Multiagent Dynamical Systems

James P. Crutchfield
Computational Science & Engineering Center and
Physics Department
University of California, Davis
cse.ucdavis.edu/∼chaos

2 June 2005

Abstract We show how to model multiagent systems using dynamical systems theory. We adapt our reinforcement-learning multiagent dynamics framework to model various kinds of agent collective. Several examples illustrate a close connection with game theory and evolutionary dynamics. We also model a collective in which N-agents service M time-sensitive tasks. We analyze, both numerically and analytically, the resulting replicator equations and the collective transient, equilibrium, and dynamical behaviors. Using these results we derive predictions for individual and collective behaviors and characterize reward structures that lead to optimal collective performance and resource use. As part of the latter, we adapt measures of synchronization and structural complexity to quantify the degree of collective coordination.

Joint work with *Dave Albers* (MPI Complex Systems, Leipzig; SFI; Physics, UWisconsin, Madison), *Manuel Sanchez-Montanes* (Computer Science, U. Autonoma Madrid), and *Yuzuru Sato* (SFI and UTokyo)

Agenda:

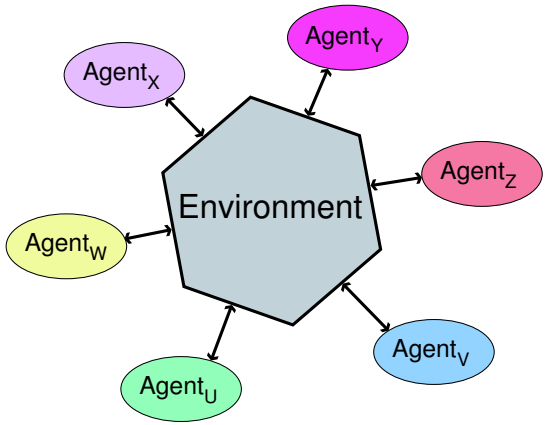
- Introduction
- Reinforcement Learning
- Multiagent Dynamical Systems
- Example Agent Collectives
  - Matching-Pennies Interaction
  - Rock-Scissors-Paper Interaction
  - Multiple Agents Servicing Multiple Tasks
- Structured MADS
- Information Space
- Discrete-Time Adaptive Maps

Central Problem:

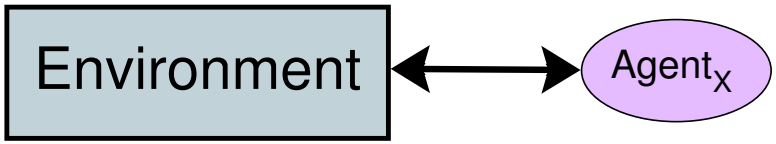
Many agents with limited information, What is the Global Behavior?

Concerns that arise for multiagent systems:

- Hierarchical:
  - adaptive agents
  - dynamic environment
  - collective behaviors differ from individual
- High-dimensional stochastic processes
- Where is a predictive theory?



Single-Agent Adaptation:



N possible actions: $i = 1, 2, \ldots, N$.

Environment: $r_i(\tau)$ is the reinforcement for taking action i.

Agent's memories are $\mathbf{Q}(\tau) = (Q_1(\tau), \ldots, Q_N(\tau))$ are updated:

$$Q_i(\tau + 1) - Q_i(\tau) = \frac{1}{T}\left[\delta_{i\tau} r_i(\tau) - \alpha Q_i(\tau)\right]$$

$\alpha = 0$: the agent has a perfect memory.
$\alpha > 0$: memory is attenuated.

Agent's *state* is its *choice distribution*: $\mathbf{x}(\tau) = (x_1(\tau), \ldots, x_N(\tau))$.

Agent translates rewards to choice probabilities using:

$$x_i(\tau) = \frac{e^{\beta Q_i(\tau)}}{\sum_k e^{\beta Q_k(\tau)}} \ ,$$

$\beta \in [0, \infty]$ controls the *adaptation sensitivity*.

$\beta = 0$: Choice unaffected by reward memory; $x_i = N^{-1}$.

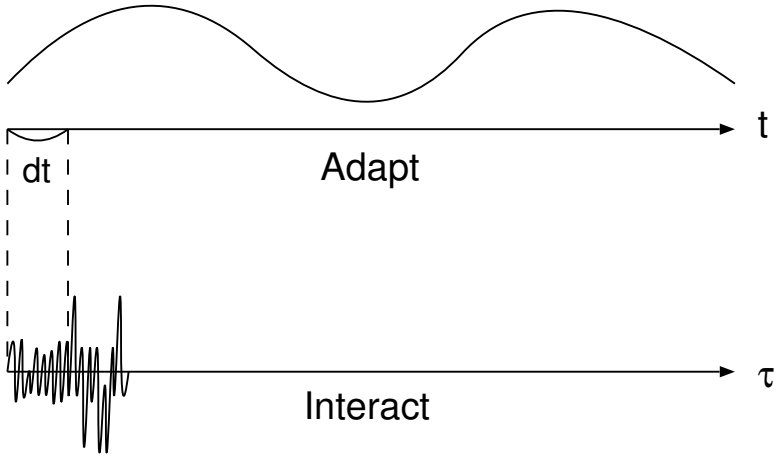
$\beta = \infty$: Always choose action with maximum reward.

Dynamics of the choice distribution is:

$$x_i(\tau + 1) = \frac{x_i(\tau) e^{\beta[Q_i(\tau+1) - Q_i(\tau)]}}{\sum_{n=1}^{N} x_n(\tau) e^{\beta[Q_n(\tau+1) - Q_n(\tau)]}}$$

Adaptation: aka Q-learning (machine learning), operant conditioning (Hebb), stochastic learning, ...

Assume adaptation is very slow, compared to interactions: $\tau \ll t$.



