Selfish Routing and Congestion Games: Towards a Game Based Analysis of the Internet

Berthold Vöcking

Department of Computer Science
RWTH Aachen

Delis/Evergrow School, Santorini 2004

Selfish Routing and the Price of Anarchy
- Wardrop's Traffic Model
- Discrete and Atomic Models

Congestion and Crowding Games
- Definition and Classification
- Relationship to Potential Games
- Relationship to Wardrop's traffic model

On the Evolution of Selfish Routing
- Evolutionary Game Theory
- Stability of Symmetric Routing Games
- Stability of Asymmetric Routing Games
- Time of Convergence

Summary

Definition of routing problem

Routing = multicommodity flow + latency functions

- directed (multi)graph $G = (V, E)$
- source-sink pairs s_i, t_i
- traffic rates $r_i \geq 0$ s.t. $\sum r_i = 1$
- latency functions $\ell_e(\cdot)$, $e \in E$
- flow on edge e is denoted by f_e
- latency on edge e is $\ell_e(f_e)$

global objective

$$\text{minimize average latency } C = \sum_{e \in E} f_e \cdot \ell_e(f_e)$$

Definition of routing problem

Routing = multicommodity flow + latency functions

- directed (multi)graph $G = (V, E)$
- source-sink pairs s_i, t_i
- traffic rates $r_i \geq 0$ s.t. $\sum r_i = 1$
- latency functions $\ell_e(\cdot)$, $e \in E$
- flow on edge e is denoted by f_e
- latency on edge e is $\ell_e(f_e)$

global objective

$$\text{minimize average latency } C = \sum_{e \in E} f_e \cdot \ell_e(f_e)$$

Definition of routing problem

Routing = multicommodity flow + latency functions

- directed (multi)graph $G = (V, E)$
- source-sink pairs s_i, t_i
- traffic rates $r_i \geq 0$ s.t. $\sum r_i = 1$
- latency functions $\ell_e(\cdot)$, $e \in E$
- flow on edge e is denoted by f_e
- latency on edge e is $\ell_e(f_e)$

global objective

$$\text{minimize average latency } C = \sum_{e \in E} f_e \cdot \ell_e(f_e)$$

Definition of routing problem

Routing = multicommodity flow + latency functions

- directed (multi)graph $G = (V, E)$
- source-sink pairs s_i, t_i
- traffic rates $r_i \geq 0$ s.t. $\sum r_i = 1$
- latency functions $\ell_e(\cdot)$, $e \in E$
- flow on edge e is denoted by f_e
- latency on edge e is $\ell_e(f_e)$

global objective

$$\text{minimize average latency } C = \sum_{e \in E} f_e \cdot \ell_e(f_e)$$

Definition of routing problem

Routing = multicommodity flow + latency functions

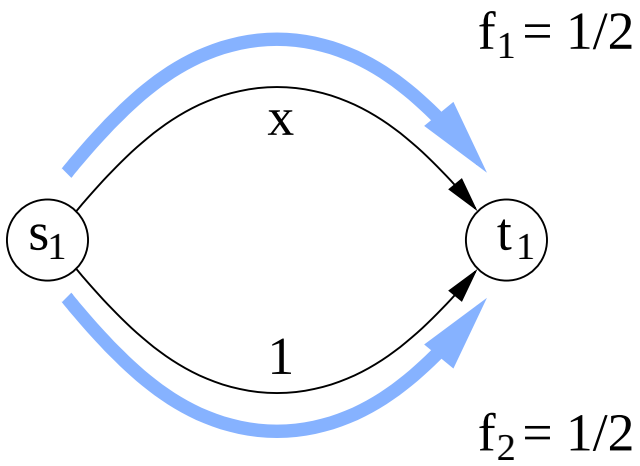
- directed (multi)graph $G = (V, E)$
- source-sink pairs s_i, t_i
- traffic rates $r_i \geq 0$ s.t. $\sum r_i = 1$
- latency functions $\ell_e(\cdot)$, $e \in E$
- flow on edge e is denoted by f_e
- latency on edge e is $\ell_e(f_e)$

global objective

$$\text{minimize average latency } C = \sum_{e \in E} f_e \cdot \ell_e(f_e)$$

Pigou's Example

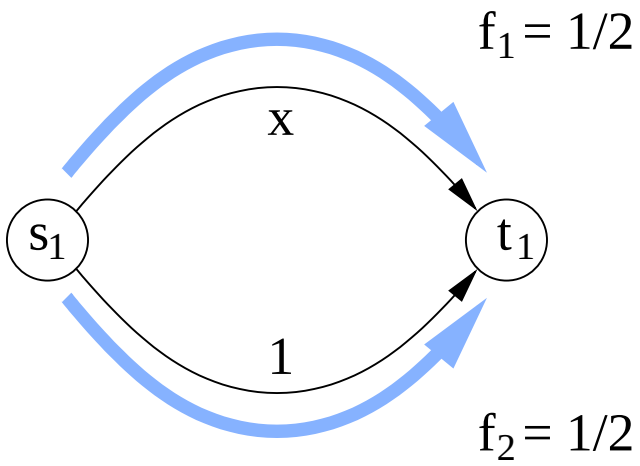
Assume $r = r_1 = 1$.
Then optimal solution is



$$C = f_1 \cdot \ell_1(f_1) + f_2 \cdot \ell_2(f_2) = \frac{3}{4}.$$

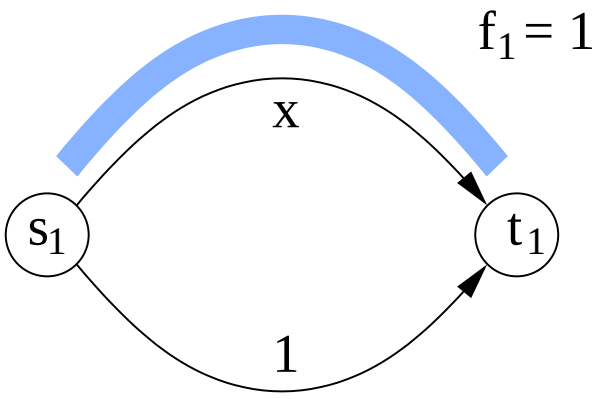
Pigou's Example

Assume $r = r_1 = 1$.
Then optimal solution is



$$C = f_1 \cdot \ell_1(f_1) + f_2 \cdot \ell_2(f_2) = \frac{3}{4}.$$

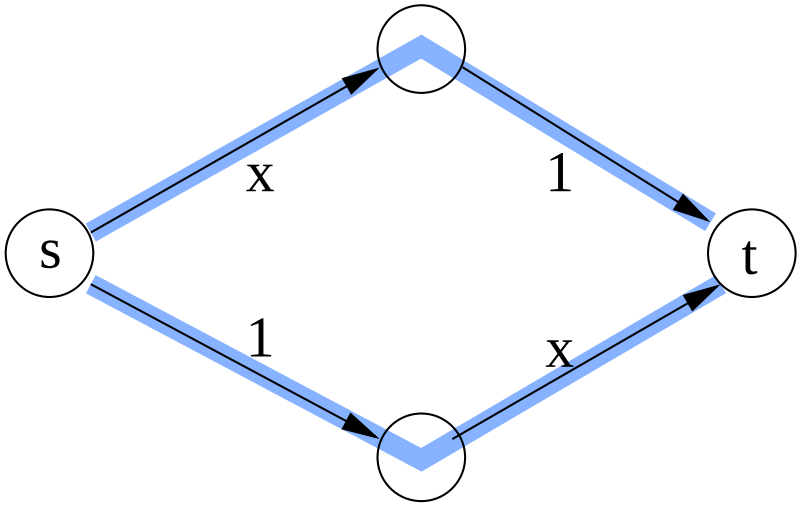
But flow on edge 2 is envious.
Moves onto edge 1.



Now $C = 1$.

Braess paradox

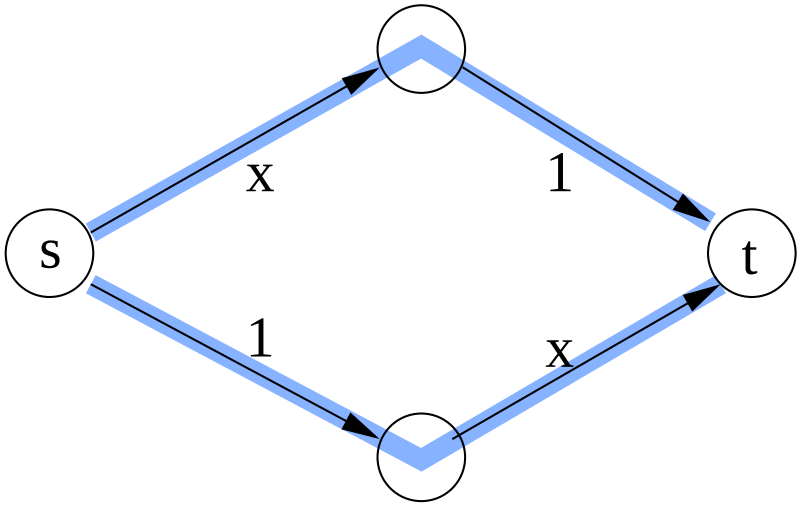
Optimal Flow:



$$C = \frac{3}{2}.$$

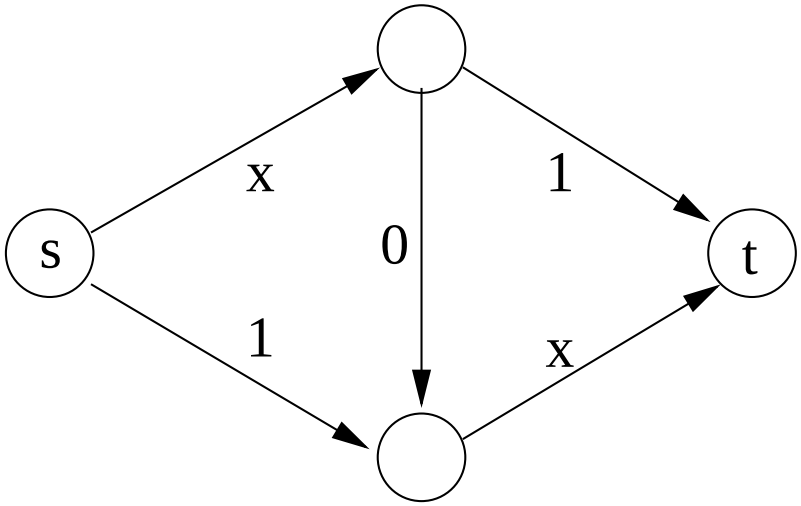
Braess paradox

Optimal Flow:



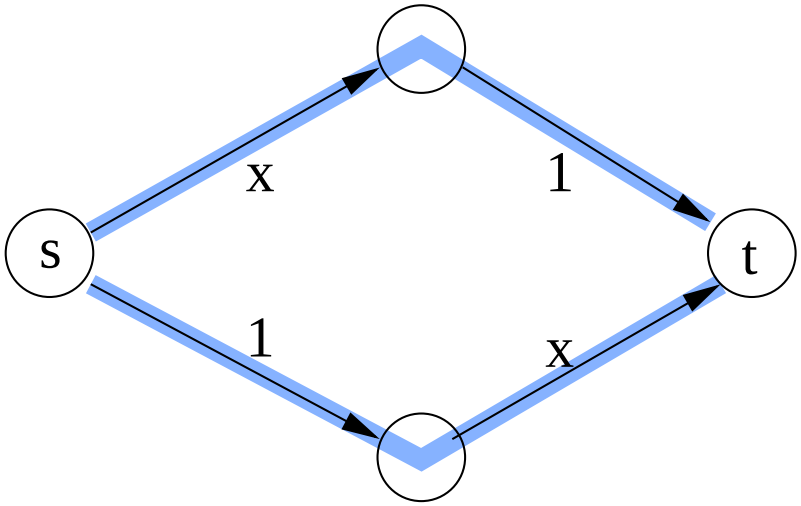
$C = \frac{3}{2}$.

Build a new road.



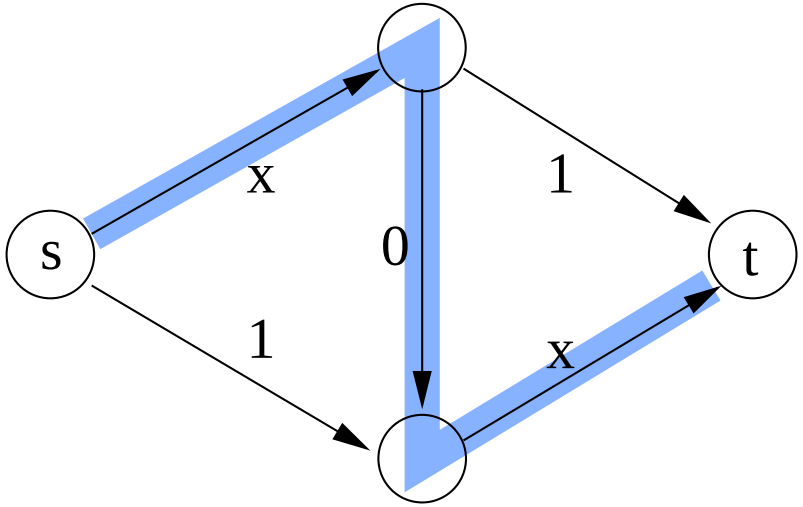
Braess paradox

Optimal Flow:



$C = \frac{3}{2}$.

Build a new road.



Now $C = 2$.

Selfish flow - intuition

- selfish flow = routes of many noncooperative, selfish agents
- flow is at Nash equilibrium if no agent can improve its latency by changing its path

Motivating Examples:

- cars in a highway system
- packets in a network at steady state

Selfish flow - intuition

- selfish flow = routes of many noncooperative, selfish agents
- flow is at Nash equilibrium if no agent can improve its latency by changing its path

Motivating Examples:

- cars in a highway system
- packets in a network at steady state

Selfish flow - formal definition of equilibrium

- For commodity $i \in \{1, \ldots, k\}$, let $\mathcal{P}_i$ denote the set of paths from s_i to t_i.
- For a path P and a flow f, define $\ell_P(f) = \sum_{e \in P} \ell_e(f)$.
- For a path P, let f_P denote the flow along P.
- A flow f is at *Nash equilibrium* if for all $i \in \{1, \ldots, k\}$ and every $P_1, P_2 \in \mathcal{P}_i$ and $\delta \in (0, f_{P_1}]$, we have $\ell_{P_1}(f) \leq \ell_{P_2}(f')$, where

$$f'_P = \begin{cases} f_P - \delta & \text{if } P = P_1 \\ f_P + \delta & \text{if } P = P_2 \\ f_P & \text{else} \end{cases}$$

Selfish flow - formal definition of equilibrium

- For commodity $i \in \{1, \ldots, k\}$, let $\mathcal{P}_i$ denote the set of paths from s_i to t_i.
- For a path P and a flow f, define $\ell_P(f) = \sum_{e \in P} \ell_e(f)$.
- For a path P, let f_P denote the flow along P.
- A flow f is at *Nash equilibrium* if for all $i \in \{1, \ldots, k\}$ and every $P_1, P_2 \in \mathcal{P}_i$ and $\delta \in (0, f_{P_1}]$, we have $\ell_{P_1}(f) \leq \ell_{P_2}(f')$, where

$$f'_P = \begin{cases} f_P - \delta & \text{if } P = P_1 \\ f_P + \delta & \text{if } P = P_2 \\ f_P & \text{else} \end{cases}$$

Selfish flow - formal definition of equilibrium

- For commodity $i \in \{1, \ldots, k\}$, let $\mathcal{P}_i$ denote the set of paths from s_i to t_i.
- For a path P and a flow f, define $\ell_P(f) = \sum_{e \in P} \ell_e(f)$.
- For a path P, let f_P denote the flow along P.
- A flow f is at *Nash equilibrium* if for all $i \in \{1, \ldots, k\}$ and every $P_1, P_2 \in \mathcal{P}_i$ and $\delta \in (0, f_{P_1}]$, we have $\ell_{P_1}(f) \leq \ell_{P_2}(f')$, where

$$f'_P = \begin{cases} f_P - \delta & \text{if } P = P_1 \\ f_P + \delta & \text{if } P = P_2 \\ f_P & \text{else} \end{cases}$$

Selfish flow - formal definition of equilibrium

- For commodity $i \in \{1, \ldots, k\}$, let $\mathcal{P}_i$ denote the set of paths from s_i to t_i.
- For a path P and a flow f, define $\ell_P(f) = \sum_{e \in P} \ell_e(f)$.
- For a path P, let f_P denote the flow along P.
- A flow f is at *Nash equilibrium* if for all $i \in \{1, \ldots, k\}$ and every $P_1, P_2 \in \mathcal{P}_i$ and $\delta \in (0, f_{P_1}]$, we have $\ell_{P_1}(f) \leq \ell_{P_2}(f')$, where

$$f'_P = \begin{cases} f_P - \delta & \text{if } P = P_1 \\ f_P + \delta & \text{if } P = P_2 \\ f_P & \text{else} \end{cases}$$

Price of anarchy

What is the ratio between *Nash cost* and *optimal cost*?

