Improving Selfish Routing

Algorithmic Game Theory '20

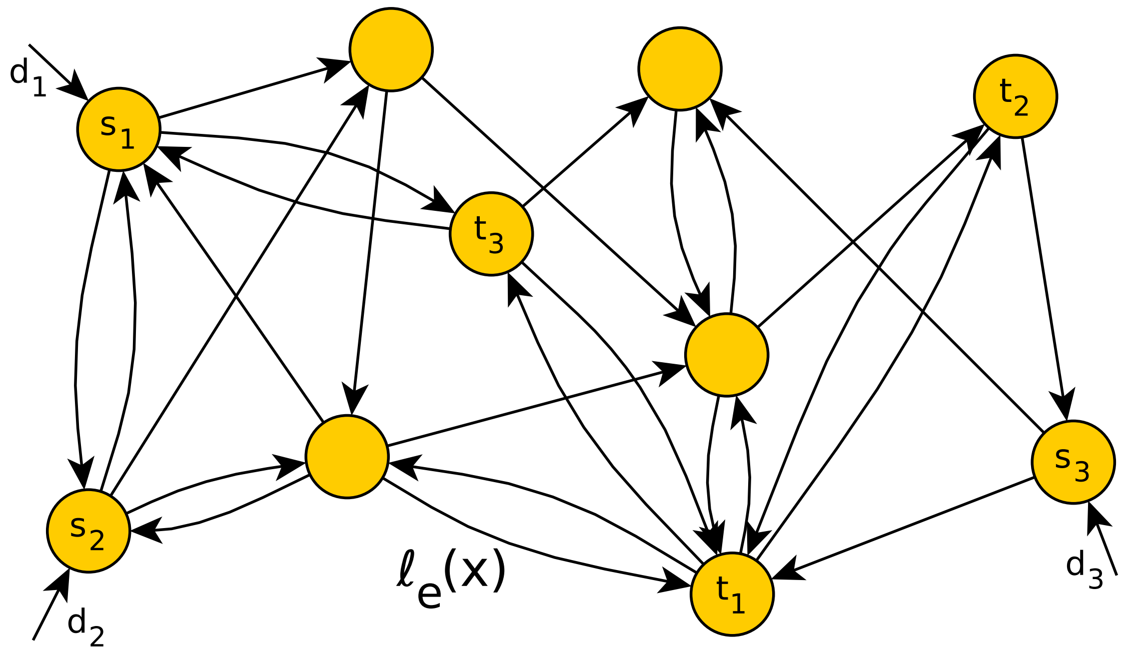
ΑΛΜΑ, ΣΗΜΜΥ

Selfish Routing

Selfish users traveling on a network

Selfish routing

Selfish users traveling on a network



- Graph $G = (V, E)$,
- Vertices $s_i, t_i \in V$,
- Edge functions $\ell_e(x)$
- Demands that consists of infinite infinitesimally small selfish players.

Users minimize their cost: $\ell_p(x) := \sum_{e \in p} \ell_e(x)$

Optimal and Equilibrium Flows

Social cost of flow x

$$SC(x) = \sum_p x_p \ell_p(x) = \sum_e x_e \ell_e(x_e)$$

Optimal flow, x^*

minimizes the social cost:

$$x^* = \arg \min_{x \text{ flow}} \{SC(x)\}$$

Equilibrium flow, f

For any commodity all positive flow paths have minimum costs. Property:

$$f = \arg \min_{x \text{ flow}} \quad \Phi(x) := \sum_{e \in E} \int_0^{x_e} \ell_e(x)\, dx$$

Optimal vs Equilibrium Flow

$$SC(x) = \sum_e x_e \ell_e(x_e) \text{ vs } \Phi(x) = \sum_{e \in E} \int_0^{x_e} \ell_e(x)\, dx$$

Feasible region:

$$\sum_{p \in P_i} x_p = d_i, \qquad \textit{commod.}$$

$$x_e = \sum_{p \in \mathcal{P}} x_p, \qquad \textit{edges}$$

$$\sum_{e \in \delta^-(u)} x_e = \sum_{e \in \delta^+(u)} x_e, \quad \textit{nodes}$$

$$x_{s_i} \in \delta^-(s_i),\ x_{t_i} \in \delta^+(t_i)$$

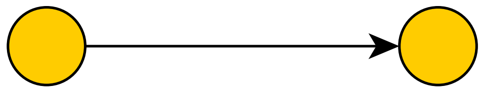
Variational Inequality → PoA bound

$$\sum_e f_e \ell_e(f_e) \leq \sum_e x^*_e \ell_e(f_e) = \sum_e x^*_e \ell_e(x^*_e) + \sum_e x^*_e(\ell_e(f_e) - \ell_e(x^*_e))$$

$$\leq \sum_e x^*_e \ell_e(x^*_e) + \beta(\mathcal{L}) \sum_e f_e \ell_e(f_e) \Rightarrow PoA(\mathcal{L}) \leq \frac{1}{1 - \beta(\mathcal{L})}$$

The Power of Tolls

Introducing tolls on edges:



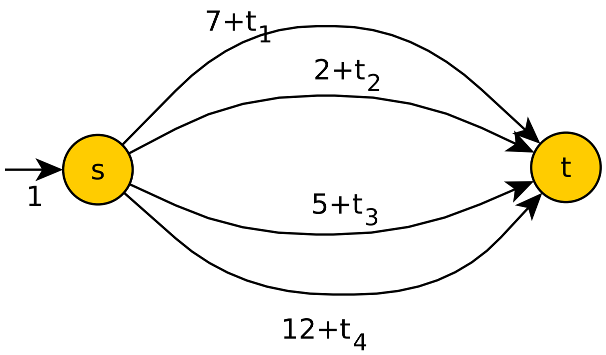
- Each user now minimizes $\ell_p(x) + \sum_{e \in p} t_e$
- Users' equilibrium minimizes

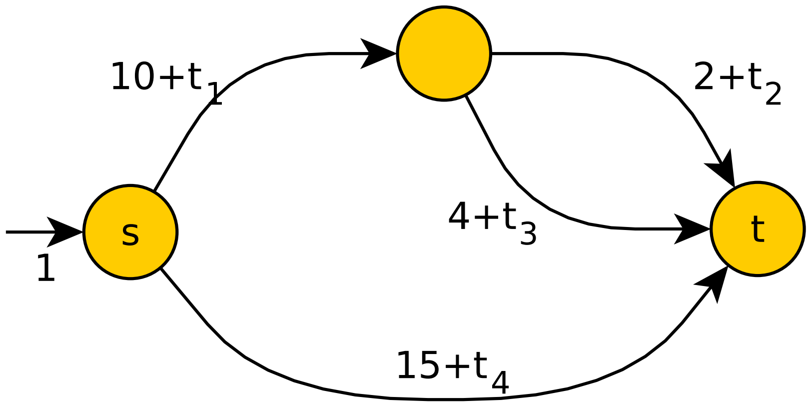
$$x(t) = \arg \min_{y \text{ flow}} \quad \Phi_t(y) := \sum_{e \in E} \int_0^{y_e} (\ell_e(y) + t_e)\, dy$$

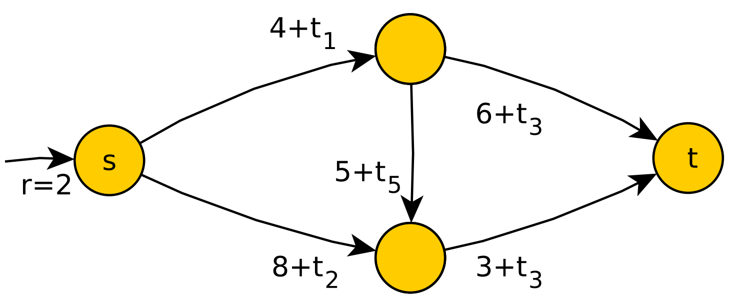
- Marginal tolls, i.e. $\hat{t}_e := x_e^* \ell_e'(x_e^*)$, are optimal:

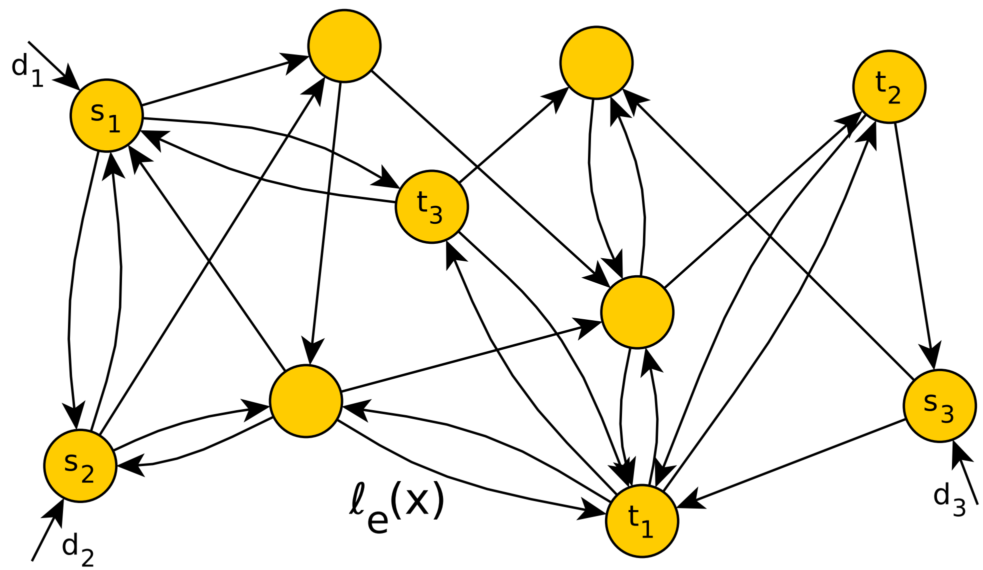
$$x^* = x(\hat{t}) = \arg \min_{y \text{ flow}} \quad \sum_{e \in E} \int_0^{y_e} (\ell_e(y) + \hat{t}_e)\, dy$$

Uniqueness of Tolls?









Uniqueness of Tolls?

