Games in Networks: the price of anarchy, stability and learning

Éva Tardos
Cornell University

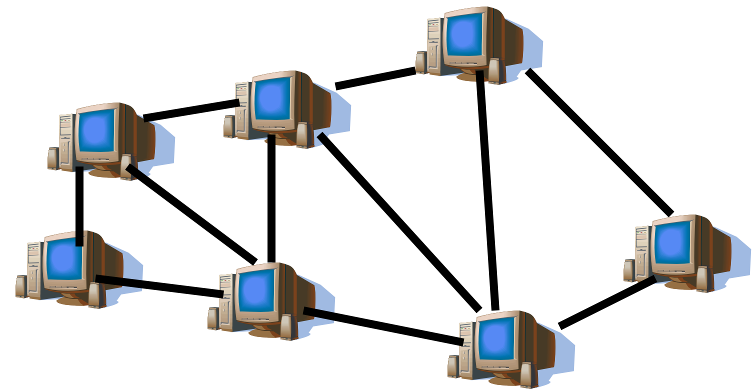
Why care about Games?

Users with a multitude of diverse economic interests sharing a Network (Internet)

- browsers
- routers
- servers

Selfishness:

Parties deviate from their protocol if it is in their interest



Model Resulting Issues as

Games on Networks

Main question:
Quality of Selfish outcome

Well known: Central design can lead to better outcome than selfishness.

e.g.: Prisoner Dilemma

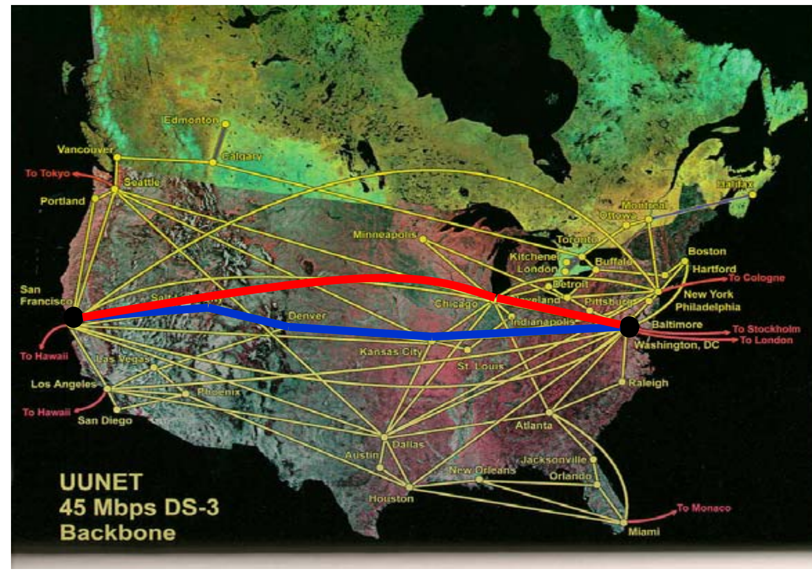
	C	D
C	2 / 2	1 / 99
D	99 / 1	98 / 98

Question: how much better?

Our Games

- Routing and Network formation: Users select paths that connects their terminals to minimize their own delay or cost

Example: Routing Game



- Traffic subject to congestion delays
- cars and packets follow shortest path

Congestion games: cost depends on congestion

includes many other games

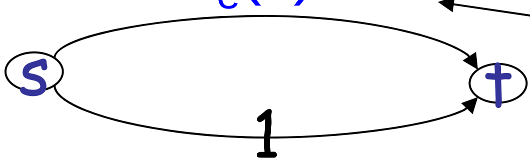
Computer Science Games

- Routing:
- routers choose path for packets though the Internet
- Bandwidth Sharing:
- routers share limited bandwidth between processes
- Facility Location:
- Decide where to host certain Web applications
- Load Balancing
- Balancing load on servers (e.g. Web servers)
- Network Design:
- Independent service providers building the Internet

