Selfish Routing

Tim Roughgarden
Cornell University

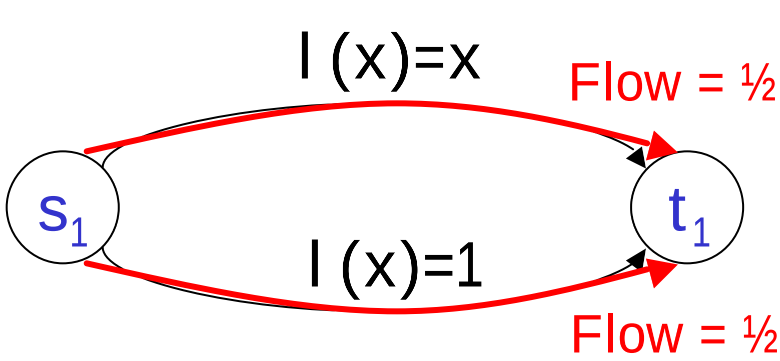
Includes joint work with Éva Tardos

Traffic in Congested Networks

The Model:

- A directed graph $G = (V,E)$
- k source-destination pairs $(s_1, t_1), \ldots, (s_k, t_k)$
- A rate r_i of traffic from s_i to t_i
- For each edge e, a latency function $l_e(\bullet)$

Example: $(k,r=1)$



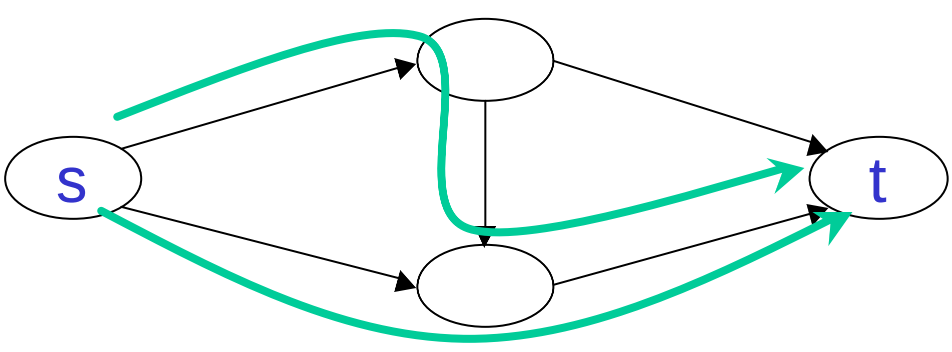
Flows and their Cost

Traffic and Flows:

- f_P = amount of traffic routed on s_i-t_i path P
- flow vector f ⇔ routing of traffic

The Cost of a Flow:

- $l_P(f)$ = sum of latencies of edges on P (w.r.t. the flow f)
- $C(f)$ = cost or total latency of flow f: $S_P\, f_P \bullet l_P(f)$

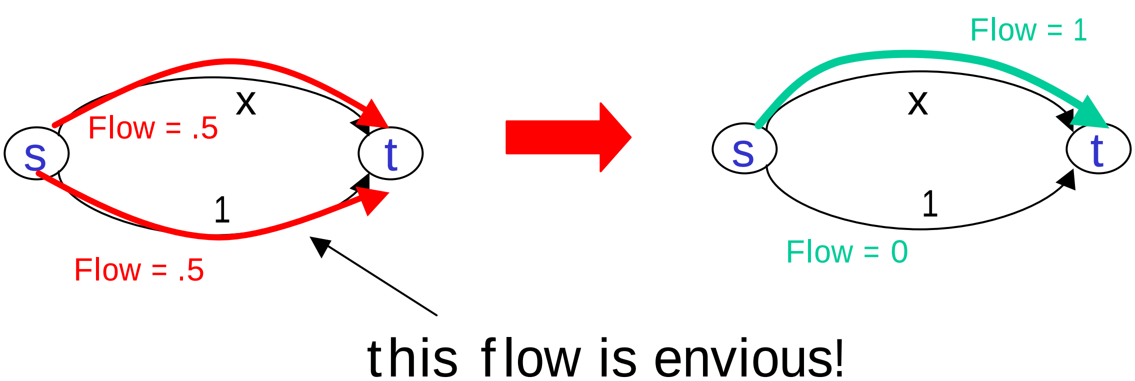


Selfish Routing

- flow = routes of many noncooperative agents
- Examples:
  - cars in a highway system [Wardrop 52]
  - packets in a network
- cost (total latency) of a flow as a measure of social welfare
- agents are selfish
  - do not care about social welfare
  - want to minimize personal latency