Theorem 1 [Roughgarden & Tardos, 2000]
For linear latency functions the cost of a Nash flow is at most $\frac{4}{3}$ times that of the optimal cost.

Theorem 2 [Roughgarden & Tardos, 2000]
Suppose

- the latency functions are continuous and non-decreasing, and
- the cost functions $C_e(f) = f \cdot \ell_e(f)$ are convex.

Then the cost of a Nash flow with rate r is at most the optimal cost for rates $2r$.

Price of anarchy

What is the ratio between *Nash cost* and *optimal cost*?

Theorem 1 [Roughgarden & Tardos, 2000]
For linear latency functions the cost of a Nash flow is at most $\frac{4}{3}$ times that of the optimal cost.

Theorem 2 [Roughgarden & Tardos, 2000]
Suppose

- the latency functions are continuous and non-decreasing, and
- the cost functions $C_e(f) = f \cdot \ell_e(f)$ are convex.

Then the cost of a Nash flow with rate r is at most the optimal cost for rates $2r$.

Price of anarchy

What is the ratio between *Nash cost* and *optimal cost*?

Theorem 1 [Roughgarden & Tardos, 2000]
For linear latency functions the cost of a Nash flow is at most $\frac{4}{3}$ times that of the optimal cost.

Theorem 2 [Roughgarden & Tardos, 2000]
Suppose

- the latency functions are continuous and non-decreasing, and
- the cost functions $C_e(f) = f \cdot \ell_e(f)$ are convex.

Then the cost of a Nash flow with rate r is at most the optimal cost for rates $2r$.

Characterization of optimal flow

- Flow problem can be described by convex program with convex objective function:

Minimize $\sum_{e \in E} f_e \cdot \ell_e(f_e)$
subject to:

$$\begin{array}{ll} \forall i & \sum_{P \in \mathcal{P}_i} f_P = r_i \\ \forall e & f_e = \sum_{P : e \in P} f_P \\ \forall P & f_P \geq 0 \end{array}$$

- Thus, local optimum = global optimum.

Characterization of optimal flow

- Flow problem can be described by convex program with convex objective function:

$$\text{Minimize } \sum_{e \in E} f_e \cdot \ell_e(f_e)$$
subject to:

$$\begin{array}{ll} \forall i & \sum_{P \in \mathcal{P}_i} f_P = r_i \\ \forall e & f_e = \sum_{P : e \in P} f_P \\ \forall P & f_P \geq 0 \end{array}$$

- Thus, local optimum = global optimum.

Characterization of optimal flow

Lemma: A flow f is optimal if and only if for each source-sink pair i and paths $P_1, P_2 \in \mathcal{P}_i$ with $f_{P_1} > 0$,

$$\sum_{e \in P_1} C'_e(f_e) \;\leq\; \sum_{e \in P_2} C'_e(f_e) \;.$$

Characterization of Nash flow

Lemma: [Wardrop's Principle]
A flow f is in Nash equilibrium if and only if for each source-sink pair i and paths $P_1, P_2 \in \mathcal{P}_i$ with $f_{P_1} > 0$,

$$\sum_{e \in P_1} \ell_e(f_e) \;\leq\; \sum_{e \in P_2} \ell_e(f_e) \;.$$

Consequences for Linear Latency Functions

Suppose $\ell_e(x) = a_e x + b_e$.

Characterization of Nash flow f:

$$\sum_{e \in P_1} a_e f_e + b_e \;\leq\; \sum_{e \in P_2} a_e f_e + b_e$$

$C_e(x) = a_e x^2 + b_e x$, $C_e'(x) = 2a_e x + b_e$.

Characterization of optimal flow f^*:

$$\sum_{e \in P_1} 2a_e f_e^* + b_e \;\leq\; \sum_{e \in P_2} 2a_e f_e^* + b_e$$

Consequences for Linear Latency Functions

Suppose $\ell_e(x) = a_e x + b_e$.

Characterization of Nash flow f:

$$\sum_{e \in P_1} a_e f_e + b_e \;\leq\; \sum_{e \in P_2} a_e f_e + b_e$$

$C_e(x) = a_e x^2 + b_e x$, $C_e'(x) = 2a_e x + b_e$.

Characterization of optimal flow f^*:

$$\sum_{e \in P_1} 2a_e f_e^* + b_e \;\leq\; \sum_{e \in P_2} 2a_e f_e^* + b_e$$

Consequences for Linear Latency Functions

Suppose $\ell_e(x) = a_e x + b_e$.

Characterization of Nash flow f:

$$\sum_{e \in P_1} a_e f_e + b_e \;\leq\; \sum_{e \in P_2} a_e f_e + b_e$$

$C_e(x) = a_e x^2 + b_e x$, $C_e'(x) = 2a_e x + b_e$.

Characterization of optimal flow f^*:

$$\sum_{e \in P_1} 2a_e f_e^* + b_e \;\leq\; \sum_{e \in P_2} 2a_e f_e^* + b_e$$

Consequences for Linear Latency Functions

Suppose $\ell_e(x) = a_e x + b_e$.

Characterization of Nash flow f:

$$\sum_{e \in P_1} a_e f_e + b_e \;\leq\; \sum_{e \in P_2} a_e f_e + b_e$$

$C_e(x) = a_e x^2 + b_e x$, $C_e'(x) = 2a_e x + b_e$.

Characterization of optimal flow f^*:

$$\sum_{e \in P_1} 2a_e f_e^* + b_e \;\leq\; \sum_{e \in P_2} 2a_e f_e^* + b_e$$

Corollary 1: Suppose $b_e = 0$, for every $e \in E$. Then

$$\text{Nash flow } f \;=\; \text{optimal flow } f^* \;.$$

Corollary 2: The Nash flow $f/2$ is optimal for rates $r_i/2$.

Corollary 1: Suppose $b_e = 0$, for every $e \in E$. Then

$$\text{Nash flow } f \;=\; \text{optimal flow } f^* \;.$$

Corollary 2: The Nash flow $f/2$ is optimal for rates $r_i/2$.

Applying Corollary 2 gives

$$
\begin{aligned}
C(f^*) &\geq C\left(\frac{f}{2}\right) \\
&= \sum_e a_e \left(\frac{f_e}{2}\right)^2 + b_e \frac{f_e}{2} \\
&\geq \frac{1}{4} \sum_e a_e f_e^2 + b_e f_e \\
&= \frac{1}{4} C(f)
\end{aligned}
$$

Thus $C(f) \leq 4C(f^*)$. □

Applying Corollary 2 gives

$$\begin{aligned} C(f^*) &\geq C\left(\frac{f}{2}\right) \\ &= \sum_e a_e \left(\frac{f_e}{2}\right)^2 + b_e \frac{f_e}{2} \\ &\geq \frac{1}{4} \sum_e a_e f_e^2 + b_e f_e \\ &= \frac{1}{4} C(f) \end{aligned}$$

Thus $C(f) \leq 4C(f^*)$. □

Applying Corollary 2 gives

$$\begin{aligned} C(f^*) &\geq C\left(\frac{f}{2}\right) \\ &= \sum_e a_e \left(\frac{f_e}{2}\right)^2 + b_e \frac{f_e}{2} \\ &\geq \frac{1}{4} \sum_e a_e f_e^2 + b_e f_e \\ &= \frac{1}{4} C(f) \end{aligned}$$

Thus $C(f) \leq 4C(f^*)$. □

Applying Corollary 2 gives

$$\begin{aligned} C(f^*) &\geq C\left(\frac{f}{2}\right) \\ &= \sum_e a_e \left(\frac{f_e}{2}\right)^2 + b_e \frac{f_e}{2} \\ &\geq \frac{1}{4} \sum_e a_e f_e^2 + b_e f_e \\ &= \frac{1}{4} C(f) \end{aligned}$$

Thus $C(f) \leq 4C(f^*)$. □

Applying Corollary 2 gives

$$\begin{aligned} C(f^*) &\geq C\left(\frac{f}{2}\right) \\ &= \sum_e a_e \left(\frac{f_e}{2}\right)^2 + b_e \frac{f_e}{2} \\ &\geq \frac{1}{4} \sum_e a_e f_e^2 + b_e f_e \\ &= \frac{1}{4} C(f) \end{aligned}$$

Thus $C(f) \leq 4C(f^*)$. $\square$

Makespan scheduling

- m servers with bandwidths $b_1 \geq b_2 \geq \cdots \geq b_m$
- n tasks with weights $w_1, \ldots, w_n$ should be assigned to servers
- decision variables $x_i^j = 1$ if $i \to j$, else 0
- cost of server j

$$C_j = \sum_{i \in [n]} \frac{w_i \cdot x_i^j}{b_j}$$

- global objective: minimize makespan

$$C = \max_{j \in [m]} C_j$$

Makespan scheduling

- m servers with bandwidths $b_1 \geq b_2 \geq \cdots \geq b_m$
- n tasks with weights $w_1, \ldots, w_n$ should be assigned to servers
- decision variables $x_i^j = 1$ if $i \to j$, else 0
- cost of server j

$$C_j = \sum_{i \in [n]} \frac{w_i \cdot x_i^j}{b_j}$$

- global objective: minimize makespan

$$C = \max_{j \in [m]} C_j$$

Makespan scheduling

- m servers with bandwidths $b_1 \geq b_2 \geq \cdots \geq b_m$
- n tasks with weights $w_1, \ldots, w_n$ should be assigned to servers
- decision variables $x_i^j = 1$ if $i \to j$, else 0
- cost of server j

$$C_j = \sum_{i \in [n]} \frac{w_i \cdot x_i^j}{b_j}$$

- global objective: minimize makespan

$$C = \max_{j \in [m]} C_j$$

Makespan scheduling

- m servers with bandwidths $b_1 \geq b_2 \geq \cdots \geq b_m$
- n tasks with weights $w_1, \ldots, w_n$ should be assigned to servers
- decision variables $x_i^j = 1$ if $i \to j$, else 0
- cost of server j

$$C_j = \sum_{i \in [n]} \frac{w_i \cdot x_i^j}{b_j}$$

- global objective: minimize makespan

$$C = \max_{j \in [m]} C_j$$

Makespan scheduling

- m servers with bandwidths $b_1 \geq b_2 \geq \cdots \geq b_m$
- n tasks with weights $w_1, \ldots, w_n$ should be assigned to servers
- decision variables $x_i^j = 1$ if $i \to j$, else 0
- cost of server j

$$C_j = \sum_{i \in [n]} \frac{w_i \cdot x_i^j}{b_j}$$

- global objective: minimize makespan

$$C = \max_{j \in [m]} C_j$$

A routing game based on makespan scheduling

[Koutsoupias & Papadimitriou, 1999]

- every task is managed by a selfish agent
- mixed strategy: probability distribution, $p_i^j = \mathbf{Pr}\left[x_i^j = 1\right]$
- expected cost of server j

$$\mathbf{E}\left[C_j\right] = \sum_{i \in [n]} \frac{w_i \cdot p_i^j}{b_j}$$

- every agent aims to minimize its expected cost
- expected cost of task i on server j

$$c_i^j = \mathbf{E}\left[C_j \mid x_i^j = 1\right] = \frac{w_i + \sum_{k \neq i} w_k \cdot p_k^j}{b_j}$$

A routing game based on makespan scheduling

[Koutsoupias & Papadimitriou, 1999]

- every task is managed by a selfish agent
- mixed strategy: probability distribution, $p_i^j = \mathbf{Pr}\left[x_i^j = 1\right]$
- expected cost of server j

$$\mathbf{E}\left[C_j\right] = \sum_{i \in [n]} \frac{w_i \cdot p_i^j}{b_j}$$

- every agent aims to minimize its expected cost
- expected cost of task i on server j

$$c_i^j = \mathbf{E}\left[C_j \mid x_i^j = 1\right] = \frac{w_i + \sum_{k \neq i} w_k \cdot p_k^j}{b_j}$$

A routing game based on makespan scheduling

[Koutsoupias & Papadimitriou, 1999]

- every task is managed by a selfish agent
- mixed strategy: probability distribution, $p_i^j = \mathbf{Pr}\left[x_i^j = 1\right]$
- expected cost of server j

$$\mathbf{E}\left[C_j\right] = \sum_{i \in [n]} \frac{w_i \cdot p_i^j}{b_j}$$

- every agent aims to minimize its expected cost
- expected cost of task i on server j

$$c_i^j = \mathbf{E}\left[C_j \mid x_i^j = 1\right] = \frac{w_i + \sum_{k \neq i} w_k \cdot p_k^j}{b_j}$$

A routing game based on makespan scheduling

[Koutsoupias & Papadimitriou, 1999]

- every task is managed by a selfish agent
- mixed strategy: probability distribution, $p_i^j = \mathbf{Pr}\left[x_i^j = 1\right]$
- expected cost of server j

$$\mathbf{E}\left[C_j\right] = \sum_{i \in [n]} \frac{w_i \cdot p_i^j}{b_j}$$

- every agent aims to minimize its expected cost
- expected cost of task i on server j

$$c_i^j = \mathbf{E}\left[C_j \mid x_i^j = 1\right] = \frac{w_i + \sum_{k \neq i} w_k \cdot p_k^j}{b_j}$$

A routing game based on makespan scheduling

[Koutsoupias & Papadimitriou, 1999]

- every task is managed by a selfish agent
- mixed strategy: probability distribution, $p_i^j = \mathbf{Pr}\left[x_i^j = 1\right]$
- expected cost of server j

$$\mathbf{E}\left[C_j\right] = \sum_{i \in [n]} \frac{w_i \cdot p_i^j}{b_j}$$

- every agent aims to minimize its expected cost
- expected cost of task i on server j

$$c_i^j = \mathbf{E}\left[C_j \mid x_i^j = 1\right] = \frac{w_i + \sum_{k \neq i} w_k \cdot p_k^j}{b_j}$$

Cost of mixed allocation

$$C = \mathbf{E}\left[\max_{j\in[m]} C_j\right]$$

Nash equilibrium

Allocation is in Nash equilibrium if

$$p_i^j > 0 \quad \Rightarrow \quad \forall k \in [m] : \; c_i^j \leq c_i^k$$

Coordination ratio

$$R = \max_{X \text{ is Nash allocation}} \frac{C(X)}{\mathrm{opt}}$$

Easy lower bounds – two servers

$R \geq \frac{3}{2}$

Example: 2 servers, 2 tasks, $b_1 = b_2 = w_1 = w_2 = 1$

$opt = 1$ $\qquad$ $C = 1 \cdot \frac{1}{2} + 2 \cdot \frac{1}{2} = \frac{3}{2}$

Easy lower bounds – m identical servers

$$R = \Omega\left(\frac{\log m}{\log\log m}\right)$$

Example:

- $n = m$ tasks
- all bandwidths and weights equal to one
- one task for each server gives $opt = 1$
- fully mixed Nash assignment: $p_i^j = \frac{1}{m}$, for $i \in [n], m \in [m]$, gives $C = \Theta\left(\frac{\log m}{\log\log m}\right)$

Results on the coordination ratio

	identical servers	general model
pure	$2 - \frac{1}{m}$	$\Theta\left(\frac{\log m}{\log\log m}\right)$
mixed	$\Theta\left(\frac{\log m}{\log\log m}\right)$	$\Theta\left(\frac{\log m}{\log\log\log m}\right)$

[Koutsoupias & Papadimitriou, 1999]
[Mavronicolas & Spirakis, 2001]
[Czumaj & Vöcking, 2002]

Proof of upper bound for pure strategies

- Fix an arbitrary pure Nash equilibrium.
- Suppose the maximum cost over all servers is $r \cdot opt$, for $r \geq 1$.
- To simplify notation, assume r is integral.

We need to show: $r = O\left(\frac{\log m}{\log \log m}\right)$.

- Recall $b_1 \geq b_2 \geq \cdots \geq b_m$.
- For $0 \leq k \leq r-1$, define group G_k to be the maximum length prefix of servers with cost $\geq k \cdot opt$.

Proof of upper bound for pure strategies

- Fix an arbitrary pure Nash equilibrium.
- Suppose the maximum cost over all servers is $r \cdot opt$, for $r \geq 1$.
- To simplify notation, assume r is integral.