Congestion sensitive load balancing

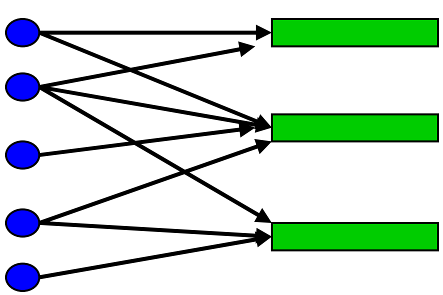
Routing network:

$\ell_e(x) = x$



Load balancing:

jobs



$\ell_e(x) = x$

machines

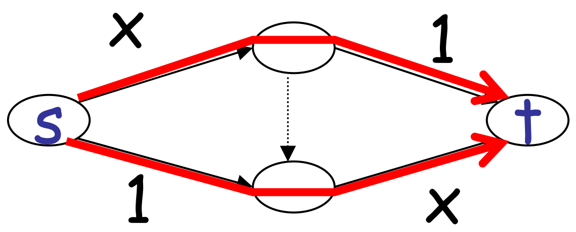
Cost/Delay/Response time as a fn of load:

x unit of load → causes delay $\ell_e(x)$

A congestion game

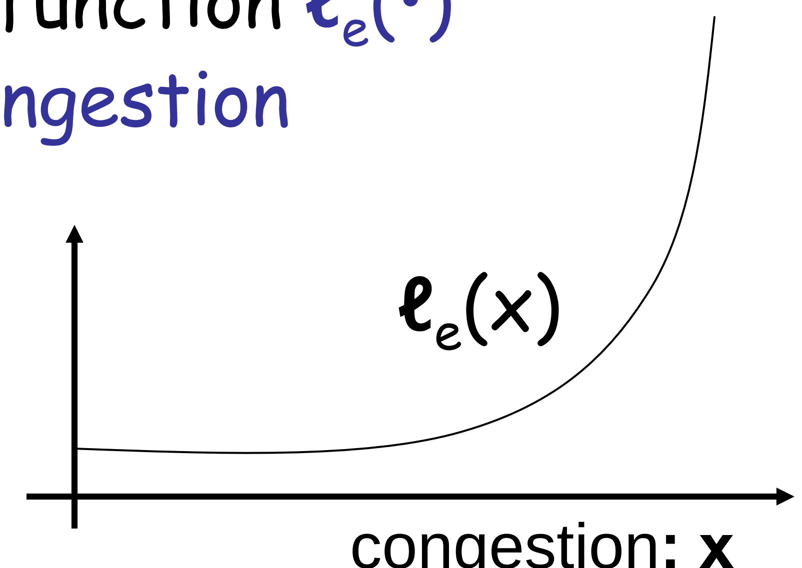
Model of Routing Game

- A directed graph $G = (V,E)$
- source-sink pairs s_i,t_i for $i=1,..,k$
- User i selects path P_i for traffic between s_i and t_i for each $i=1,..,k$



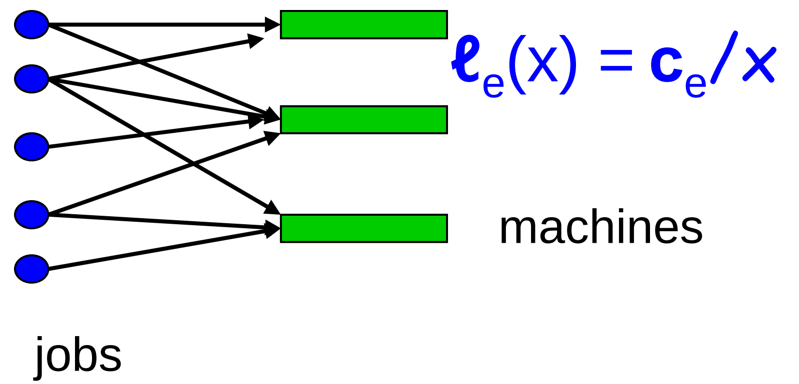
For each edge e a latency function $\ell_e(\cdot)$

