Learning In Distributed Systems: A Game Theoretic View

Malik Magdon-Ismail
Rensselaer Polytechnic Institute

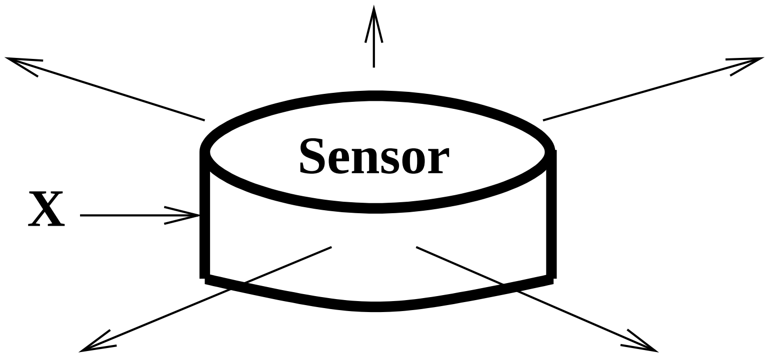
November 9, 2004.

Example

Sensor Network Admission Control

(Joint work with Carothers, Yener)

Sensors – A Simple Model



Sensor Capabilities:

- Can **sense** some particular event X and **transmit**.
- Can **receive** a neighbor transmission and **re-transmit**.
- No buffer to store events.
- Small CPU, small RAM (can store and execute a small program).
- Has access to a random number generator.
- Low power FM antenna to receive (centralized) information.

A Simple Learning Task

A sensor senses an event and/or receives a transmission.

$\boxed{\text{transmit sensed event } (\textbf{local job})}$

XOR

$\boxed{\text{transmit received event } (\textbf{remote job})}$

p_{route} = probability to process remote job

If no remote job exists, only option is local job.

Feedback

Events have a value $\boldsymbol{X}$ (**cookie**).

Individual sensors have "bank accounts" (**bank**).

$\boxed{\text{ROUTE (remote job): bank } \epsilon.}$

(**immediate reward**)

$\boxed{\text{SENSE (local job): bank } \beta X.}$

(**potential delayed reward**)

– Delayed reward only if job arrives at destination.
– Bank is communicated by central authority.
– Sensors occasionally check their "bank accounts"

A Simple Learning Algorithm

Learning Task (for each sensor): Determine p_{route}.

Gradient Based algorithm:

```
Initialize: p_route(0);
Obtain change in bank after τ periods: ΔB(0);
while Alive do
    Perturb: p_route(t+1) ← p_route(t) + Δp;
    Obtain change in bank after τ periods: ΔB(t+1);
    if ΔB(t+1) > ΔB(0) then
        Keep search direction;
    else
        Change search direction: Δp ← −Δp;
    end if
end while
```

A Simple Learning Algorithm

Learning Task (for each sensor): Determine p_{route}.

Gradient Based algorithm:

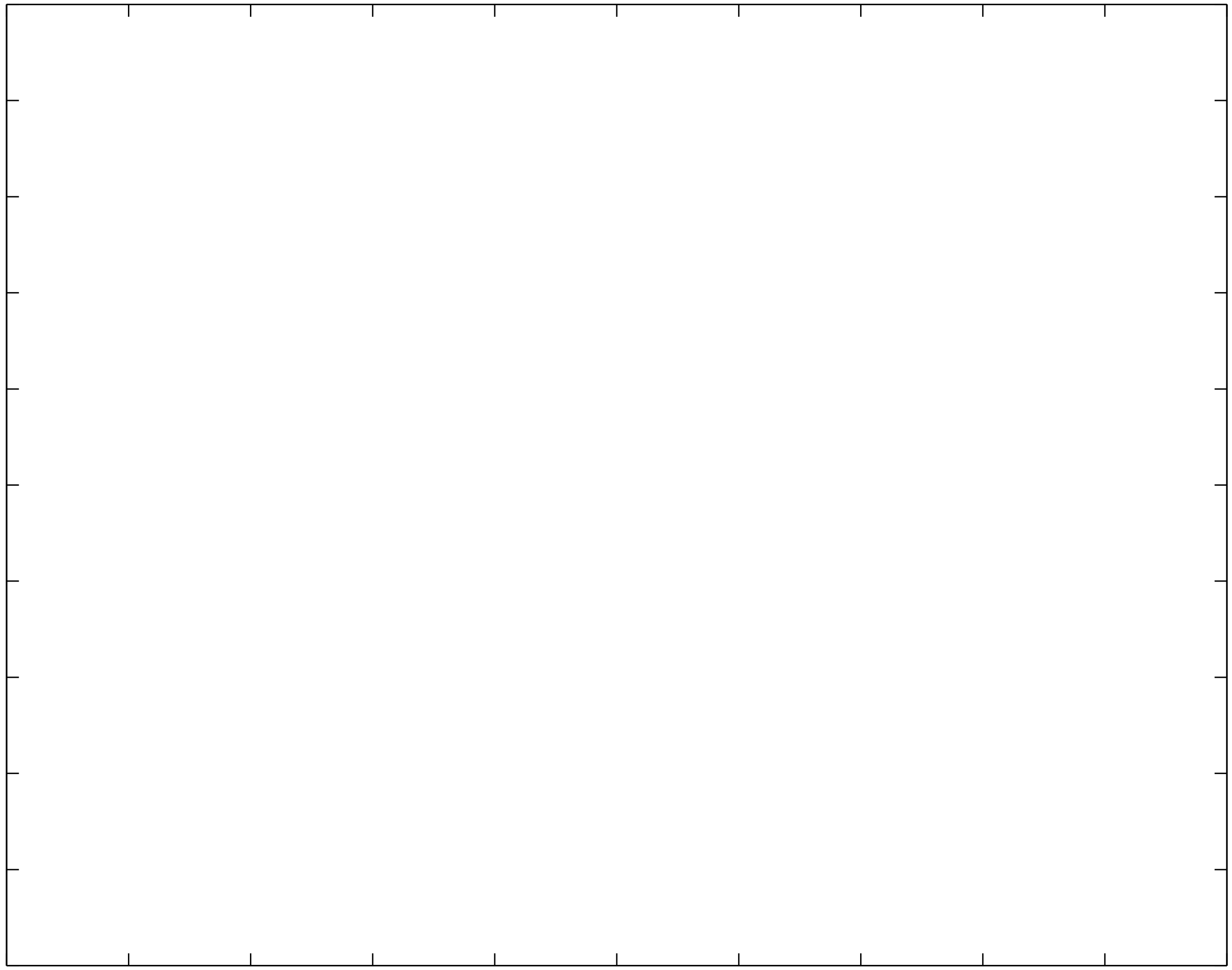
Want to learn p_{route}.
Start somewhere (say) $p_{route} = 0.5$.
How good is p_{route}?
Change p_{route} a little.
Is it better?
if yes **then**
    Keep going.
else
    Reverse direction.
end if
Repeat until "convergence".