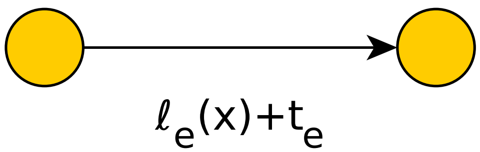
Goal: Minimize the payments while inducing the optimal flow at NE.

$$\min \sum_{e \in E} x_e^* t_e$$

$$\begin{aligned}
\nu_u - \nu_v + t_e &= -\ell_e(x_e^*) \quad \forall e = (u, v) : x_e^* > 0 \\
\nu_u - \nu_v + t_e &\geq -\ell_e(x_e^*), \quad \forall e = (u, v) : x_e^* = 0 \\
t &\geq 0
\end{aligned}$$

Tolls for Heterogeneous Users

Introducing tolls on edges:



- User of sensitivity a_i minimizes $\ell_p(x) + a_i \sum_{e \in p} t_e$
  (or $\frac{1}{a_i}\ell_p(x) + \sum_{e \in p} t_e$)
- Users' equilibrium minimizes ????
- Marginal tolls are no more optimal (in general)

A Magic LP

Let g be a flow to be enforced.

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_i a_i \sum_{p \in P_i} c_p(g) f_p^i \\ \text{so that} \quad & \\ \forall e \in E: \quad & \textstyle\sum_i \sum_{p \in P: e \in p} f_p^i \le g_e & (1) \\ \forall i: \quad & \textstyle\sum_{p \in P_i} f_p^i = d_i & (2) \\ \forall i \forall p \in P_i: \quad & f_p^i \ge 0 & (3) \end{aligned}$$

$$\begin{aligned} \text{maximize} \quad & \textstyle\sum_i d_i z_i - \sum_{e \in E} g_e t_e \\ \text{so that} \quad & \\ \forall i \forall p \in P_i: \quad & z_i - \textstyle\sum_{e \in p} t_e \le a_i c_p(g) \\ \forall e \in E: \quad & t_e \ge 0 \end{aligned}$$

- (feasible) g is minimal if inequality 1 is tight (for all e)
- g is enforceable if there are tolls to enforce it on equilibrium.

g minimal iff g enforceable

"$\Rightarrow$": $f_e = g_e$ and $f_p^i > 0 \Rightarrow z_i = a_i c_p(g) + \sum_{e \in p} t_e$

"$\Leftarrow$": There are tolls for which g is Nash, thus
$g_p^i > 0 \Rightarrow z_i := a_i c_p(g) + \sum_{e \in p} t_e$
$\Rightarrow$ g and (z,t) complementary

A Detail and Generalizations

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_i a_i \sum_{p \in P_i} c_p(g) f_p^i & \\ \text{so that} \quad & & \\ \forall e \in E: \quad & \textstyle\sum_i \sum_{p \in P: e \in p} f_p^i \leq g_e & (1) \\ \forall i: \quad & \textstyle\sum_{p \in P_i} f_p^i = d_i & (2) \\ \forall i \forall p \in P_i: \quad & f_p^i \geq 0 & (3) \end{aligned}$$

$$\begin{aligned} \text{maximize} \quad & \textstyle\sum_i d_i z_i - \sum_{e \in E} g_e t_e \\ \text{so that} \quad & \\ \forall i \forall p \in P_i: \quad & z_i - \textstyle\sum_{e \in p} t_e \leq a_i c_p(g) \\ \forall e \in E: \quad & t_e \geq 0 \end{aligned}$$

Is optimal g minimal??

If not, reduce g_e's up to right before losing feasibility: $C(g^*) \leq C(g)$

Generalizations:

- g can minimize any non-decreasing function, not only the Social Cost
- player specific latencies
- proves existence of tolls for continuous heterogeneity

Other Toll Directions

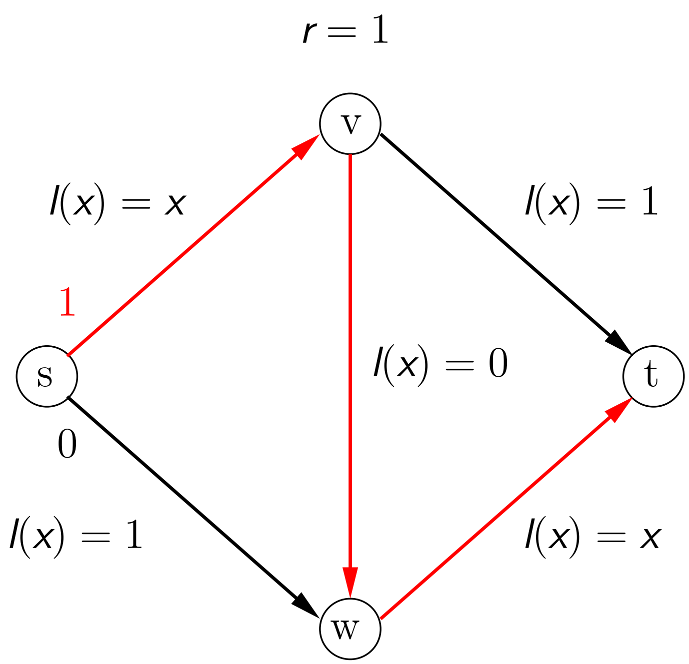
- Tolls affect the Social Cost
- Upper bounds on the tolls
- Use tolls on the minimum number of edges
- Profit maximizers operate tolls
    - Existence of equilibria?
    - Optimality?

And of course atomic players!!

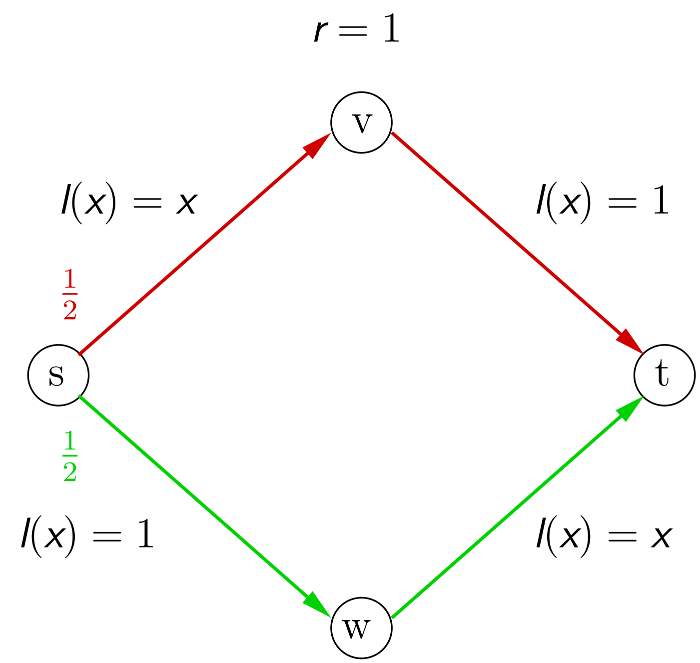
Braess Paradox and Network Design

- Problem: route traffic in a network of selfish non-cooperative players.
- Motivation: simple examples show that Nash equilibria can be inefficient (Price of Anarchy).
- Question: which subnetwork will exhibit the best performance when used selfishly?

Braess's Paradox



$$L = 1 + 0 + 1 = 2$$



$$L = \frac{1}{2} + 1 = \frac{3}{2}$$

Formalizing our Problem

Problem

Given an instance (G, r, l), find a subgraph H of G that minimizes $L(H, r, l)$.

Properties of Nash Flows

Lemma

For every instance (G, r, l), $L(G, r, l)$ *is a non-decreasing function of* r.

Lemma

Let f *be a flow feasible for* (G, r, l). *For a vertex* v *in* G, *let* $d(v)$ *denote the length, with respect to edge lengths* $\{l_e(f_e)\}_{e \in E}$ *of a shortest* $s - v$ *path in* G. *Then* f *is at Nash equilibrium iff*

$$d(w) - d(v) \leq l_e(f_e)$$

for all edges $e = (v, w)$, *with equality holding whenever* $f_e > 0$.

Lemma

If f *is a flow at NE for* (G, r, l), *then* $C(f) = r \cdot L(G, r, l)$.

Linear Latency Functions

We consider latency functions of the form $l_e(x) = a_e x + b_e$, $a_e, b_e \geq 0$. We then call the problem the LINEAR LATENCY NETWORK DESIGN. It is known that the price of anarchy in such networks is at most $\frac{4}{3}$.

Trivial Algorithm

Given an instance (G, r, l), build the whole network G.

Lemma (Roughgarden - Tardos)

Let f^* and f be feasible and Nash flows, respectively, for an instance (G, r, l) with linear latency functions. Then,

$$C(f) \leq \frac{4}{3} \cdot C(f^*).$$

Upper Bound of Trivial Algorithm

Corollary

The trivial algorithm is a $\frac{4}{3}$-approximation algorithm for LINEAR LATENCY NETWORK DESIGN.

Απόδειξη.

- Let H be the subgraph that minimizes $L(H, r, l)$, and f and f^* be the flows at NE for (G, r, l) and (H, r, l).
- $C(f) = r \cdot L(G, r, l)$ and $C(f^*) = r \cdot L(H, r, l)$.
- f^* feasible for (G, r, l), thus $C(f) \leq \frac{4}{3} C(f^*)$.
- Hence, $L(G, r, l) \leq \frac{4}{3} L(H, r, l)$.

□

Optimality of the Trivial Algorithm (1 / 3)

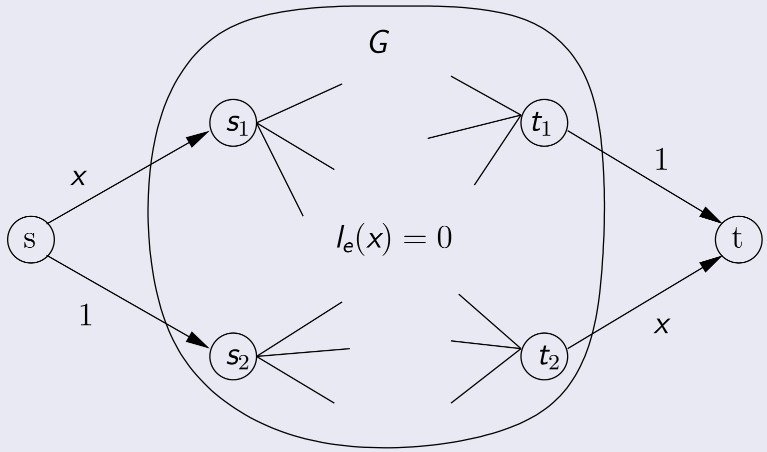
Theorem

For every $\epsilon > 0$, there is no $\frac{4}{3} - \epsilon$ -approximation algorithm for LINEAR LATENCY NETWORK DESIGN, assuming $\mathrm{P} \neq \mathrm{NP}$.