Continuous-time limit:

$$\dot{Q}_i(t) = R_i + \alpha Q_i(t), \text{ where } R_i = \langle r_i(\tau) \rangle_{\tau \in [t, t+dt]}$$

$$\dot{x}_i(t) = \beta x_i(t) \left[\dot{Q}_i(t) - \sum_{n=1}^{N} \dot{Q}_n(t) x_n(t) \right]$$

Choice distribution dynamic (remove memory variable):

$$\frac{\dot{x}_i}{x_i} = \beta \left[R_i - \sum_{n=1}^{N} x_n R_n \right] + \alpha \left[-\log x_i + \sum_{n=1}^{N} x_n \log x_n \right]$$

In continuous-time limit, adaptation is governed by

$$\frac{\dot{x_i}}{x_i} = \beta(R_i - \Sigma_{k=1}^{N} x_k R_k) + \alpha(H_i - H) \ ,$$

where R_i is the integrated reinforcement and $H_i = -\log x_i$.

$R_i - \Sigma_k x_k R_k$ is the relative benefit compared to the mean.

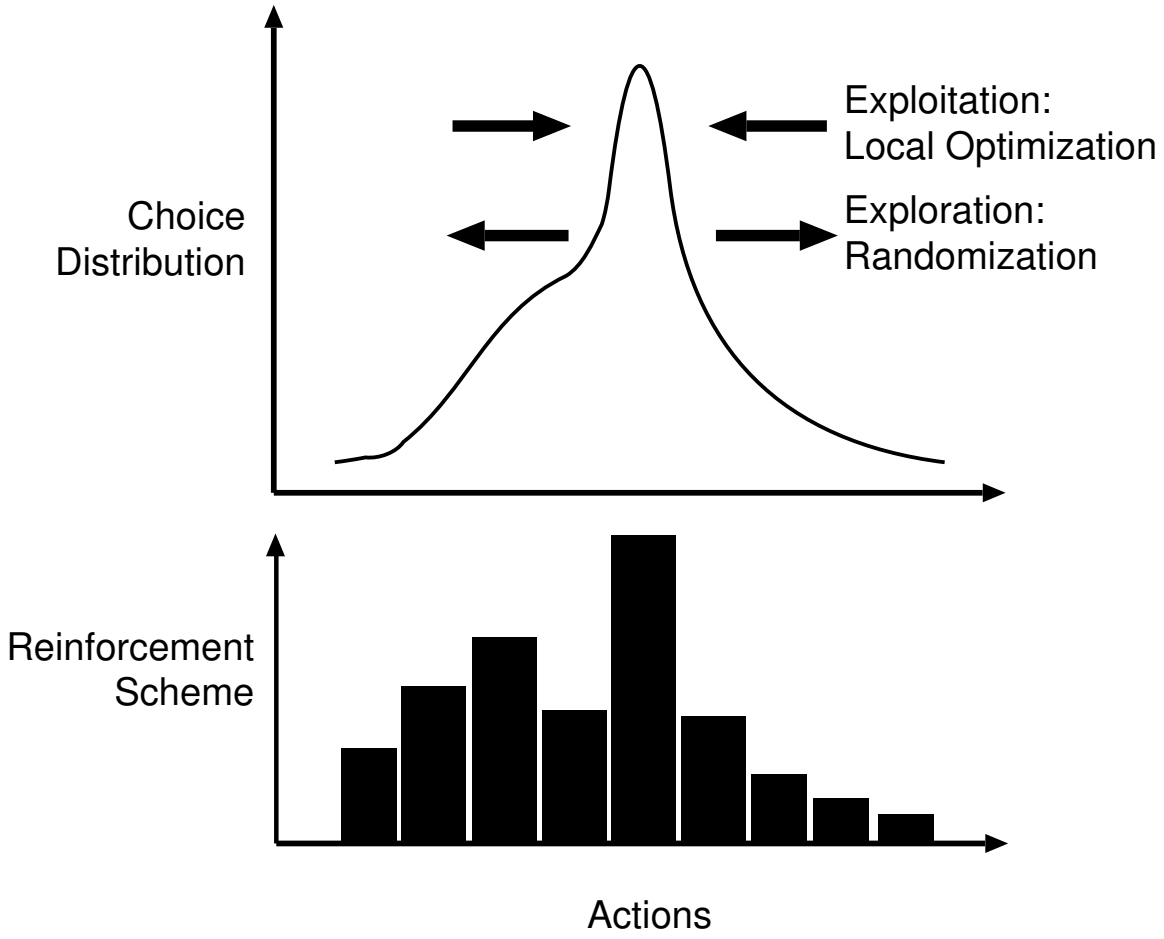
$H_i - H$ is relative informativeness compared to the average surprise H.