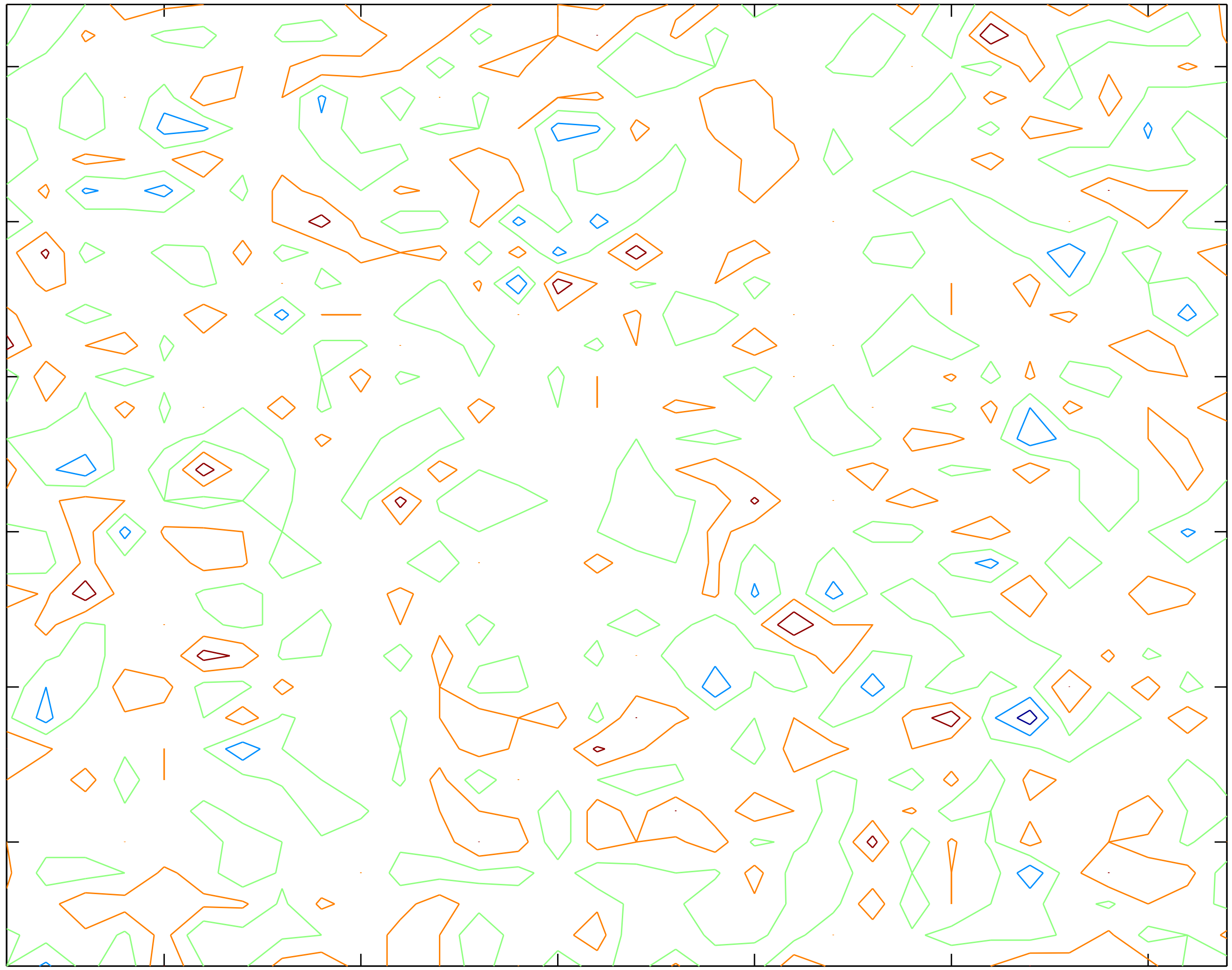
A Simulation

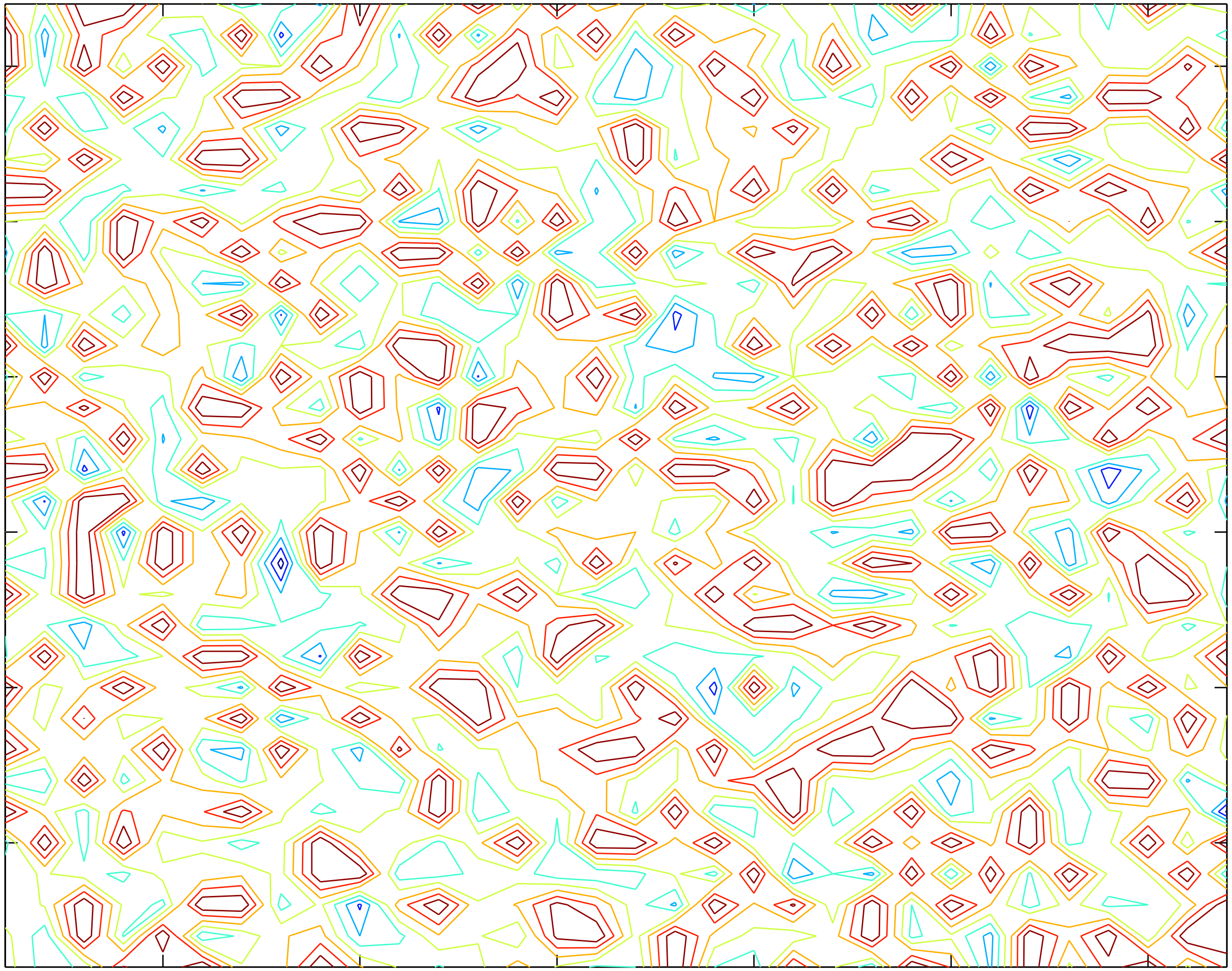
– Each sensor has a continuous stream of sensing events.