Latency increasing with congestion



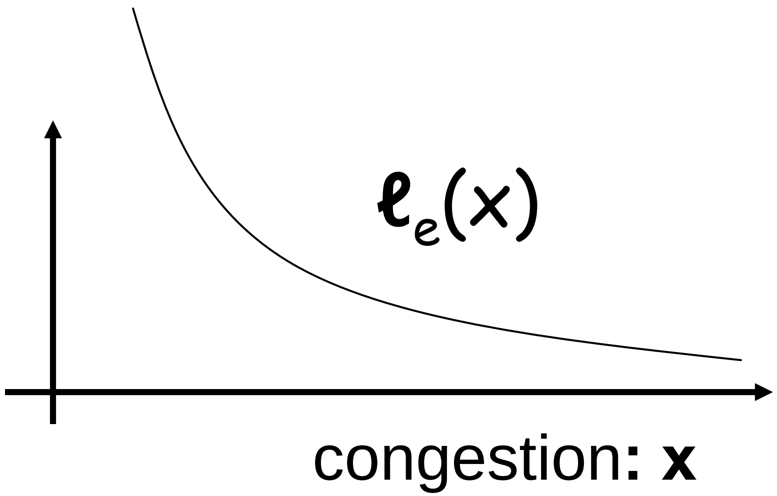
Cost-sharing: a Coordination Game

- jobs $i=1,..,k$
- For each machine e a cost function $\ell_e(\cdot)$
  - E.g. cloud computing



- Cost decreasing with congestion (decreasing marginal cost)

$\ell_e(x)= c_e/x$



Goal's of the Game

Personal objective: minimize

$\ell_P(\mathbf{x})$ = sum of latencies or costs of edges along the chosen path **P** (with respect to flow **x**)

Overall objective:

$\mathbf{C}(\mathbf{x})$ = total latency/cost of a flow **x**: $= \Sigma_P \, x_P \cdot \ell_P(\mathbf{x})$

delay summed over all paths used, where x_P is the amount of flow carried by path **P**.

What is Selfish Outcome (1)?

Traditionally: Nash equilibrium

- Current strategy "best response" for all players (no incentive to deviate)

Theorem [Nash 1952]:

- Always exists if we allow randomized strategies

Price of Anarchy:

$$\frac{\text{cost of worst (pure) Nash}}{\text{“socially optimum” cost}}$$

Price of Stability: worst $\rightarrow$ best

Selfish Outcome (2)?

- Does natural behavior lead no Nash?
- Which Nash?
- Finding Nash is hard in many games…
- What is natural behavior?
  - Best response?
  - learning?

Games with good Price of Anarchy/Stability

- Routing and load balancing: routers choose path [Koutsoupias-Papadimitriou '99], [Roughgarden-Tardos 02] , etc
- Network Design:

  [Fabrikant et al'03], [Anshelevich et al'04], etc
- Facility location Game

Placing servers (e.g. Web) to extract income

  [Vetta '02] and [Devanur-Garg-Khandekar-Pandit-Saberi-Vazirani'04]
- Bandwidth Sharing:

routers decide how to share limited bandwidth between many processes [Kelly'97, Johari-Tsitsiklis 04]