We will use a reduction from the 2 DIRECTED DISJOINT PATHS (2DDP) problem: given a directed graph $G = (V, E)$ and distinct vertices $s_1, s_2, t_1, t_2 \in V$, are there $s_i - t_i$ paths P_i for $i = 1, 2$, such that P_1 and P_2 are vertex-disjoint?
2DDP is NP-complete.

Optimality of the Trivial Algorithm (2 / 3)

Απόδειξη.



$(G', 1, l)$

- If algorithm returns a subgraph H with $L(H, 1, l) < 2$, then "yes" instance of 2DDP, else "no".
- If "yes" instance, let P_1 and P_2 be vertext disjoint $s_1 - t_1$ and $s_2 - t_2$ paths. Obtain H by deleting all other edges. Observe now that $L(H, 1, l) = \frac{3}{2}$ (1/2 routed in $s_1 \to t_1 \to t$ and 1/2 in $s_2 \to t_2 \to t$). So, $ALG \leq \left(\frac{4}{3} - \epsilon\right) \cdot \frac{3}{2} < 2$.

Optimality of the Trivial Algorithm (3 / 3)

Proof (continued).

- We will prove that if "no" instance, then $L(H, 1, l) \geq 2$ for all subgraphs of G', and so $ALG \geq 2$.
- Split subgraphs of G' in 3 groups: (i) those with an $s_2 - t_1$ path, (ii) those with an $s_1 - t_2$ path and (iii) those with an $s_i - t_i$ path for exactly one $i \in \{1, 2\}$.
- In all cases, routing flow in such a path gives NE and $L(H) = 2$.
- Thus, $ALG \geq OPT \geq 2$, and so we solve 2DDP.

□

Interpretation of Results

- Efficiently detecting Braess's Paradox in networks with linear latency functions is impossible (i.e. NP-hard). This holds even in the most severe cases, where $PoA = \frac{4}{3}$.
- However, by restricting our linear latency functions only to strictly increasing ones, we can get positive results!

Towards some Positive Results

For instances with strictly increasing linear latencies, the optimal flow is **unique** and can be efficiently computed.

Definition

An instance (G, r, l) is called *paradox-free* if for every subgraph H of G, $L(H, r, l) \geq L(G, r, l)$. An instance (G, r, l) is called *paradox-ridden* if there is a subgraph H of G, such that
$L(H, r, l) = L^*(G, r, l) = L(G, r, l)/PoA(G, r, l) \leq L(G, r, l)$.

Note: In a paradox-free instance PoA cannot be improved by edge removal.

Lemma

An instance (G, r, l) with $G = (V, E)$ is paradox-ridden iff there is an optimal flow f^* that is a Nash flow on the subgraph $G^*(V, E^*)$, where
$E^* = \{e \in E : f_e^* > 0\}$.

Detecting Paradox-Ridden Networks

Theorem (Fotakis, Kaporis, Spirakis)

Given an instance (G, r, l) with strictly increasing linear latencies, one can decide in polynomial time whether the instance is paradox-ridden or not.

Απόδειξη.

- We can efficiently compute the *unique* optimal flow f^*.
- We then compute the length $d(v)$ of a shortest $s - v$ path wrt edge lengths $\{l_e(f_e^*)\}_{e \in E^*}$ for all $v \in V$.
- f^* Nash flow $\Leftrightarrow \forall (u, v) \in E^*, d(v) = d(u) + l_{(u,v)}(f^*_{(u,v)})$.

□

Towards a Positive Result for Arbitrary Linear Latencies

- As already stated, we cannot decide whether an instance with arbitrary linear latencies is paradox-ridden or not.
- However, we can reach sufficient conditions under which we can answer the above question.
- Let (G, r, l) be an instance with $l_e(x) = a_e(x) + b_e$ and $E^c = \{e \in E : a_e = 0\}$. Let $E^i = E \setminus E^c$ and let $\mathcal{O}$ be the set of optimal flows.

Note: All optimal flows assign the same traffic to the edges with strictly increasing latencies, and can differ only on edges with constant latencies. This motivates the following LP formulation, given a **fixed** optimal flow o.

An LP formulation

(LP):

$$\min \sum_{e\in E^c} f_e b_e, \quad s.t.:$$

$$\sum_{u:(v,u)\in E^i} o_{(v,u)} + \sum_{u:(v,u)\in E^c} f_{(v,u)} = \sum_{u:(u,v)\in E^i} o_{(u,v)} + \sum_{u:(u,v)\in E^c} f_{(u,v)} \qquad \forall v \in V\setminus\{s,t\},$$

$$\sum_{u:(s,u)\in E^i} o_{(s,u)} + \sum_{u:(s,u)\in E^c} f_{(s,u)} = r,$$

$$\sum_{u:(u,t)\in E^i} o_{(u,t)} + \sum_{u:(u,t)\in E^c} f_{(u,t)} = r,$$

$$f_e \geq 0 \qquad \forall e \in E^c.$$

Notes on the LP

- An optimal solution to (LP) corresponds to a feasible flow for (G, r, l) that agrees with o on all edges in E^i and allocates traffic to the edges in E^c so that the total latency is minimized.
- Optimal solutions to (LP) $\leftrightarrow$ $1-1$ Optimal flows in $\mathcal{O}$.
- Given an optimal flow o, the problem of checking if there is a $o \in \mathcal{O}$ that is a Nash flow on G_o reduces to the problem of generating all optimal solutions of (LP) and checking whether some of them can be translated into a Nash flow on the corresponding subnetwork.
- This can be performed in polynomial time if (LP)'s optimal solution is unique.

A Positive Result for Arbitrary Linear Latencies (1 / 2)

Theorem

Given an instance (G, r, l) with arbitrary linear latencies where the corresponding (LP) has a unique optimal solution, one can decide in polynomial time whether the instance is paradox-ridden or not.

Note: In fact, it suffices to generate all optimal basic feasible solutions, as the (LP) allocates traffic to constant latency edges. Observe that if a feasible flow f is a Nash flow, then any solution f' with $\{e : f'_e > 0\} \subseteq \{e : f_e > 0\}$ is a Nash flow, too.

A Positive Result for Arbitrary Linear Latencies (2 / 2)

Theorem

Given an instance (G, r, l) with arbitrary linear latencies where the corresponding (LP) has a polynomial number of basic feasible solutions, one can decide in polynomial time whether the instance is paradox-ridden or not.

Note: The above class includes instances with a constant number of constant latency edges.

Finding Near-Optimal Subnetworks

- In general, finding optimal subnetworks in paradox-ridden instances is NP-hard.
- However, we can reach a subexponential-time approximation scheme on networks with polynomially many paths, each of polylogarithmic length.
- For this purpose, we need to turn our attention to "sparse" flows and ε-Nash flows.

Definition (ε-Nash flow)

For some $\varepsilon > 0$, a flow f is an ε-Nash flow if for every path P and P' with $f_P > 0$, $l_P(f) \leq l_{P'}(f) + \varepsilon$.

Making a Flow "Sparse" (1 / 3)

Lemma (Fotakis, Kaporis, Spirakis)

Let $(G, 1, l)$ *be an instance on a graph* $G = (V, E)$*, and let* f *be any feasible flow. For any* $\varepsilon > 0$*, there exists a feasible flow* $\tilde{f}$ *that assigns positive traffic to at most* $\lfloor \log(2m)/(2\varepsilon^2) \rfloor + 1$ *paths, such that* $|\tilde{f}_e - f_e| \le \varepsilon, \ \forall e \in E$*.*

Απόδειξη.

- Let $\mu = |\mathcal{P}|$, and we index the $s - t$ paths by integers in $[\mu]$.
- Flow f can be seen as a probability distribution on $\mathcal{P}$.
- We prove that if we select $k > \log(2m)/(2\varepsilon^2)$ paths uniformly at random with replacement according to f, and assign to each path j a flow equal to the number of times j is selected divided by k, we obtain a flow that is an ε-approximation to f with positive probability. (Probabilistic Method)

Making a Flow "Sparse" (2 / 3)

Proof (continued).

- Fix ε and let $k = \lfloor \log(2m)/(2\varepsilon^2) \rfloor + 1$.
- Define random variables $P_1, \dots P_k \in [\mu]$, i.i.d., such that $\mathrm{P}[P_i = j] = f_j$.
- For each path $j \in [\mu]$, $F_j = |\{i \in [k] : P_i = j\}| \,/\, k$. Note that $\mathbf{E}[F_j] = f_j$.
- For each edge e and random variable P_i, define the independent indicator variables $F_{e,i} = 1$ if e in path P_i, otherwise 0.
- Let $F_e = \frac{1}{k} \sum_{i=1}^{k} F_{e,i}$. Observe that $F_e = \sum_{j:e \in j} F_j$ and $\mathbf{E}[F_e] = f_e$.

Making a Flow "Sparse" (3 / 3)

Proof (continued).

- Note that $\sum_{j=1}^{\mu} F_j = 1$. Thus, $F_1, \ldots, F_\mu$ define a feasible flow that assignes positive traffic to at most k paths and "agrees" with f on expectation.
- By the Chernoff-Hoeffding bound we get that for every edge e

$$\mathrm{P}[|F_e - f_e| > \varepsilon] \leq 2e^{-2\varepsilon^2 k} < 1/m$$

- Thus, by union bound, $\mathrm{P}[\exists e : |F_e - f_e| > \varepsilon] < m(1/m) = 1$.
- So, there is positive probability that the flow $(F_1, \ldots, F_\mu)$ satisfies $|F_e - f_e| \leq \varepsilon$, $\forall e \in E$. Thus, there exists a flow $\tilde{f}$ with the properties of $(F_1, \ldots, F_\mu)$.

□

Finding a Near-Optimal Subnetwork

Theorem

Let $(G(V,E), 1, \{a_e x + b_e\}_{e \in E})$ be an instance, $\alpha = \max_{e \in E}\{a_e\}$, and let H^B be the best subnetwork of G. For some constants $d_1, d_2 > 0$, let $|\mathcal{P}| \leq m^{d_1}$ and $|P| \leq \log^{d_2} m$, for all $P \in \mathcal{P}$. Then, for any $\varepsilon > 0$, we can compute in time $m^{O(d_1 \alpha^2 \log^{2d_2+1}(2m)/\varepsilon^2)}$ a flow $\tilde{f}$ that is an ε-Nash flow on $G_{\tilde{f}}$ and satisfies $l_P(\tilde{f}) \leq L(H^B, 1, \{a_e x + b_e\}_{e \in E(H^B)}) + \varepsilon/2$, for all paths $P \in G_{\tilde{f}}$.

