

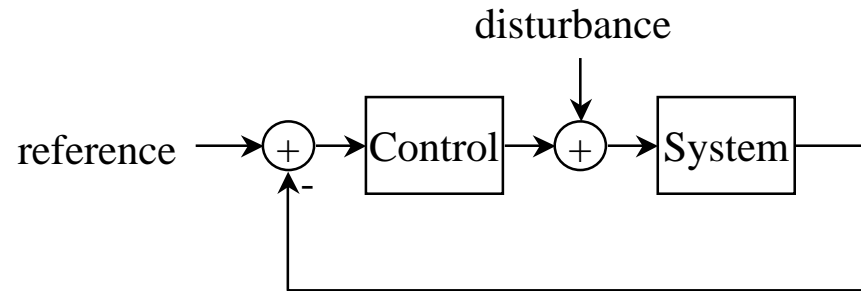
# Exploring Interfaces between CS and CDS

Mani Chandy  
Sayan Mitra  
Concetta Pilotto  
Jerome White

# Overview

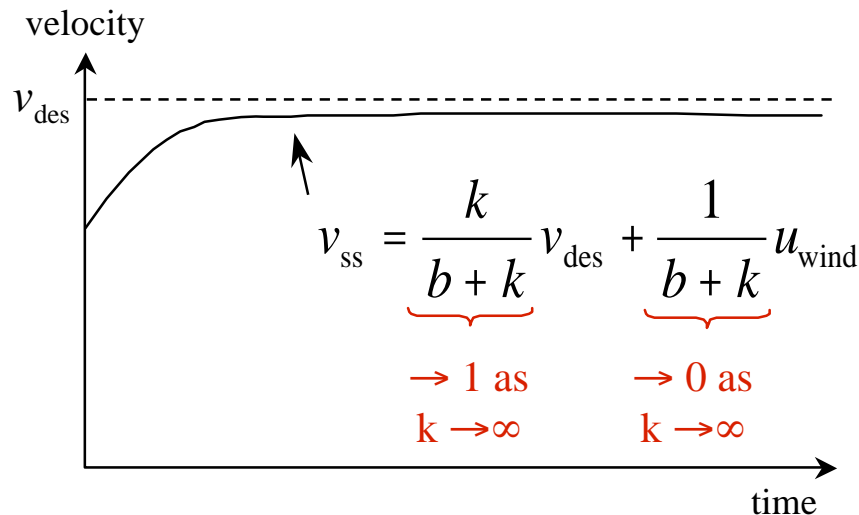
1. Control theory
2. Distributed computing theory
3. Differences
4. Graph dynamics: CDS & CS viewpoints.
5. Example: Mobile agents in formation from the CDS and CS viewpoints.
6. Caltech joint CDS/CS research: MURI

# Example #1: Cruise Control



$$m\dot{v} = -bv + u_{\text{engine}} + u_{\text{hill}}$$

$$u_{\text{engine}} = k(v_{\text{des}} - v)$$



## Stability/performance

- Steady state velocity approaches desired velocity as  $k \rightarrow \infty$
- Smooth response; no overshoot or oscillations

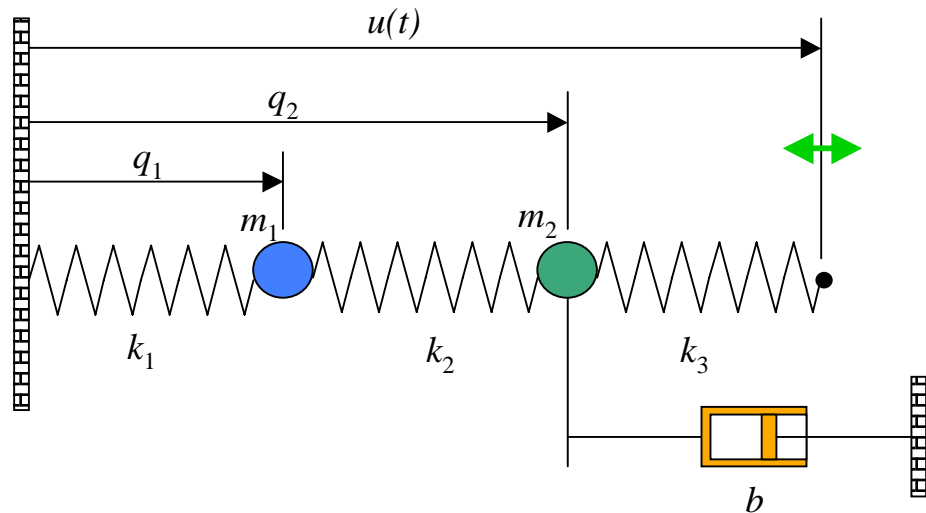
## Disturbance rejection

- Effect of disturbances (hills) approaches zero as  $k \rightarrow \infty$

## Robustness

- None of these results depend on the specific values of  $b$ ,  $m$ , or  $k$  for  $k$  sufficiently large

# Frequency Response for a Mass Spring System



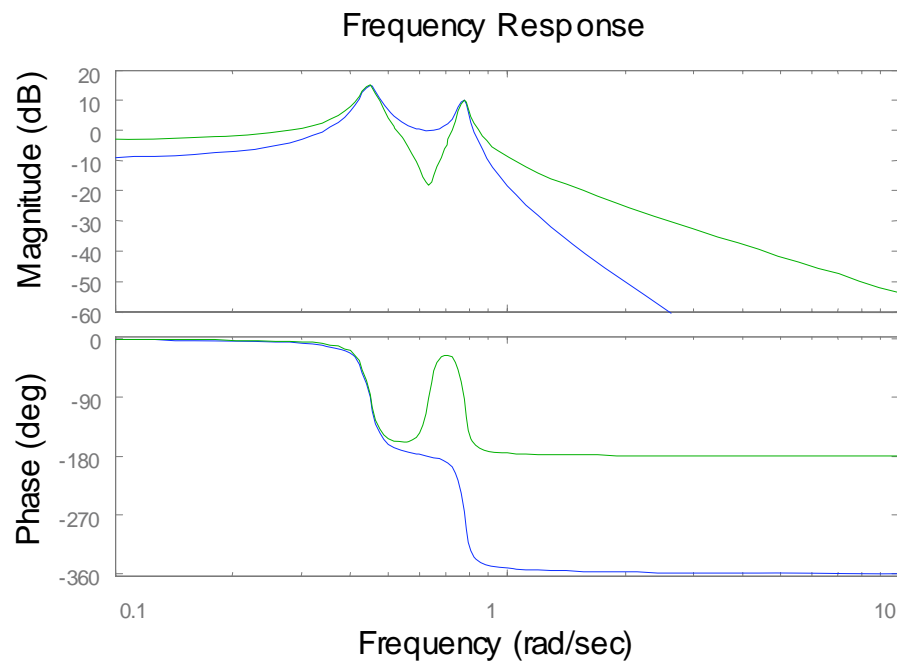
## Steady state frequency response

- Force the system with a sinusoid
- Plot the “steady state” response, after transients have died out
- Plot relative magnitude and phase of output versus input (more later)

## Matlab simulation

```
function dydt = f(t, y, ...)
u = 0.00315*cos(omega*t);
dydt = [
    y(3);
    y(4);
    -(k1+k2)/m1*y(1) + k2/m1*y(2);
    k2/m2*y(1) - (k2+k3)/m2*y(2)
    - b/m2*y(4) + k3/m2*u ];

t,y] = ode45(dydt,tspan,y0,[], k1, k2,
k3, m1, m2, b, omega);
```



# Difference Equations

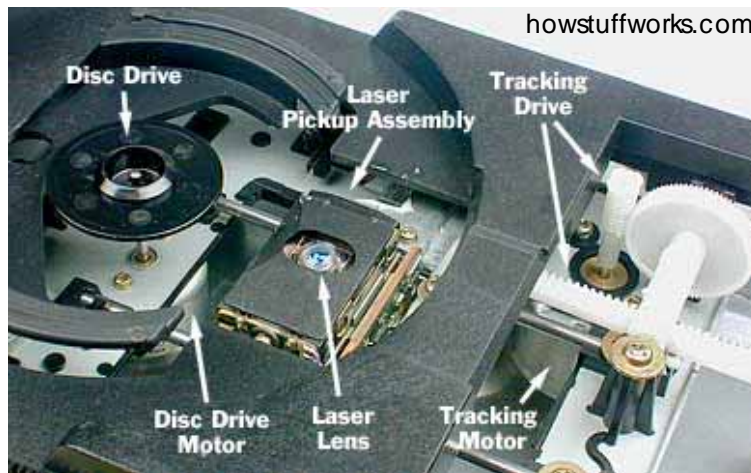
**Difference eqs model discrete transitions between continuous variables**