Adaptation is a dynamic balance between

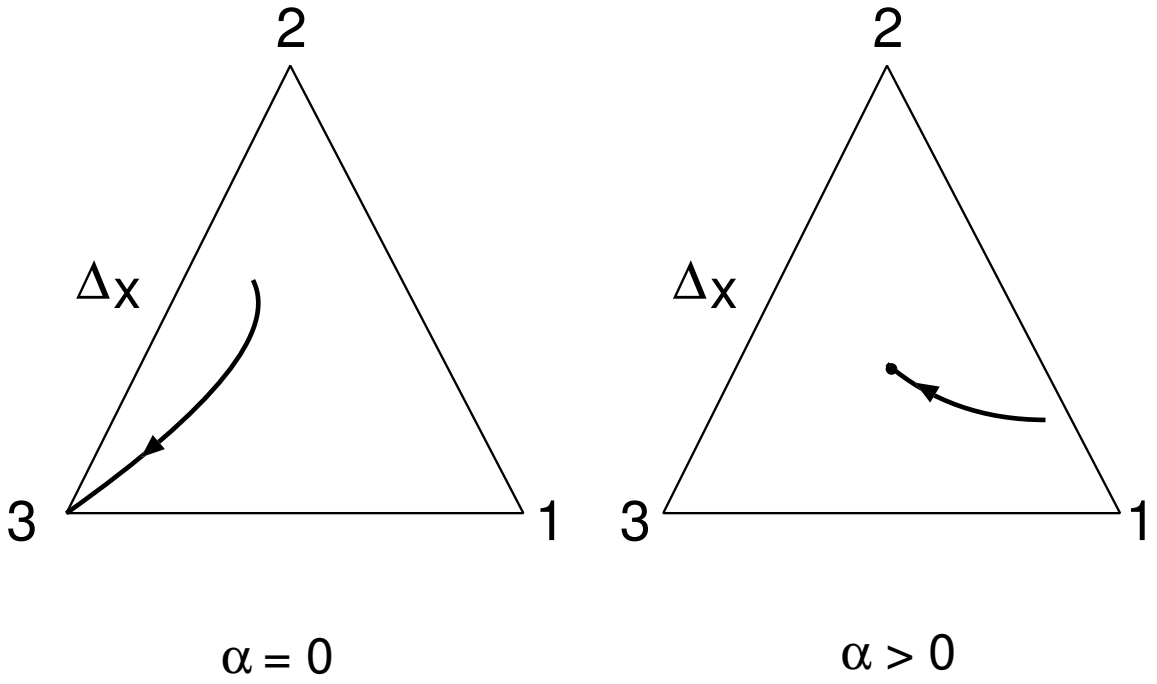
- *Exploitation* occurs when the agent adapts to the environmental constraints: specializes on the action with the highest reward.

- *Exploration* occurs when the entropy term dominates and equalizes the choice probabilities.

Adaptation as a Dynamic Balance of Exploitation and Exploration



The Extremes: Exploit versus Explore



Single Agent with 3 Actions: $\{1, 2, 3\}$; $\mathbf{R} = \left(\frac{1}{3}, -\frac{7}{6}, \frac{5}{6}\right)$; $\beta = 1.0$.

$\alpha = 0 \Rightarrow \vec{x} = (0, 0, 1)$.

$\alpha > 0 \Rightarrow \vec{x} = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$.

Multiagent Dynamical Systems

Introduced: Y. Sato, E. Akiyama, & JPC, Physical Review E **67**:1 (2003) 40–43.
Review: Y. Sato, E. Akiyama, & JPC, Physica D (2005) submitted.

Two agents, X and Y:

$$\begin{aligned} \frac{\dot{x}_i}{x_i} &= \beta_X(R_i^X - R^X) + \alpha_X(H_i^X - H^X) \\ \frac{\dot{y}_j}{y_j} &= \beta_Y(R_j^Y - R^Y) + \alpha_Y(H_j^Y - H^Y) \ , \end{aligned} \tag{1}$$

Fixed relationship between actions and rewards, then

$$\begin{aligned} \frac{\dot{x}_i}{x_i} &= \beta_X[(A\mathbf{y})_i - \mathbf{x}\cdot A\mathbf{y}] + \alpha_X[H_i^X - H^X] \\ \frac{\dot{y}_j}{y_j} &= \beta_Y[(B\mathbf{x})_j - \mathbf{y}\cdot B\mathbf{x}] + \alpha_Y[H_j^Y - H^Y] \ , \end{aligned} \tag{2}$$

A and B are matrices.

Replicator equations (pop. & ev. biology), but with the memory term.

Describes an effective game-interaction between agents.

Matching Pennies

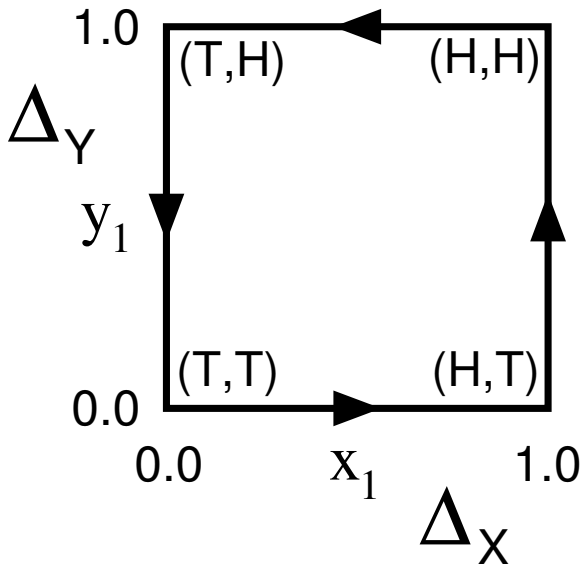
Y wins if pennies match; otherwise X wins

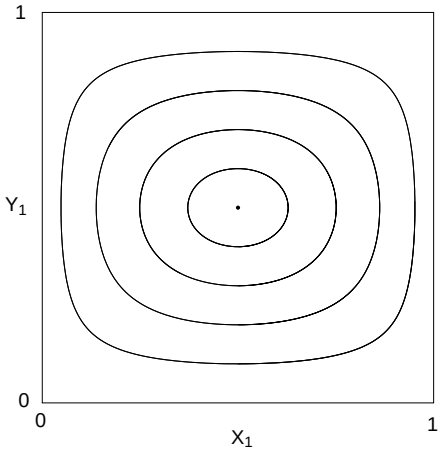
Interaction matrices:

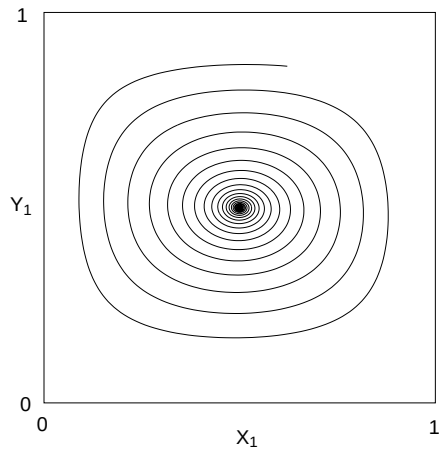
$$A = \begin{bmatrix} -\epsilon_X & \epsilon_X \\ \epsilon_X & -\epsilon_X \end{bmatrix} \text{ and } B = \begin{bmatrix} -\epsilon_Y & \epsilon_Y \\ \epsilon_Y & -\epsilon_Y \end{bmatrix} , \tag{3}$$

$\epsilon_X \in (0.0, 1.0]$ gives agent X's reward for winning.