We need to show: $r = O\left(\frac{\log m}{\log \log m}\right)$.

- Recall $b_1 \geq b_2 \geq \cdots \geq b_m$.
- For $0 \leq k \leq r-1$, define group G_k to be the maximum length prefix of servers with cost $\geq k \cdot opt$.

Proof of upper bound for pure strategies

- Fix an arbitrary pure Nash equilibrium.
- Suppose the maximum cost over all servers is $r \cdot opt$, for $r \geq 1$.
- To simplify notation, assume r is integral.

We need to show: $r = O\left(\frac{\log m}{\log \log m}\right)$.

- Recall $b_1 \geq b_2 \geq \cdots \geq b_m$.
- For $0 \leq k \leq r-1$, define group G_k to be the maximum length prefix of servers with cost $\geq k \cdot opt$.

Proof of upper bound for pure strategies

- Fix an arbitrary pure Nash equilibrium.
- Suppose the maximum cost over all servers is $r \cdot opt$, for $r \geq 1$.
- To simplify notation, assume r is integral.

$$\boxed{\text{We need to show: } r = O\left(\frac{\log m}{\log \log m}\right).}$$

- Recall $b_1 \geq b_2 \geq \cdots \geq b_m$.
- For $0 \leq k \leq r-1$, define group G_k to be the maximum length prefix of servers with cost $\geq k \cdot opt$.

Proof of upper bound for pure strategies

- Fix an arbitrary pure Nash equilibrium.
- Suppose the maximum cost over all servers is $r \cdot opt$, for $r \geq 1$.
- To simplify notation, assume r is integral.

$$\boxed{\text{We need to show: } r = O\left(\frac{\log m}{\log \log m}\right).}$$

- Recall $b_1 \geq b_2 \geq \cdots \geq b_m$.
- For $0 \leq k \leq r-1$, define group G_k to be the maximum length prefix of servers with cost $\geq k \cdot opt$.

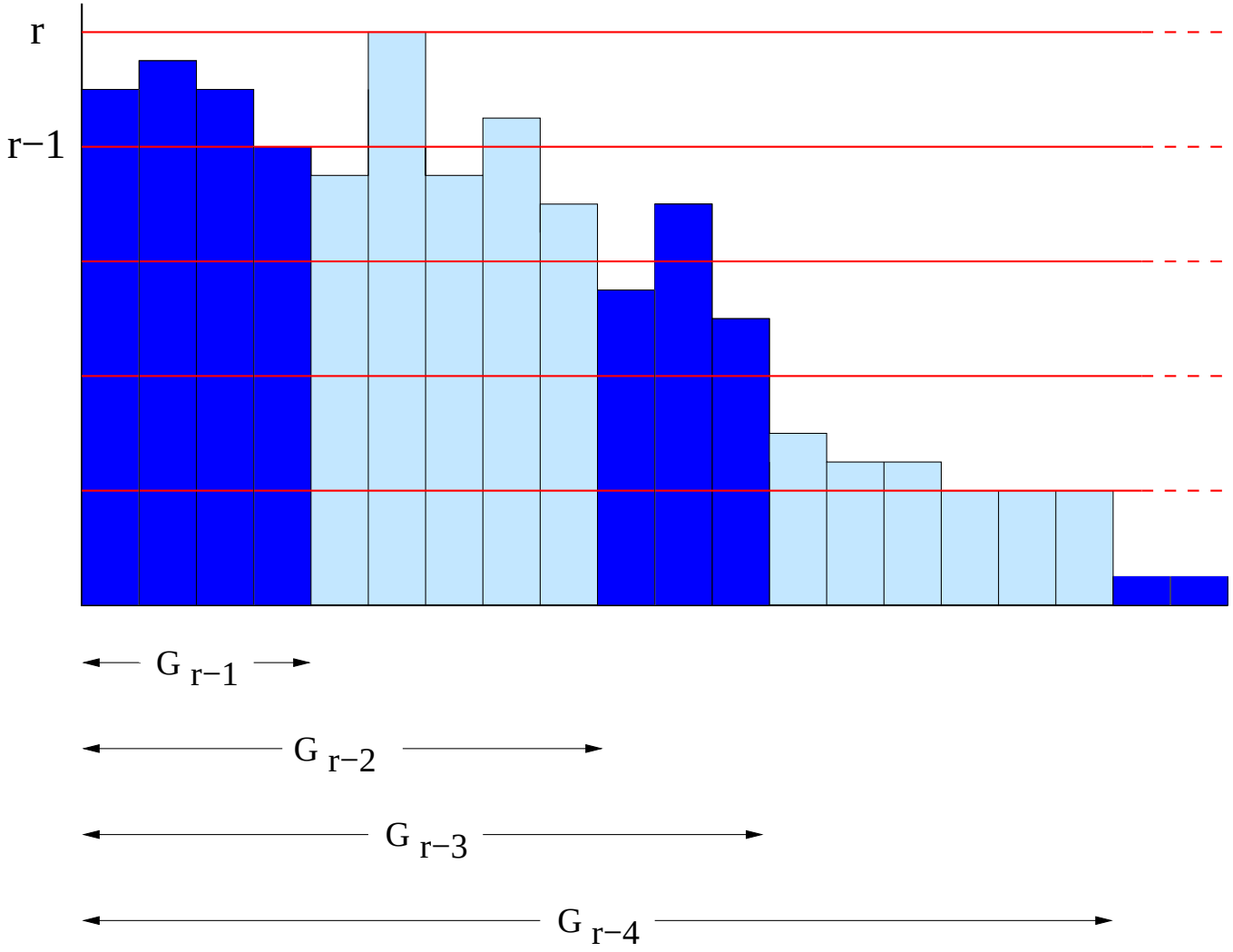
Proof of upper bound for pure strategies

- Fix an arbitrary pure Nash equilibrium.
- Suppose the maximum cost over all servers is $r \cdot opt$, for $r \geq 1$.
- To simplify notation, assume r is integral.

$$\boxed{\text{We need to show: } r = O\left(\frac{\log m}{\log \log m}\right).}$$

- Recall $b_1 \geq b_2 \geq \cdots \geq b_m$.
- For $0 \leq k \leq r-1$, define group G_k to be the maximum length prefix of servers with cost $\geq k \cdot opt$.

Definition of groups



We will prove the following recurrence

$$\boxed{\begin{aligned} |G_k| &\geq (k+1)\,|G_{k+1}| \quad \text{, for } 0 \leq k \leq r-2 \\ |G_{r-1}| &\geq 1 \ . \end{aligned}}$$

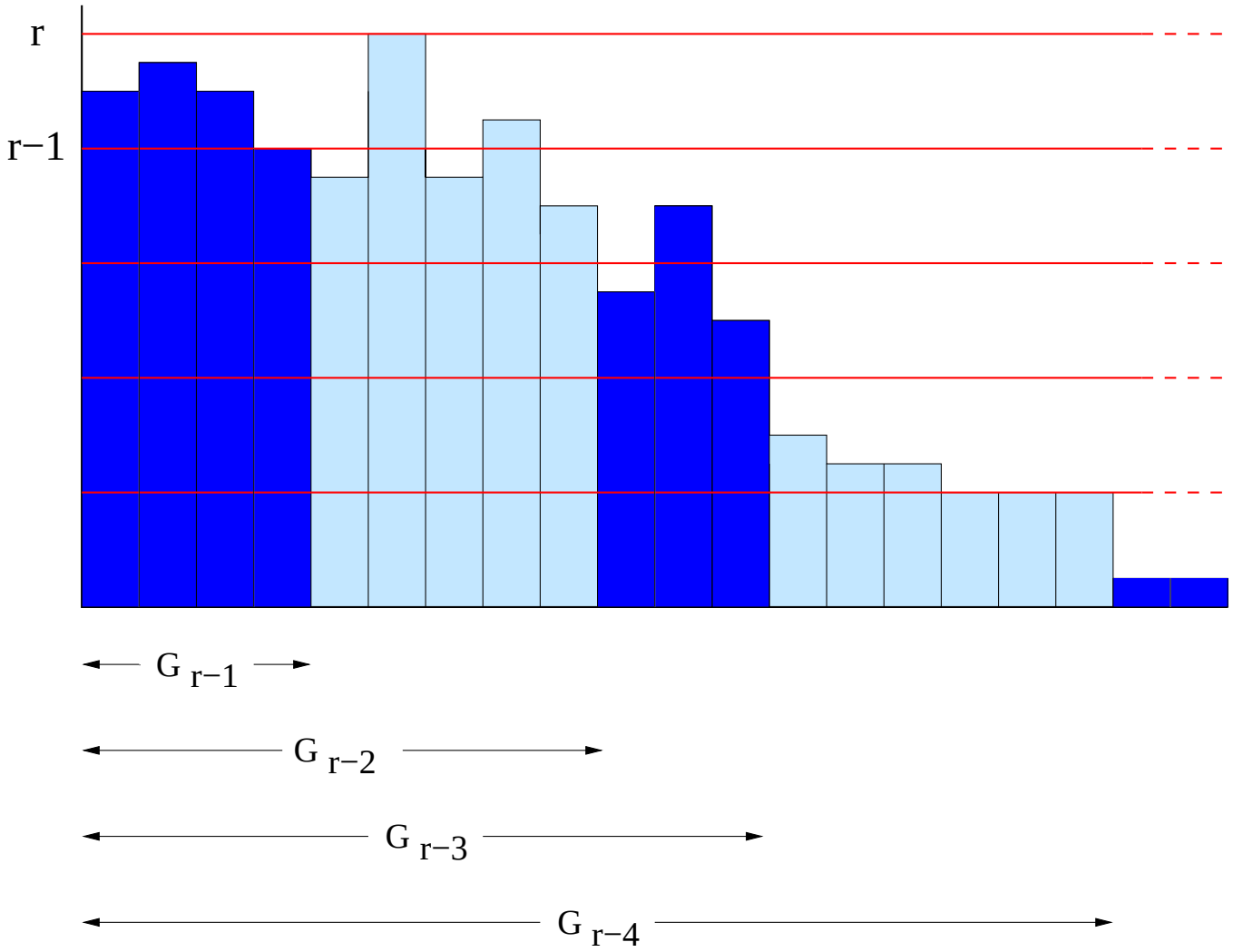
As a consequence,

$$m = |G_0| \geq (r-1)! = \Gamma(r)$$

Solving for r gives

$$r \leq \Gamma^{-1}(m) = O\left(\frac{\log m}{\log \log m}\right) \ .$$

First, let us prove $|G_{r-1}| \geq 1$



- For contradiction, assume $|G_{r-1}| = 0$, that is, server 1 has cost *less than* $(r-1)opt$.
- Let $i \in [n]$ be a task placed on the server with maximum cost, that is, task i has cost r in the Nash equilibrium.
- Moving i to server 1 gives cost *less than*

$$(r-1)opt + \frac{w_i}{b_1} \quad \leq \quad (r-1)opt + opt \quad \leq \quad r \cdot opt \ .$$

- This contradicts the Nash equilibrium assumption.

- For contradiction, assume $|G_{r-1}| = 0$, that is, server 1 has cost *less than* $(r-1)opt$.
- Let $i \in [n]$ be a task placed on the server with maximum cost, that is, task i has cost r in the Nash equilibrium.
- Moving i to server 1 gives cost *less than*

$$(r-1)opt + \frac{w_i}{b_1} \quad \leq \quad (r-1)opt + opt \quad \leq \quad r \cdot opt \ .$$

- This contradicts the Nash equilibrium assumption.

- For contradiction, assume $|G_{r-1}| = 0$, that is, server 1 has cost *less than* $(r-1)opt$.
- Let $i \in [n]$ be a task placed on the server with maximum cost, that is, task i has cost r in the Nash equilibrium.
- Moving i to server 1 gives cost *less than*

$$(r-1)opt + \frac{w_i}{b_1} \quad \leq \quad (r-1)opt + opt \quad \leq \quad r \cdot opt \ .$$

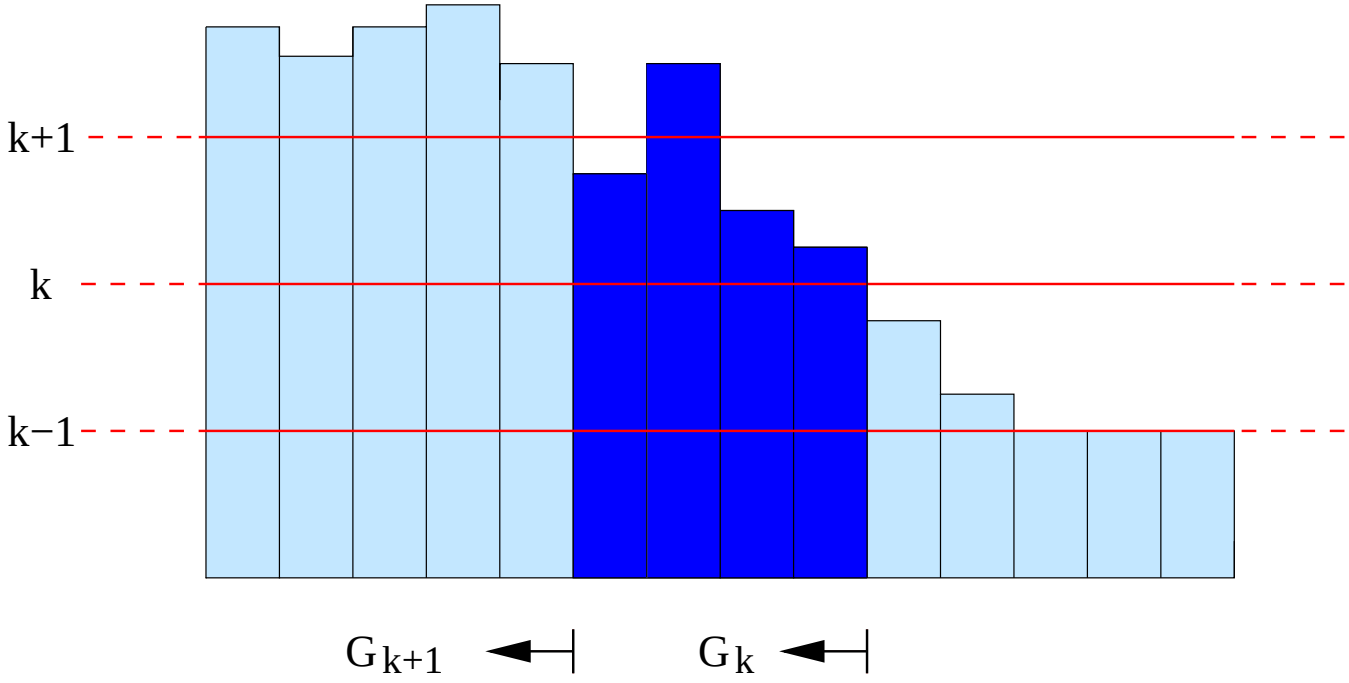
- This contradicts the Nash equilibrium assumption.