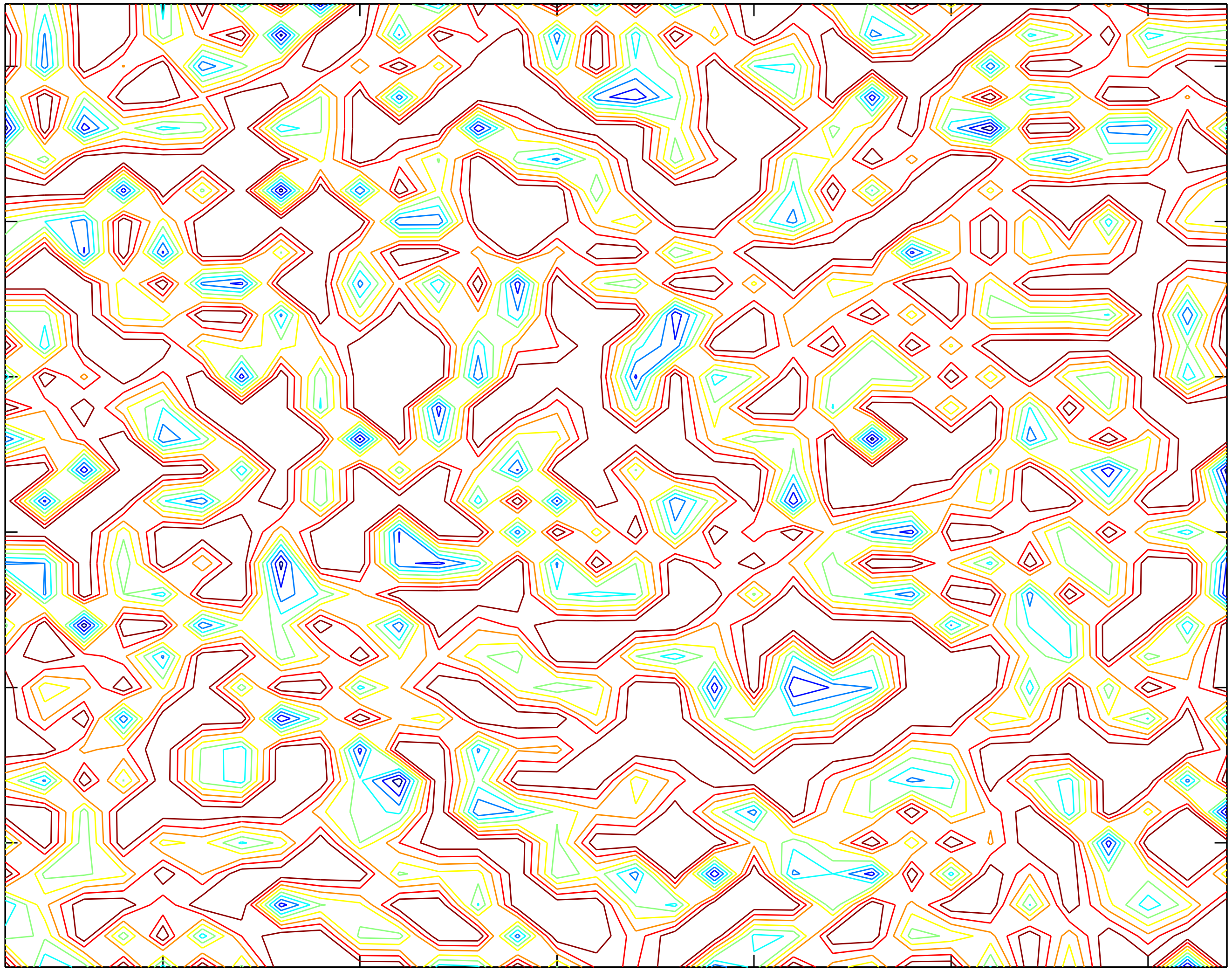
– Cookies drawn from a stationary distribution.

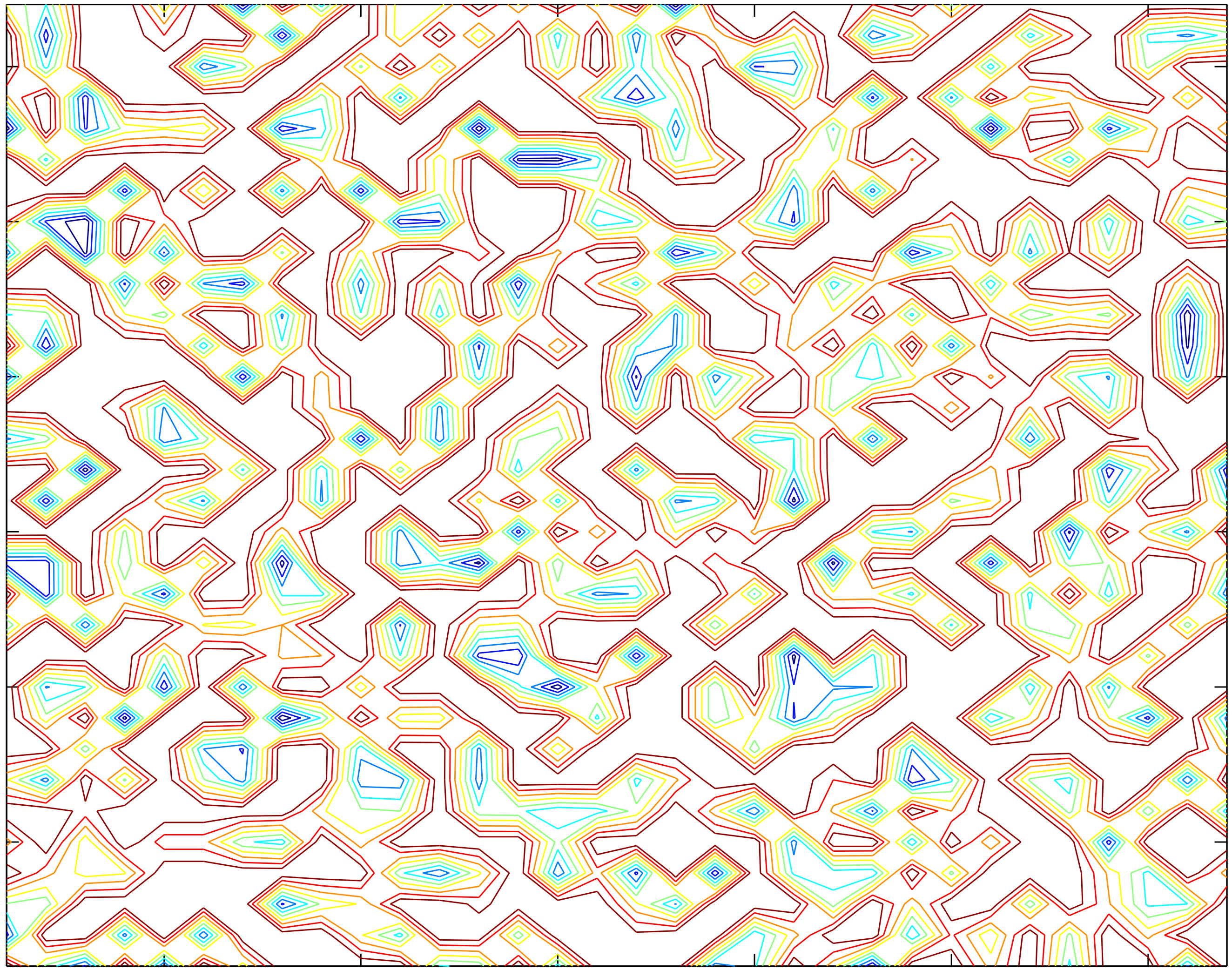
– ϵ, β fixed.