- “Discrete time” description (clocked transitions)
- New state is function of current state + inputs
- State is represented as a *continuous* variable

$$x_{k+1} = f(x_k, u_k)$$

$$y_{k+1} = h(x_{k+1})$$

**Example: CD read/write head *controller* (implemented on DSP)**



**Controller operation (every 1/44,100 sec)**

- Get analog signal from read head
- Determine the data (1/0) plus estimate the location of the track center
- Update estimate of “wobble”
- Compute where to position disk head for next read (limited by motor torque)

**Performance specification**

- Keep disk head on track center
- Reject disturbances due to disk shape, shaking and bumps, etc

**State:** estimated center, wobble

**Inputs:** read head signal

**Outputs:** commanded motion

# Distributed Computing Example

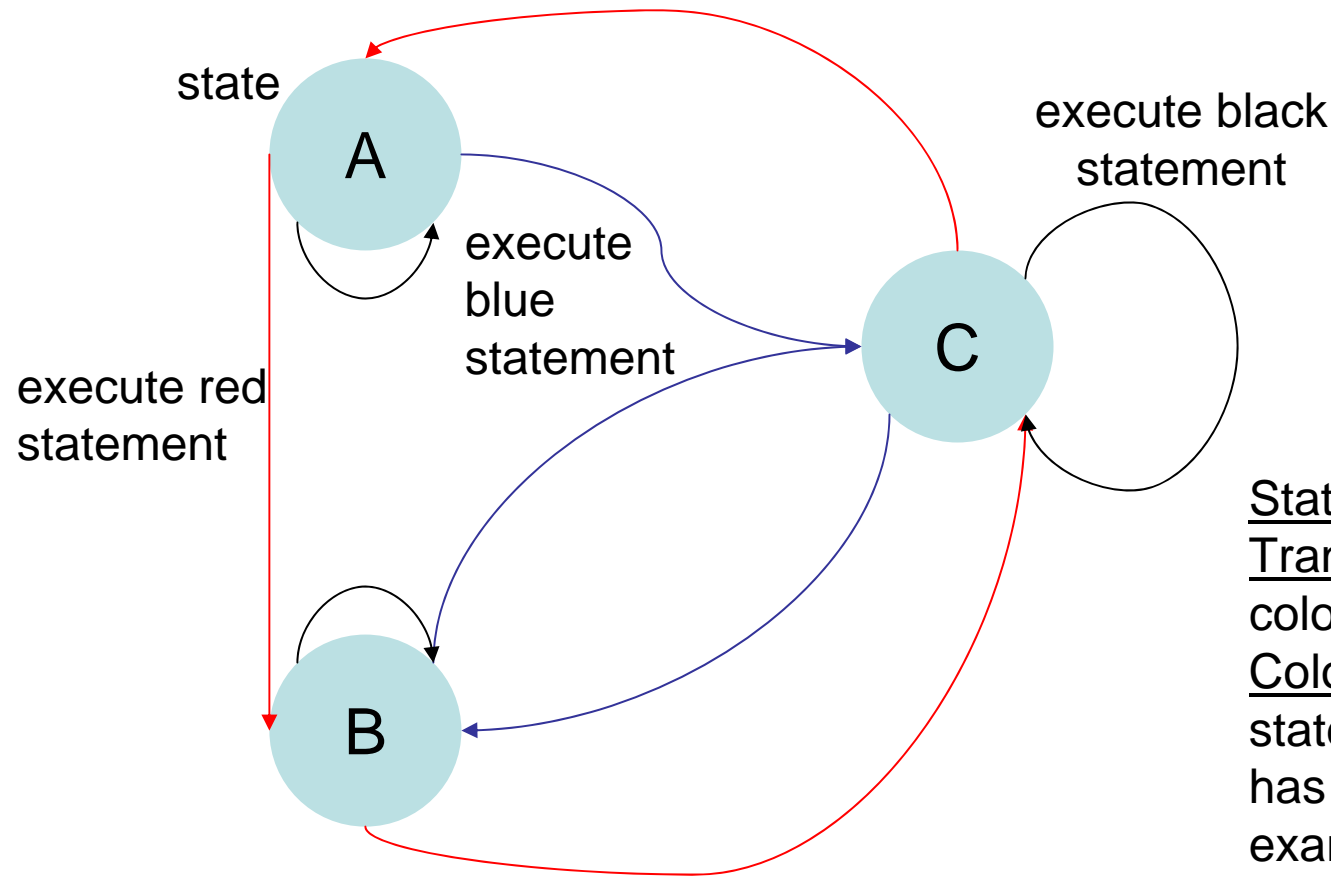
## Managing Shared Resources

### Process lifecycle

1. Computing
2. Waiting for access to shared files
3. Using shared files (for finite time)
  - Back to computing

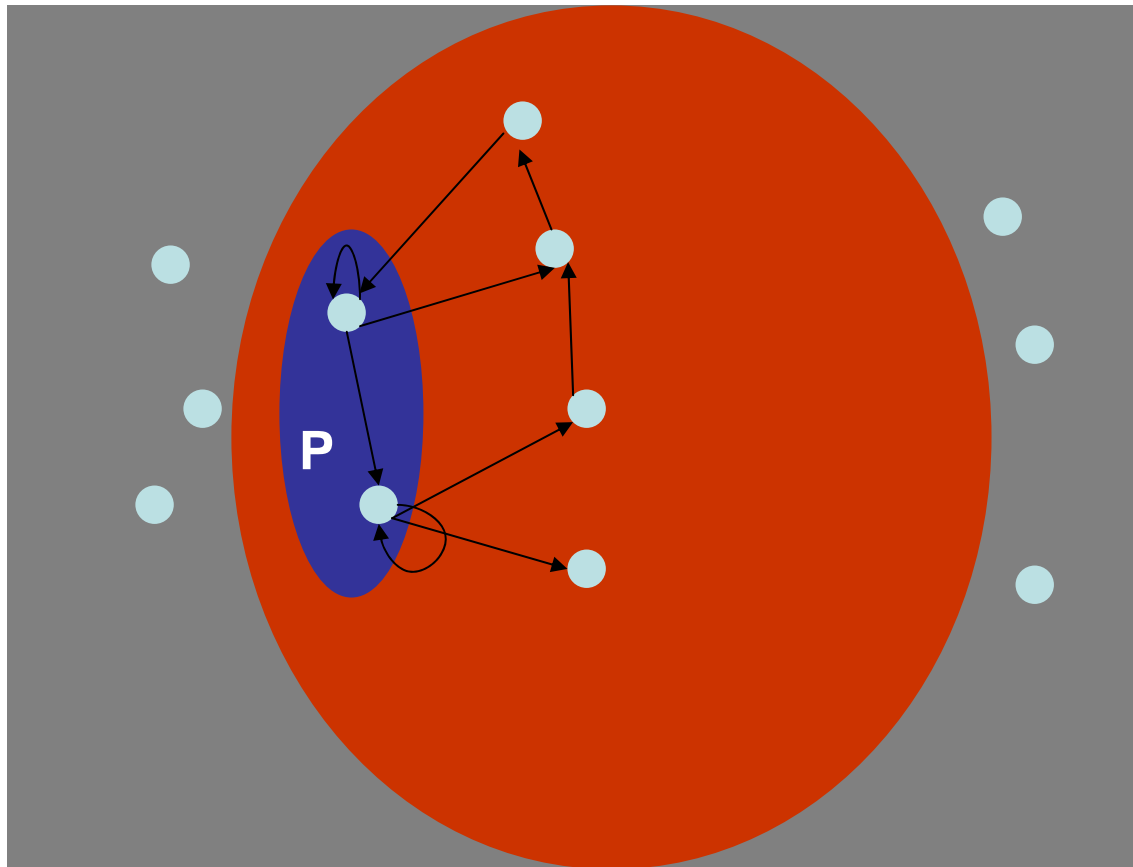
Ensure that waiting is finite,  
(i.e., processes are never stuck forever).

# Graph Representation of State Transitions



States: Vertices  
Transitions: Directed colored edges  
Color: Represents statement – each statement has its own color. In this example there are three statements (colors): red, blue and black (skip).