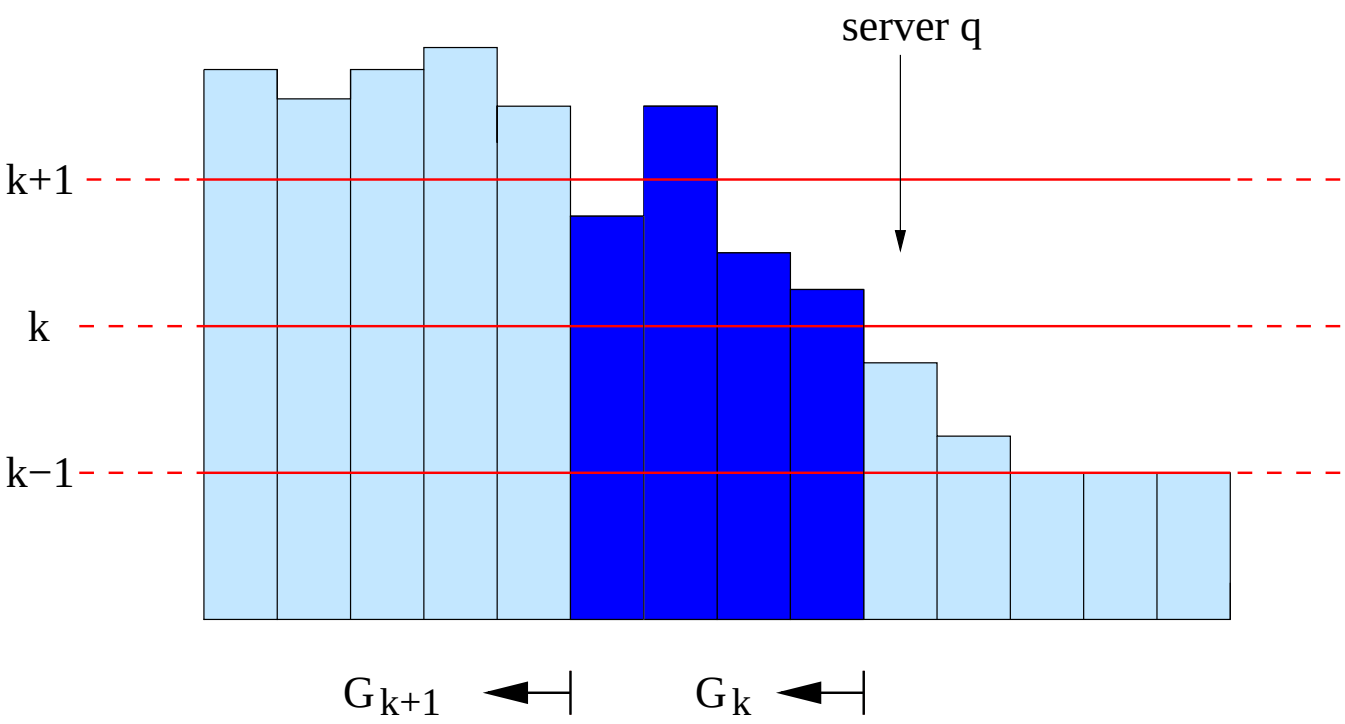
- For contradiction, assume $|G_{r-1}| = 0$, that is, server 1 has cost *less than* $(r-1)opt$.
- Let $i \in [n]$ be a task placed on the server with maximum cost, that is, task i has cost r in the Nash equilibrium.
- Moving i to server 1 gives cost *less than*

$$(r-1)opt + \frac{w_i}{b_1} \quad \leq \quad (r-1)opt + opt \quad \leq \quad r \cdot opt \ .$$

- This contradicts the Nash equilibrium assumption.

It remains to show $|G_k| \geq (k+1)\,|G_{k+1}|$, for $0 \leq k \leq r-2$.



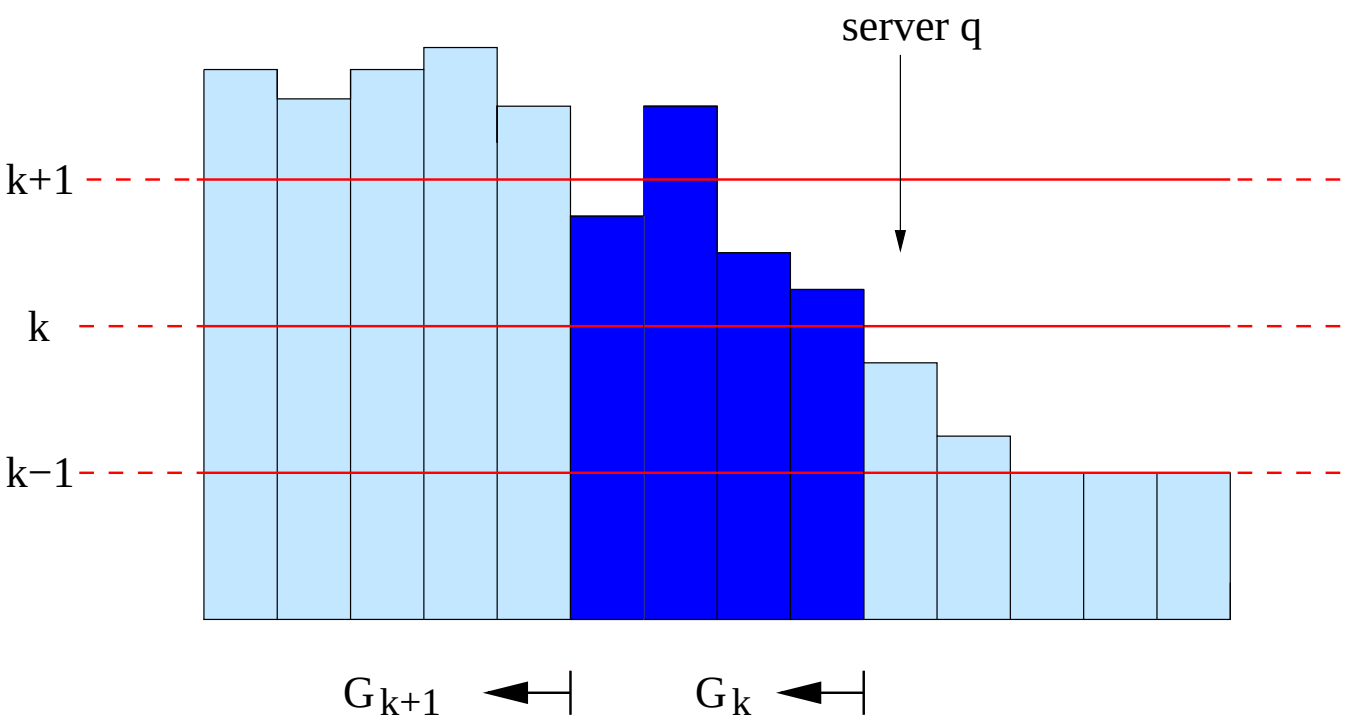


all tasks in G_{k+1} have weight larger than $b_q \cdot opt$

$\Rightarrow$ OPT must give these tasks to servers with speed $> b_q$

$\Rightarrow$ OPT needs to assign these tasks to servers in G_k

$\Rightarrow$ $|G_k| \geq (k+1)|G_{k+1}|$. □

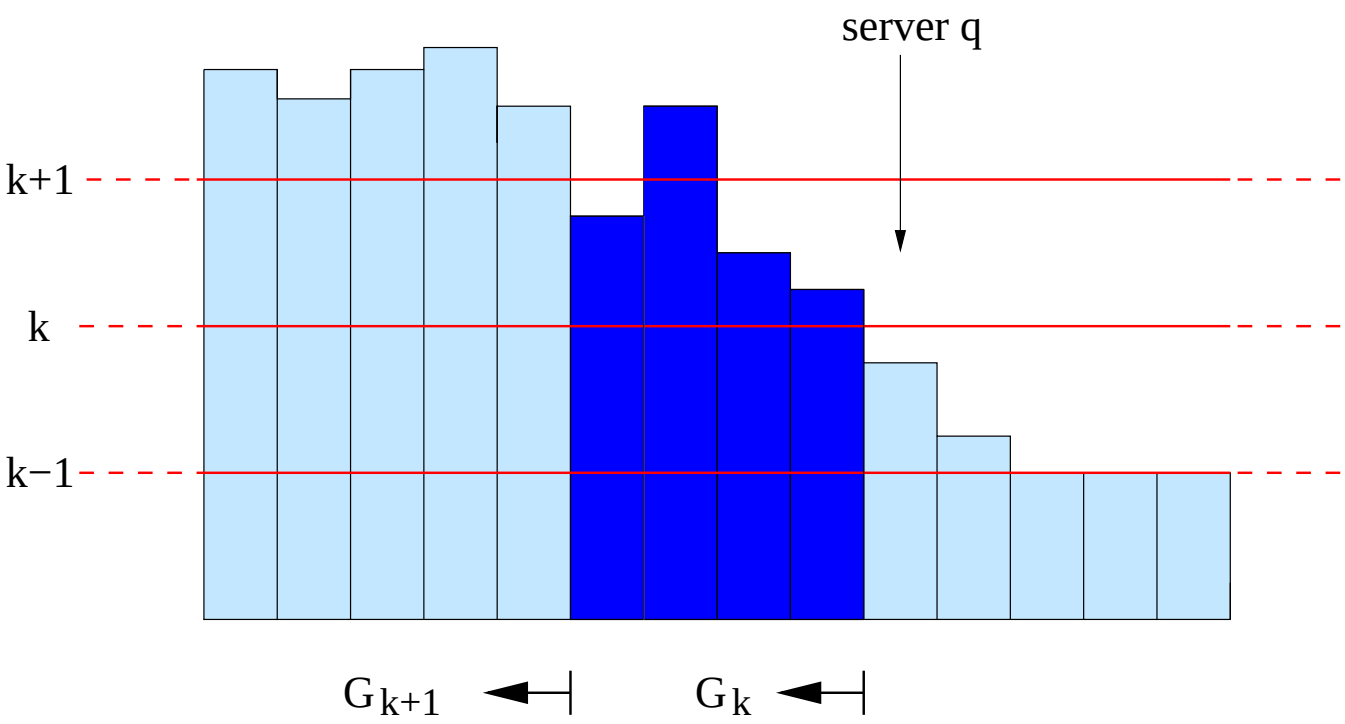


all tasks in G_{k+1} have weight larger than $b_q \cdot opt$

$\Rightarrow$ OPT must give these tasks to servers with speed $> b_q$

$\Rightarrow$ OPT needs to assign these tasks to servers in G_k

$\Rightarrow$ $|G_k| \geq (k+1)\,|G_{k+1}|$. □

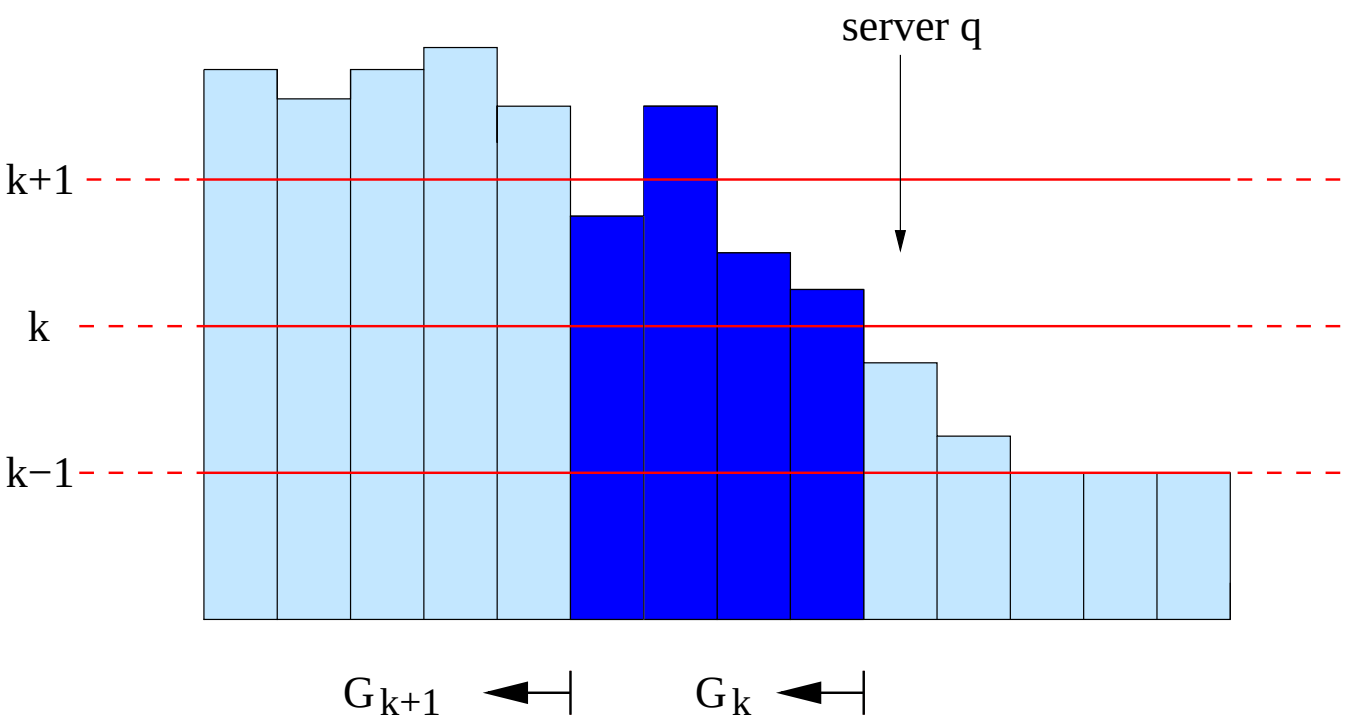


all tasks in G_{k+1} have weight larger than $b_q \cdot opt$

$\Rightarrow$ OPT must give these tasks to servers with speed $> b_q$

$\Rightarrow$ OPT needs to assign these tasks to servers in G_k

$\Rightarrow |G_k| \geq (k+1)|G_{k+1}|$. □



all tasks in G_{k+1} have weight larger than $b_q \cdot opt$

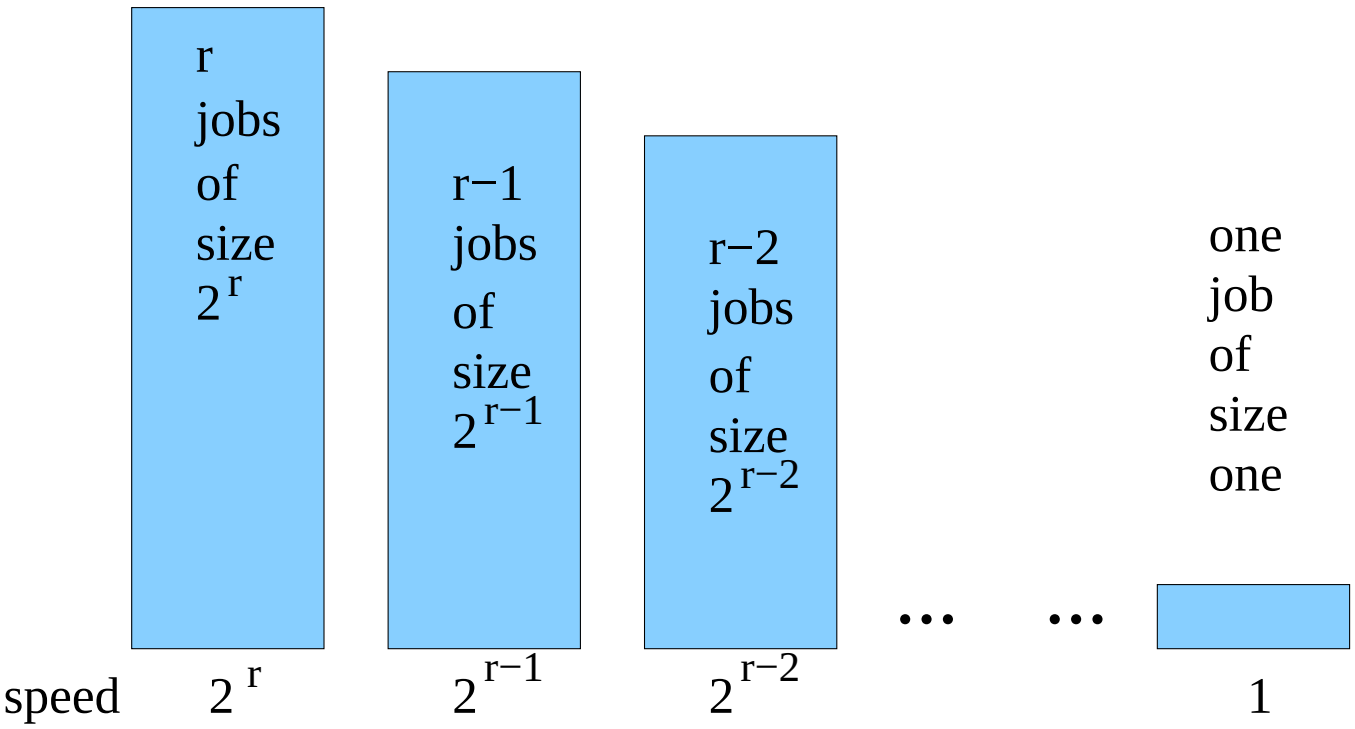
$\Rightarrow$ OPT must give these tasks to servers with speed $> b_q$

$\Rightarrow$ OPT needs to assign these tasks to servers in G_k

$\Rightarrow$ $|G_k| \geq (k+1)\,|G_{k+1}|$. $\square$

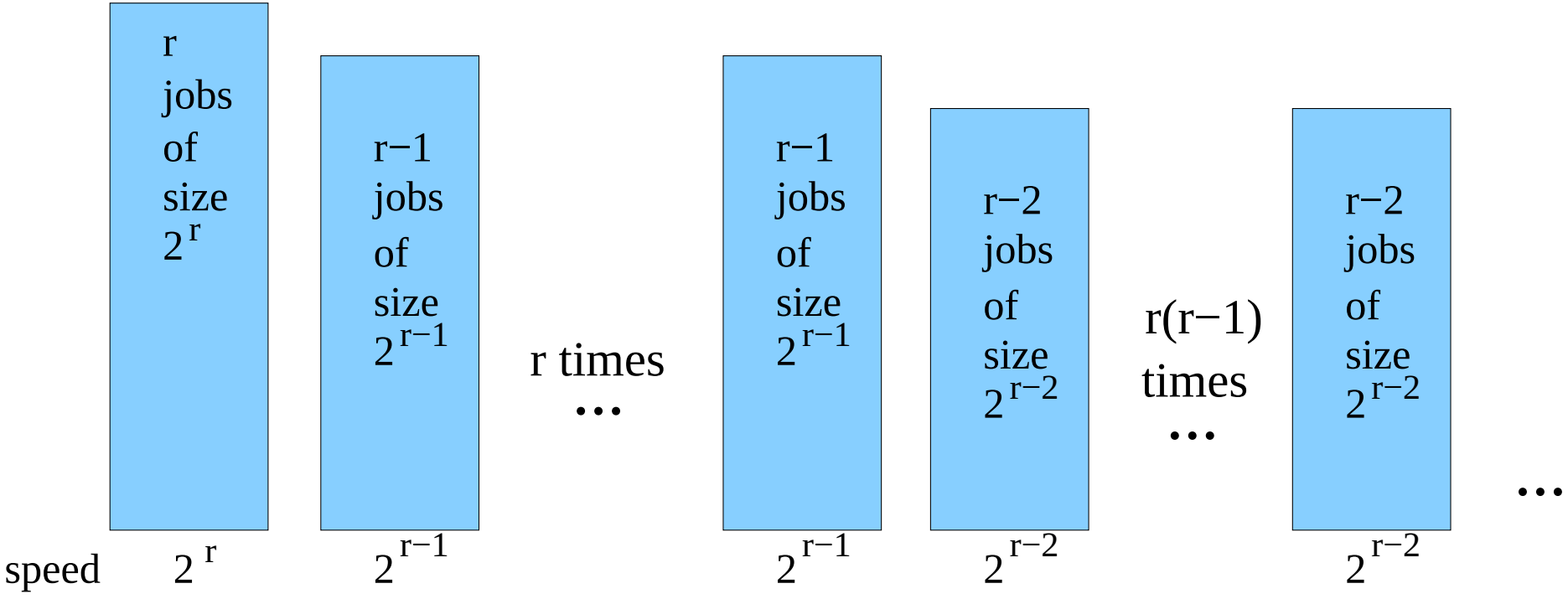
Construction of lower bound

Nash Equilibrium for r machines



Construction of lower bound

Replicate machines and their jobs in such a way that opt = O(1).