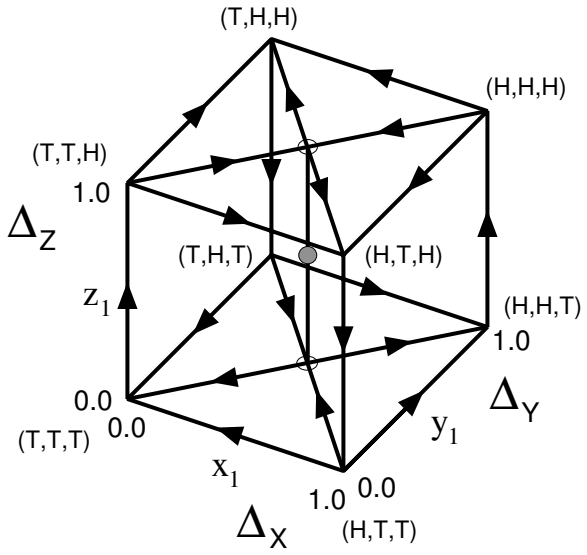
$-\epsilon_Y \in (0.0, 1.0]$ gives agent Y's reward for winning.

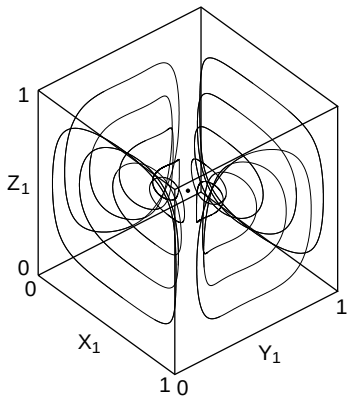

1.0
(T,H)
(H,H)
Δ_Y
y_1
(T,T)
(H,T)
0.0
0.0
x_1
1.0
Δ_X

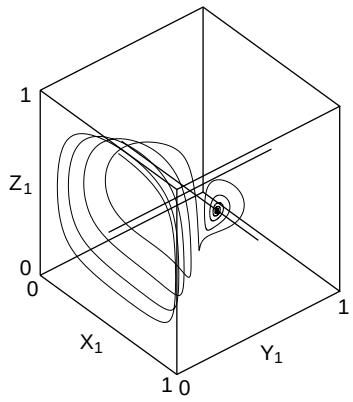

1
Y_1
0
0
1
X_1


1
Y_1
0
0
1
X_1

Three Agents in the Even-Odd Pennies Game







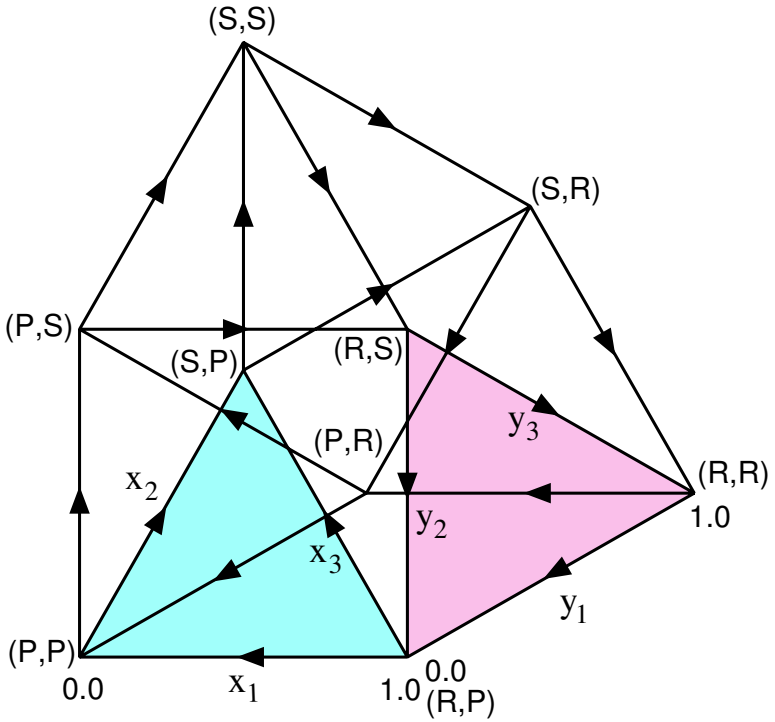
Rock-Scissors-Paper

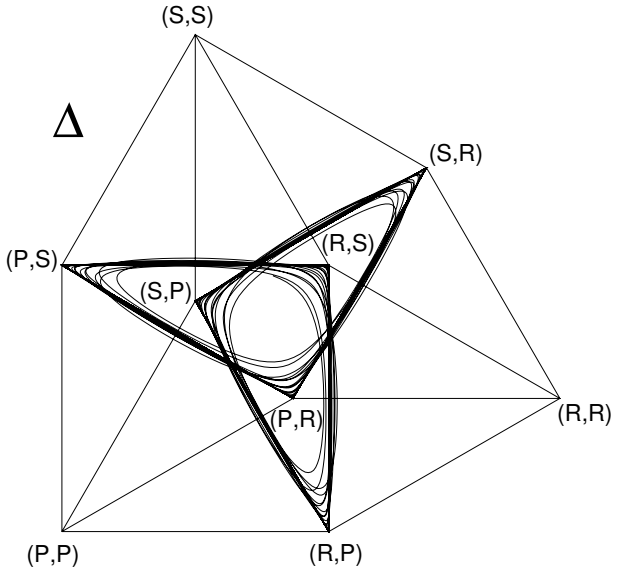
Rock beats Scissors beats Paper beats Rock.