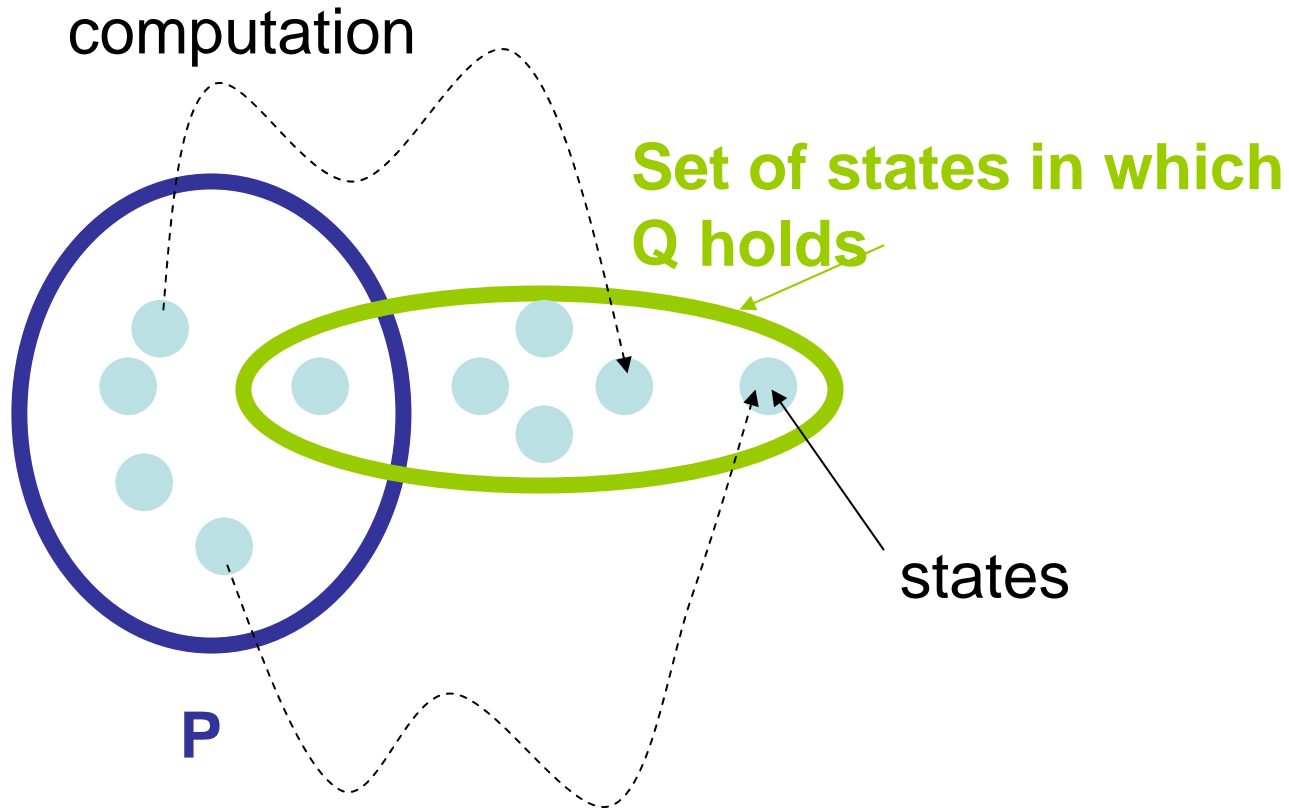
# P Implies Always.X



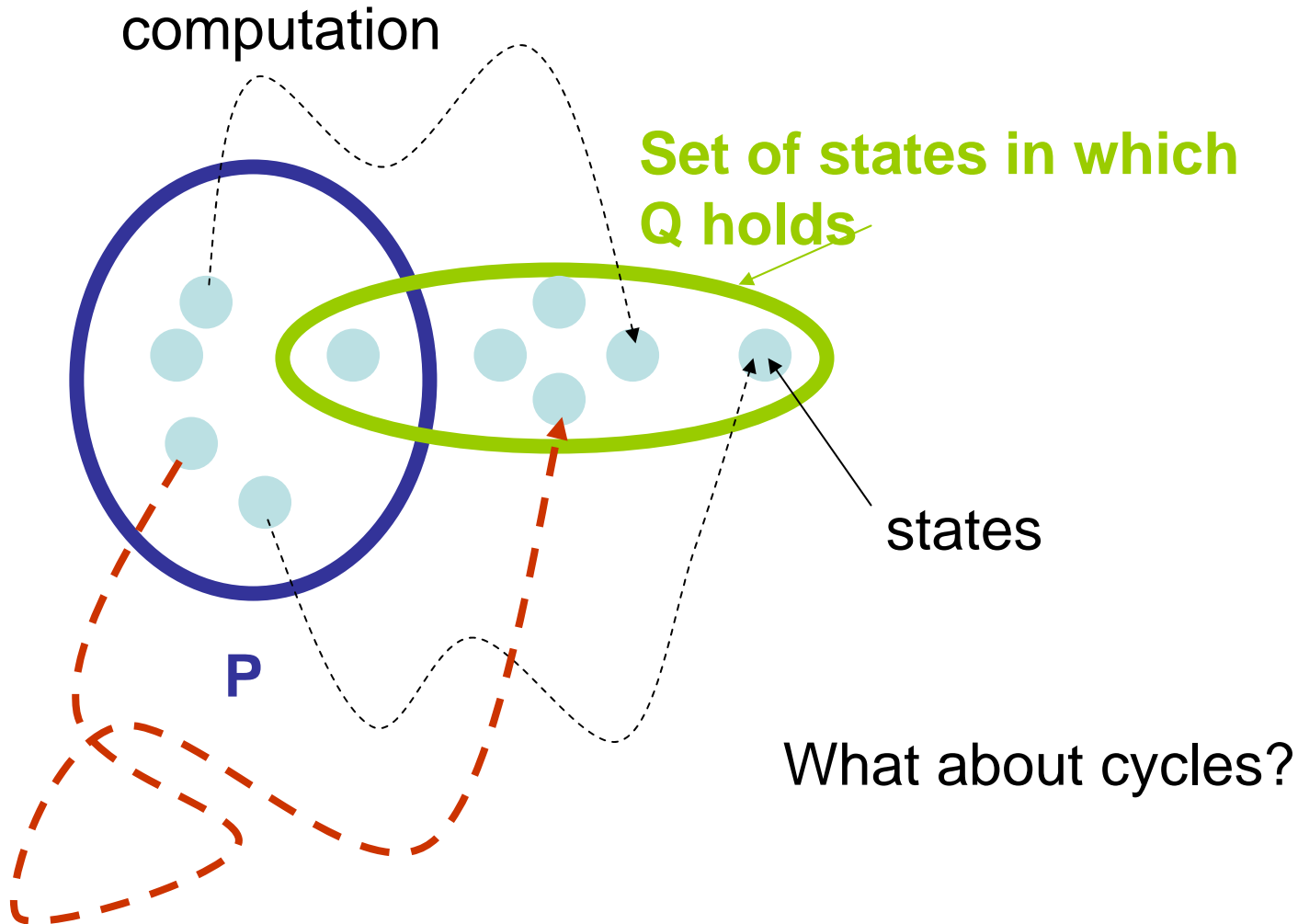
X is a predicate on states.

P Implies Always.X  
means:  
all states reachable  
from states for  
which P holds  
satisfy X.