Another "Sparse" Flow

Theorem (Barman: approximate version of Caratheodory's Theorem)

Let X be a set of vectors $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ and $\epsilon > 0$.
For every $\mu \in conv(X)$ and $2 \leq p \leq \inf$ there exist an $O(\frac{p\gamma^2}{\epsilon^2})$ uniform vector $\mu' \in conv(X)$ such that $||\mu - \mu'||_p \leq \epsilon$, where $\gamma = \max_{x \in X} ||x||_p$.

How to apply:

- Let X be the set of different paths described by an "edge"vector: path containing e_1, e_2 and e_6 out of 7 edges corresponds to $(1, 1, 0, 0, 0, 1, 0)$.
- Any flow can be seen as a convex combination of the x_i's and vice versa.
- There are $|X|^{O(\frac{p\gamma^2}{\epsilon^2})}$ different $O(\frac{p\gamma^2}{\epsilon^2})$ uniform vectors.
- Enumerate, evaluate and keep the one of lowest cost

General Latency Functions

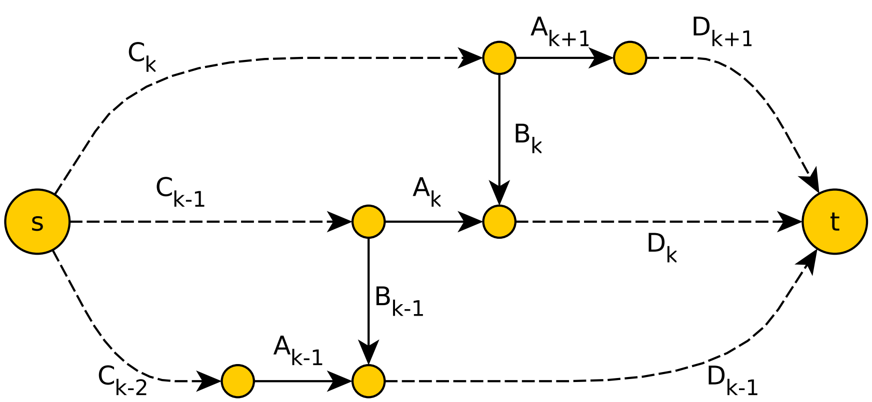
- We will now consider general (continuous, non-decreasing) latency functions (we call this problem the GENERAL LATENCY NETWORK DESIGN).
- We will see that the trivial algorithm is still the best thing we can do. However, the approximation factor gets worse.
- In order to prove the above, we will need new techniques, as in networks with general latency functions, a Nash flow can be arbitrarily more costly than other feasible flows.

The Trivial Algorithm for GENERAL LATENCY NETWORK DESIGN

Theorem

The trivial algorithm is a $\lfloor n/2 \rfloor$-approximation algorithm for GENERAL LATENCY NETWORK DESIGN.

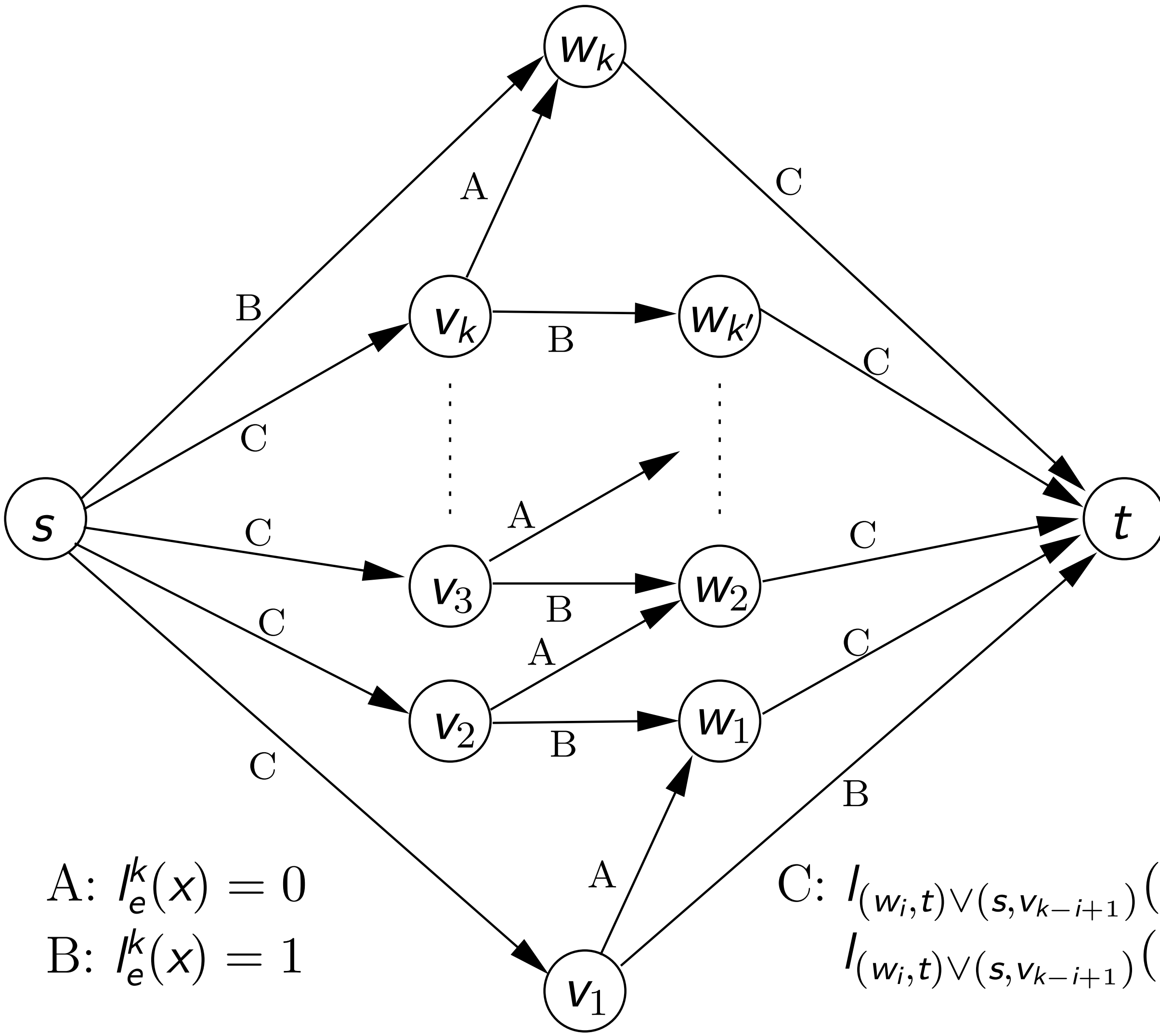
- f Nash flow, o best subnetwork's Nash flow
- $A = \{e : o_e \geq f_e\}$ and $B = \{e : o_e < f_e\}$



For the cost of f:

- $C_k^f + B_k^f \leq C_{k-1}^f + A_k^f$
- $C_k^f + A_{k+1}^f \leq \sum_i A_i^f - \sum_i B_i^f$
- $L(f) \leq C_m^f + A_{m+1}^f \leq \sum_i A_i^f \leq \sum_i A_i^o \leq \frac{n}{2} L(o)$

Tightness of the $\lfloor n/2 \rfloor$ bound: the B^k Braess Graph



A: $l_e^k(x) = 0$
B: $l_e^k(x) = 1$

C: $l_{(w_i,t)\vee(s,v_{k-i+1})}(k/(k+1)) = 0$
$l_{(w_i,t)\vee(s,v_{k-i+1})}(1) = i$

Tightness of the $\lfloor n/2 \rfloor$ bound (1 / 2)

Theorem

For every integer $n \geq 2$, there is an instance (G, r, l) in which G has n vertices and a subgraph H with

$$L(G, r, l) = \left\lfloor \frac{n}{2} \right\rfloor \cdot L(H, r, l).$$

Απόδειξη.

- Assume that $n \geq 4$ is even (otherwise, add an isolated vertex).
- So, $n = 2k + 2$ and we consider the instance (B^k, k, l^k).
- NE for (B^k, k, l^k): 1 unit on each path $s \to v_i \to w_i \to t$, and $L(B^k, k, l^k) = k + 1$.

Tightness of the $\lfloor n/2 \rfloor$ bound (2 / 2)

Proof (continued).

- We now remove all A-type edges and obtain H.
- Routing $k/(k+1)$ units on paths $s \to v_1 \to t$, $s \to w_k \to t$ and $\{s \to v_i \to w_{i-1} \to t\}_{(i=2,\ldots,k)}$, we get a NE with $L(H, k, l^k) = 1$.
- Thus, $L(G)/L(H) = k + 1 = n/2$.

□

Hardness of approximation for GENERAL LATENCY NETWORK DESIGN

Theorem (Roughgarden)

For every $\epsilon > 0$, there is no $(\lfloor n/2 \rfloor - \epsilon)$-approximation algorithm for GENERAL LATENCY NETWORK DESIGN, assuming $\mathrm{P} \neq \mathrm{NP}$.

Proof is based on a reduction from the NP-complete problem PARTITION.

How often does Braess's paradox occur?

Is Braess's paradox often in practical networks or is it just a theoretical curiosity?

Braess Paradox in real life

- Stuttgart Germany - In 1969 a newly constructed road worsened traffic. Traffic improved when the road was closed.
- New York City - Earth Day 1990 Traffic improved when 42nd St was closed
- Seoul, Korea - A 6 lane road that was perpetually jammed was closed and removed, traffic improved.

Valiant and Roughgarden: occurs in many networks by utilizing random graph models.