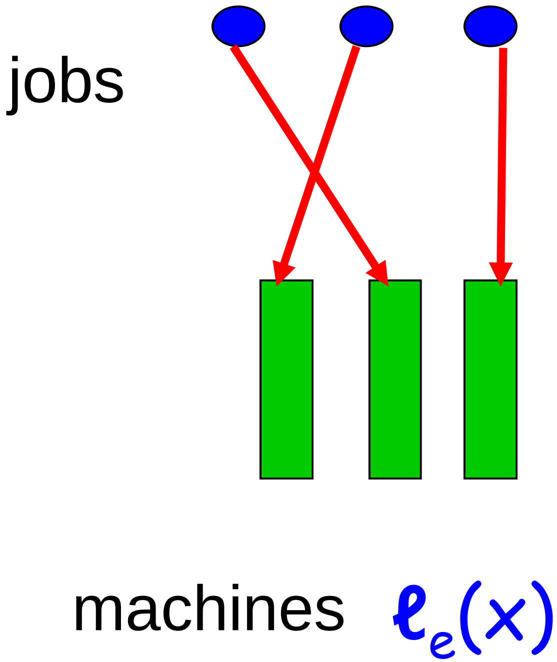
Example: Atomic Game (pure Nash)

n jobs and n machines with identical $\ell_e(x)$ functions

Pure Nash: each job selects a different machine, load = $\ell_e(1)$:

Optimal…

Load balancing:



Example: Atomic Game (mixed Nash)

n jobs and n machines with identical $\ell_e(x)$ functions

Mixed Nash: e.g. each job selects uniformly random:

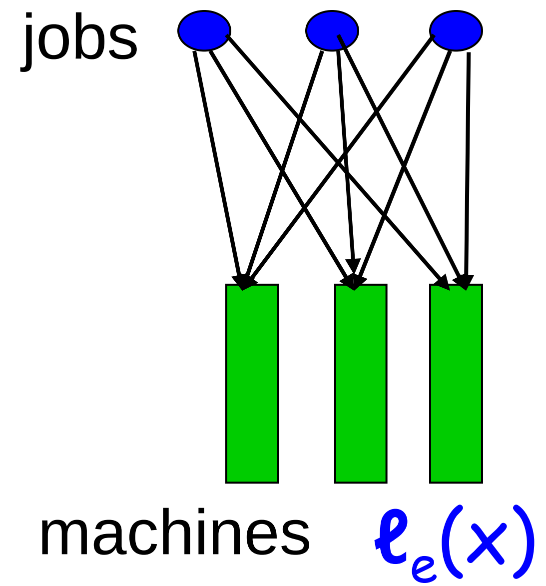
With high prob.
max load ~ log n/loglog n

$\Rightarrow$ expected load is approx

> ~ $\ell_e(1) + \ell_e(\log n)/n$

a lot more when $\ell_e(x)$ grows fast

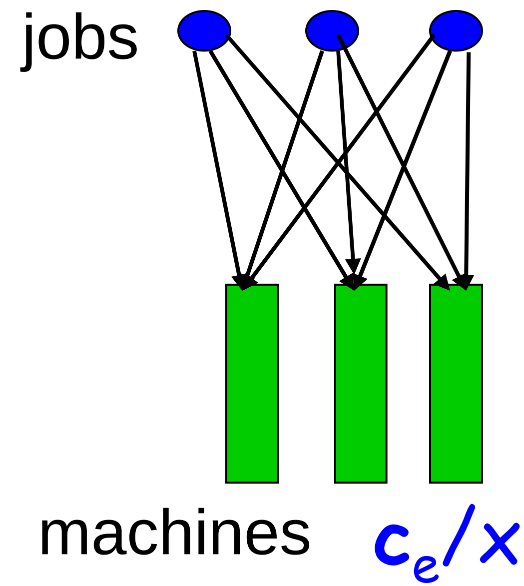
Load balancing:



Example: Cost-sharing (mixed vs pure)

- n jobs and n machines with identical costs c_e/x functions

Cost-sharing:



- Pure Nash: select one machine to use. Total cost c_e
- Mixed Nash: e.g. each job selects uniformly random:

With high prob.

expected cost ~ $\Omega(n\, c_e)$

$\Omega(n)$ times more than pure Nash

Learning?

Iterated play where users update play based on experience

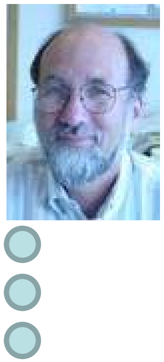
Traditional Setting: stock market

m experts   N options

Goal: can we do as well as the best expert?