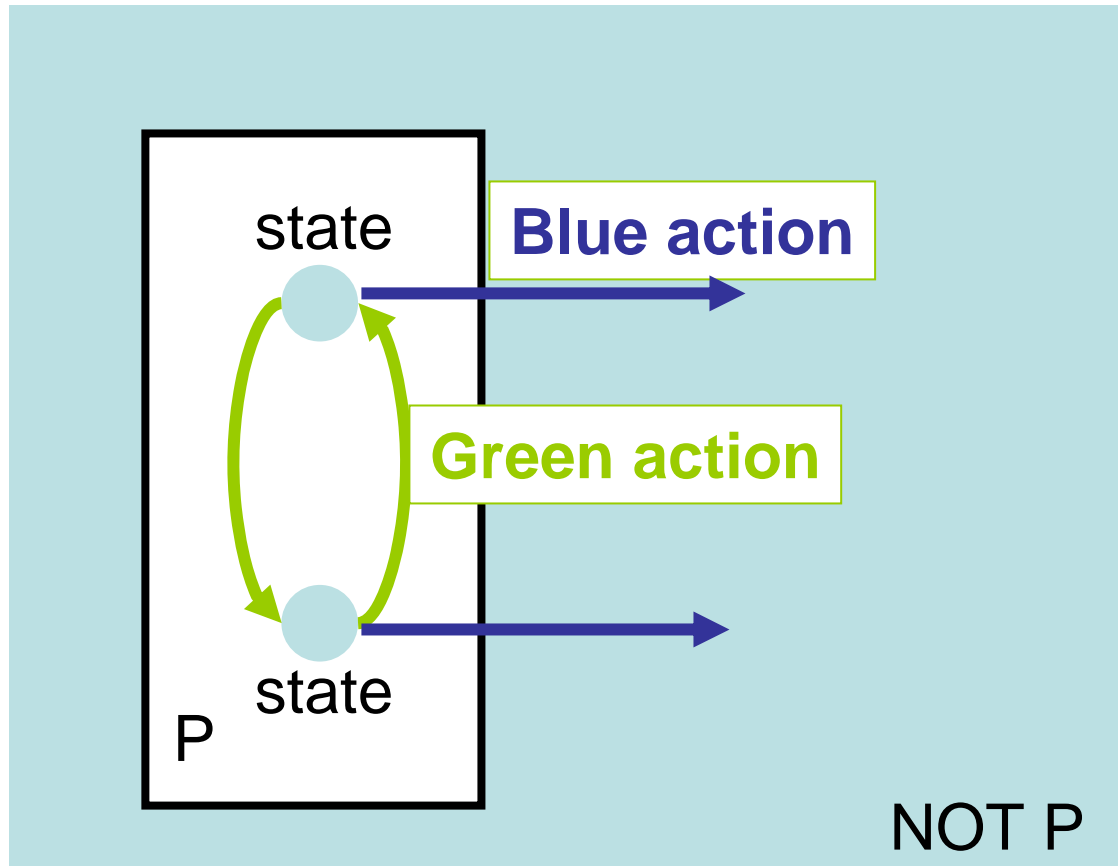
# P Implies Eventually Q



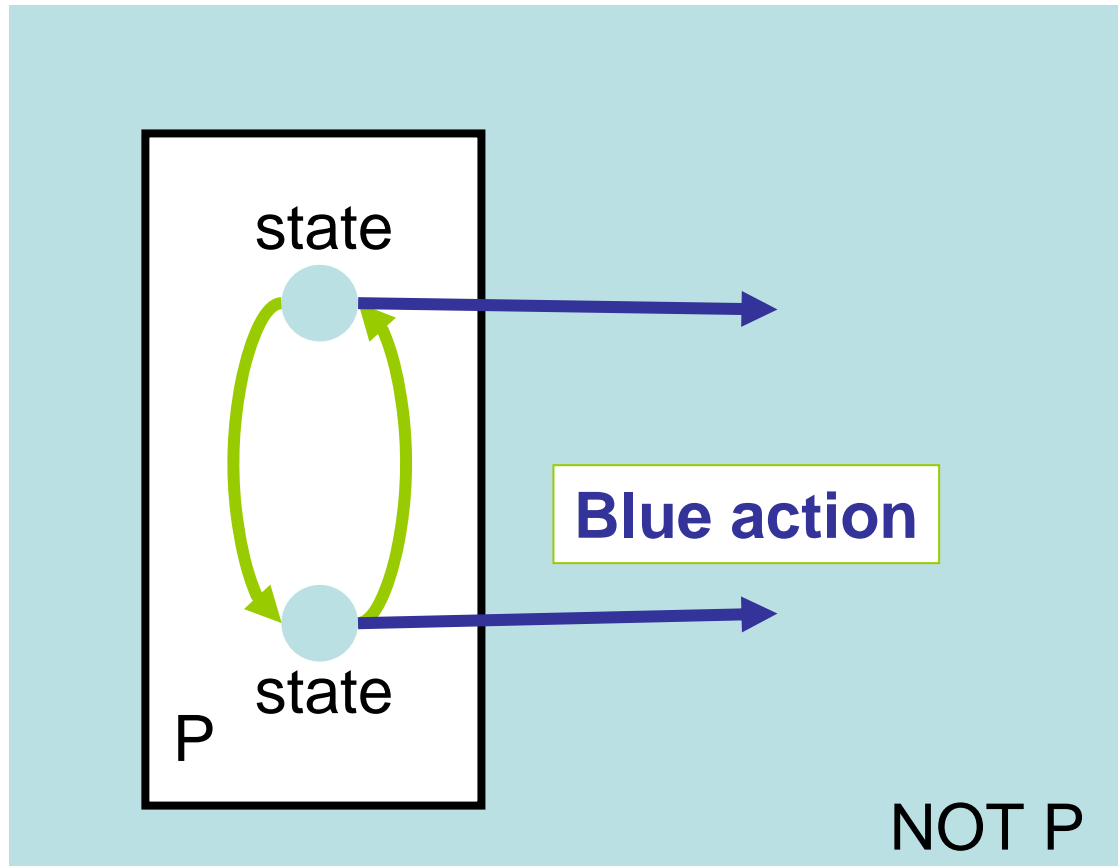
# Pictorial Representation



# Can the System Remain in this Loop Forever?



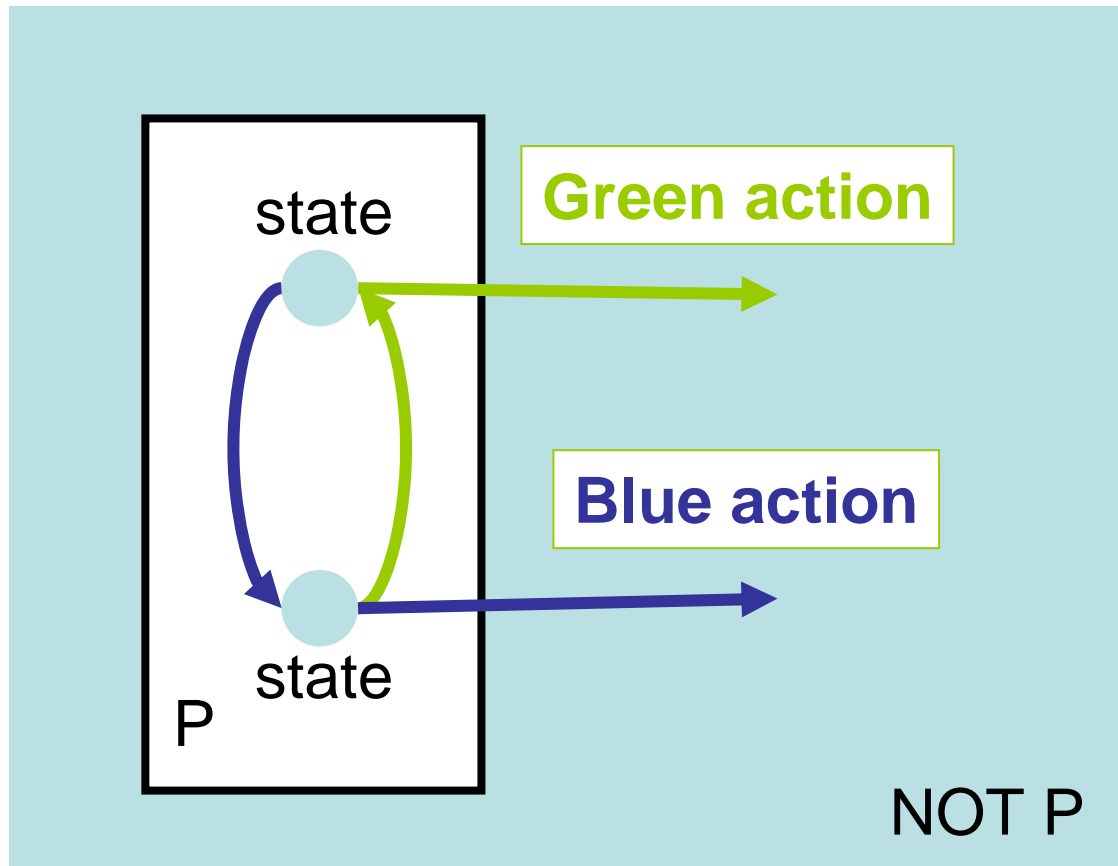
# Can the System Remain in this Loop Forever?