The model

- Probability distribution oven graphs and edge latency functions.
- Graph G distributed according to the standard Erdös-Renyi $\mathcal{G}(n, p)$ model. For a fixed $n \geq 2$, each edge is present independently with probability p. We assume that $p = \Omega(n^{-1/2+\epsilon})$ for some $\epsilon > 0$.
- Source s and destination t are chosen randomly or arbitrarily. (we assume that there is no edge (s, t)).
- Linear latency functions $l(x) = ax + b$, $a, b \geq 0$:
  1. *Independent coefficients* model: two fixed distributions $\mathcal{A}$ and $\mathcal{B}$, and each edge is independently given a latency function $l(x) = ax + b$, where a and b are drawn independently from $\mathcal{A}$ and $\mathcal{B}$, respectively.
  2. 1/x model: each edge present in the graph (independently) has the latency function $l(x) = x$ with probability q and $l(x) = 1$ with probability $1 - q$.

Main results

Theorem (Independent coefficients model)

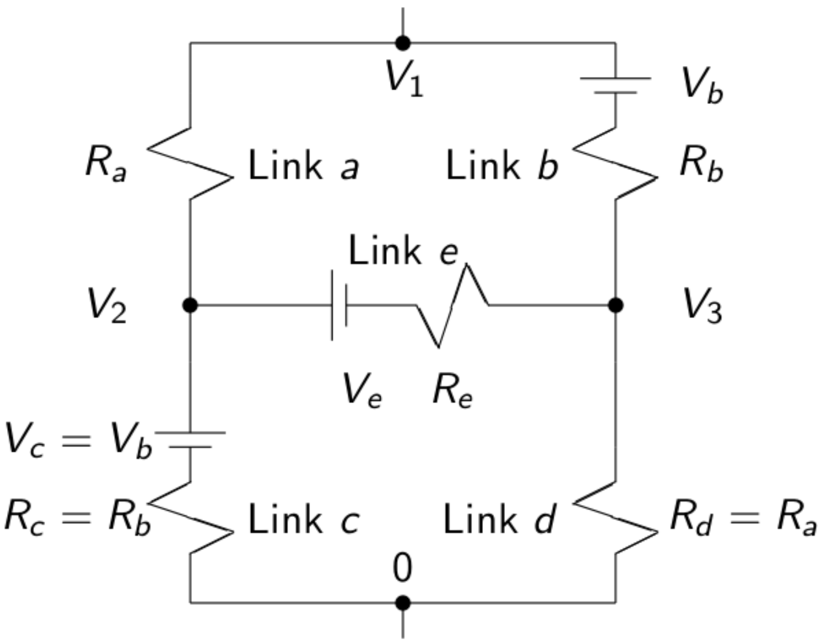
Let $\mathcal{A}$ and $\mathcal{B}$ be reasonable distributions. There is a constant $p = p(\mathcal{A}, \mathcal{B}) > 1$ such that, with high probability, a random network (G, l) admits a choice of traffic rate r such that the Braess ration of the instance (G, r, l) is at least p.

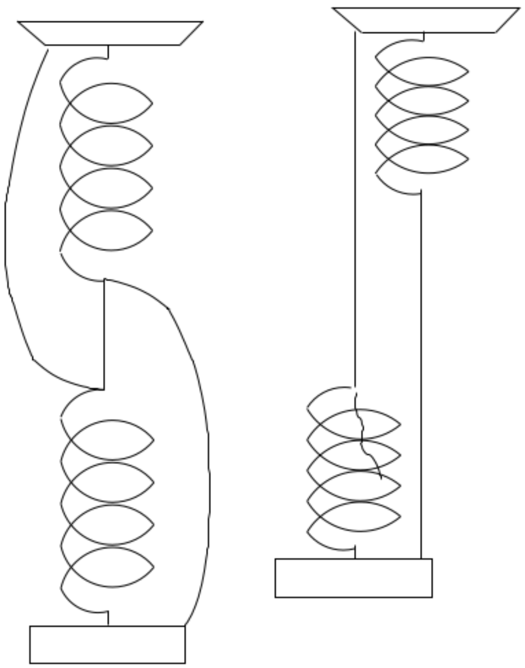
Theorem (The 1/x model)

There is a traffic rate $R = R(n, p, q)$ such that, with high probability as $n \to \infty$, the Braess ratio of a random n-node network from $\mathcal{G}(n, p, q)$ with traffic rate R is at least

$$\frac{4 - 3pq}{3 - 2pq}.$$

Braess Paradox Everywhere!





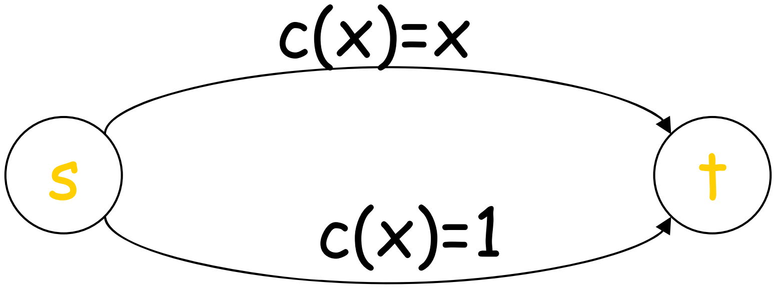
(a) Before (b) After

Stackelberg Routing

- In (classic) selfish routing all players act **selfishly**.
- In Stackelberg routing there exist players willing to **cooperate** for social welfare (a fraction of the total players).
  - Both Selfish and Cooperative players are present.
  - Leader determines the paths of the coordinated players.
  - Selfish players (followers) minimize their own cost.
- **Nash Equilibria** are considered as the possible outcomes of the game.
- A **Stackelberg Strategy** is an algorithm that allocates paths to coordinated players so as to lead selfish players to a good Nash Equilibrium.

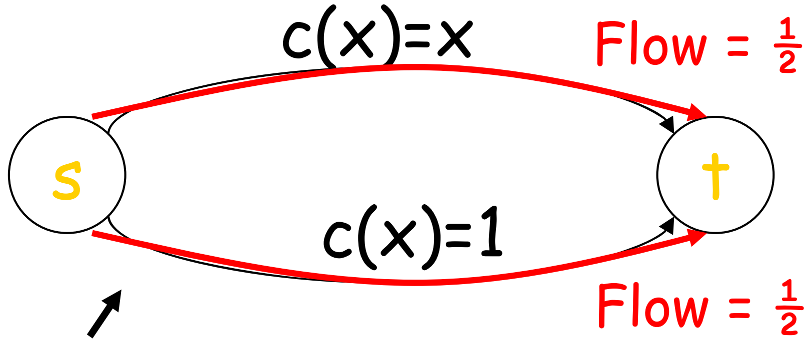
Example: Pigou’s Network

One unit of flow is to be routed from s to t



Example: Pigou's Network

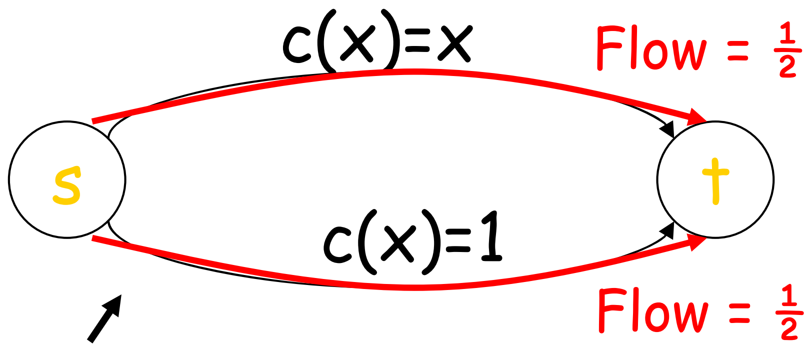
One unit of flow is to be routed from s to t