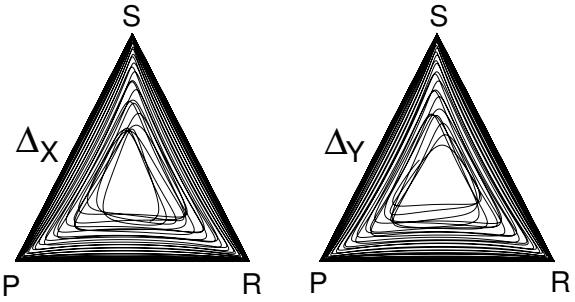
Interaction matrices for two agents:

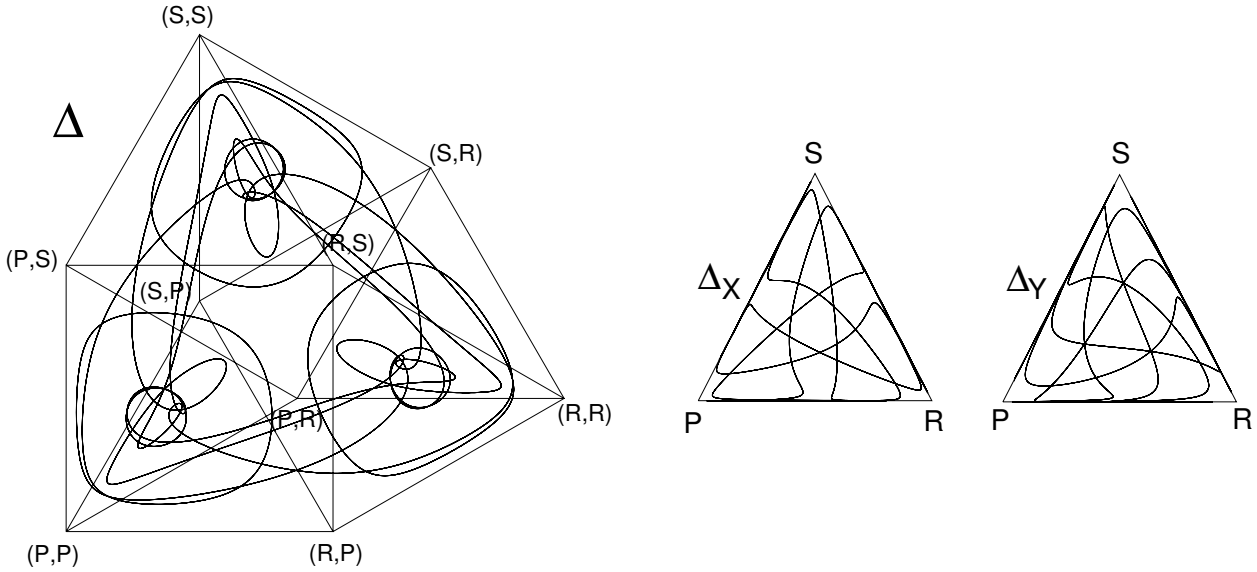
$$A' = \begin{bmatrix} \epsilon_X & 1 & -1 \\ -1 & \epsilon_X & 1 \\ 1 & -1 & \epsilon_X \end{bmatrix} \text{ and } B' = \begin{bmatrix} \epsilon_Y & 1 & -1 \\ -1 & \epsilon_Y & 1 \\ 1 & -1 & \epsilon_Y \end{bmatrix}, \qquad (4)$$

$\epsilon_X, \epsilon_Y \in [-1.0, 1.0]$ are the rewards for ties.

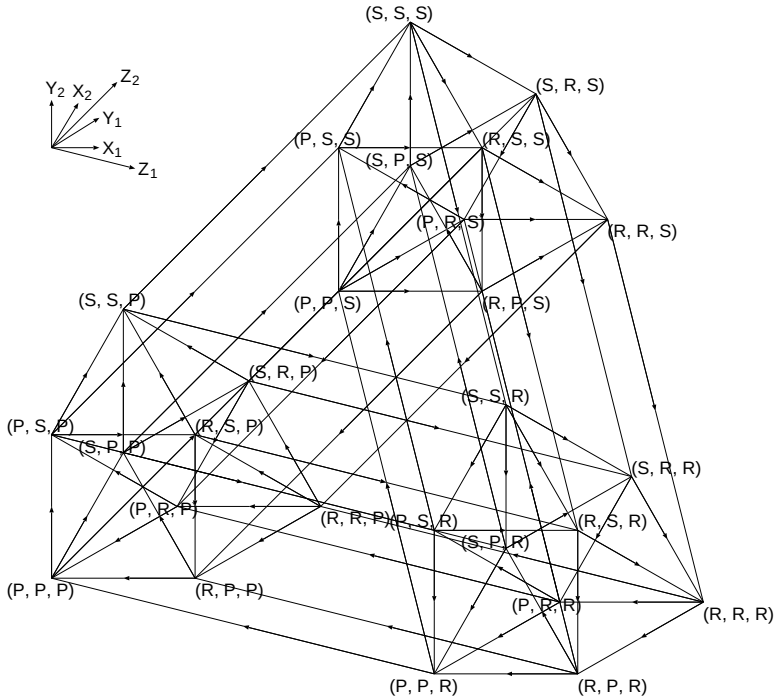

(S,S)
(S,R)
(P,S)
(S,P)
(R,S)
(P,R)
(R,R)
1.0
x_2
x_3
y_3
y_2
y_1
(P,P)
0.0
x_1
1.0
0.0
(R,P)