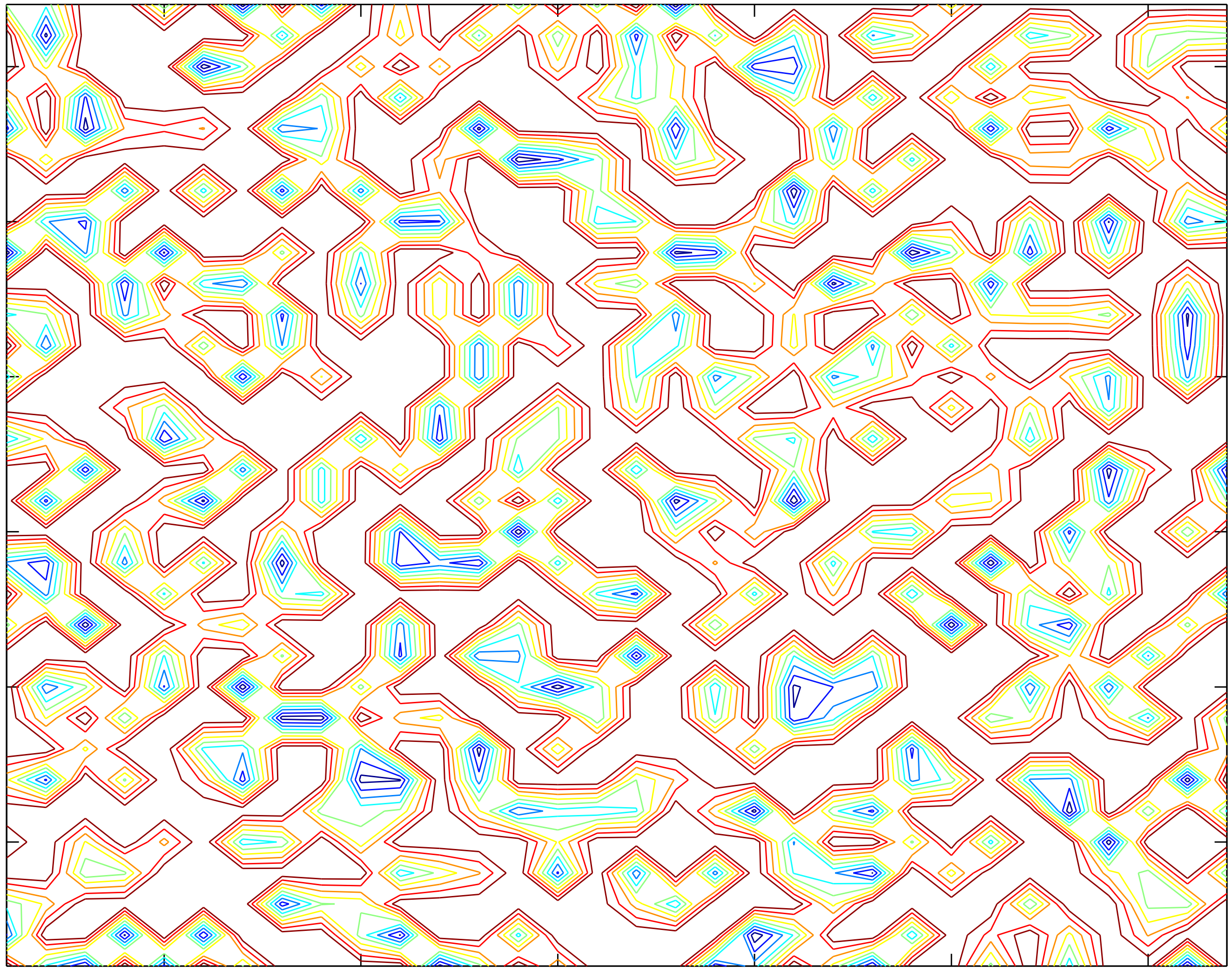


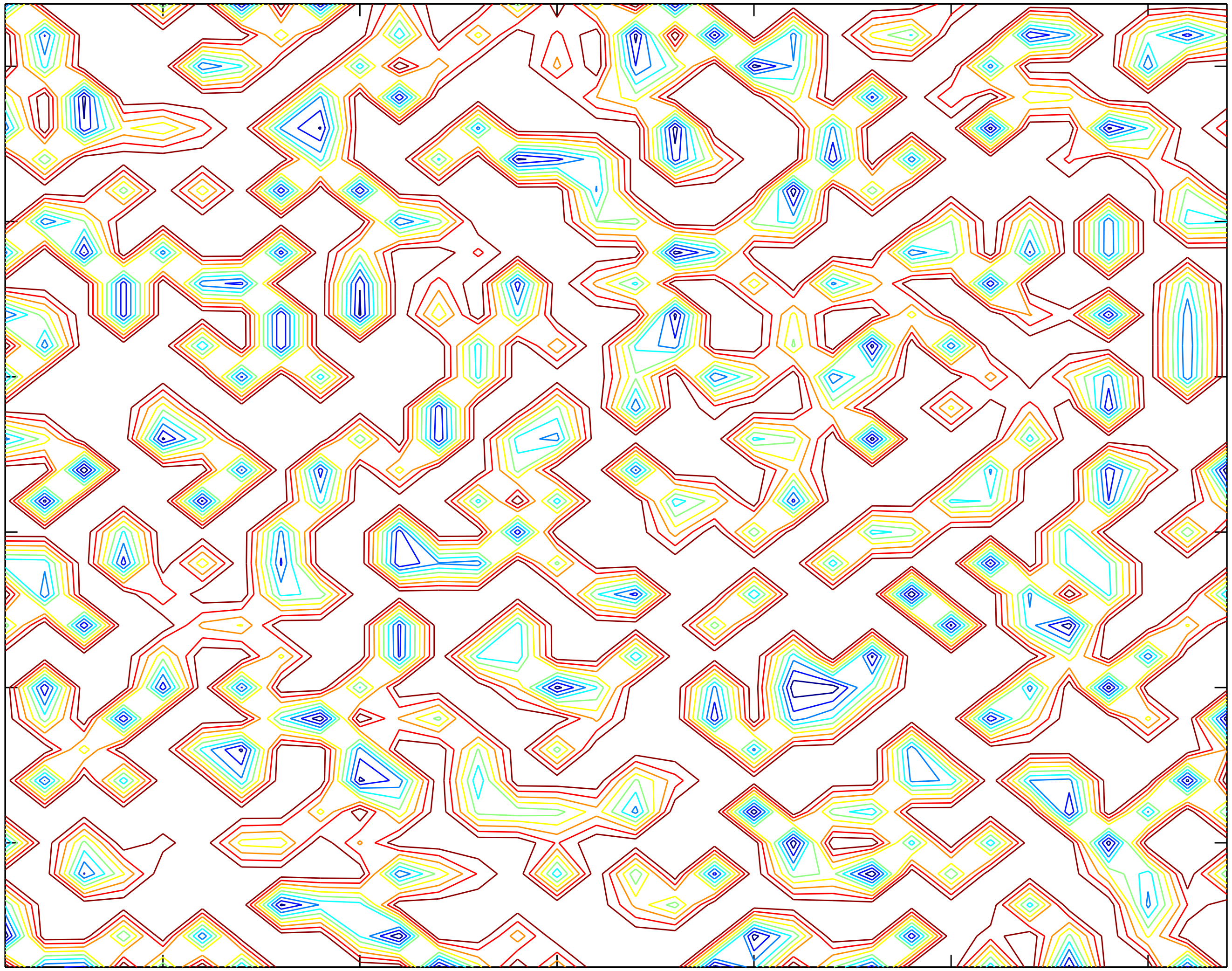












Convergence?

Given: $\boldsymbol{\epsilon}, \boldsymbol{\beta}$ and traffic (cookie) distribution ($\boldsymbol{X}$),

– convergence to a "self organized" **steady state**.

$\boxed{\textbf{What are the properties of the steady state?}}$

Reinforcement Learning?

Similar to setup of Reinforcement Learning:

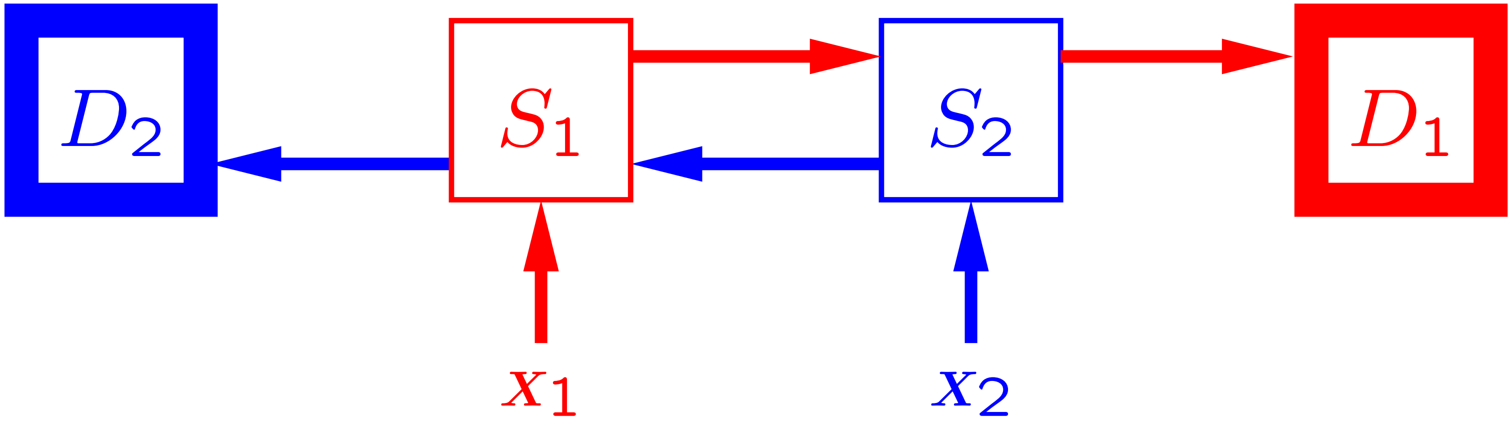