Optimal flow

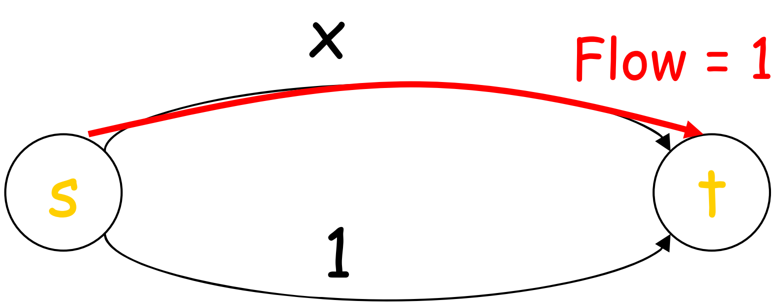
Example: Pigou's Network

One unit of flow is to be routed from s to t



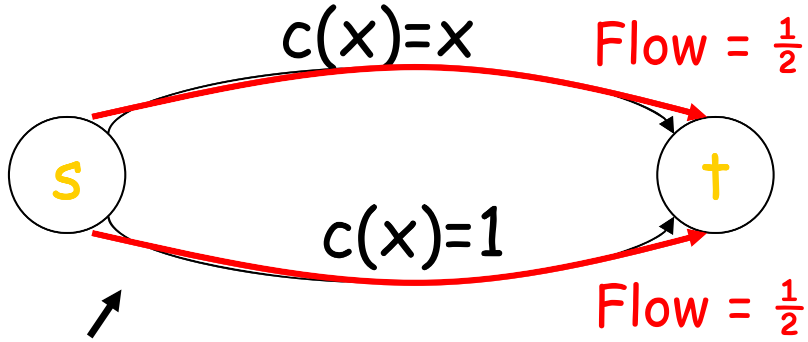
Optimal flow

(Classic) Nash flow



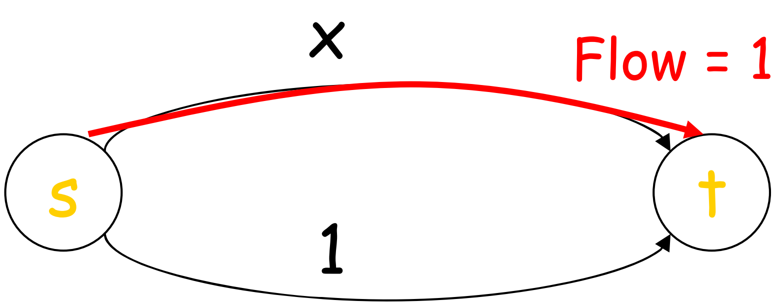
Example: Pigou's Network

One unit of flow is to be routed from s to t

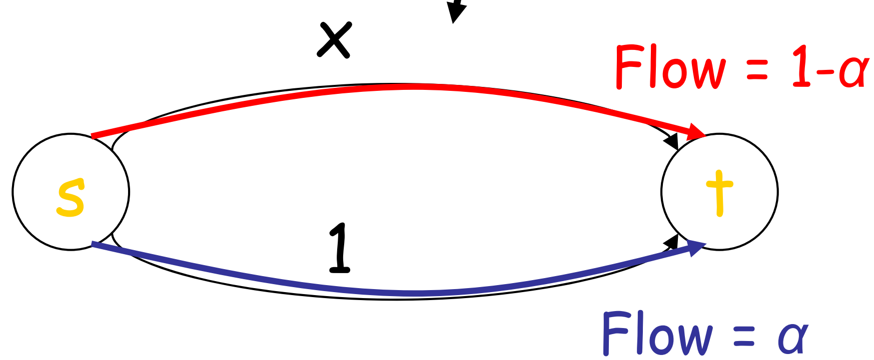


Optimal flow

(Classic) Nash flow

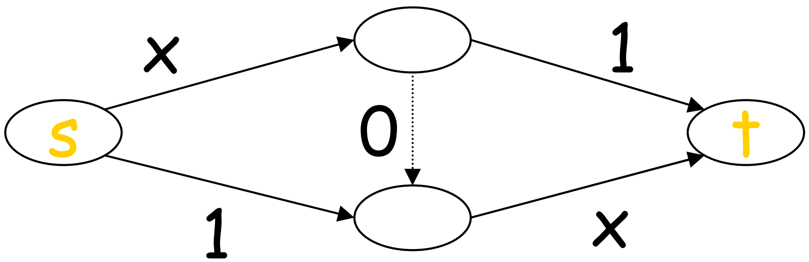


Nash flow when **a fraction α of (coordinated) players** is sent through the lower edge



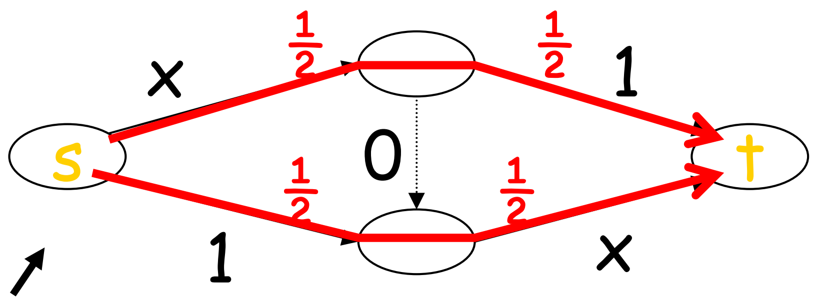
Example: Braess's Network

One unit of flow is to be routed from s to t



Example: Braess's Network

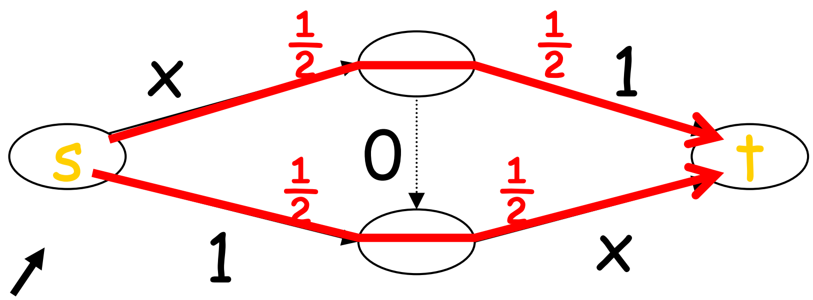
One unit of flow is to be routed from s to t



Optimal flow

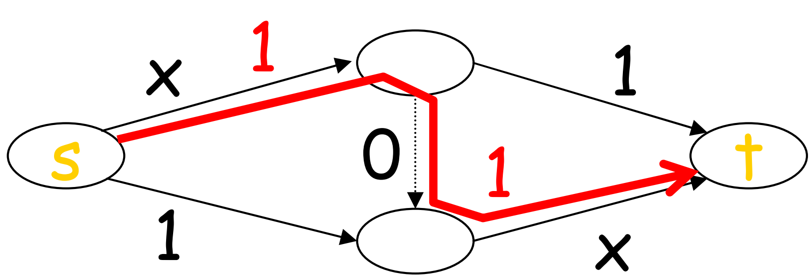
Example: Braess's Network

One unit of flow is to be routed from s to t



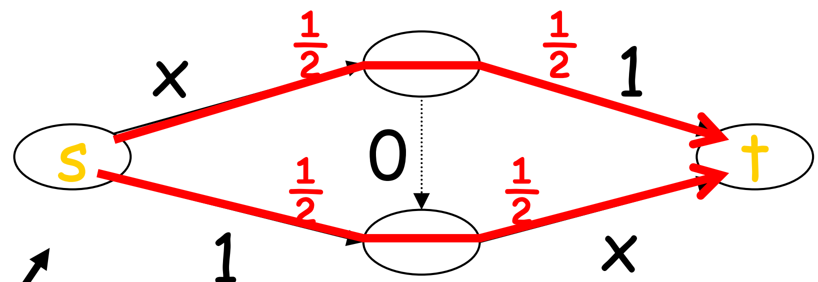
Optimal flow

(Classic) Nash flow



Example: Braess's Network

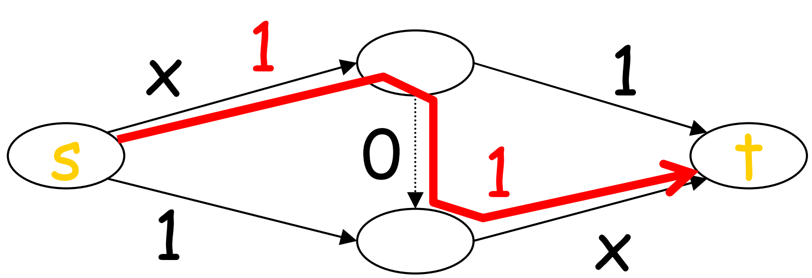
One unit of flow is to be routed from s to t

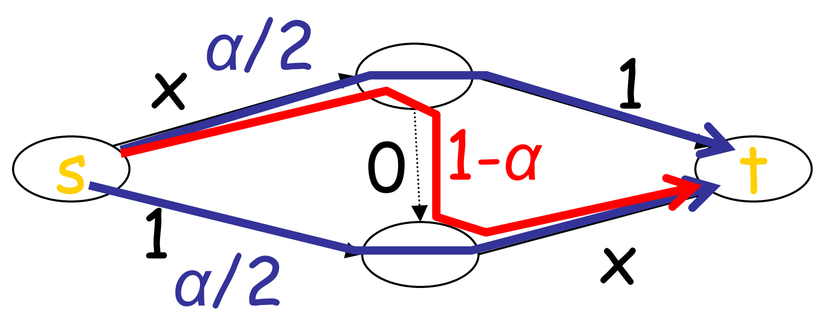


Optimal flow

(Classic) Nash flow

Nash flow when **a fraction α of coordinated players** is sent through the lower edge





Slightly more formal

- We will consider single commodity networks.
- An instance in such networks: (G, c_e, r)
- Assume that a fraction α of the players are cooperative. (G, c_e, r, α)
- A Stackelberg strategy assigns cooperative players to paths.
  - They induce a congestion $s = \{s_e\}_{e \in E}$
- A new game is "created": $(G, c'_e, (1-\alpha)r)$
  - Where $c'_e(x) = c_e(x + s_e)$

In the "new" game

- Selfish players choose paths (as usual), and Nash flows are considered as the possible outcomes of the game (as usual).
- On Equilibrium, selfish players induce a congestion $\sigma = \{\sigma_e\}_{e \in E}$
- The Price of Anarchy is $PoA = \frac{C(\sigma + s)}{OPT}$

The Central Questions

- Given a Stackelberg routing instance, we can ask:
  - Among all Stackelberg strategies, can we characterize and/or compute the strategy inducing the **Stackelberg equilibrium** - i.e., the eq. of minimum total latency?
  - What is the worst-case ratio between the total latency of the Stackelberg eq. and that of the optimal assignment of users to paths?

Finding best strategy: NP-hard

Reduction from $\frac{1}{3}$-$\frac{2}{3}$ $\mathrm{Partition}$ problem:

Given *n* positive integers $a_1, \ldots, a_n$ is there an $S \subseteq \{1, \ldots, n\}$ satisfying:

$$\sum_{i \in S} a_i = \frac{1}{3} \sum_{i=1}^{n} a_i$$

Given an instance of $\frac{1}{3}$-$\frac{2}{3}$ $\mathrm{Partition}$ create an instance of stackelberg routing:

- A network *G* with *n*+1 parallel links
- Demand: $2 \sum_{i=1}^{n} a_i = 2A$
- ¼ of the players are followers
- Cost functions: $c_i(x) = \frac{x}{a_i} + 4, i \leq n \text{ and } c_{n+1}(x) = \frac{x}{A}$

$$\text{"yes" instance} \Leftrightarrow \text{there exist a strategy with average cost} = \frac{35}{4} A$$

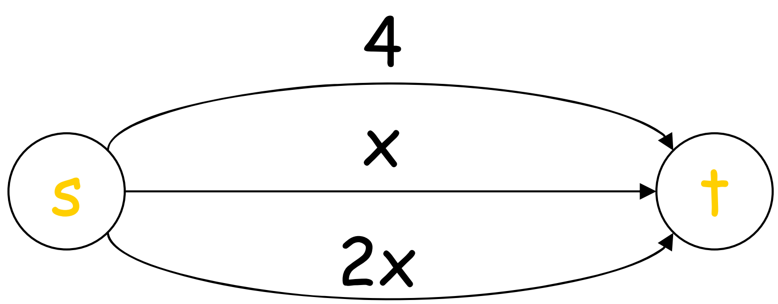
LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

6 units to be routed.



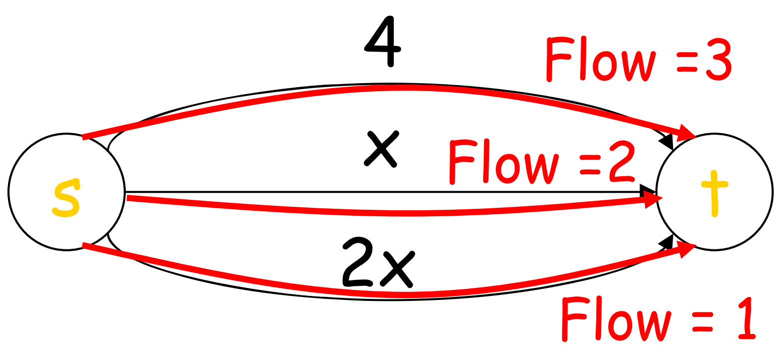
LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

6 units to be routed.