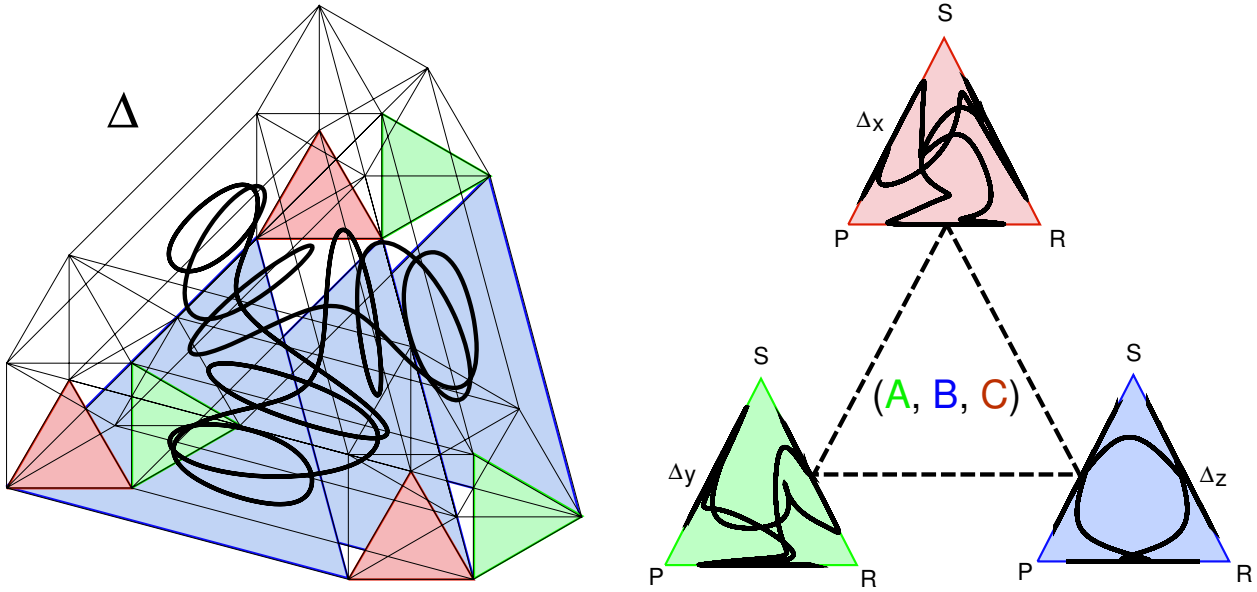

(S,S)
Δ
(S,R)
(P,S)
(R,S)
(S,P)
(P,R)
(R,R)
(P,P)
(R,P)


S
Δ_X
P
R
S
Δ_Y
P
R


(S,S)
Δ
(S,R)
(P,S)
(R,S)
(S,P)
(P,R)
(R,R)
(P,P)
(R,P)
S
Δ_X
P
R
S
Δ_Y
P
R

Three agents interacting via generalized RSP




Δ
S
Δx
P
R
S
(A, B, C)
S
Δy
Δz
P
R
P
R

Structured Multiagent Systems

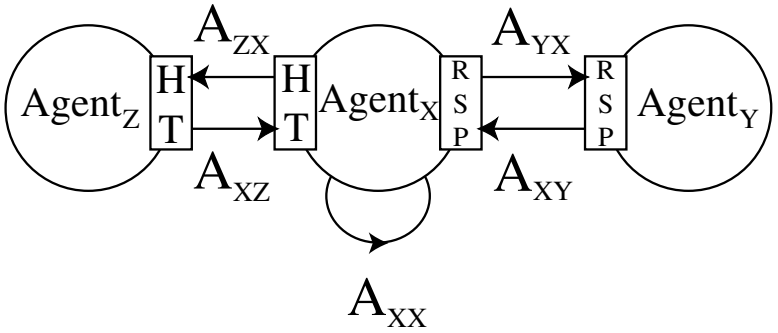
Problem: Model requires all agents communicate to all

- For large collectives, not realistic;
- |Interactions| $\propto exp(S)$; and
- Agent's interactions with others always of same type.

Solution: Extend models to include

- Role Playing and
- Communication Networks.

Example of Role Playing and Communication Networks

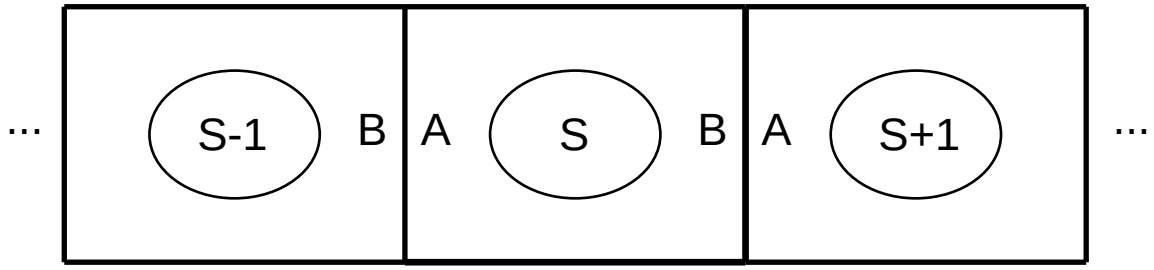


$$
\begin{aligned}
\frac{\dot{x}_i}{x_i} &= \beta_X \left[(A^{XX}\mathbf{x})_i - \mathbf{x} \cdot A^{XX}\mathbf{x} + (A^{XZ}\mathbf{z})_i - \mathbf{p} \cdot A^{XZ}\mathbf{z} + (A^{XY}\mathbf{y})_i - \mathbf{q} \cdot A^{XY}\mathbf{y} \right] \\
&+ \alpha_X (H(x_i) - H[\mathbf{x}]) \\
\frac{\dot{y}_i}{y_i} &= \beta_Y \left[(A^{YX}\mathbf{x})_i - \mathbf{y} \cdot A^{YX}\mathbf{x} \right] + \alpha_Y (H(y_i) - H[\mathbf{y}]) \\
\frac{\dot{z}_i}{z_i} &= \beta_Z \left[(A^{ZX}\mathbf{x})_i - \mathbf{y} \cdot A^{ZX}\mathbf{x} \right] + \alpha_Z (H(z_i) - H[\mathbf{z}])
\end{aligned}
$$