Other latency functions

Theorem (Krysta, Czmuaj, Vöcking, 2002)
The coordination ratio for the server game can be bounded polynomially in m if and only if the latency functions satisfy

$$\exists \alpha \geq 1 : \quad \forall \lambda > 0 : \quad \ell(2\,\lambda) \leq \alpha \cdot \ell(\lambda) \ .$$

Otherwise the coordination ratio is unbounded.

bounded	unbounded
linear functions polynomial functions	exponential functions queueing functions Erlang loss

Other latency functions

Theorem (Krysta, Czmuaj, Vöcking, 2002)
The coordination ratio for the server game can be bounded polynomially in m if and only if the latency functions satisfy

$$\exists \alpha \geq 1 : \quad \forall \lambda > 0 : \quad \ell(2\,\lambda) \leq \alpha \cdot \ell(\lambda) \ .$$

Otherwise the coordination ratio is unbounded.

bounded	unbounded
linear functions polynomial functions	exponential functions queueing functions Erlang loss

Discrete and atomic routing games in networks

- Price of anarchy for unweighted games with linear latency functions is 2.5
- Price of anarchy for weighted games with linear latency functions is 2.618
- If latency functions are polynomials of degree d then the price of anarchy is $O(d^d)$.
- Results hold for atomic and non-atomic games.

[Awerbuch, Azar, Epstein, unpublished work]

Discrete and atomic routing games in networks

- Price of anarchy for unweighted games with linear latency functions is 2.5
- Price of anarchy for weighted games with linear latency functions is 2.618
- If latency functions are polynomials of degree d then the price of anarchy is $O(d^d)$.
- Results hold for atomic and non-atomic games.

[Awerbuch, Azar, Epstein, unpublished work]

Discrete and atomic routing games in networks

- Price of anarchy for unweighted games with linear latency functions is 2.5
- Price of anarchy for weighted games with linear latency functions is 2.618
- If latency functions are polynomials of degree d then the price of anarchy is $O(d^d)$.
- Results hold for atomic and non-atomic games.

[Awerbuch, Azar, Epstein, unpublished work]

Discrete and atomic routing games in networks

- Price of anarchy for unweighted games with linear latency functions is 2.5
- Price of anarchy for weighted games with linear latency functions is 2.618
- If latency functions are polynomials of degree d then the price of anarchy is $O(d^d)$.
- Results hold for atomic and non-atomic games.

[Awerbuch, Azar, Epstein, unpublished work]

Congestion Games

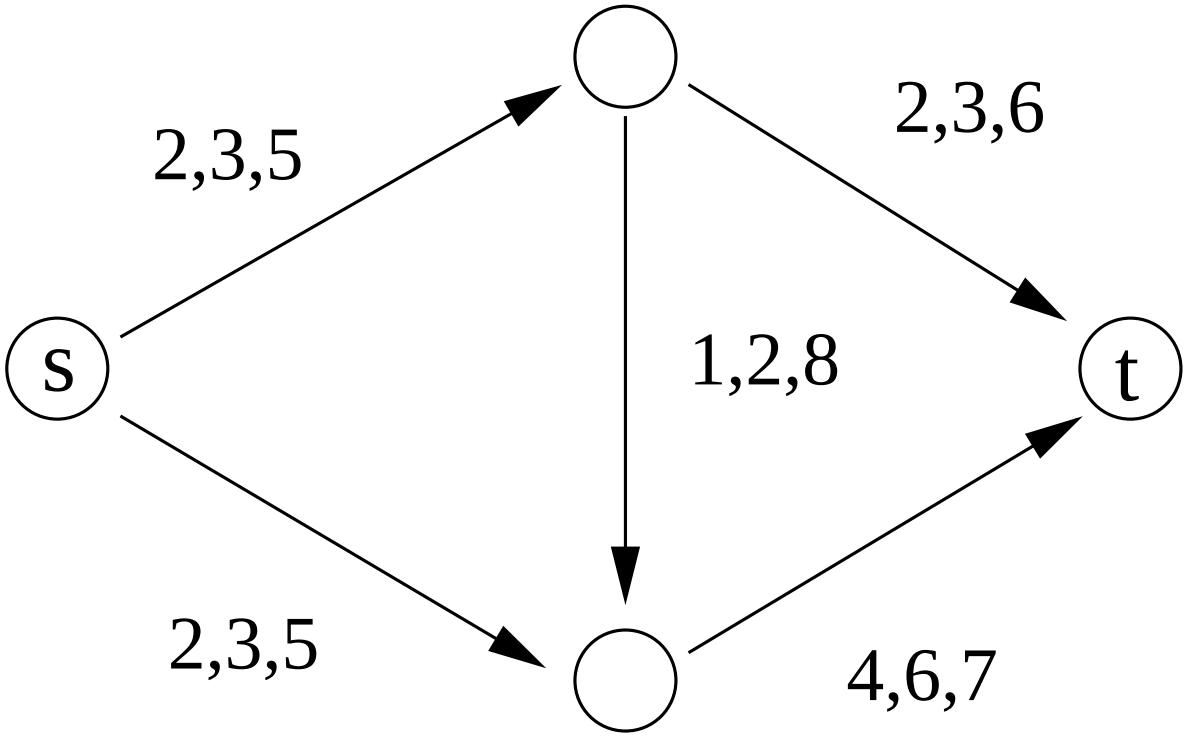
Notation:

- number of agents is denoted by n
- finite set of resources, edges E
- strategy space of player i is $S_i \subseteq 2^E$
- latency function for edge $e \in E$ is $\ell_e : [n] \to \mathcal{N}$

Subclasses:

- network congestion games
- symmetric congestion games (identical strategy spaces)
- single-choice congestion games (like parallel links)

A symmetric network congestion game



Congestion Games

Generalizations:

- weighted congestion games
- subjective congestion games

Studied variants:

- server game = weighted single-choice game
- crowding game = subjective single-choice congestion game

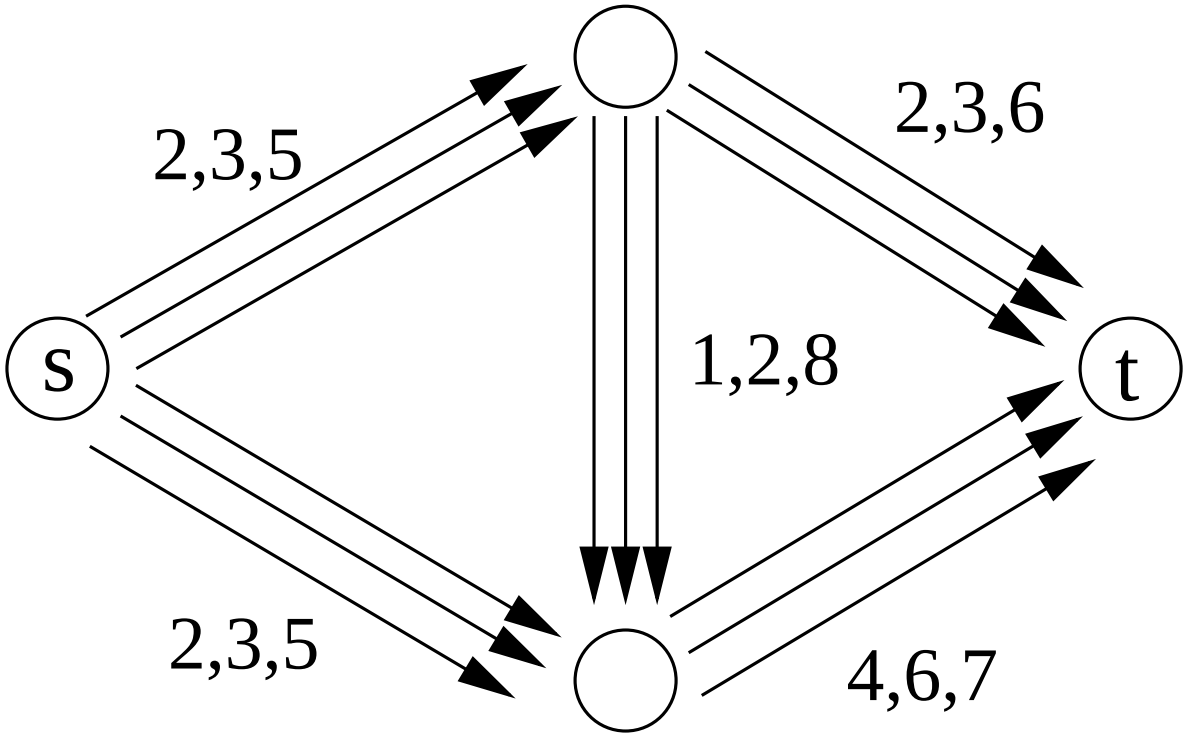
Congestion games – pure equilibria

Theorem: (Rosenthal 1973)
Every congestion game admits a pure Nash equilibrium.

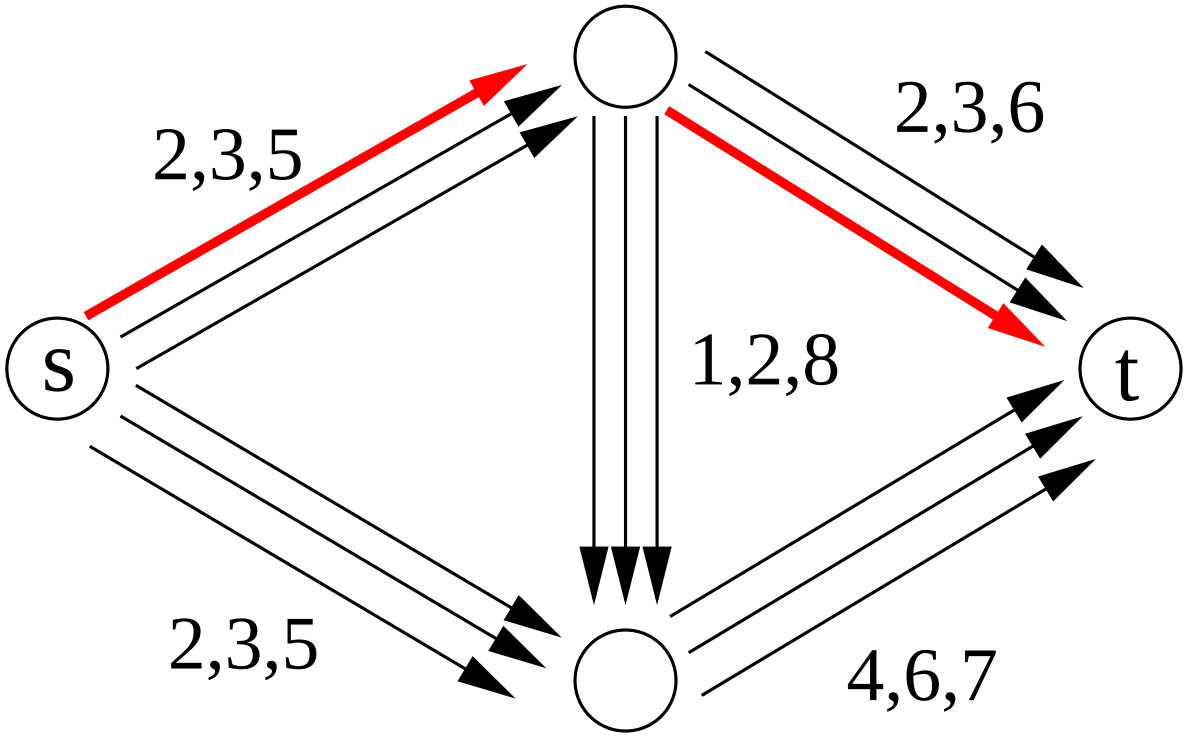
The theorem follows by a nice potential function argument:

$$\phi(s) = \sum_{e \in E} \sum_{i=1}^{n(e,s)} \ell_e(i)$$

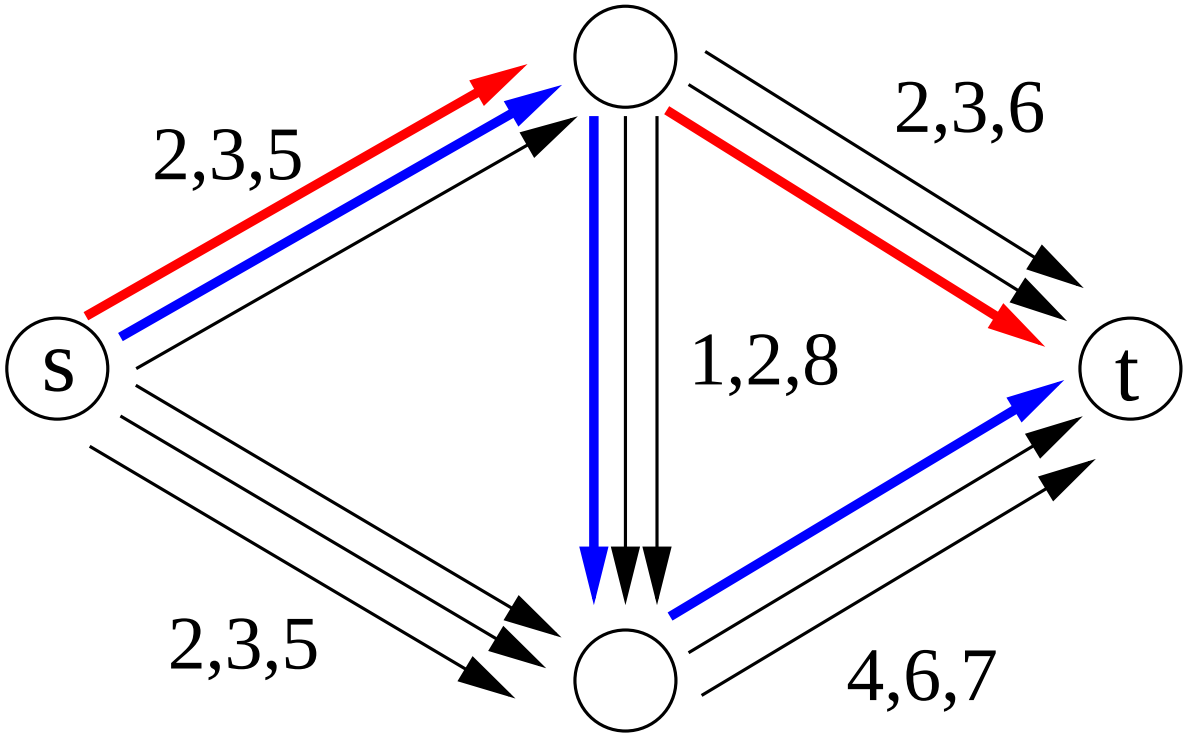
Rosenthal's potential function



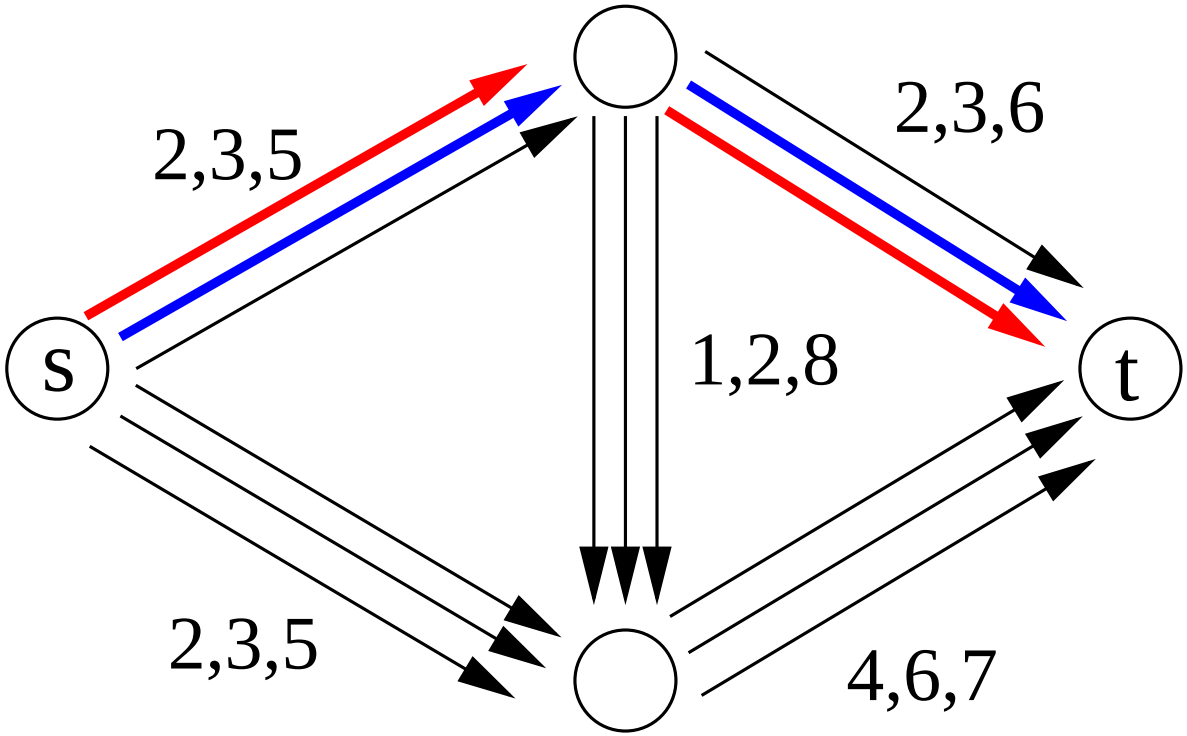
Rosenthal's potential function



Rosenthal's potential function



Rosenthal's potential function



Complexity issues:

- Potential function directly implies a pseudopolynomial algorithm for congestion games.
- There is a polynomial time algorithm for computing pure equilibria in symmetric network congestion games.
- Computing pure equilibria for more general variants is PLS-complete.

[Fabrikant, Papadimitriou, Talwar 2004]

PLS (Polynomial Local Search)

Problems in PLS:

- optimization problem Π with neighborhood Γ
- for every solution s, $\Gamma(s)$ denotes neighborhood of s
- neibhorhood can be „evaluated in polynomial time“
- objective: find local optimum wrt Γ

PLS reductions:

- Given two PLS problems Π_1 and Π_2 find
- poly-time mapping of an instance of Π_1 to an instance of Π_2
- such that local optima are preserved.

NAE-SAT (Not-All-Equal-SAT)

- **Input:** A list of 3-clauses $c_1, \ldots, c_m$, $c_i = (x_i^1, x_i^2, x_i^3)$, only positite literals, each of which having a weight w_i
- **Objective:** Maximize the weighted number of clauses in which not all literals have the same value
- **Neibhorhood:** Flips of single variables.

Theorem: NAE-SAT is PLS-complete.

NAE-SAT (Not-All-Equal-SAT)

- **Input:** A list of 3-clauses $c_1, \ldots, c_m$, $c_i = (x_i^1, x_i^2, x_i^3)$, only positite literals, each of which having a weight w_i
- **Objective:** Maximize the weighted number of clauses in which not all literals have the same value
- **Neibhorhood:** Flips of single variables.

Theorem: NAE-SAT is PLS-complete.

Hardness of computing equilibria

Theorem: Finding a Nash equilibrium in a general congestion game is PLS-complete.

Proof idea: Reduction from NAE-SAT.

- Agents correspond to variables.
- For each clause c_i spent two resources $e_i(0)$ and $e_i(1)$.
- An agent chooses a resource $e_i(0)$ or $e_i(1)$.

This is a one-to-one mapping between the solution spaces.
Agent improves its latency $\Leftrightarrow$ agent improves NAE value.
Thus, Nash equilibrium = local optimum of NAE. $\square$

Hardness of computing equilibria

Theorem: Finding a Nash equilibrium in a general congestion game is PLS-complete.

Proof idea: Reduction from NAE-SAT.

- Agents correspond to variables.
- For each clause c_i spent two resources $e_i(0)$ and $e_i(1)$.
- All latency functions are of the form $(0, 0, w_i)$.
- An agent whose variable is contained in c_i uses $e_i(0)$ if the variable has value 0, otherwise $e_i(1)$.

This is a one-to-one mapping between the solution spaces.
Agent improves its latency $\Leftrightarrow$ agent improves NAE value.
Thus, Nash equilibrium = local optimum of NAE. $\square$

Hardness of computing equilibria

Theorem: Finding a Nash equilibrium in a general congestion game is PLS-complete.

Proof idea: Reduction from NAE-SAT.

- Agents correspond to variables.
- For each clause c_i spent two resources $e_i(0)$ and $e_i(1)$.
- All latency functions are of the form $(0, 0, w_i)$.
- An agent whose variable is contained in c_i uses $e_i(0)$ if the variable has value 0, otherwise $e_i(1)$.

This is a one-to-one mapping between the solution spaces.
Agent improves its latency $\Leftrightarrow$ agent improves NAE value.
Thus, Nash equilibrium = local optimum of NAE. $\square$

Hardness of computing equilibria

Theorem: Finding a Nash equilibrium in a general congestion game is PLS-complete.

Proof idea: Reduction from NAE-SAT.

- Agents correspond to variables.
- For each clause c_i spent two resources $e_i(0)$ and $e_i(1)$.
- All latency functions are of the form $(0, 0, w_i)$.
- An agent whose variable is contained in c_i uses $e_i(0)$ if the variable has value 0, otherwise $e_i(1)$.

This is a one-to-one mapping between the solution spaces.
Agent improves its latency $\Leftrightarrow$ agent improves NAE value.
Thus, Nash equilibrium = local optimum of NAE. $\square$

Hardness of computing equilibria

Theorem: Finding a Nash equilibrium in a general congestion game is PLS-complete.

Proof idea: Reduction from NAE-SAT.

- Agents correspond to variables.
- For each clause c_i spent two resources $e_i(0)$ and $e_i(1)$.
- All latency functions are of the form $(0, 0, w_i)$.
- An agent whose variable is contained in c_i uses $e_i(0)$ if the variable has value 0, otherwise $e_i(1)$.

This is a one-to-one mapping between the solution spaces.
Agent improves its latency $\Leftrightarrow$ agent improves NAE value.
Thus, Nash equilibrium = local optimum of NAE. $\square$

Hardness of computing equilibria

Theorem: Finding a Nash equilibrium in a general congestion game is PLS-complete.

Proof idea: Reduction from NAE-SAT.

- Agents correspond to variables.
- For each clause c_i spent two resources $e_i(0)$ and $e_i(1)$.
- All latency functions are of the form $(0, 0, w_i)$.
- An agent whose variable is contained in c_i uses $e_i(0)$ if the variable has value 0, otherwise $e_i(1)$.

This is a one-to-one mapping between the solution spaces.
Agent improves its latency $\Leftrightarrow$ agent improves NAE value.
Thus, Nash equilibrium = local optimum of NAE. $\square$

Some open problems

- Find an efficient algorithm for computing a $(1+\epsilon)$-*approximate equilibrium*, i.e., no agent can improve its latency by a factor of more than $(1+\epsilon)$.
- Show existence of a polynomial length local improvement path leading to a $(1+\epsilon)$-approximate equilibrium.
- Find a simple (randomized) process that quickly converges to a $(1+\epsilon)$-approximate equilibrium.

Some open problems

- Find an efficient algorithm for computing a $(1+\epsilon)$-*approximate equilibrium*, i.e., no agent can improve its latency by a factor of more than $(1+\epsilon)$.
- Show existence of a polynomial length local improvement path leading to a $(1+\epsilon)$-approximate equilibrium.
- Find a simple (randomized) process that quickly converges to a $(1+\epsilon)$-approximate equilibrium.

Some open problems

- Find an efficient algorithm for computing a $(1+\epsilon)$-*approximate equilibrium*, i.e., no agent can improve its latency by a factor of more than $(1+\epsilon)$.
- Show existence of a polynomial length local improvement path leading to a $(1+\epsilon)$-approximate equilibrium.
- Find a simple (randomized) process that quickly converges to a $(1+\epsilon)$-approximate equilibrium.

Relationship to Wardrop's traffic model

- Wardrop's traffic model is a continuous variant of a network congestion game.
- Same kind of potential function

$$\phi(f) = \sum_{e \in E} \int_0^{f(e)} \ell_e(s) ds$$

ensures existence and essential uniqueness of equilibria via convex programming.

Critical thoughts

- We studied large networks using game theory.
- Our results are mainly concerned with properties of equilibria.
- These equilibria are based on fundamental game theoretic assumptions like unbounded rationality and full knowledge.

Both of these assumptions cannot be justified when considering large networks like the Internet.

Evolutionary game theory

- Suppose a symmetric two player game is played by a large population.
- Different players are paired up at random over and over again.
- Examples for different behavioral models:
  - *Success-driven:* At a fixed rate each player selects another player at random, and switches to other player's strategy with probability proportional to other player's payoff.
  - *Disappointment-driven:* The smaller the payoff of a player, the faster she switches to the strategy of another randomly selected player. W.l.o.g., the rate at which players migrate decreases linear in their payoff.

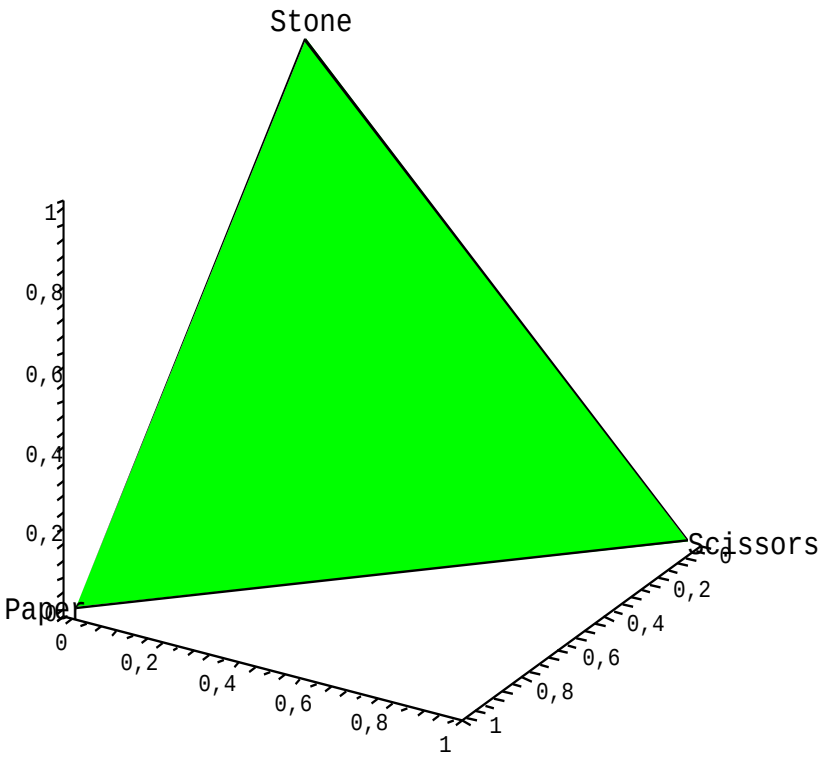
The replicator dynamics

- We model the *dynamics* of the population vector $\mathbf{x}(t)$ in a so-called fluid limit model.
- Both suggested behavorial models (and others) lead to the same kind of dynamics!

$$\dot{x}_i = \lambda(\mathbf{x}) \cdot x_i \cdot ((\mathbf{A}\mathbf{x})_i - \mathbf{x} \cdot \mathbf{A}\mathbf{x})$$

Example: Paper-Scissors-Stone

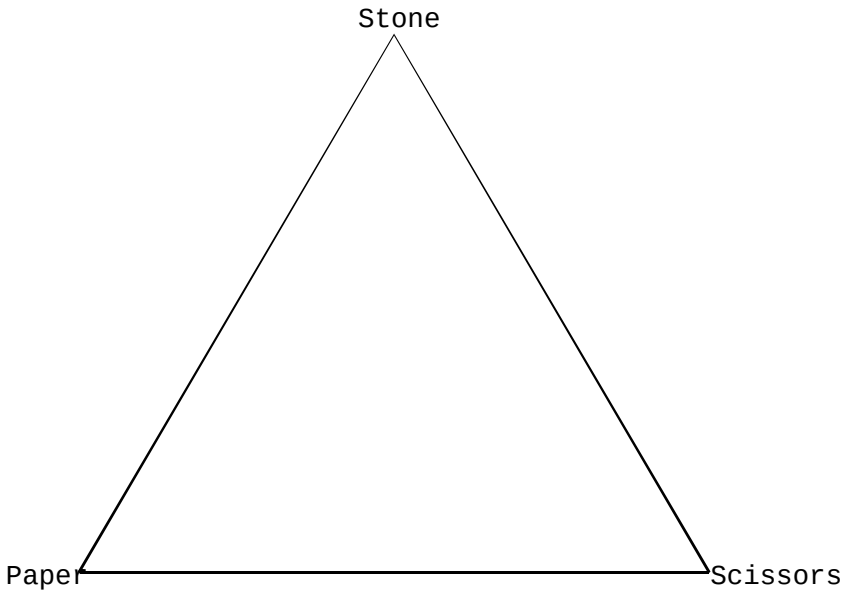
$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 2 \\ 2 & 0 & 1 \end{bmatrix}$$



- Where is the Nash equilibrium? What should one expect?
- Stable orbit for the standard case
- Convergent for $\epsilon > 0$
- Divergent for $\epsilon < 0$

Example: Paper-Scissors-Stone

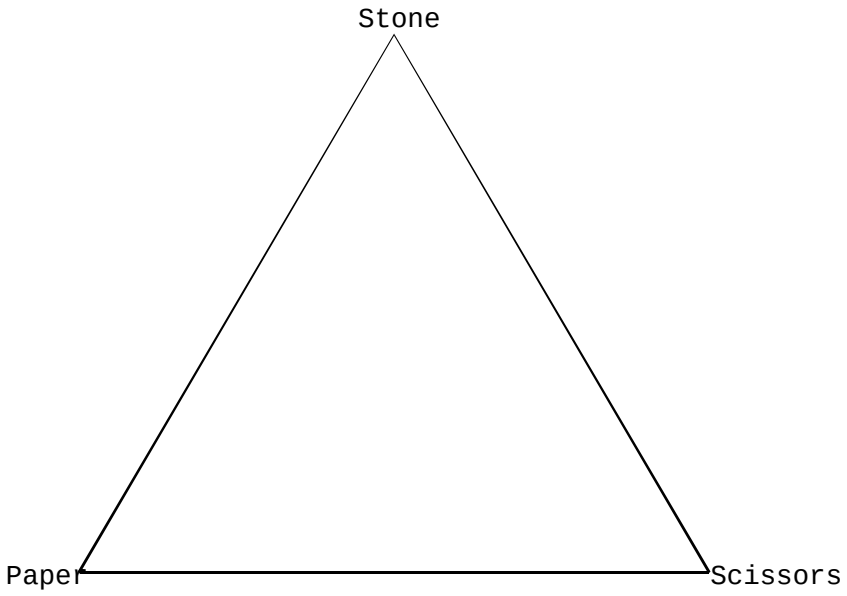
$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 2 \\ 2 & 0 & 1 \end{bmatrix}$$