– There is an environment $(\boldsymbol{\epsilon}, \boldsymbol{\beta}, \boldsymbol{X})$.
– Node performs action (picks a particular p_{route}).
– Node receives reward (increase in bank account).

Except . . .

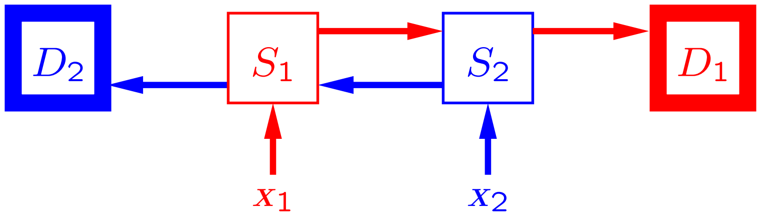
. . . there are other “reinforcement learners” that affect your reward.

– Adversarial.

Two Node System



Deterministic Actions



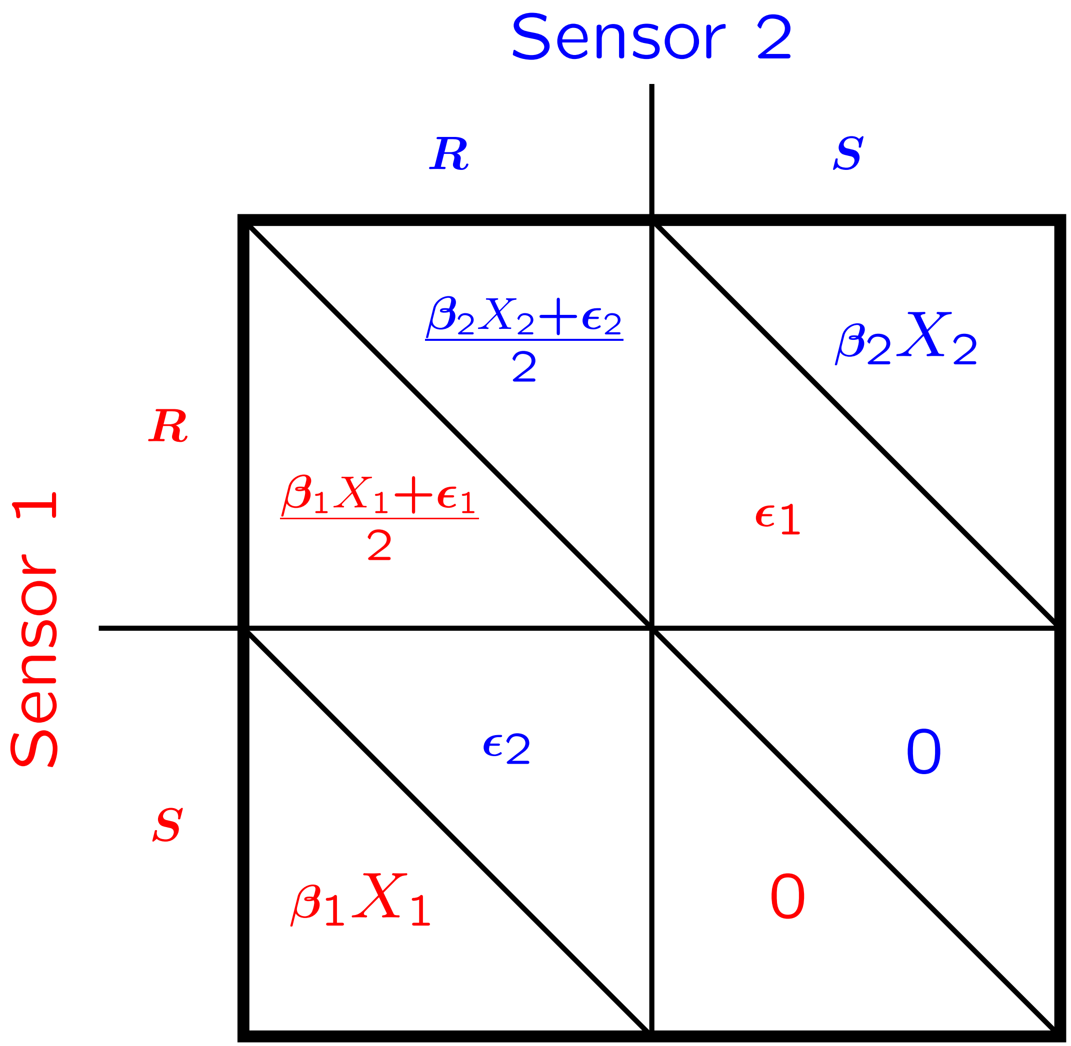
S_1: route $\boldsymbol{R}$ or sense $\boldsymbol{S}$.