where

- $\Sigma x_i = \Sigma y_i = \Sigma z_i = 1$,
- $\mathbf{x} = (x_{\mathsf{HR}}, x_{\mathsf{HS}}, x_{\mathsf{HP}}, x_{\mathsf{TR}}, x_{\mathsf{TS}}, x_{\mathsf{TP}})$, $\mathbf{y} = (y_{\mathsf{R}}, y_{\mathsf{S}}, y_{\mathsf{P}})$, and $\mathbf{z} = (z_{\mathsf{H}}, z_{\mathsf{T}})$,
- $\mathbf{p} = (x_{\mathsf{HR}} + x_{\mathsf{HS}} + x_{\mathsf{HP}}, x_{\mathsf{TR}} + x_{\mathsf{TS}} + x_{\mathsf{TP}})$,
- $\mathbf{q} = (x_{\mathsf{HR}} + x_{\mathsf{TR}}, x_{\mathsf{HS}} + x_{\mathsf{TS}}, x_{\mathsf{HP}} + x_{\mathsf{TP}})$.

Example of Spatially Distributed Collective



S agents on a one-dimensional lattice:

$$\frac{\dot{x_i^s}}{x_i^s} = \beta_s \left[(A^s \mathbf{x}^{s-1})_i - \mathbf{p}^s \cdot A^s \mathbf{x}^{s-1} + (B^s \mathbf{x}^{s+1})_i - \mathbf{q}^s \cdot B^s \mathbf{x}^{s+1} \right] + \alpha_s (-\log x_i^s - \Sigma_n^4 x_n^s \log x_n^s)$$

where

- $\Sigma x_i^s = 1$,
- $\mathbf{p}^s = (x_1^s + x_2^s, x_3^s + x_4^s)$, and
- $\mathbf{q}^s = (x_1^s + x_3^s, x_2^s + x_4^s)$.

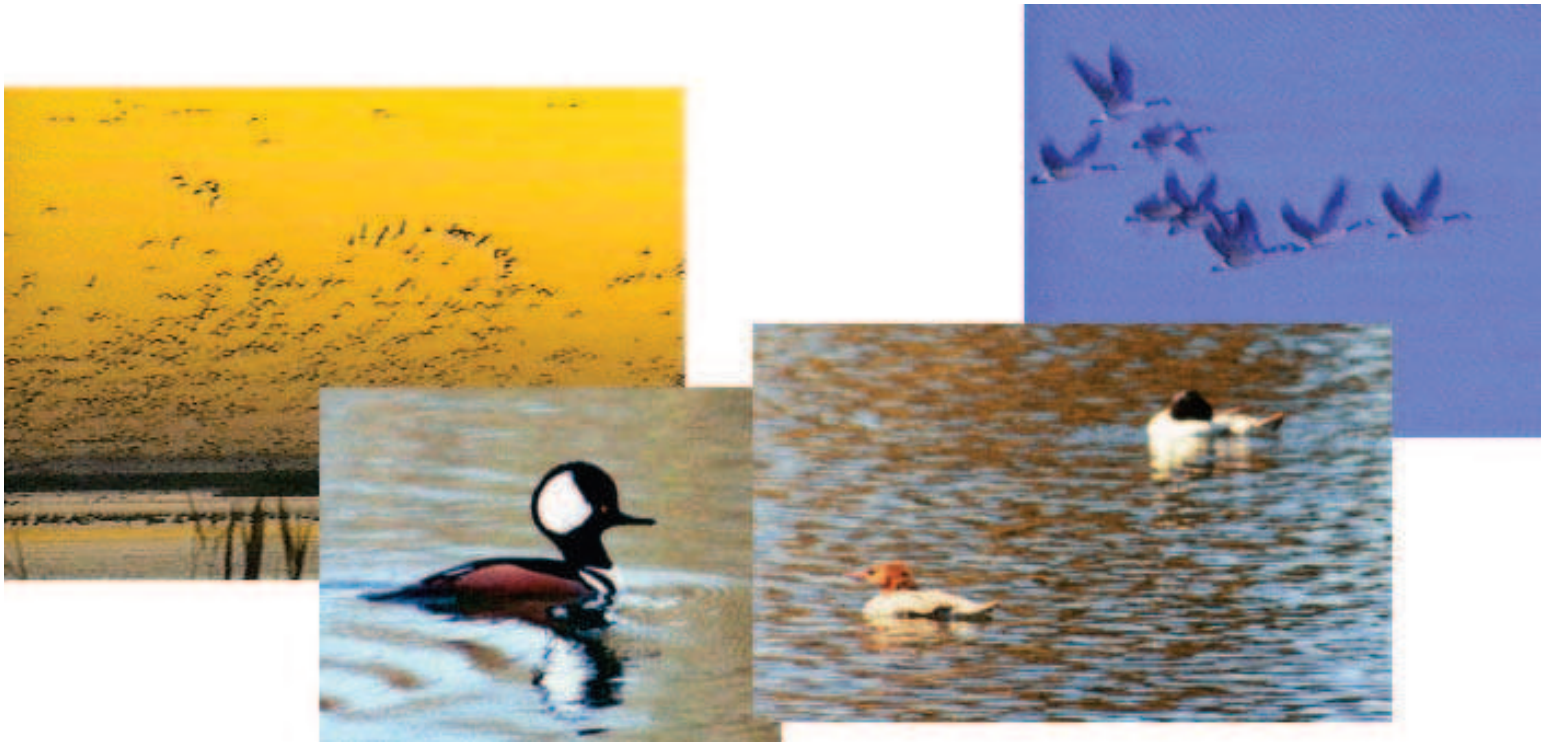
Multiple Agents Servicing Multiple Tasks (MASMT)

N-agent collective responsible for servicing M distributed tasks:

