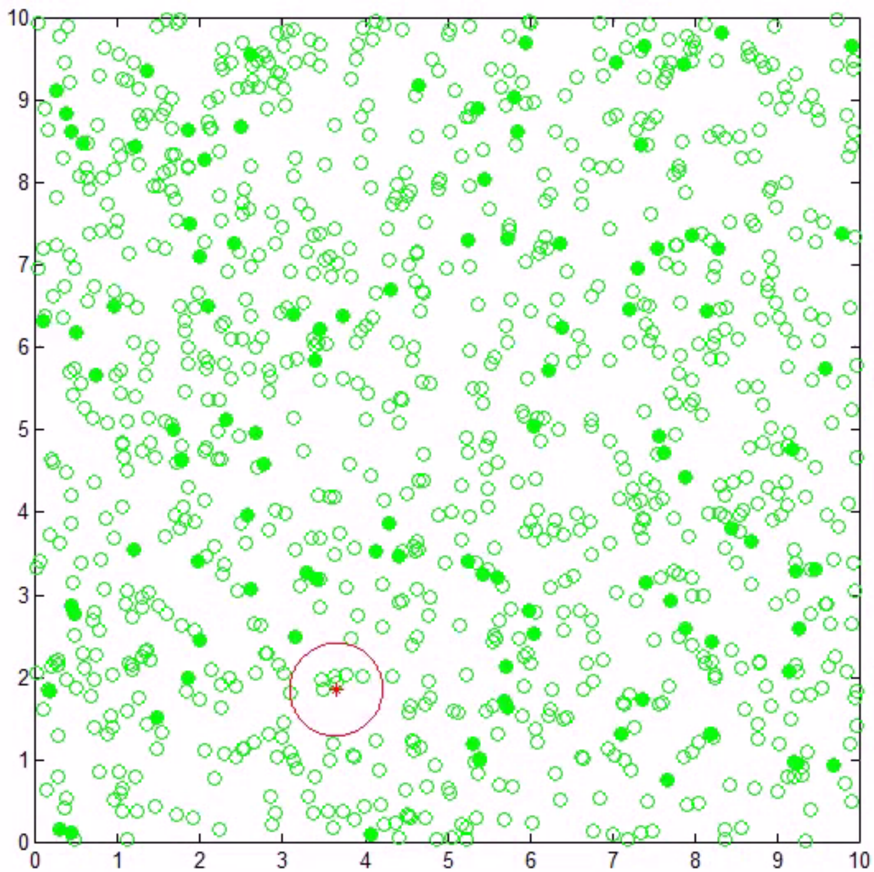


Probability of sensor being *on q vs time t*

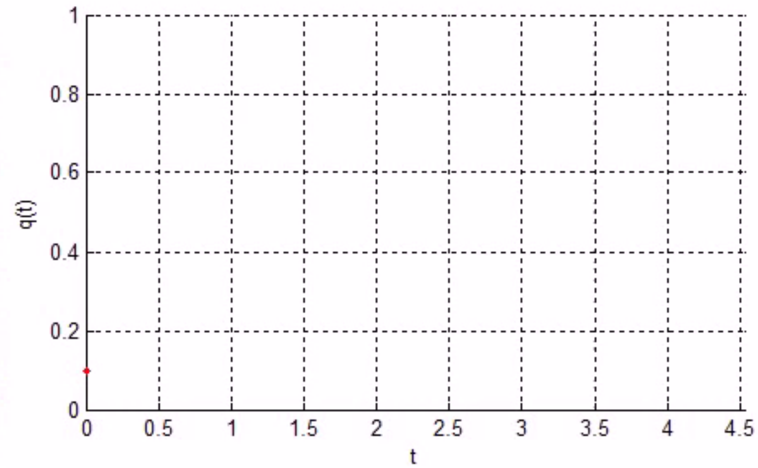
Decaying Footprints With Feedback



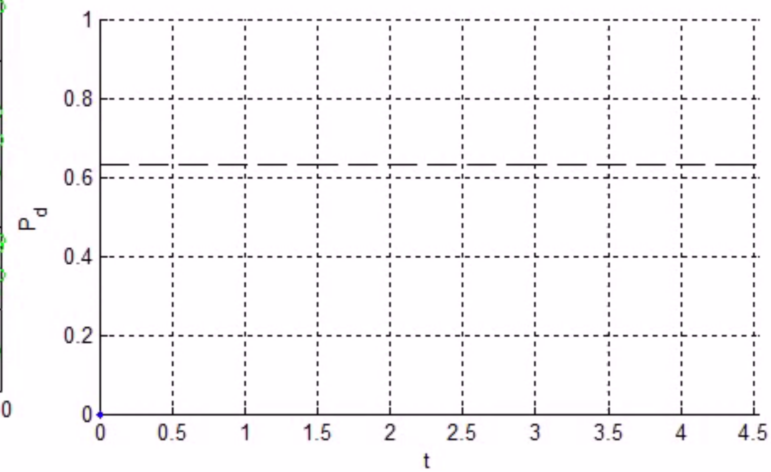
Event detection probability P_d vs time t for decaying networks with given $P_d = 0.63$; with scheduling scheme (solid line) and without scheduling scheme (dashed line)



10
9
8
7
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
8
9
10



q(t)
1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
t



P_d
1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
t

Issues

- We may still not have the right problem:
  - No on/off cost
  - No consideration of the decreasing communications capabilities
- What we do know about the hard problem
  - Rendezvous with shrinking footprints while maintaining connectivity?
- **Big question**: Mobility vs. Sensing vs. Communications vs. Computation???