S_2: route $\boldsymbol{R}$ or sense $\boldsymbol{S}$.

$\boxed{\text{4 state system}}$

$(\boldsymbol{R}, \boldsymbol{R})$, $(\boldsymbol{R}, \boldsymbol{S})$, $(\boldsymbol{S}, \boldsymbol{R})$, $(\boldsymbol{S}, \boldsymbol{S})$.

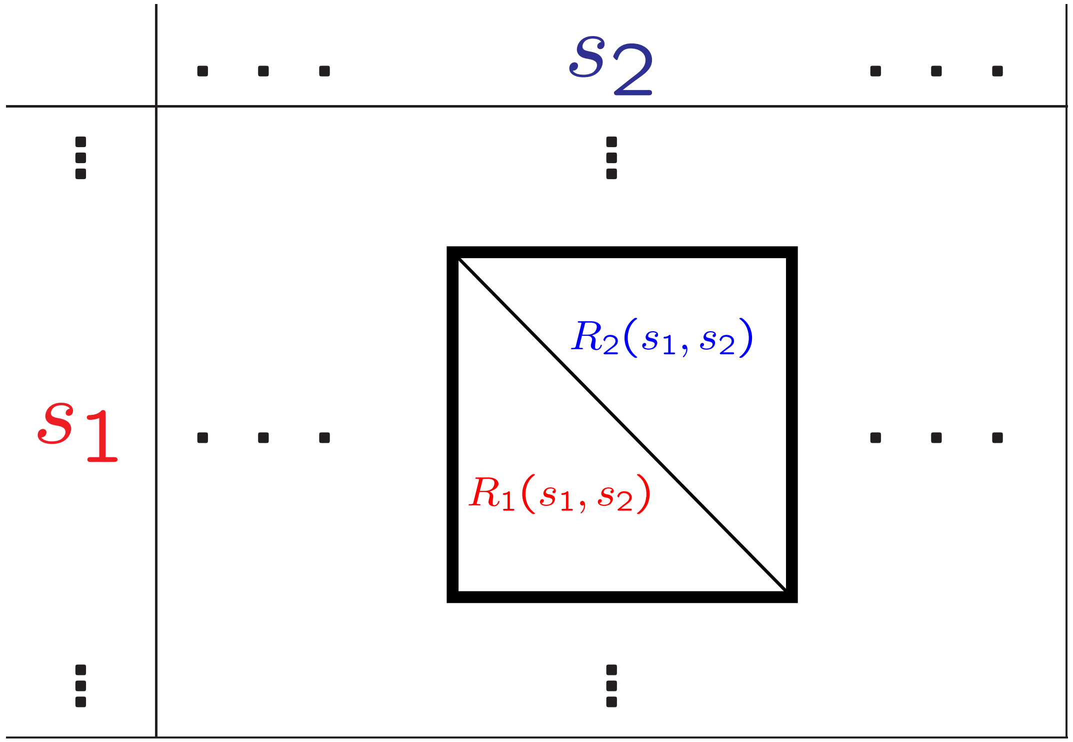
Payoffs



Sensor 2
R
S
Sensor 1
R
S
$\frac{\beta_2 X_2 + \epsilon_2}{2}$
$\frac{\beta_1 X_1 + \epsilon_1}{2}$
$\beta_2 X_2$
ϵ_1
ϵ_2
$\beta_1 X_1$
0
0

Non-Cooperative Games

Two Player Matrix Games



$R_1(s_1, s_2)$ is reward to player $\mathbf{1}$ when (s_1, s_2) is played.
$R_2(s_1, s_2)$ is reward to player $\mathbf{2}$ when (s_1, s_2) is played.

Nash Equilibrium

(s_1, s_2) is a **Nash Equilibrium (NE)** if
neither player wishes to switch strategies:

$R_1(s_1, s_2) \geq R_1(\boldsymbol{u}, s_2)$ for all other possible plays $\boldsymbol{u}$ of player 1.
$R_2(s_1, s_2) \geq R_1(s_1, \boldsymbol{v})$ for all other possible plays $\boldsymbol{v}$ of player 1.

Some Two Person Games

Prisoners Dilemma

	Quiet	Rat
Quiet	(-3,-3)	(-10,-1)
Rat	(-1,-10)	$\boxed{(-7,-7)}$

NE: (Rat,Rat)

Matching Pennies

	Heads	Tails
Heads	(1,-1)	(-1,1)
Tails	(-1,1)	(1,-1)

NE: None

If probabilistic choices are allowed (**mixed strategies**), there is always a Nash Equilibrium [Nash, 1951].

Repeated Games (Non-Perfect Information)

Players play multiple times (not necessarily against the same players).

Varying degrees of information (feedback):

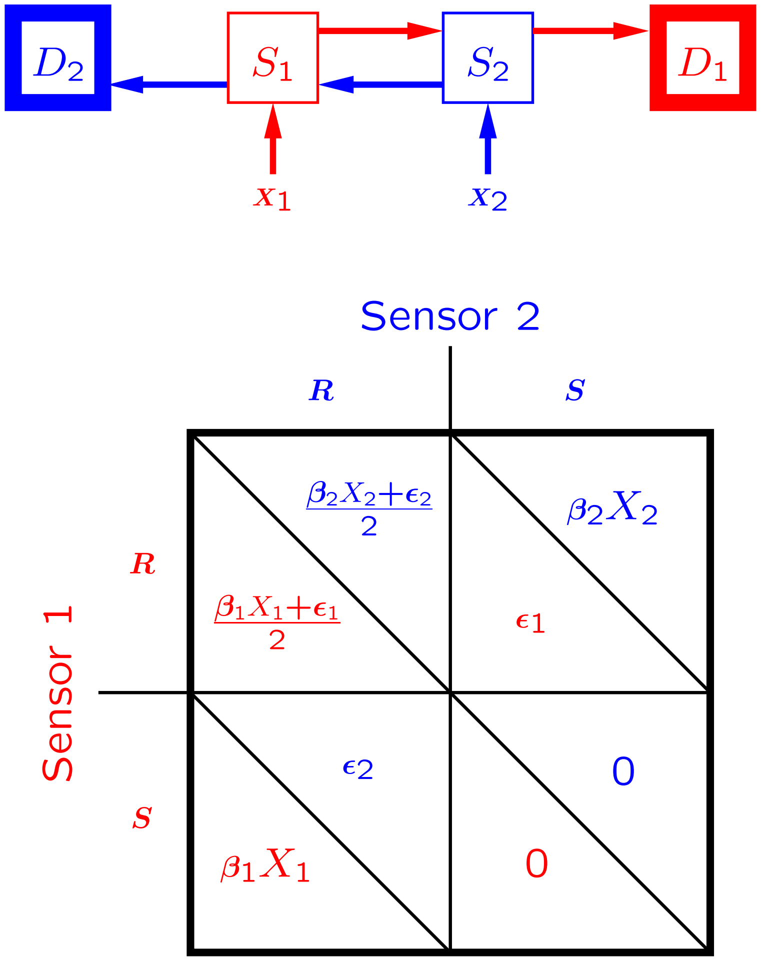
Know the strategy spaces;