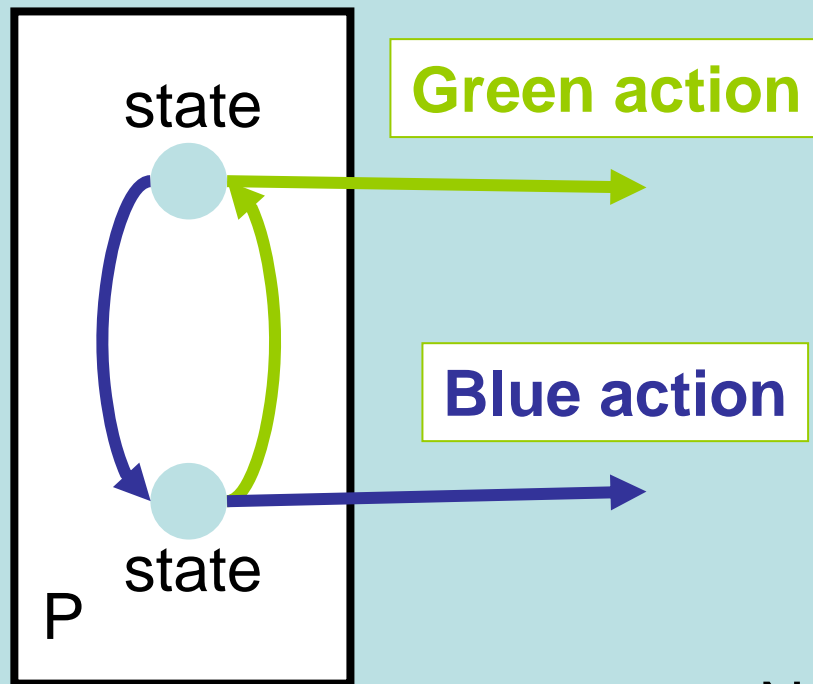
Not if the blue action is executed eventually, i.e., not if the system is “weakly fair.”

NOT P

# Can the System Remain in This Loop Forever?



# Can the System Remain in This Loop Forever?

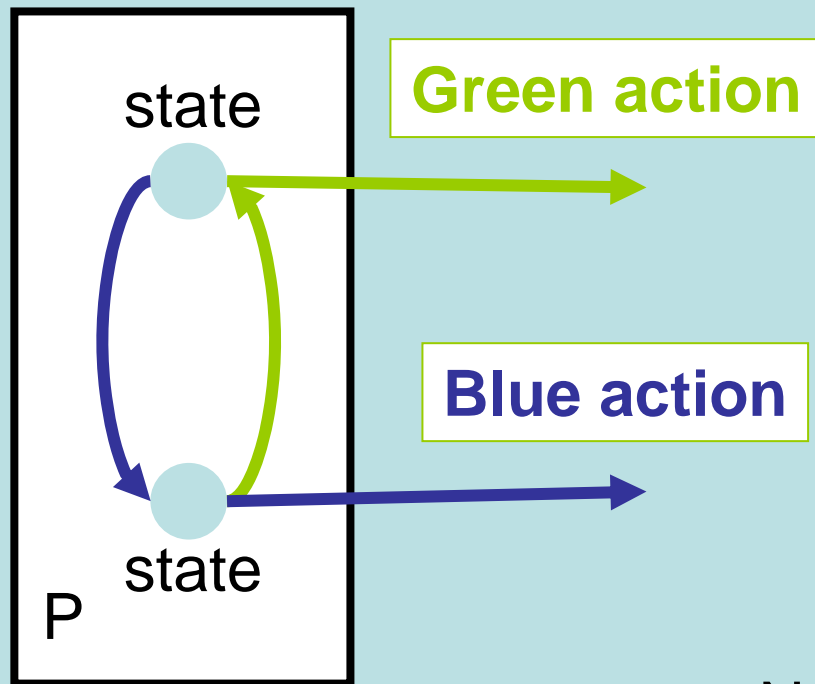


NOT P

Yes, even if the green action is picked eventually and the blue action is picked eventually.

So picking each color infinitely often isn't enough to guarantee exit from the loop.

# Can the System Remain in This Loop Forever?



NOT P

Not if the probability that a color is picked (independently) is at least  $p$  for some positive constant  $p$ , i.e., not if the system is “strongly fair.”



# Overview

1. Control theory approaches
2. Distributed computing theory
3. What are the significant differences? ←  
And, why are there differences?
4. Example: Graph algorithms from the CDS & CS viewpoints.
5. Example: Mobile agents in formation from the CDS and CS viewpoints.
6. Caltech joint CDS/CS research: MURI

# Differences: CDS, CS

- Distributed computing assumes very little: arbitrary numbers of agents, arbitrary message delays, arbitrary failure periods (but finite).
- Control theory assumes a lot more: continuous interaction, bounded delays

# Differences: CDS & CS

- Distributed computing theory proves relatively little about progress: eventually good things happen.
- Control theory proves a lot about dynamics.

# Differences: CDS & CS

Distributed computing theories are partitioned into theories about:

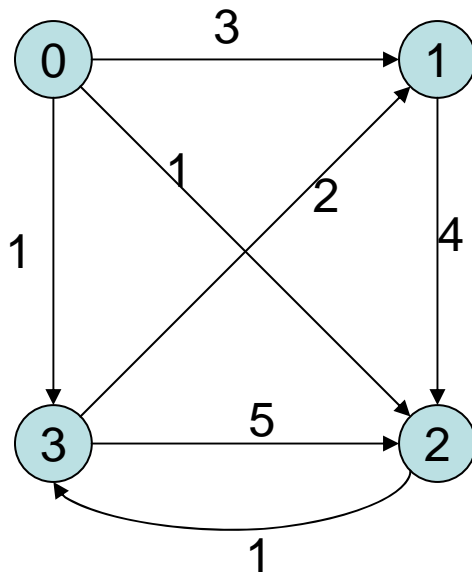
1. Correctness: **Boolean**
2. “ilities”: Reliability, maintainability, performance, ... that have **continuous measures**.

# Overview

1. Control theory approaches
2. Distributed computing theory
3. What are the significant differences?  
And, why are there differences?
4. **Example: Graph algorithms from the  
CDS & CS viewpoints. ←**
5. Example: Mobile agents in formation  
from the CDS and CS viewpoints.
6. Caltech joint CDS/CS research: MURI

# Control Theory: Graph Dynamics

$$\frac{dX}{dt} = \Lambda \cdot X$$



$$\Lambda = \begin{pmatrix} -5 & 3 & 1 & 1 \\ 0 & -4 & 4 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 2 & 5 & -7 \end{pmatrix}$$

$$\frac{dX[j]}{dt} = \sum W[j, k] \cdot (X[k] - X[j])$$

↑  
Weight on edge from j to k

# Graph Dynamics: Consensus

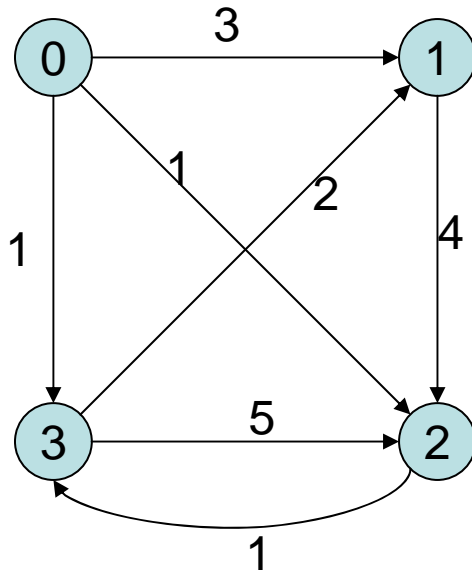
**Equilibrium:  $dX/dt = \Lambda \cdot X = 0$**

Notation:  $\underline{k} = [k, k, \dots, k]^T$  is a consensus vector: “all agents come to a consensus  $k$ ”

$\Lambda \cdot \underline{k} = \mathbf{0}$  because rows of  $\Lambda$  sum to 1

# Discrete State, Continuous Time, Markov Processes

Probability of transition from state  $j$  to state  $k$  in interval  $[t, t+dt]$  given system is in state  $j$  at  $t$  is  $\Lambda[j,k].dt$



$$\Lambda = \begin{pmatrix} -5 & 3 & 1 & 1 \\ 0 & -4 & 4 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 2 & 5 & -7 \end{pmatrix}$$

# Graph and Markov Processes

$P$  is a row vector,  $X$  is a column vector

**Markov Dynamics:**  $dP/dt = P \cdot \Lambda$

**Graph Dynamics:**  $dX/dt = \Lambda \cdot X$

# Conservation Law

- Restrict attention to ergodic Markov processes
- Let  $\Pi$  be the equilibrium probability (row) vector for the Markov process.