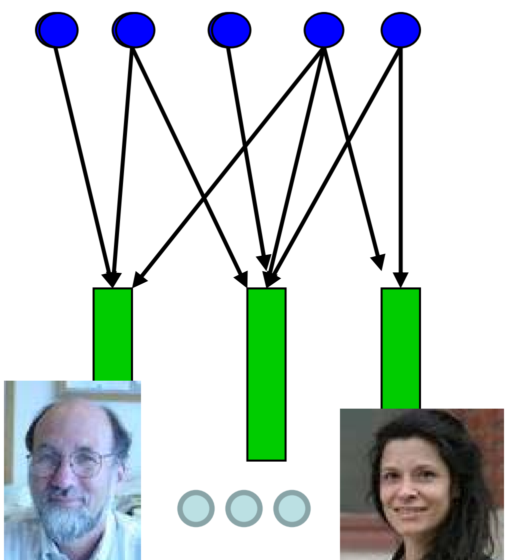
Regret = long term average cost – average cost of single best strategy with hindsight.

Learning and Games



Goal: can we do as well as the best expert?

- As the single stock in hindsight?



Focus on a single player:

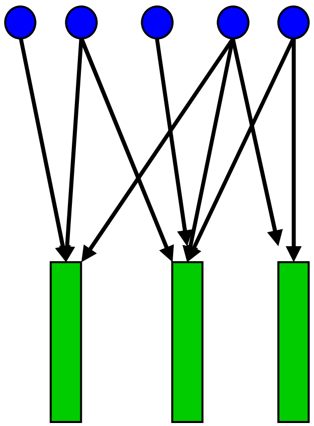
experts = strategies to play

Learn to play the best strategy with hindsight?

Best depends on others

A Natural Learning Process

Iterated play where users update probability distributions based on experience



Example: Multiplicative update (Hedge) strategies 1,…,n

Maintain weights $w_e \geq 0$

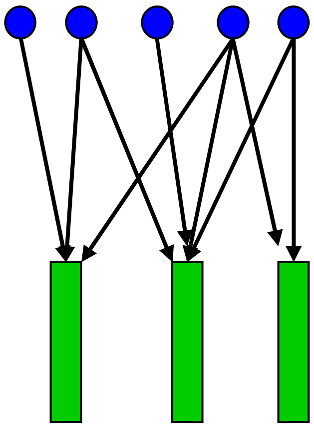
probability $p_e \sim w_e$ all e

Update w_e to $w_e (1-\varepsilon)^{cost(e)}$

$\alpha=1-\varepsilon$ think of $\varepsilon \sim$ learning rate

Learning and Games

Regret = long term average cost – average cost of single best strategy with hindsight.



Nash = all players have no regret

Hart & Mas-Colell: general games → Long term average play is (coarse) correlated equilibrium

Correlated? Correlate on history of play

(Coarse) correlated equilibrium

Coarse correlated equilibrium: probability distribution of outcomes such that for all players

expected cost $\leq$ exp. cost of any fixed strategy

Correlated eq. & players independent = Nash

Learning:

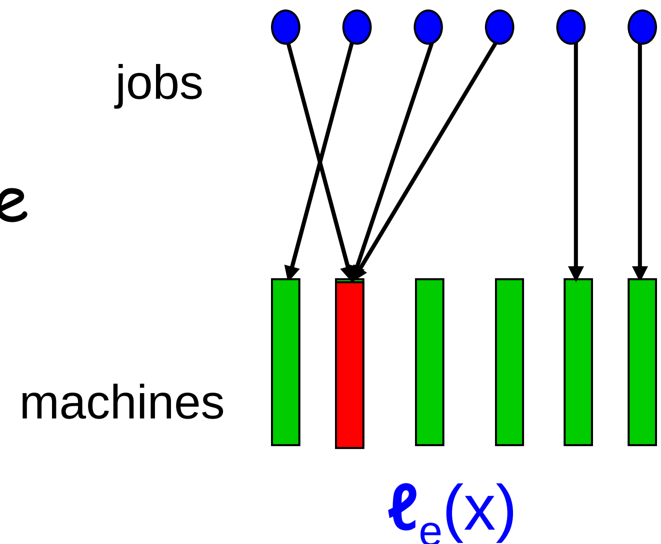
Players update independently, but correlate on shared history