- Where is the Nash equilibrium? What should one expect?
- Stable orbit for the standard case
- Convergent for $\epsilon > 0$
- Divergent for $\epsilon < 0$

Example: Paper-Scissors-Stone

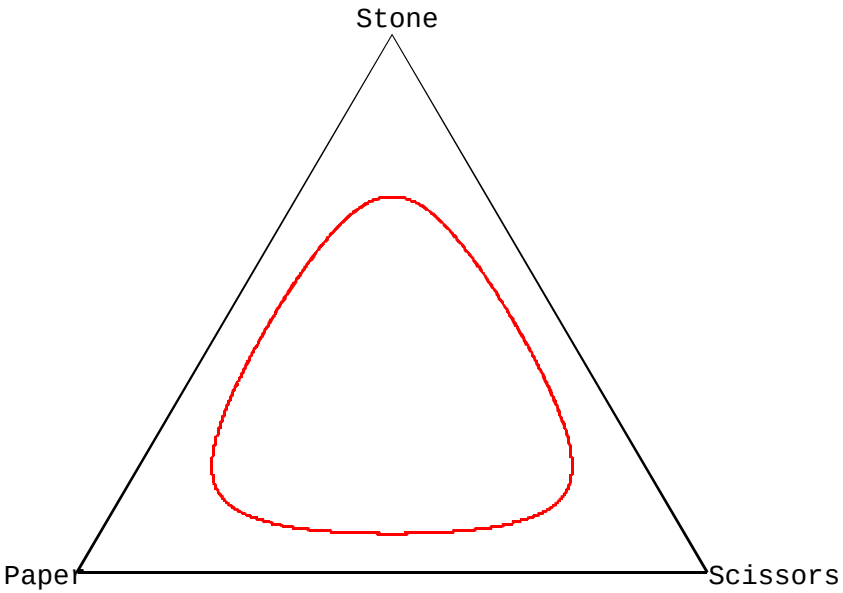
$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 2 \\ 2 & 0 & 1 \end{bmatrix}$$



- Where is the Nash equilibrium? What should one expect?
- Stable orbit for the standard case
- Convergent for $\epsilon > 0$
- Divergent for $\epsilon < 0$

Example: Paper-Scissors-Stone

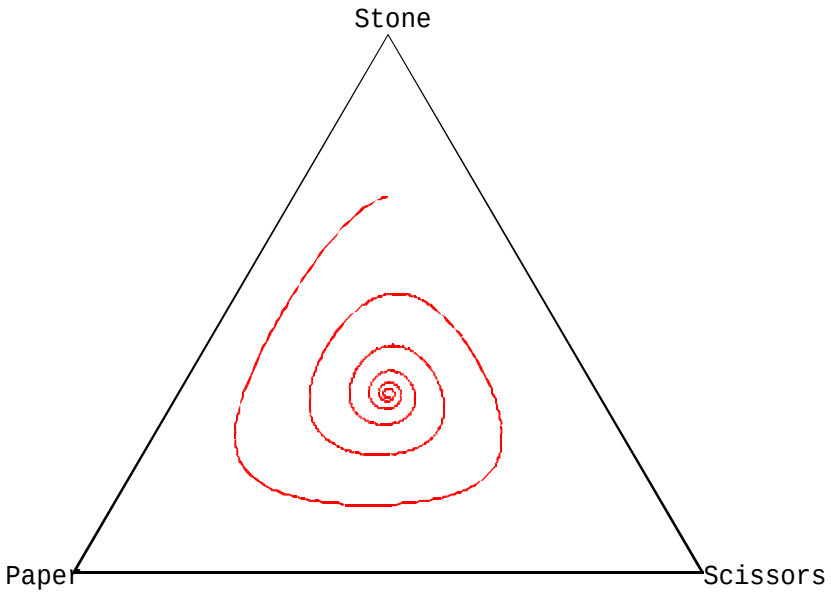
$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 2 \\ 2 & 0 & 1 \end{bmatrix}$$



- Where is the Nash equilibrium? What should one expect?
- Stable orbit for the standard case
- Convergent for $\epsilon > 0$
- Divergent for $\epsilon < 0$

Example: Paper-Scissors-Stone

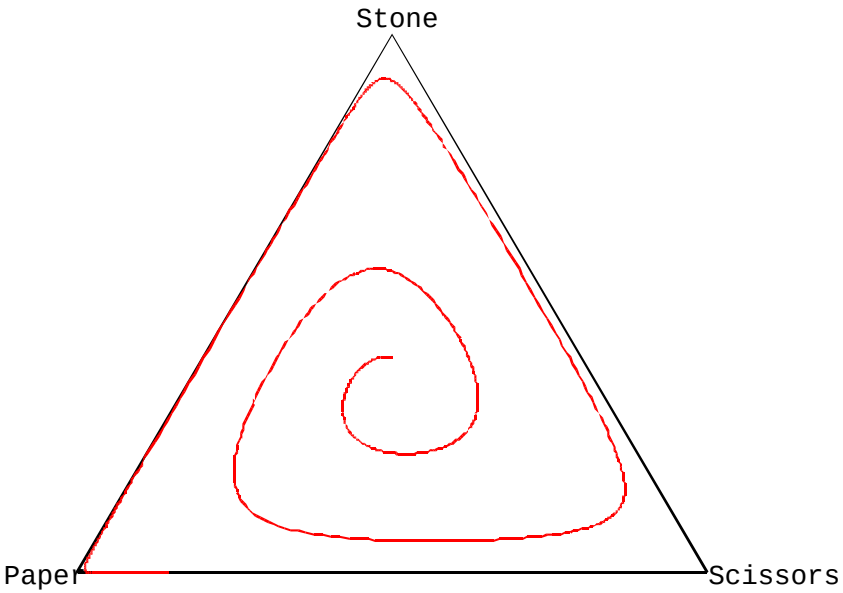
$$\mathbf{A} = \begin{bmatrix} 1 & 2+\epsilon & 0 \\ 0 & 1 & 2+\epsilon \\ 2+\epsilon & 0 & 1 \end{bmatrix}$$



- Where is the Nash equilibrium? What should one expect?
- Stable orbit for the standard case
- Convergent for $\epsilon > 0$
- Divergent for $\epsilon < 0$

Example: Paper-Scissors-Stone

$$\mathbf{A} = \begin{bmatrix} 1 & 2-\epsilon & 0 \\ 0 & 1 & 2-\epsilon \\ 2-\epsilon & 0 & 1 \end{bmatrix}$$



- Where is the Nash equilibrium? What should one expect?
- Stable orbit for the standard case
- Convergent for $\epsilon > 0$
- Divergent for $\epsilon < 0$

The replicator dynamics

$$\dot{x}_i = x_i \cdot \underbrace{\lambda(\mathbf{x})((\mathbf{A}\mathbf{x})_i - \mathbf{x} \cdot \mathbf{A}\mathbf{x})}_{=:g_i(x)}$$

Desirable properties:

- *payoff monotonicity:* $(\mathbf{A}\mathbf{x})_i > (\mathbf{A}\mathbf{x})_j \;\Leftrightarrow\; g_i(\mathbf{x}) > g_j(\mathbf{x})$
- *aggregate monotonicity:* $\mathbf{y} \cdot \mathbf{A}\mathbf{x} > \mathbf{z} \cdot \mathbf{A}\mathbf{x} \;\Leftrightarrow\; \mathbf{y} \cdot g(\mathbf{x}) > \mathbf{z} \cdot g(\mathbf{x})$

Any aggregate monotone dynamics can be represented in form of the replicator dynamics!

The replicator dynamics

$$\dot{x}_i = x_i \cdot \underbrace{\lambda(\mathbf{x})((\mathbf{A}\mathbf{x})_i - \mathbf{x} \cdot \mathbf{A}\mathbf{x})}_{=:g_i(x)}$$

Further observations:

- Replicator dynamics do not discover initially unused strategies.
- A strategy that is initially in the support will never die out completely.

In the following, we assume that all strategies are in the initial support and, hence, belong to the support at any time.

Evolutionary stability

Definition (evolutionary stable)

A strategy $\mathbf{x}$ is evolutionary stable if for any other strategy $\mathbf{y} \neq \mathbf{x}$ there exists an $\epsilon > 0$ such that for $\mathbf{w} = \epsilon\mathbf{y} + (1-\epsilon)\mathbf{x}$ it holds

$$\mathbf{xAw} > \mathbf{yAw} \ .$$

Proposition

A strategy $\mathbf{x}$ *is evolutionary stable if there exists* $\epsilon > 0$ *such that for every* y *with* $||y - x|| \leq \epsilon$ *it holds*

$$\mathbf{xAw} > \mathbf{yAw} \ .$$

Evolutionary stability

Definition (evolutionary stable)

A strategy $\mathbf{x}$ is evolutionary stable if for any other strategy $\mathbf{y} \neq \mathbf{x}$ there exists an $\epsilon > 0$ such that for $\mathbf{w} = \epsilon\mathbf{y} + (1-\epsilon)\mathbf{x}$ it holds

$$\mathbf{xAw} > \mathbf{yAw} \ .$$

Proposition

A strategy $\mathbf{x}$ *is evolutionary stable if there exists* $\epsilon > 0$ *such that for every* y *with* $||y - x|| \leq \epsilon$ *it holds*

$$\mathbf{xAw} > \mathbf{yAw} \ .$$

Theorem (Taylor and Jonker, 1987)

Evolutionary stability implies asymptotic stability in the local sense, that is, starting close at a Nash equilibrium, the replicatior dynamics converges towards the Nash equilibrium.

Proof by Luapunov method.

Selfish routing and evolutionary game theory

Single commodity flow – symmetric games:

- Every strategy corresponds to a path $p \in \mathcal{P}$
- Payoff corresponds to negative latency $-\ell_p(\mathbf{x})$
- Average latency is denoted by $\bar{\ell}(\mathbf{x})$

Behavioral model: Suppose agents change their strategy at a rate proportional to their latency, and adopt the strategy of another randomly selectd player.

This results in the following **network dynamics:**

$$\dot{x}_p = \lambda(\mathbf{x}) \cdot x_p \cdot (\bar{\ell}(\mathbf{x}) - \ell_p(\mathbf{x})) \ .$$

Selfish routing and evolutionary game theory

Single commodity flow – symmetric games:

- Every strategy corresponds to a path $p \in \mathcal{P}$
- Payoff corresponds to negative latency $-\ell_p(\mathbf{x})$
- Average latency is denoted by $\bar{\ell}(\mathbf{x})$

Behavioral model: Suppose agents change their strategy at a rate proportional to their latency, and adopt the strategy of another randomly selectd player.

This results in the following **network dynamics:**

$$\dot{x}_p = \lambda(\mathbf{x}) \cdot x_p \cdot (\bar{\ell}(\mathbf{x}) - \ell_p(\mathbf{x})) \ .$$

Selfish routing and evolutionary game theory

Single commodity flow – symmetric games:

- Every strategy corresponds to a path $p \in \mathcal{P}$
- Payoff corresponds to negative latency $-\ell_p(\mathbf{x})$
- Average latency is denoted by $\bar{\ell}(\mathbf{x})$

Behavioral model: Suppose agents change their strategy at a rate proportional to their latency, and adopt the strategy of another randomly selectd player.

This results in the following **network dynamics:**

$$\dot{x}_p = \lambda(\mathbf{x}) \cdot x_p \cdot (\bar{\ell}(\mathbf{x}) - \ell_p(\mathbf{x})) \ .$$

Let us assume strictly increasing latency functions.
Then there is a unique Wardrop equilibrium $\mathbf{x}^*$.

Lemma

For every $\mathbf{y} \neq \mathbf{x}^*$, it holds $\mathbf{x}^* \cdot \ell(\mathbf{y}) < \mathbf{y} \cdot \ell(\mathbf{y})$.

Corollary

The Wardrop equilibrium is evolutionary stable.

Proof of Lemma:

- In a Wardrop equilibrium $\mathbf{x}$, all latencies of used paths are equal, implying $\mathbf{x} \cdot \ell(\mathbf{x}) \leq \mathbf{y} \cdot \ell(\mathbf{x})$ and, hence,

$$\begin{aligned} \mathbf{y} \cdot \ell(\mathbf{y}) &\geq \mathbf{y} \cdot \ell(\mathbf{y}) + \mathbf{x} \cdot \ell(\mathbf{x}) - \mathbf{y} \cdot \ell(\mathbf{x}) \\ &= \mathbf{x} \cdot \ell(\mathbf{x}) + \mathbf{y} \cdot (\ell(\mathbf{y}) - \ell(\mathbf{x})) \\ &= \mathbf{x} \cdot \ell(\mathbf{x}) + \sum_{e \in E} y_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})). \end{aligned}$$

- Consider the system $\dot{y}_e = y_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x}))$ for all $e \in E$ that starts with strictly positive flow.

Proof of Lemma:

- In a Wardrop equilibrium $\mathbf{x}$, all latencies of used paths are equal, implying $\mathbf{x} \cdot \ell(\mathbf{x}) \leq \mathbf{y} \cdot \ell(\mathbf{x})$ and, hence,

$$\begin{aligned} \mathbf{y} \cdot \ell(\mathbf{y}) &\geq \mathbf{y} \cdot \ell(\mathbf{y}) + \mathbf{x} \cdot \ell(\mathbf{x}) - \mathbf{y} \cdot \ell(\mathbf{x}) \\ &= \mathbf{x} \cdot \ell(\mathbf{x}) + \mathbf{y} \cdot (\ell(\mathbf{y}) - \ell(\mathbf{x})) \\ &= \mathbf{x} \cdot \ell(\mathbf{x}) + \sum_{e \in E} y_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})). \end{aligned}$$

- For each edge e $y_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})) \geq x_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x}))$, at least once with strict inequality.

Proof of Lemma:

- In a Wardrop equilibrium $\mathbf{x}$, all latencies of used paths are equal, implying $\mathbf{x} \cdot \ell(\mathbf{x}) \leq \mathbf{y} \cdot \ell(\mathbf{x})$ and, hence,

$$\begin{aligned} \mathbf{y} \cdot \ell(\mathbf{y}) &\geq \mathbf{y} \cdot \ell(\mathbf{y}) + \mathbf{x} \cdot \ell(\mathbf{x}) - \mathbf{y} \cdot \ell(\mathbf{x}) \\ &= \mathbf{x} \cdot \ell(\mathbf{x}) + \mathbf{y} \cdot (\ell(\mathbf{y}) - \ell(\mathbf{x})) \\ &= \mathbf{x} \cdot \ell(\mathbf{x}) + \sum_{e \in E} y_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})). \end{aligned}$$

- For each edge e $y_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})) \geq x_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x}))$, at least once with strict inequality.

- Thus, $\mathbf{y} \cdot \ell(\mathbf{y}) > \mathbf{x} \cdot \ell(\mathbf{x}) + \sum_{e \in E} x_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})) = \mathbf{x} \cdot \ell(\mathbf{y})$. $\square$

Proof of Lemma:

- In a Wardrop equilibrium $\mathbf{x}$, all latencies of used paths are equal, implying $\mathbf{x} \cdot \ell(\mathbf{x}) \leq \mathbf{y} \cdot \ell(\mathbf{x})$ and, hence,

$$\begin{aligned} \mathbf{y} \cdot \ell(\mathbf{y}) &\geq \mathbf{y} \cdot \ell(\mathbf{y}) + \mathbf{x} \cdot \ell(\mathbf{x}) - \mathbf{y} \cdot \ell(\mathbf{x}) \\ &= \mathbf{x} \cdot \ell(\mathbf{x}) + \mathbf{y} \cdot (\ell(\mathbf{y}) - \ell(\mathbf{x})) \\ &= \mathbf{x} \cdot \ell(\mathbf{x}) + \sum_{e \in E} y_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})). \end{aligned}$$

- For each edge e $y_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})) \geq x_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x}))$, at least once with strict inequality.