## Theorems:

1. Conservation Law:  $\Pi.X = \text{constant}$ .
2. Unique equilibrium for  $X$  is a consensus  $\underline{k}$

## Corollary:

Unique equilibrium: for all  $j$ :  $X[j] = (\Pi.X^{(0)})$

# Graph Dynamics: Computing Averages

Theorem: All agents converge to the average of initial values if  $\Lambda$  is doubly stochastic.

Proof:

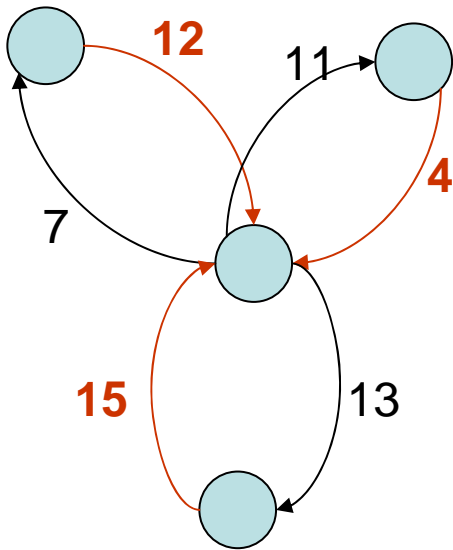
Unique equilibrium: for all  $j$ :  $X[j] = (\Pi.X^{(0)})$

For average we want: for all  $j$ :  $X[j] = (\sum_k X_k^{(0)})/N$

Hence,  $\Pi = [1/N, 1/N, \dots, 1/N]$  is a solution

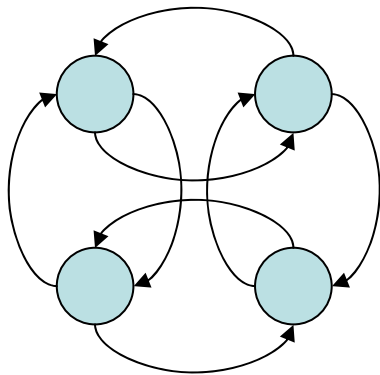
This solution is obtained when  $\Lambda$  is doubly stochastic

# Doubly Stochastic

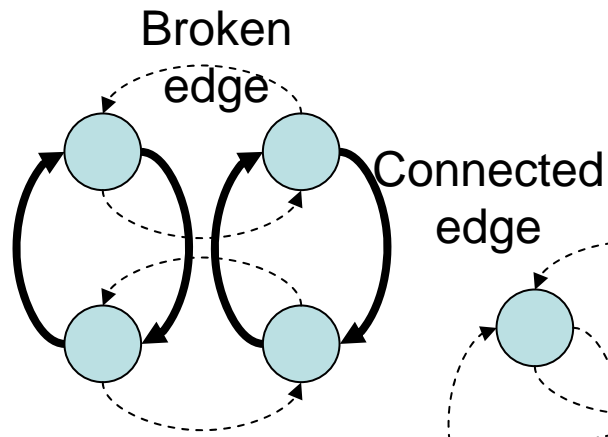


Sum of outgoing weights  
=  
sum of incoming weights

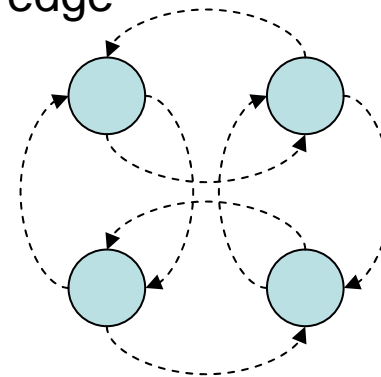
# CDS Questions: What if edge weights change?



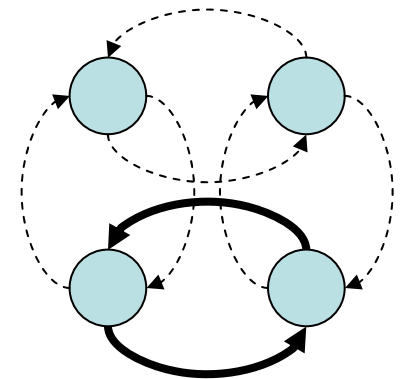
Phase 1



Phase 2



Phase 3



Phase 4

- Opponent picks a phase (edge weights) and specifies the duration for each phase.
- What constraints on the opponent guarantee convergence to the same results?

# A CDS Approach: Tsitsiklis

Phases defined as matrices (or operators)

$$dX/dt = \Lambda^{(k)} \cdot X$$

where  $\Lambda^{(k)}$  is a member of a set  $Z$  of matrices.

Opponent picks any matrix from the given set for each interval of time.

# CDS & CS Joint Research Questions

- What does convergence mean in this context?
  - **Eventually**, no matter what the opponent does, the system state converges.
  - Example: Initial condition P guarantees convergence to the origin is:
    - For all  $\varepsilon > 0$ :
    - P implies eventually always**  $\|x\| < \varepsilon$

# CDS & CS Joint Research Questions

- What do the (different) definitions of stability mean in this context?
- The origin is stable means:
  - for every  $\varepsilon > 0$  there exists a  $\delta = \delta(\varepsilon) > 0$ , such that:  
 $\|x(t)\| < \varepsilon$  for all  $t$  if  $\|x(0)\| < \delta$   
no matter what the opponent does
- $\|x\| < \delta$  **implies** *always*  $\|x\| < \varepsilon$

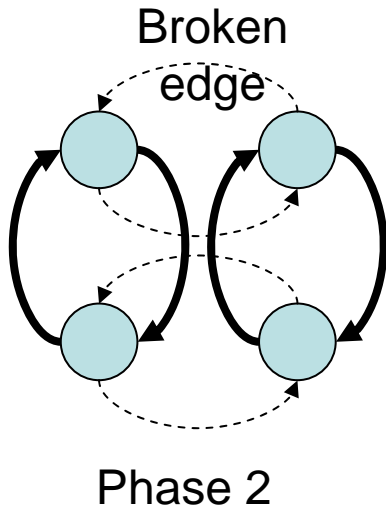
# CDS & CS Joint Research Questions

How are constraints on the opponent specified?

- Example: System is never permanently partitioned into non-communicating subsets.
- **Eventually**, for every subset  $S$ , an agent in  $S$  communicates with an agent in its complement.

# CS & CDS Joint Research Questions

Questions: What role does the graph play?



Three types of roles:

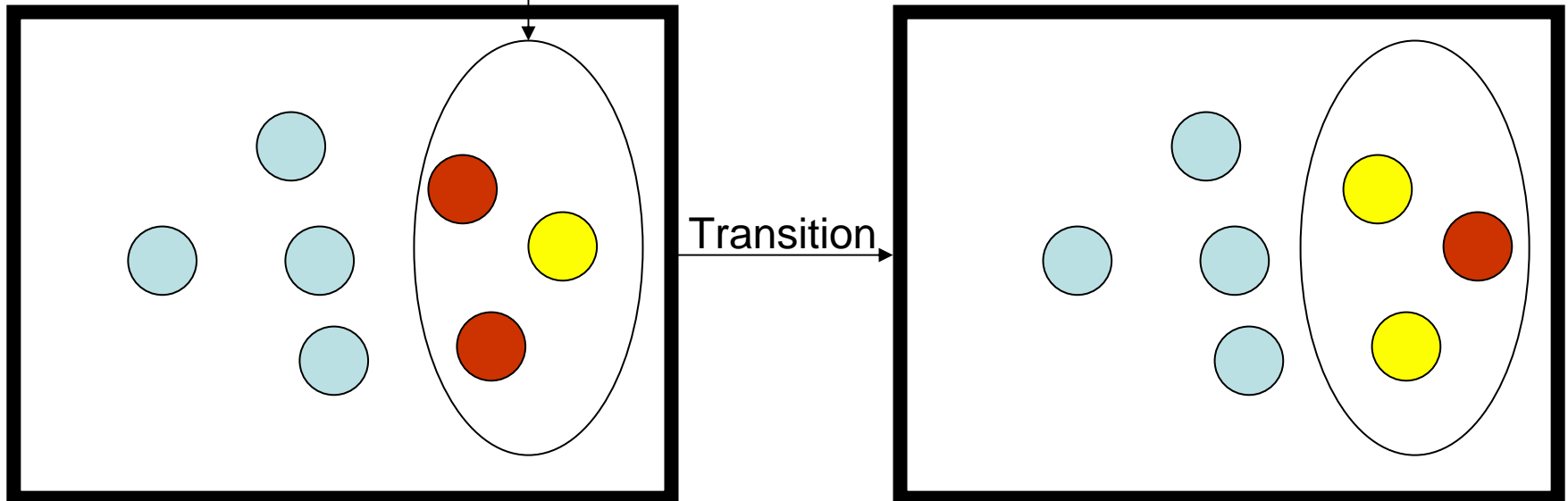
1. Independent subsets:  
Partitioning the system into non-communicating subsets of agents.
2. State change of each subset:  
Graphs specify the dynamics within each subset.
3. Compositional structure of interactions

# Challenges:

## How to represent dynamics?

Represent dynamics abstractly: A collection  $C$  of agents changes their states while states of other agents remain unchanged and the state change satisfies some relation  $R_C$ .