Example Correlated Equilibrium: Load Balancing

n jobs and n machines with identical $\ell_e(x)$ functions

- Select a k jobs and 1 machine at random and send all k jobs to the one machine.
- Send all remaining jobs to different machines

Load balancing:



Correlated equilibrium if two costs same

- Correlated play cost: $\sim \ell_e(1) + k/n\ \ell_e(k)$
- Fixed other strategy cost $\sim \ell_e(2)$

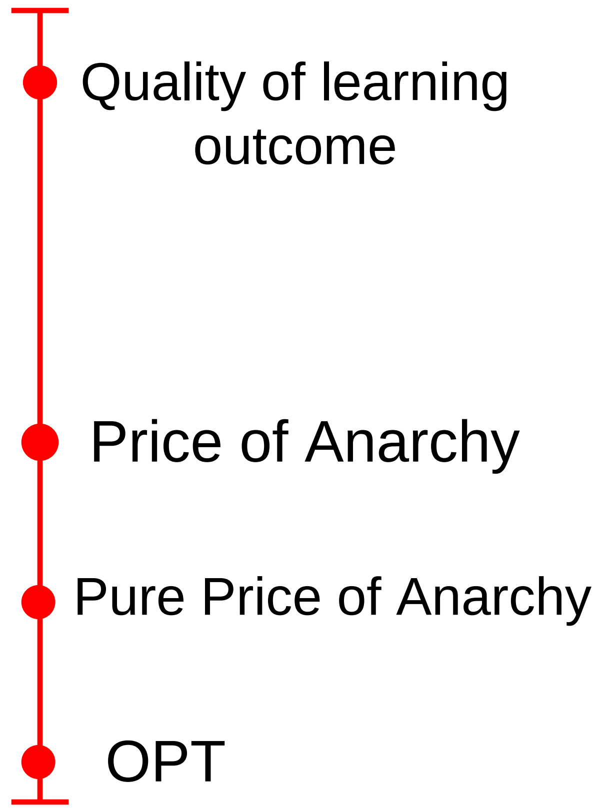
When $\ell_e(x)$ costs balance when $k=\sqrt{n}$: bad congestion

What are learning outcomes?

Blum, Even-Dar, Ligett'06: In **non-atomic** congestion games Routing without regret $\Rightarrow$ learning converge to Nash equilibria 2006.

What about atomic games?

Hope: learning will not make users coordinate on bad equilibria



Main question:
Quality of Selfish outcome

Answer: depends on which learning…

Theorem: $\forall$ correlated equilibrium is the limit point of no-regret play

Intelligent designer algorithm is no regret:

- Follow the designed sequence as long as all other players do.

Hope: natural learning process (Hedge) coordinates on good quality solutions

Quality of learning outcome

Roughgarden 2009

- In congestion games with any class of latency functions the worst price equilibrium same as quality loss in worst pure equilibrium

Yet in load balancing games…

R. Kleinberg-Piliouras-Tardos 2009

- natural learning process converges to pure Nash in almost all congestion games

Summary

We talked about Congestion Games (Routing)

- Learning (via Hedge algorithm) results in a weakly stable fixed point
- Almost always $\Rightarrow$ weakly stable = pure Nash

Many natural questions:

- Other learning methods?
- Outcome of natural learning in other games?

Note: finding Nash can be hard

- what does learning converge to?