Experience only reward;

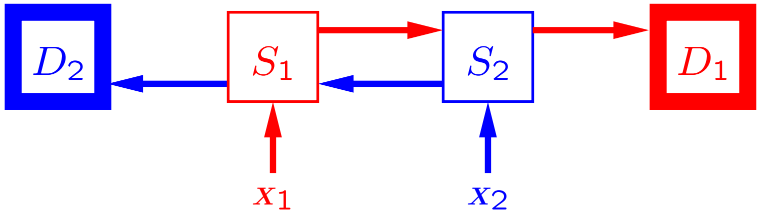
. . .

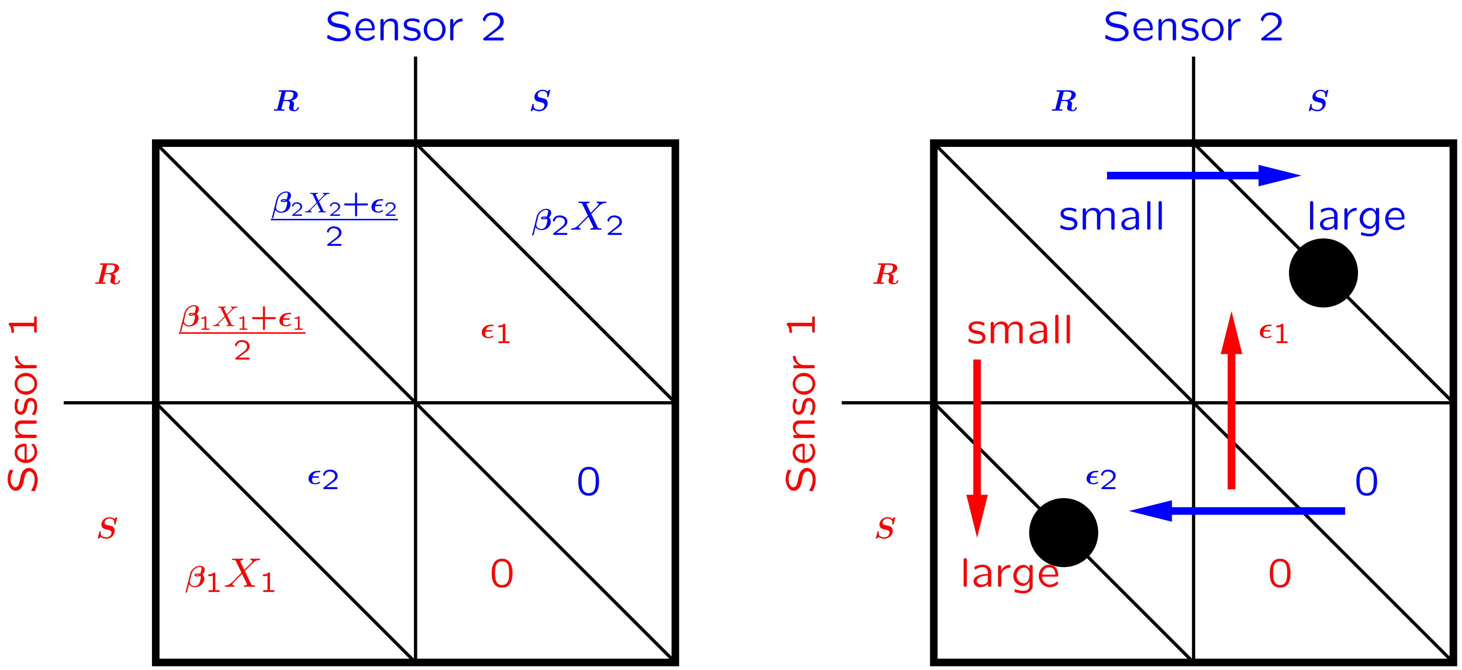
Convergence: for a large class of repeated games, and imperfect information settings, gradient based (**learning**) methods converge to a Nash Equilibrium [Arslan & Shamma, 2003,2004].