Meeting of group of agents



BEFORE MEETING

AFTER MEETING

# Challenge: Representing Dynamics

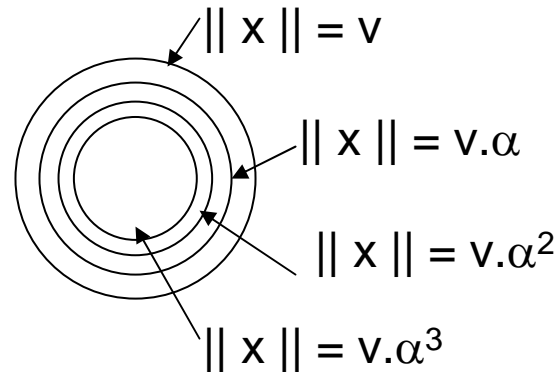
- $f$  is a local-to-global monotone function means, for any bags  $a, b, c$  where  $|a| = |b|$ :
  - $f(a) < f(b)$  IMPLIES  $f(a \cup c) \leq f(b \cup c)$
  - e.g., min, max, sum, average,...
- $f$  is a local-to-global strongly monotone function means, for any bags  $a, b, c$  where  $|a| = |b|$ :
  - $f(a) < f(b)$  IMPLIES  $f(a \cup c) < f(b \cup c)$
  - e.g., sum, average,...

# Combining Dynamics with Temporal Logic

How can we prove, for all  $\varepsilon > 0$ :  
eventually always  $\|x\| < \varepsilon$  ?

We show that there exists an  $\alpha$  where  $0 \leq \alpha < 1$  such that:  
for all  $v$ :

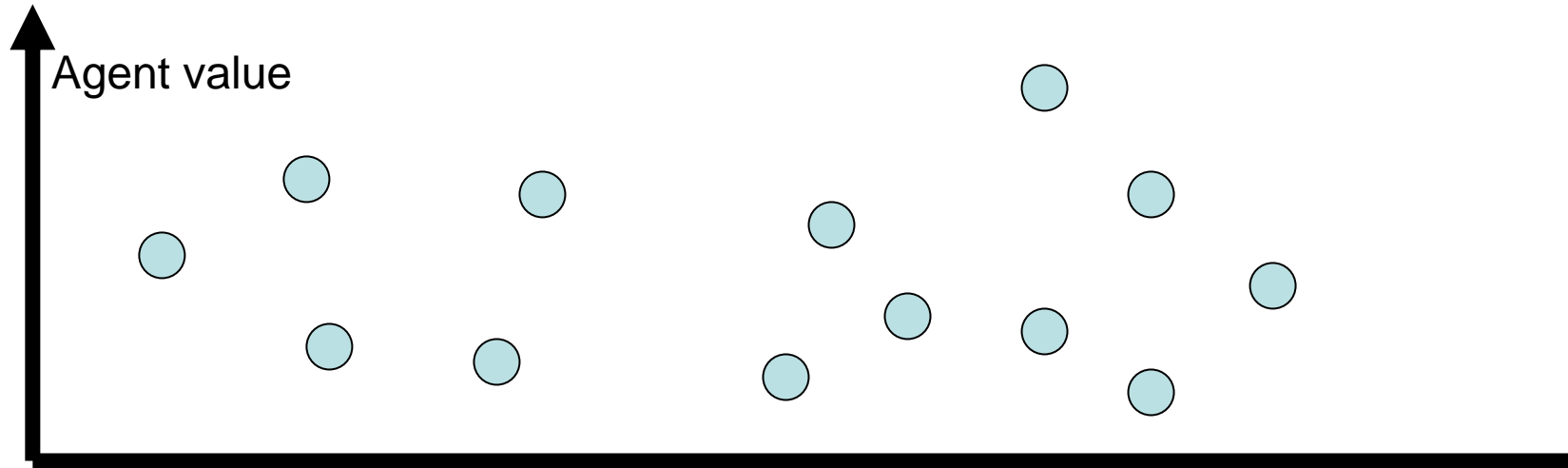
1.  $\|x\| \leq v$  **implies eventually**  $\|x\| \leq v.\alpha$ , and
2.  $\|x\| \leq v$  **implies always**  $\|x\| \leq v$



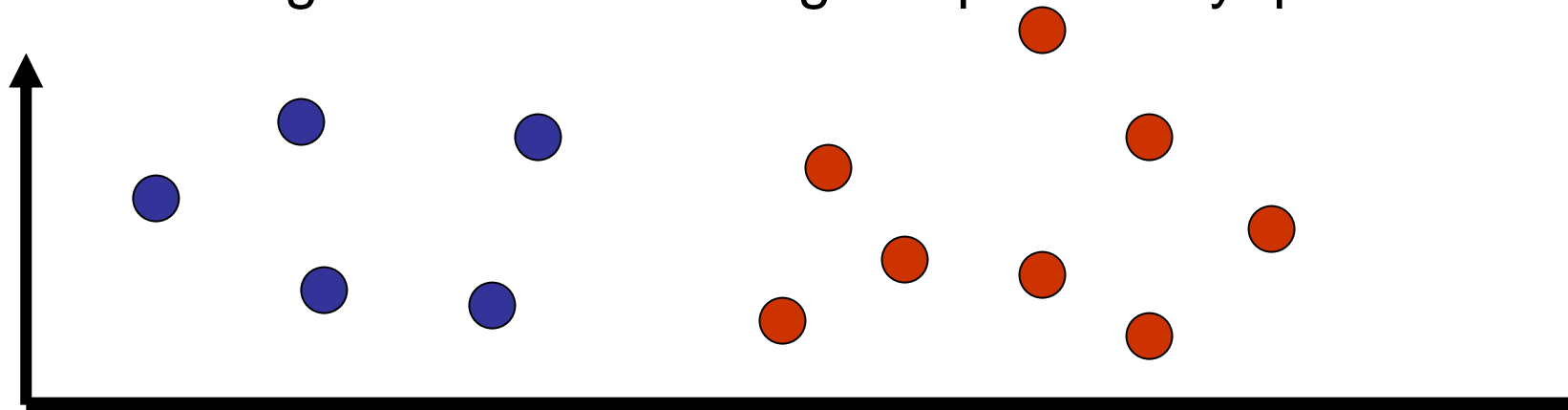
# Example: Averages with Arbitrary Communication Failure

- Conservation Law: Any operation that conserves averages locally, conserves averages globally.
- For Lyapunov function:  $\sum x[j]^2$ 
  1.  $\sum x[j]^2 \leq V$  **implies** *always*  $\sum x[j]^2 \leq V$
  2. Find  $\alpha$  where  $0 \leq \alpha < 1$  such that:  
 $\sum x[j]^2 = V$  **implies** *eventually*  $\sum x[j]^2 \leq V \cdot \alpha$