Opt routes:

- 3 to upper edge
- 2 to middle edge
- 1 to lower edge



LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

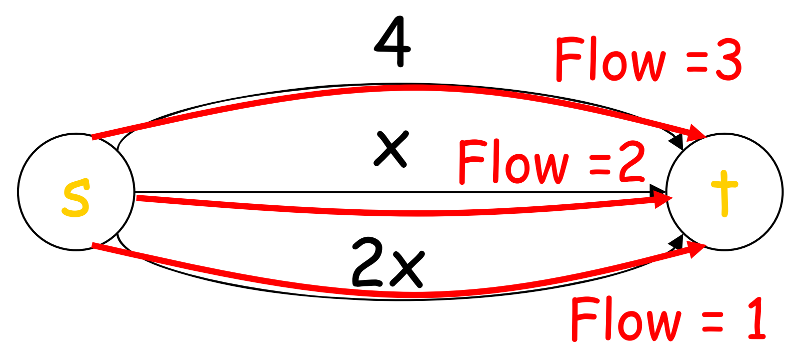
6 units to be routed.

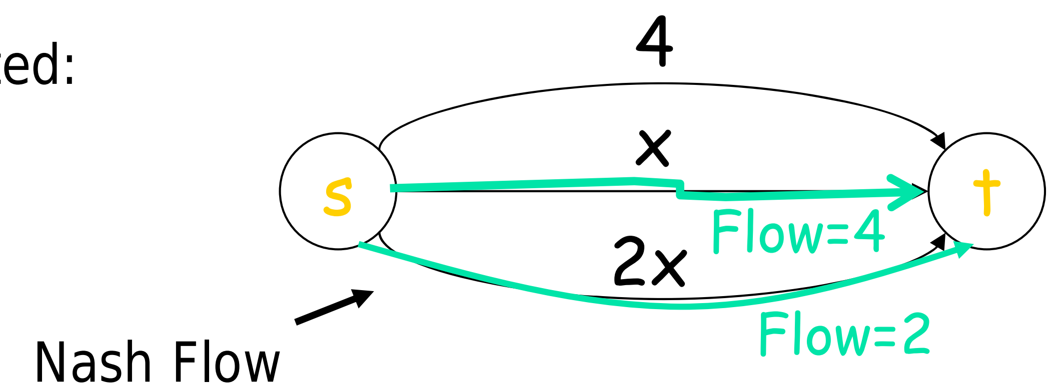
Opt routes:

- 3 to upper edge
- 2 to middle edge
- 1 to lower edge

In Nash Flow players are routed:

- 4 to middle edge
- 2 to lower edge





LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

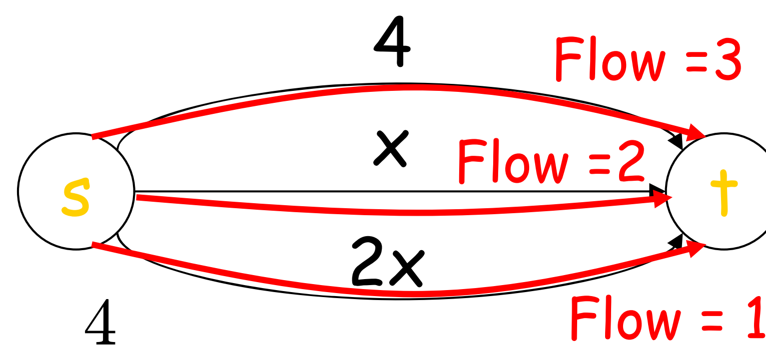
6 units to be routed.

Opt routes:

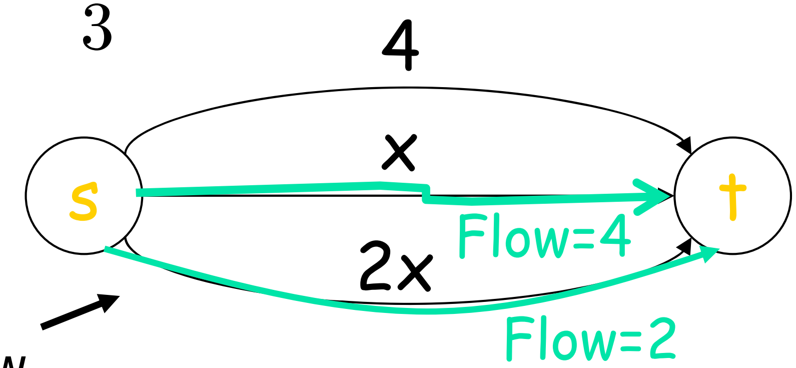
- 3 to upper edge
- 2 to middle edge
- 1 to lower edge

In Nash Flow players are routed:

- 4 to middle edge
- 2 to lower edge



$$PoA = \frac{4}{3}$$



Nash Flow

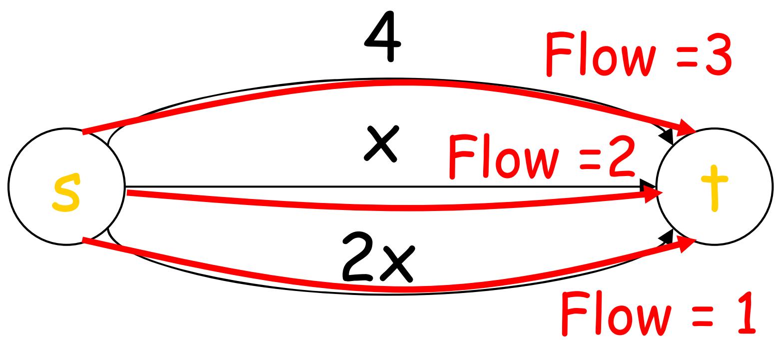
LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

6 units to be routed.

Opt routes:

- 3 to upper edge
- 2 to middle edge
- 1 to lower edge



LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

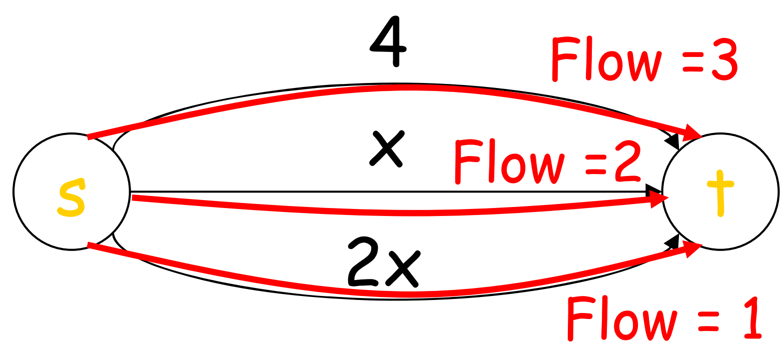
6 units to be routed.

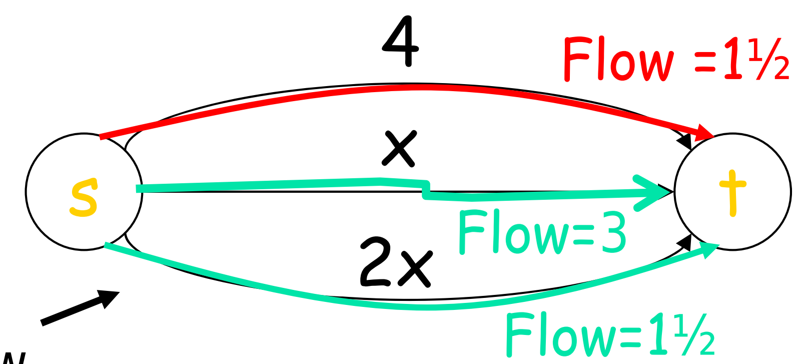
Opt routes:

- 3 to upper edge
- 2 to middle edge
- 1 to lower edge

LLF controlling ¼ players,
e.g. 1½ units, routes:

- 1½ to upper edge





Nash Flow

LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

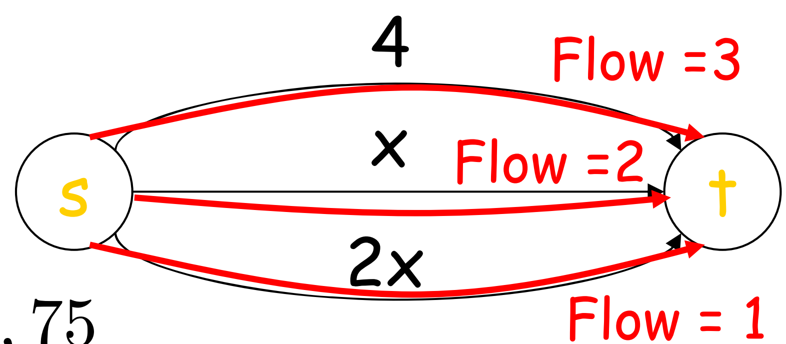
6 units to be routed.

Opt routes:

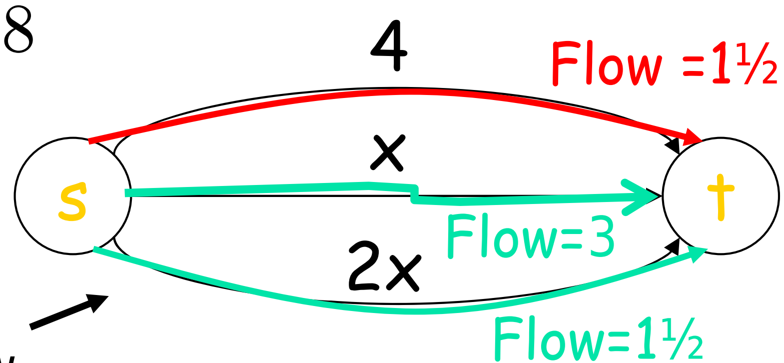
- 3 to upper edge
- 2 to middle edge
- 1 to lower edge

LLF controlling ¼ players, e.g. 1½ units, routes:

- 1½ to upper edge



$$PoA = \frac{18,75}{18}$$



Nash Flow

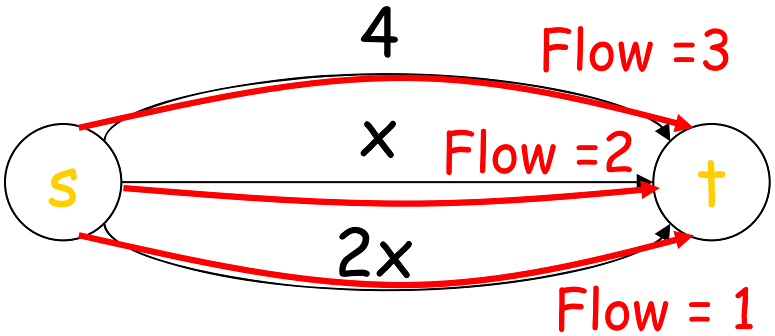
LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

6 units to be routed.