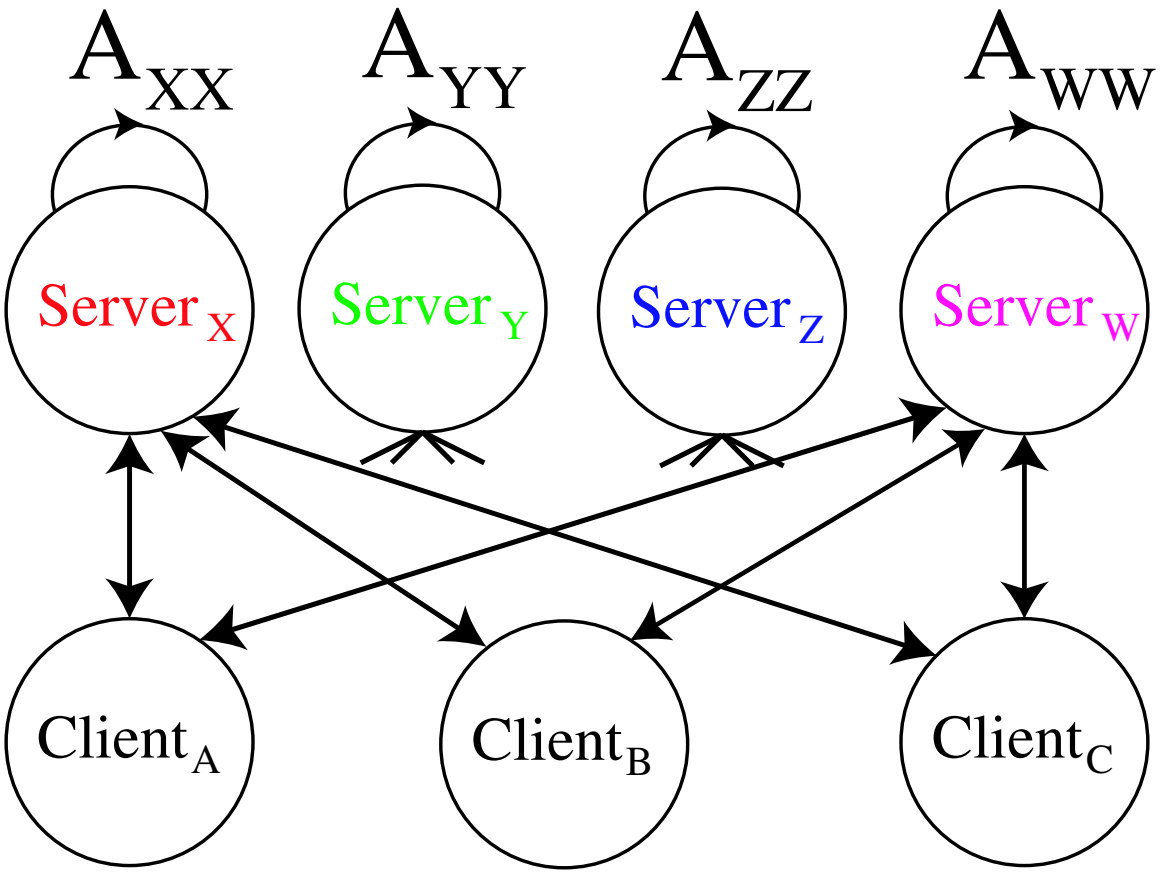
- Each task has a desired *service rate* σ_i that agents attempt to achieve.
- If a task is serviced too often, the servicing agent is punished, as wasting resources.
- If an agent services a task that is being serviced at a rate below σ_i, however, the agent is rewarded, as the agent is servicing a pending task that other agents failed to address.
- In this case, in addition, other agents are punished in proportion to their failure in allowing the task to go unserviced.
- If the task is serviced at the correct rate, the agent is neither rewarded nor punished.

Wetlands

A wetlands supporting a population of ducks with ponds that provide distinct nutrients. The birds are the agents, each with a set of nutrient requirements characteristic of the species—such as water, food, minerals, and so on. Tasks are identified as a combination of the birds' needs coupled with what the ponds provide. The timeliness of servicing the tasks corresponds to the birds using up their stored nutrients and needing to replenish them.



Example Architecture



Information Space

Self-informations of agents X and Y choosing actions i and j:

$$\begin{aligned} \xi_i &= -\log x_i \ , \\ \eta_j &= -\log y_j \ , \end{aligned}$$

Exploitation and exploration terms are differences from means: so we use

$$\begin{aligned} u_i &= \xi_i - N^{-1} \sum_{k=1}^{N} \xi_k \ , \\ v_j &= \eta_j - M^{-1} \sum_{k=1}^{M} \eta_k \ , \end{aligned}$$

with $\sum_{k=1}^{N} u_k = \sum_{k=1}^{M} v_k = 0$.

MADS equations simplify greatly

$$\begin{aligned} \dot{\mathbf{u}} &= -\beta_X A\mathbf{y} - \alpha_X \mathbf{u} \ , \\ \dot{\mathbf{v}} &= -\beta_Y B\mathbf{x} - \alpha_Y \mathbf{v} \ . \end{aligned}$$

with normalized interactions, $x_i = e^{-u_i}/\Sigma_k^N e^{-u_k}$, and $y_i = e^{-v_i}/\Sigma_k^M e^{-v_k}$.

Discrete-Time Adaptive Maps

$$x_i(t+1) = \frac{x_i^{1-\alpha_X}(t) e^{\beta_X R_i^X(t)}}{\sum_k x_k^{1-\alpha_X}(t) e^{\beta_X R_k^X(t)}}$$

$$y_i(t+1) = \frac{y_i^{1-\alpha_Y}(t) e^{\beta_Y R_i^Y(t)}}{\sum_k y_k^{1-\alpha_Y}(t) e^{\beta_Y R_k^Y(t)}}$$

The Future

1. Natural extensions:

   (a) Communication costs
   (b) Continuous actions (e.g., geographic position): PDEs
   (c) State-dependent agents (with memory)

2. Design:

   (a) Evolutionary search over space of reinforcement schemes
   (b) Optimize performance
   (c) Robustness