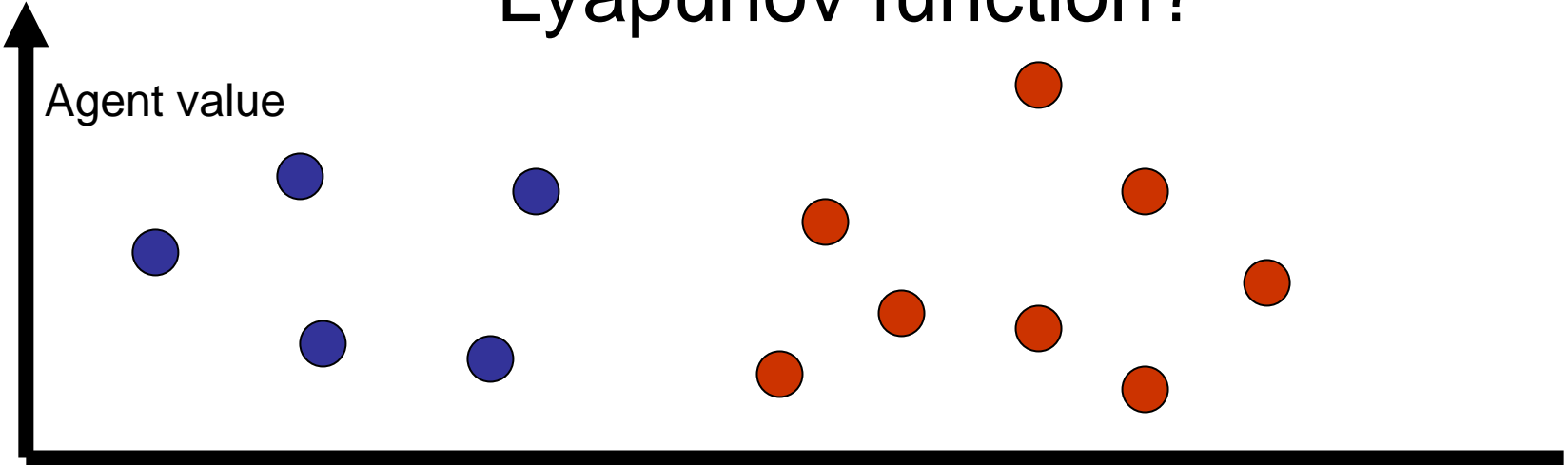
# From Continuous to Discrete



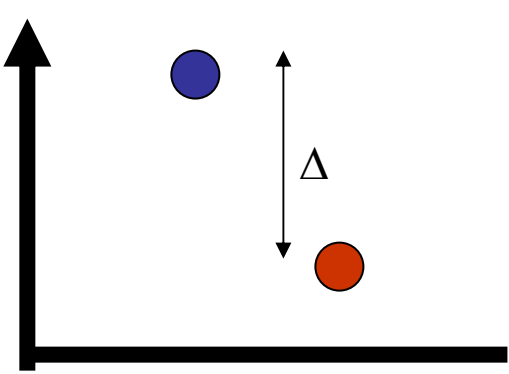
Find a red-blue partition such that any operation between red and blue agents causes a "big" drop in the Lyapunov fn.



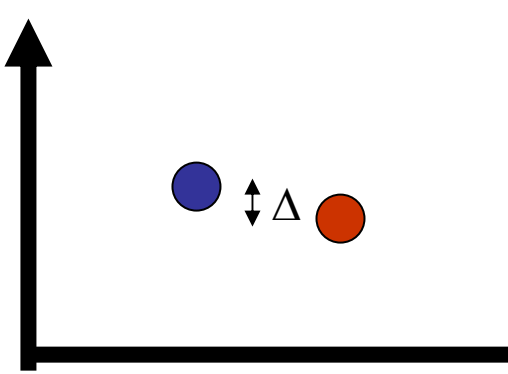
# What partition results in “big” drop of the Lyapunov function?



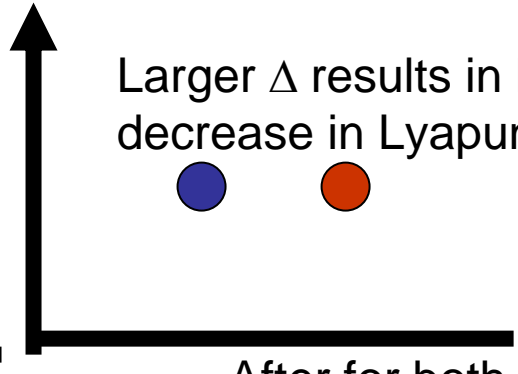
Find a partition such that any operation between red and blue agents causes a big drop in the Lyapunov function



Before 1



Before 2

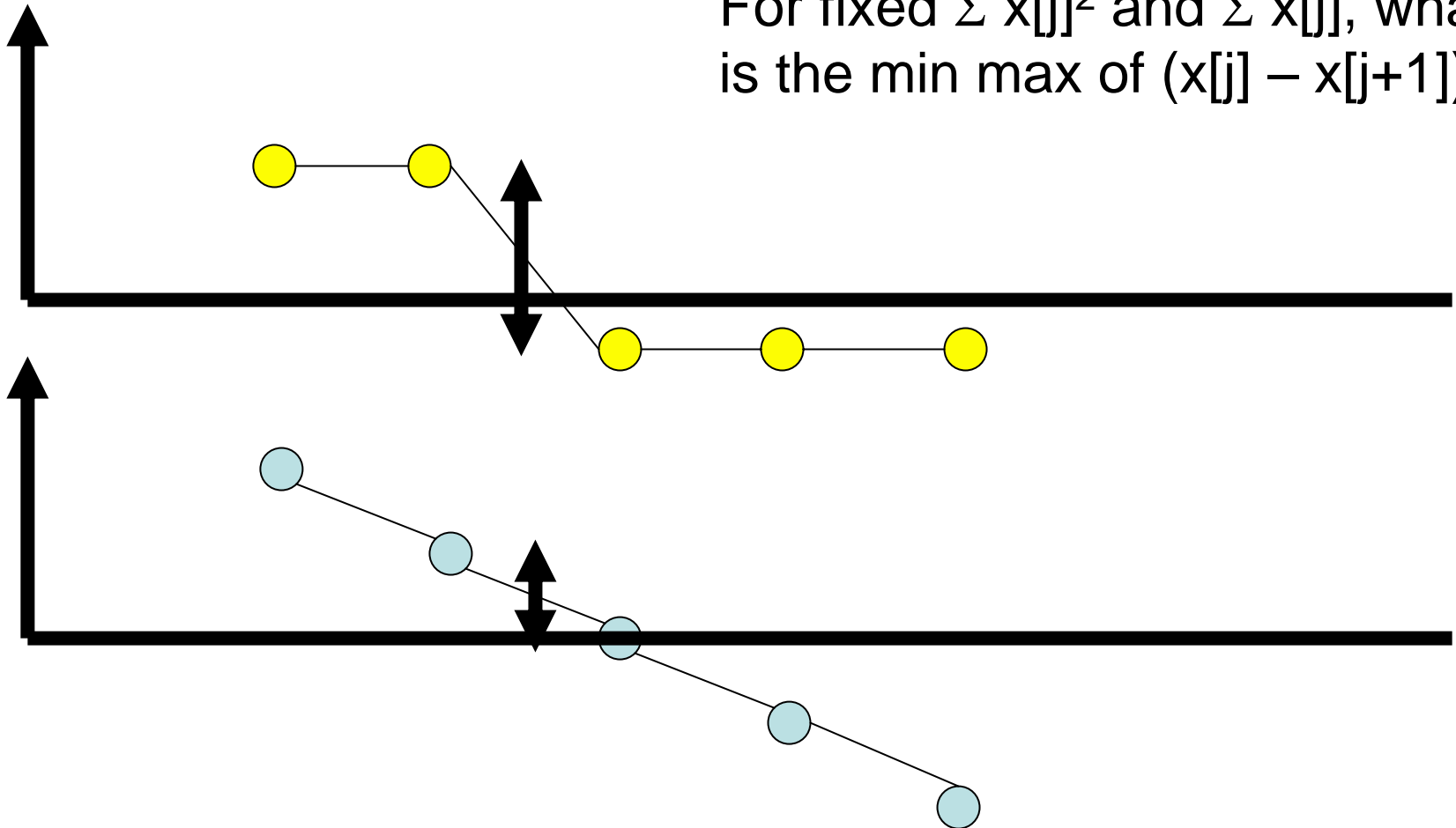


After for both cases

Larger  $\Delta$  results in larger decrease in Lyapunov fn.

# What partition results in “big” drop of the Lyapunov function?

For fixed  $\sum x[j]^2$  and  $\sum x[j]$ , what is the min max of  $(x[j] - x[j+1])$  ?



# Current Work

- Formations of mobile agents with messages
- Automatic theorem proving for continuous systems (PVS)
- Libraries of theorems for dynamics
- Design refinement