Opt routes:

- 3 to upper edge
- 2 to middle edge
- 1 to lower edge



LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

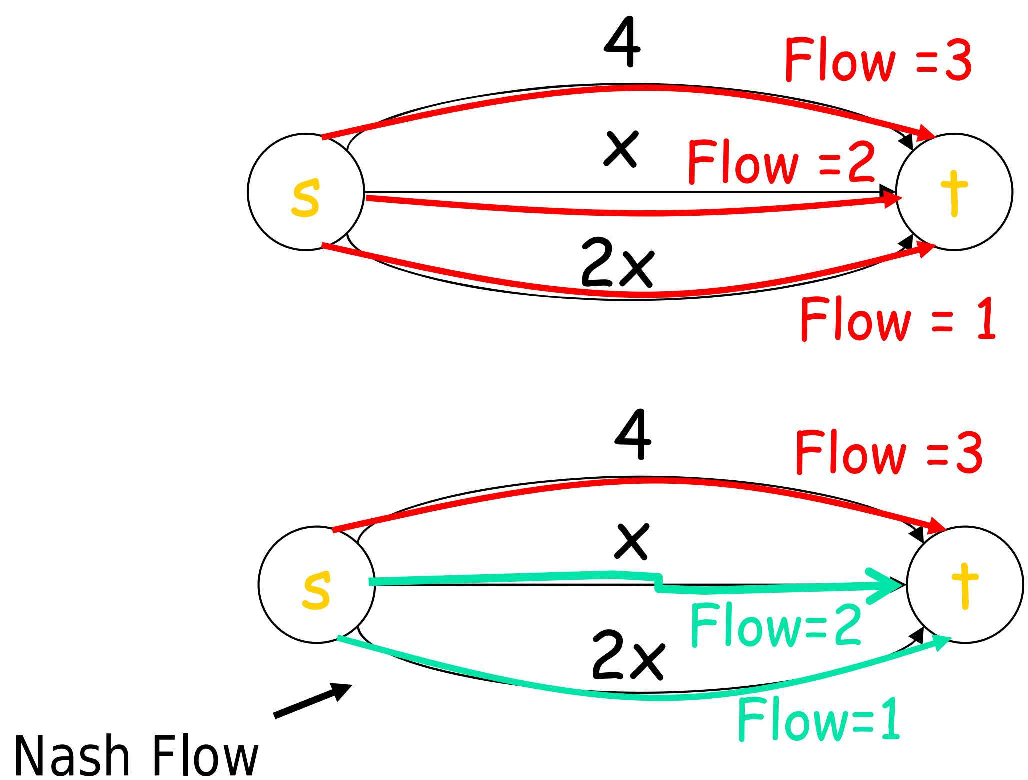
6 units to be routed.

Opt routes:

- 3 to upper edge
- 2 to middle edge
- 1 to lower edge

LLF controlling ½ players,
e.g. 3 units, routes:

- 3 to upper edge



LLF Strategy

- Largest Latency First (LLF):
  - Compute an optimal configuration
  - Assign coordinated players to optimal paths of largest latency

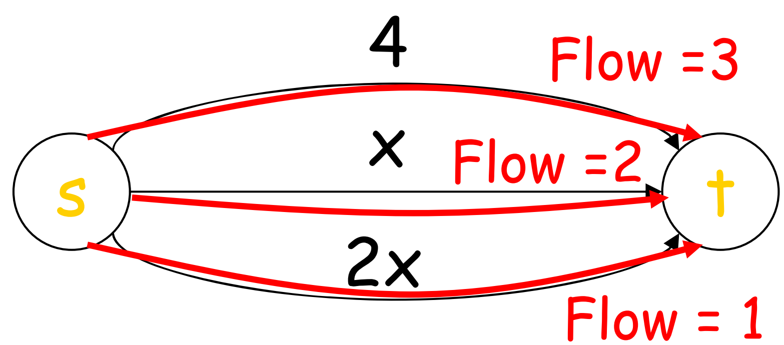
6 units to be routed.

Opt routes:

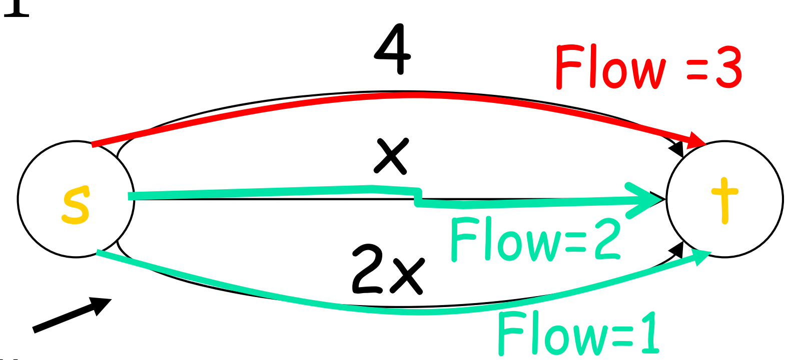
- 3 to upper edge
- 2 to middle edge
- 1 to lower edge

LLF controlling ½ players, e.g. 3 units, routes:

- 3 to upper edge



$$PoA = 1$$



Nash Flow

LLF in parallel links

Let α be the fraction of the cooperative players.

Theorem 1: In parallel links LLF induces an assignment of cost no more than $1/\alpha$ times the OPT:

$$PoA_{LLF} \leq \frac{1}{\alpha}$$

Proof by induction: When LLF saturates a link we can restrict to the instance that has:

- this link deleted and
- fraction of players the “remainders” of the previous instance.

Some problems:

- LLF may fail to saturate any link. No problem: Let m be the “heaviest” link. If L is the cost of selfish players and x* is the optimal assignment, it is

$$OPT \geq x^* c_m(x^*_m) \geq \alpha L = \alpha C(s + \sigma)$$

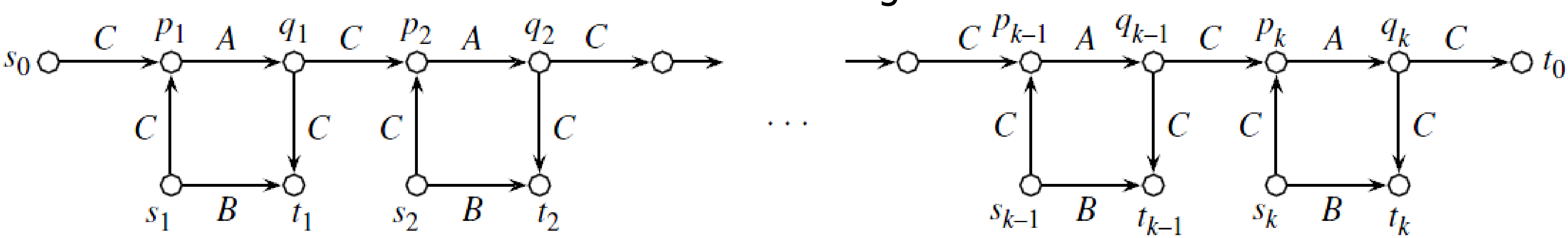
- When a link gets saturated selfish users could use it. No problem: There is an induced equilibrium that doesn’t use it.

Networks with Unbounded PoA

Theorem: Let $M > 0$ and $\alpha \in (0,1)$. There is an instance (G, c_e, r, α) such that for any Stackelberg strategy inducing *s*, it is:

$$C(s+\sigma) \geq M \cdot OPT$$

Proof: The network is the following



The demands are: $r_0 = \frac{1-\alpha}{2}$ and $r_i = \frac{1+\alpha}{2k}, i \geq 1$ (total flow=1)

Cost functions: B=1, C=0 and A is $c_\epsilon(x) = \begin{cases} 0, & \text{if } x \leq r_0; \\ 1 - \frac{r_0 + r_1 - x}{(1-\epsilon)r_1}, & \text{if } x \geq r_0 + 2\epsilon r_1. \end{cases}$

LLF in parallel links

Let o_e denote the optimal congestion

Lemma:

$$\text{i) } C(s+\sigma) = \sum (s_e + \sigma_e) c_e(s_e + \sigma_e) \leq \rho \cdot OPT$$

$$\text{ii) } \sum \sigma_e c_e(s_e + \sigma_e) \leq \rho \cdot \sum (o_e - s_e) c_e(o_e)$$

The proof follows from the variational inequality, similar to the "classic" result.

LLF in parallel links

Let o_e denote the optimal congestion

Lemma: i) $C(s+\sigma) = \sum (s_e+\sigma_e) c_e(s_e+\sigma_e) \le \rho \cdot OPT$

ii) $\sum \sigma_e c_e(s_e+\sigma_e) \le \rho \cdot \sum (o_e - s_e) c_e(o_e)$

The proof follows from the variational inequality, similar to the "classic" result.

Theorem 2: $PoA_{LLF} \le \alpha + (1-\alpha)\cdot\rho$

Proof: $OPT = \overbrace{\sum s_e c_e(o_e)}^{A} + \overbrace{\sum (o_e - s_e) c_e(o_e)}^{B}$ and $\frac{A}{B} \ge \frac{\alpha}{1-\alpha}$.

It is $C(s+\sigma) = \sum s_e c_e(s_e+\sigma_e) + \sum \sigma_e c_e(s_e+\sigma_e) \le A + \rho \cdot B$

This is maximized for $\frac{A}{B} = \frac{\alpha}{1-\alpha}$ with maximum value $\alpha + (1-\alpha)\cdot\rho$

благодаря

谢谢 (or 謝謝)

धन्यवाद (or তোমাকে ধন্যবাদ or ਤੁਹਾਡਾ ਧੰਨਵਾਦ ਹੈ)

با تشکر از شما

Ευχαριστώ

(also to Haris Angelidakis)