- Thus, $\mathbf{y} \cdot \ell(\mathbf{y}) > \mathbf{x} \cdot \ell(\mathbf{x}) + \sum_{e \in E} x_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})) = \mathbf{x} \cdot \ell(\mathbf{y})$. $\square$

Proof of Lemma:

- In a Wardrop equilibrium $\mathbf{x}$, all latencies of used paths are equal, implying $\mathbf{x} \cdot \ell(\mathbf{x}) \leq \mathbf{y} \cdot \ell(\mathbf{x})$ and, hence,

$$\begin{aligned} \mathbf{y} \cdot \ell(\mathbf{y}) &\geq \mathbf{y} \cdot \ell(\mathbf{y}) + \mathbf{x} \cdot \ell(\mathbf{x}) - \mathbf{y} \cdot \ell(\mathbf{x}) \\ &= \mathbf{x} \cdot \ell(\mathbf{x}) + \mathbf{y} \cdot (\ell(\mathbf{y}) - \ell(\mathbf{x})) \\ &= \mathbf{x} \cdot \ell(\mathbf{x}) + \sum_{e \in E} y_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})). \end{aligned}$$

- For each edge e $y_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})) \geq x_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x}))$, at least once with strict inequality.

- Thus, $\mathbf{y} \cdot \ell(\mathbf{y}) > \mathbf{x} \cdot \ell(\mathbf{x}) + \sum_{e \in E} x_e(\ell_e(\mathbf{y}) - \ell_e(\mathbf{x})) = \mathbf{x} \cdot \ell(\mathbf{y})$. $\square$

The lemma implies asymptotical stability in the global sense.

Theorem

The networks dynamics converge to the Wardrop equilibrium $\mathbf{x}^*$ *starting from any point of the interior of the simplex.*

Proof by Luapunov method

- As Luapunov function use relative entropy

$$H_{\mathbf{x}^*}(\mathbf{y}) := \sum_i x_i^* \ln \frac{x_i^*}{y_i}$$

- Entropy is continuous and differentiable.
- $H_{\mathbf{x}^*}(\mathbf{x}^*) = 0$ and $H_{\mathbf{x}^*}(\mathbf{y}) > 0$, for all $\mathbf{y} \neq \mathbf{x}^*$.
- Derivative is $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = -\sum_i x_i^* \frac{\dot{y}_i}{y_i}$.
- Substituting the replicator dynamics yields the continuous function $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = \lambda(\mathbf{y}) \cdot (\mathbf{x}^* - \mathbf{y}) \cdot \ell(\mathbf{y})$.
- Our lemma shows that this term is negative. $\square$

Proof by Luapunov method

- As Luapunov function use relative entropy

$$H_{\mathbf{x}^*}(\mathbf{y}) := \sum_i x_i^* \ln \frac{x_i^*}{y_i}$$

- Entropy is continuous and differentiable.
- $H_{\mathbf{x}^*}(\mathbf{x}^*) = 0$ and $H_{\mathbf{x}^*}(\mathbf{y}) > 0$, for all $\mathbf{y} \neq \mathbf{x}^*$.
- Derivative is $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = -\sum_i x_i^* \frac{\dot{y}_i}{y_i}$.
- Substituting the replicator dynamics yields the continuous function $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = \lambda(\mathbf{y}) \cdot (\mathbf{x}^* - \mathbf{y}) \cdot \ell(\mathbf{y})$.
- Our lemma shows that this term is negative. $\square$

Proof by Luapunov method

- As Luapunov function use relative entropy

$$H_{\mathbf{x}^*}(\mathbf{y}) := \sum_i x_i^* \ln \frac{x_i^*}{y_i}$$

- Entropy is continuous and differentiable.
- $H_{\mathbf{x}^*}(\mathbf{x}^*) = 0$ and $H_{\mathbf{x}^*}(\mathbf{y}) > 0$, for all $\mathbf{y} \neq \mathbf{x}^*$.
- Derivative is $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = -\sum_i x_i^* \frac{\dot{y}_i}{y_i}$.
- Substituting the replicator dynamics yields the continuous function $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = \lambda(\mathbf{y}) \cdot (\mathbf{x}^* - \mathbf{y}) \cdot \ell(\mathbf{y})$.
- Our lemma shows that this term is negative. □

Proof by Luapunov method

- As Luapunov function use relative entropy

$$H_{\mathbf{x}^*}(\mathbf{y}) := \sum_i x_i^* \ln \frac{x_i^*}{y_i}$$

- Entropy is continuous and differentiable.
- $H_{\mathbf{x}^*}(\mathbf{x}^*) = 0$ and $H_{\mathbf{x}^*}(\mathbf{y}) > 0$, for all $\mathbf{y} \neq \mathbf{x}^*$.
- Derivative is $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = -\sum_i x_i^* \frac{\dot{y}_i}{y_i}$.
- Substituting the replicator dynamics yields the continuous function $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = \lambda(\mathbf{y}) \cdot (\mathbf{x}^* - \mathbf{y}) \cdot \ell(\mathbf{y})$.
- Our lemma shows that this term is negative. □

Proof by Luapunov method

- As Luapunov function use relative entropy

$$H_{\mathbf{x}^*}(\mathbf{y}) := \sum_i x_i^* \ln \frac{x_i^*}{y_i}$$

- Entropy is continuous and differentiable.
- $H_{\mathbf{x}^*}(\mathbf{x}^*) = 0$ and $H_{\mathbf{x}^*}(\mathbf{y}) > 0$, for all $\mathbf{y} \neq \mathbf{x}^*$.
- Derivative is $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = -\sum_i x_i^* \frac{\dot{y}_i}{y_i}$.
- Substituting the replicator dynamics yields the continuous function $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = \lambda(\mathbf{y}) \cdot (\mathbf{x}^* - \mathbf{y}) \cdot \ell(\mathbf{y})$.
- Our lemma shows that this term is negative. □

Proof by Luapunov method

- As Luapunov function use relative entropy

$$H_{\mathbf{x}^*}(\mathbf{y}) := \sum_i x_i^* \ln \frac{x_i^*}{y_i}$$

- Entropy is continuous and differentiable.
- $H_{\mathbf{x}^*}(\mathbf{x}^*) = 0$ and $H_{\mathbf{x}^*}(\mathbf{y}) > 0$, for all $\mathbf{y} \neq \mathbf{x}^*$.
- Derivative is $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = -\sum_i x_i^* \frac{\dot{y}_i}{y_i}$.
- Substituting the replicator dynamics yields the continuous function $\dot{H}_{\mathbf{x}^*}(\mathbf{y}) = \lambda(\mathbf{y}) \cdot (\mathbf{x}^* - \mathbf{y}) \cdot \ell(\mathbf{y})$.
- Our lemma shows that this term is negative. $\square$

Multicommodity Flow – Asymmetric Games

Recall: $\mathcal{P} = \mathcal{P}_1 \dot\cup \mathcal{P}_2 \dot\cup \cdots \dot\cup \mathcal{P}_k$

Definition (Asymmetric Network dynamics)
For $p \in \mathcal{P}_i$, set $\dot{x}_p = \lambda_i(\mathbf{x}) \cdot x_p \cdot (\bar{\ell}_i(\mathbf{x}) - \ell_p(\mathbf{x}))$.

Entropy argument works only if we assume that the dynamics for all commodities behave in exactly the same way, that is, if the leading λ-factors are identical for all commodities.

However, there is a Luapunov function that yields convergence also for heterogenous mix of dynamics.

Multicommodity Flow – Asymmetric Games

Recall: $\mathcal{P} = \mathcal{P}_1 \dot\cup \mathcal{P}_2 \dot\cup \cdots \dot\cup \mathcal{P}_k$

Definition (Asymmetric Network dynamics)
For $p \in \mathcal{P}_i$, set $\dot{x}_p = \lambda_i(\mathbf{x}) \cdot x_p \cdot (\bar{\ell}_i(\mathbf{x}) - \ell_p(\mathbf{x}))$.

Entropy argument works only if we assume that the dynamics for all commodities behave in exactly the same way, that is, if the leading λ-factors are identical for all commodities.

However, there is a Luapunov function that yields convergence also for heterogenous mix of dynamics.

Multicommodity Flow – Asymmetric Games

Recall: $\mathcal{P} = \mathcal{P}_1 \dot\cup \mathcal{P}_2 \dot\cup \cdots \dot\cup \mathcal{P}_k$

Definition (Asymmetric Network dynamics)
For $p \in \mathcal{P}_i$, set $\dot{x}_p = \lambda_i(\mathbf{x}) \cdot x_p \cdot (\bar{\ell}_i(\mathbf{x}) - \ell_p(\mathbf{x}))$.

Entropy argument works only if we assume that the dynamics for all commodities behave in exactly the same way, that is, if the leading λ-factors are identical for all commodities.

However, there is a Luapunov function that yields convergence also for heterogenous mix of dynamics.

Luapunov method based on Rosenthals potential function

$$\Phi(\mathbf{x}) := \left(\sum_{e \in E} \int_0^{x_e} \ell_e(s)\, ds \right)$$

- Calculate derivative and plug in replicator dynamics:

$$\dot{\Phi} = \sum_i \lambda_i(\mathbf{x}) \left(\bar{\ell}_i(\mathbf{x})^2 - \sum_{p \in \mathcal{P}_i} x_p \ell_p(\mathbf{x})^2 \right)$$

- This derivative is negative because of Jensen's inequality, unless $\mathbf{x}$ satisfies the Wardrop equilibrium property. □

Luapunov method based on Rosenthals potential function

$$\Phi(\mathbf{x}) := \left(\sum_{e \in E} \int_0^{x_e} \ell_e(s)\, ds \right)$$

- Calculate derivative and plug in replicator dynamics:

$$\dot{\Phi} = \sum_i \lambda_i(\mathbf{x}) \left(\bar{\ell}_i(\mathbf{x})^2 - \sum_{p \in \mathcal{P}_i} x_p \ell_p(\mathbf{x})^2 \right)$$

- This derivative is negative because of Jensen's inequality, unless $\mathbf{x}$ satisfies the Wardrop equilibrium property. □

Luapunov method based on Rosenthals potential function

$$\Phi(\mathbf{x}) := \left(\sum_{e \in E} \int_0^{x_e} \ell_e(s)\, ds \right)$$

- Calculate derivative and plug in replicator dynamics:

$$\dot{\Phi} = \sum_i \lambda_i(\mathbf{x}) \left(\bar{\ell}_i(\mathbf{x})^2 - \sum_{p \in \mathcal{P}_i} x_p \ell_p(\mathbf{x})^2 \right)$$

- This derivative is negative because of Jensen's inequality, unless $\mathbf{x}$ satisfies the Wardrop equilibrium property. □

Analyzing the time of convergence

Suppose every player relates its latency to the average latency within each commodity, and switches her strategy at a rate proportional to the ratio between her own latency and the average latency.

This gives the following variant of the network dynamics.

$$\dot{x}_p = x_p \cdot \frac{\bar{\ell}_i(\mathbf{x}) - \ell_p(\mathbf{x})}{\bar{\ell}_i(\mathbf{x})}$$

How long does it take to come close to equilibrium?

Analyzing the time of convergence

Suppose every player relates its latency to the average latency within each commodity, and switches her strategy at a rate proportional to the ratio between her own latency and the average latency.

This gives the following variant of the network dynamics.

$$\dot{x}_p = x_p \cdot \frac{\bar{\ell}_i(\mathbf{x}) - \ell_p(\mathbf{x})}{\bar{\ell}_i(\mathbf{x})}$$

How long does it take to come close to equilibrium?

Analyzing the time of convergence

Suppose every player relates its latency to the average latency within each commodity, and switches her strategy at a rate proportional to the ratio between her own latency and the average latency.

This gives the following variant of the network dynamics.

$$\dot{x}_p = x_p \cdot \frac{\bar{\ell}_i(\mathbf{x}) - \ell_p(\mathbf{x})}{\bar{\ell}_i(\mathbf{x})}$$

How long does it take to come close to equilibrium?

Approximate equilibria

- Need to define right notion of "approximate equilibrium".
- Euclidian distance not suitable here as very poor strategies, i.e., very slow paths, will never die out completely.

Definition (relaxed $(1+\epsilon)$-equilibrium)
A network game is in a relaxed $(1+\epsilon)$-equilibrium if less than an ϵ-fraction of the flow uses paths with latency larger than $(1+\epsilon)\cdot\bar{\ell}$.

Approximate equilibria

- Need to define right notion of "approximate equilibrium".
- Euclidian distance not suitable here as very poor strategies, i.e., very slow paths, will never die out completely.

Definition (relaxed $(1+\epsilon)$-equilibrium)
A network game is in a relaxed $(1+\epsilon)$-equilibrium if less than an ϵ-fraction of the flow uses paths with latency larger than $(1+\epsilon)\cdot\bar{\ell}$.

Approximate equilibria

- Need to define right notion of "approximate equilibrium".
- Euclidian distance not suitable here as very poor strategies, i.e., very slow paths, will never die out completely.

Definition (relaxed $(1+\epsilon)$-equilibrium)
A network game is in a relaxed $(1+\epsilon)$-equilibrium if less than an ϵ-fraction of the flow uses paths with latency larger than $(1+\epsilon)\cdot\bar{\ell}$.

Bounds on the time of convergence

Starting from any point of the simplex the network dynamics reaches a relaxed $(1+\epsilon)$-equilibrium in time

- $\mathcal{O}(\epsilon^{-3} \cdot \ln(\ell_{\max}/\ell^*))$, for symmetric games on general networks
- $\mathcal{O}(\epsilon^{-3} \cdot \ell_{\max}/\ell^*)$, for asymmetric games on general networks

There is a symmetric game on parallel links in which the time to reach an n ϵ-equilibrium is $\Omega(\epsilon^{-2} \cdot \ln(\ell_{\max}/\ell^*))$.

Bounds on the time of convergence

Starting from any point of the simplex the network dynamics reaches a relaxed $(1+\epsilon)$-equilibrium in time

- $\mathcal{O}(\epsilon^{-3} \cdot \ln(\ell_{\max}/\ell^*))$, for symmetric games on general networks
- $\mathcal{O}(\epsilon^{-3} \cdot \ell_{\max}/\ell^*)$, for asymmetric games on general networks

There is a symmetric game on parallel links in which the time to reach an n ϵ-equilibrium is $\Omega(\epsilon^{-2} \cdot \ln(\ell_{\max}/\ell^*))$.

Bounds on the time of convergence

Starting from any point of the simplex the network dynamics reaches a relaxed $(1+\epsilon)$-equilibrium in time

- $\mathcal{O}(\epsilon^{-3} \cdot \ln(\ell_{\max}/\ell^*))$, for symmetric games on general networks
- $\mathcal{O}(\epsilon^{-3} \cdot \ell_{\max}/\ell^*)$, for asymmetric games on general networks

There is a symmetric game on parallel links in which the time to reach an n ϵ-equilibrium is $\Omega(\epsilon^{-2} \cdot \ln(\ell_{\max}/\ell^*))$.

Summary

- different models for selfish routing
  - Wardrop's continuous traffic model
  - discrete server game
- price of anarchy
  - almost independent of the network structure
  - heaviliy depending on the class of latency functions
- congestion games
  - local improvement steps lead to pure Nash equilibria
  - symmetric network games solvable via min-cost flow
  - other variants are PLS-complete
- evolutionary games
  - adaption to selfish routing (joint work with Simon Fischer)
  - evolutionary stability implies asymptotic stability
  - very fast convergence in case of symmetric games
  - fast convergence for asymmetric games

Summary

- different models for selfish routing
  - Wardrop's continuous traffic model
  - discrete server game
- price of anarchy
  - almost independent of the network structure
  - heaviliy depending on the class of latency functions
- congestion games
  - local improvement steps lead to pure Nash equilibria
  - symmetric network games solvable via min-cost flow
  - other variants are PLS-complete
- evolutionary games
  - adaption to selfish routing (joint work with Simon Fischer)
  - evolutionary stability implies asymptotic stability
  - very fast convergence in case of symmetric games
  - fast convergence for asymmetric games

Summary

- different models for selfish routing
  - Wardrop's continuous traffic model
  - discrete server game
- price of anarchy
  - almost independent of the network structure
  - heaviliy depending on the class of latency functions
- congestion games
  - local improvement steps lead to pure Nash equilibria
  - symmetric network games solvable via min-cost flow
  - other variants are PLS-complete
- evolutionary games
  - adaption to selfish routing (joint work with Simon Fischer)
  - evolutionary stability implies asymptotic stability
  - very fast convergence in case of symmetric games
  - fast convergence for asymmetric games

Summary

- different models for selfish routing
  - Wardrop's continuous traffic model
  - discrete server game
- price of anarchy
  - almost independent of the network structure
  - heaviliy depending on the class of latency functions
- congestion games
  - local improvement steps lead to pure Nash equilibria
  - symmetric network games solvable via min-cost flow
  - other variants are PLS-complete
- evolutionary games
  - adaption to selfish routing (joint work with Simon Fischer)
  - evolutionary stability implies asymptotic stability
  - very fast convergence in case of symmetric games
  - fast convergence for asymmetric games

Thanks ...