Summary IV

- By introducing power considerations into the formulation of the coverage problem, a new set of issues arise
- Life-time problems
- Shrinking footprints
- Ensemble vs. node-level design
- **Big question**: Mobility vs. Sensing vs. Communications vs. Computation???

Conclusions

- The graph is a useful and natural abstraction of the interactions in networked control systems
- By introducing leader-nodes, the network can be "reprogrammed" to perform multiple tasks such as move between different spatial domains
- Controllability based on graph-theoretic properties was introduced through external equitable partitions
- Life-time problems in sensor networks

Acknowledgements

Joint work with

– Meng Ji, Simone Martini, Musad Haque, Abubakr Muhammad, Brian Smith, Mauro Franceschelli, Philip Twu, Peter Kingston, Waseem Abbas (current and previous students)

– **Giancarlo Ferrari-Trecate, Annalisa Buffa** (containment control)

– **J.M. McNew, Eric Klavins** (coverage control)

– **Mehran Mesbahi, Amirreza Rahmani** (controllability)

This work was sponsored by the **US National Science Foundation (NSF)** [ants], the **US Army Research Office** [formations], the **National Aeronautics and Space Administration (NASA)** [coverage], **Rockwell Collins** [HSI], and **Office for Naval Research** [heterogeneity]

References

[1] S. Martinez, J. Cortes, and F. Bullo. Motion coordination with distributed information. *IEEE Control Systems Magazine*, 27 (4): 75-88, 2007.

[2] M. Mesbahi and M. Egerstedt. *Graph Theoretic Methods for Multiagent Networks*, Princeton University Press, Princeton, NJ, Sept. 2010.

[3] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and Cooperation in Networked Multi-Agent Systems, *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215-233, Jan. 2007.

[4] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48 (6): 988–1001, 2003.

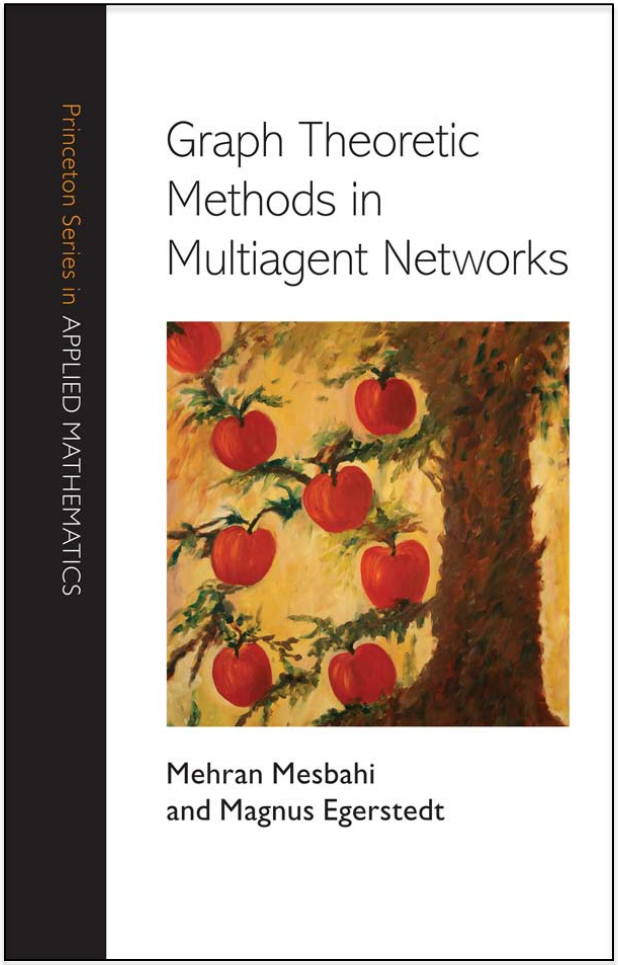
[5] M. Ji and M. Egerstedt. Distributed Coordination Control of Multi-Agent Systems While Preserving Connectedness. *IEEE Transactions on Robotics*, Vol. 23, No. 4, pp. 693-703, Aug. 2007.

[6] J.M. McNew, E. Klavins, and M. Egerstedt. Solving Coverage Problems with Embedded Graph Grammars. *Hybrid Systems: Computation and Control*, Springer-Verlag, pp. 413-427, Pisa, Italy April 2007.

[7] A. Rahmani, M. Ji, M. Mesbahi, and M. Egerstedt. Controllability of Multi-Agent Systems from a Graph-Theoretic Perspective. *SIAM Journal on Control and Optimization*, Vol. 48, No. 1, pp. 162-186, Feb. 2009.

[8] M. Egerstedt. Controllability of Networked Systems. *Mathematical Theory of Networks and Systems*, Budapest, Hungary, 2010.

[9] P. Dayawansa and C. F. Martin. A converse Lyapunov theorem for a class of dynamical systems which undergo switching, *IEEE Transactions on Automatic Control*, 44 (4): 751–760, 1999.


Princeton Series in APPLIED MATHEMATICS
Graph Theoretic
Methods in
Multiagent Networks
Mehran Mesbahi
and Magnus Egerstedt

THANK YOU!