Two Node Sensor Game



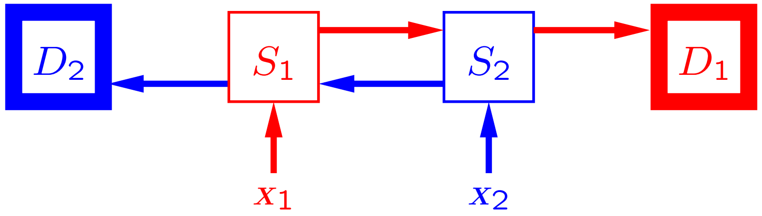
Large Sensing Reward

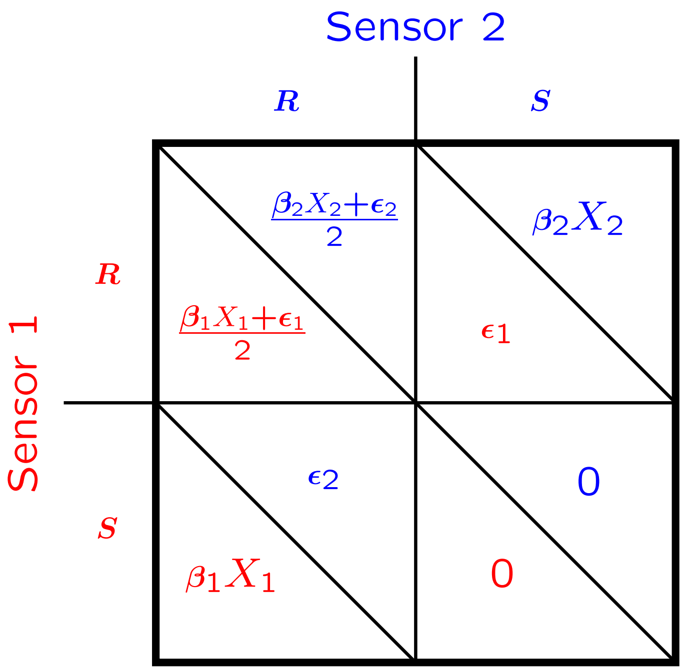


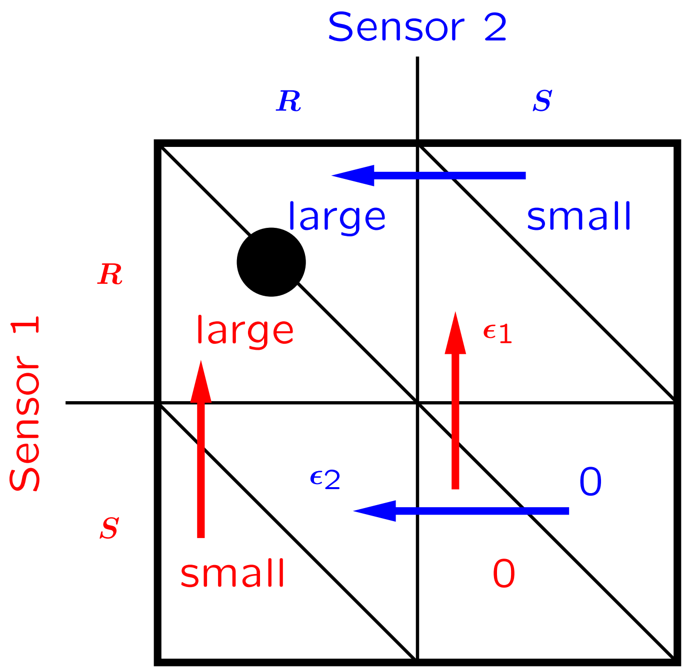


Nash Equilibria: $(\boldsymbol{S},\boldsymbol{R})$, $(\boldsymbol{R},\boldsymbol{S})$

Large Routing Reward

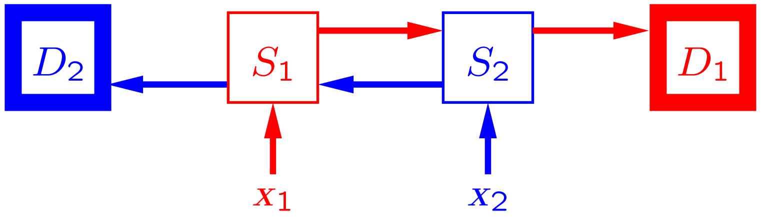


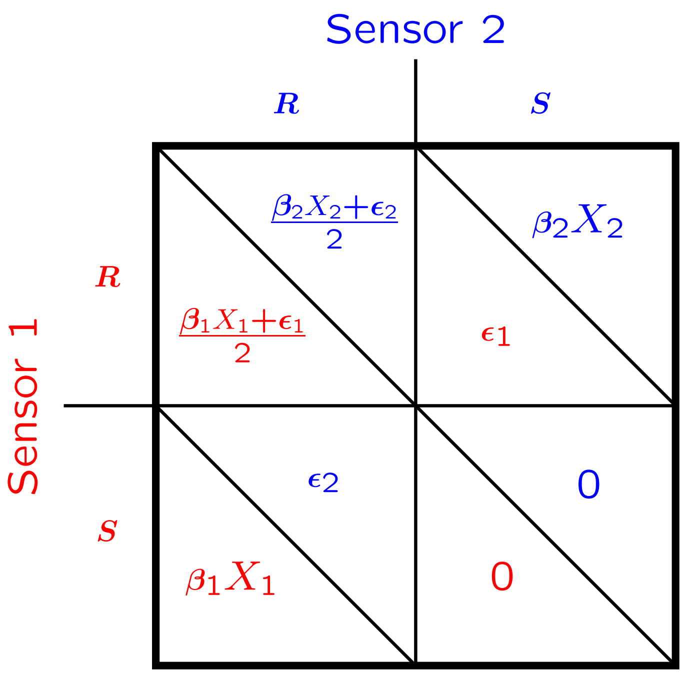


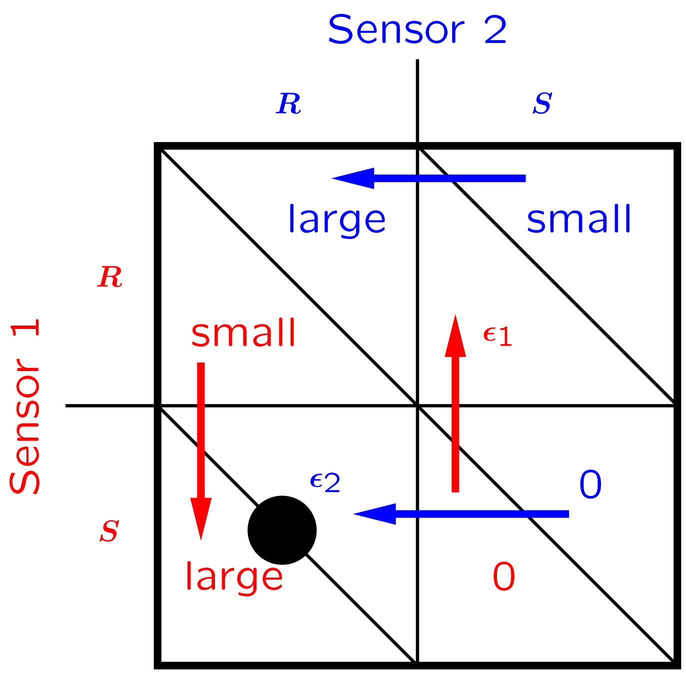


Nash Equilibria: (**R**,**R**)

Mixed Reward







Nash Equilibria: (S, R)

Summary of NE

OSAMA vs. osama

One will make it through.

Zebra vs. Zebra

Both zebra's make it through half time.

Osama vs. Zebra

Osama makes it through.

Why NE

System settles in a **NE**.

NE is a function of control parameters.

Control problem:
– Have some global criterion to optimize.
– Selfish learners reach some steady state.
– How bad is this steady state?

The Price of Anarchy

[Koutsoupias & Papadimitriou, 1999]

Anarchy: A bunch of selfish "individuals".

Price of Anarchy: How bad is where the selfish individuals settle compared with the optimal for the "society".

$$\boxed{\text{Price of Anarchy} = \frac{\text{"Cost" of } \textbf{worst NE}}{\text{"Cost" of optimal}}}$$

A Reasonable Cost Function

$$1 - Pr(X_1)Pr(X_2)$$

Theorem. Price of Anarchy $\leq \frac{4}{3}$.

Learning in **this distributed system** is not too bad, with respect to **this cost function**.

Wrapping Up

– Many situations where learning in distributed systems occurs:
  Learning gives adaptivity.

– What happens in such systems: **Nash Equilibria**
  Taylor the individual objective so that desired **NE** are “good”.
  Control parameters can also affect **NE**.

– Learning can be bad.
  There can be bad **NE**: that’s the **price of anarchy**.

Some Context

[Wardrop, Beckman, 1950's], Nash Equilibria in selfish traffic flow.

[Korilis & Lazar]
[Nisan & Ronen]
[Koutsoupias,Papadimitriou]
[Roughgarden & Tardos]
[Shenker]
⋮

, Internet protocols
(selfish routing of flows)

[Kearns, et al.], Finding (approx) Nash equilibria in graphical games.

[Bush & Magdon-Ismail], Network games for load distribution (in progress).
[Magdon-Ismail et al.], Sensor network admission control (in progress).

Thank You!