Flows at Nash Equilibrium

Def: A flow is at Nash equilibrium (is a Nash flow) if no agent can improve its latency by changing its path



Assumption: edge latency functions are continuous, nondecreasing

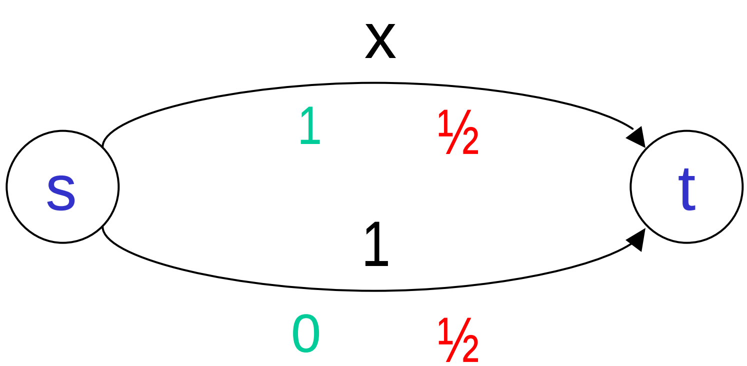
Lemma: f is a Nash flow $\Leftrightarrow$ all flow on minimum-latency paths (w.r.t. f)

Fact: have existence, uniqueness

The Inefficiency of Nash Flows

Fact: Nash flows do not optimize total latency [Pigou 1920]

Þ lack of coordination leads to inefficiency

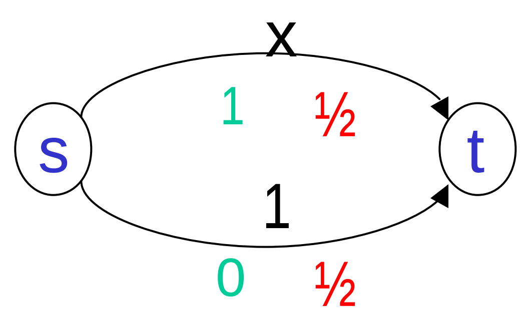


Cost of Nash flow = 1•1 + 0•1 = 1

Cost of optimal (min-cost) flow = ½•½ +½•1 = ¾

How Bad is Selfish Routing?

Pigou's example is simple…



How inefficient are Nash flows:

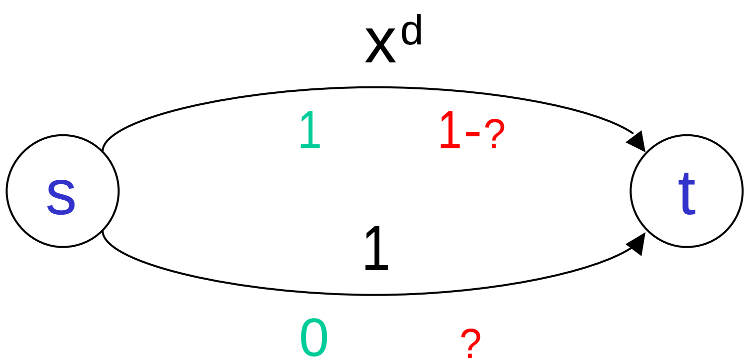
- with more realistic latency fns?
- in more realistic networks?

Goal: prove that Nash flows are near-optimal

- want a laissez-faire approach to managing networks
  - also [Koutsoupias/Papadimitriou 99]

The Bad News

Bad Example: (r = 1, d large)



Nash flow has cost 1, min cost ≈ 0

Þ Nash flow can cost arbitrarily more than the optimal (min-cost) flow

– even if latency functions are polynomials

A Bicriteria Bound

Approach #1: settle for weaker type of guarantee

Theorem: [Roughgarden/Tardos 00] network w/cts, nondecreasing latency functions Þ

cost of Nash at rate r = cost of opt at rate 2r

Corollary: M/M/1 delay fns (l(x)=1/(u-x), u = capacity) Þ

Nash cost w/ capacities 2u = opt cost w/ capacities u

Linear Latency Functions

Approach #2: restrict class of allowable latency functions

Def: a linear latency function is of the form $l_e(x)=a_e x+b_e$

Theorem: [Roughgarden/Tardos 00]

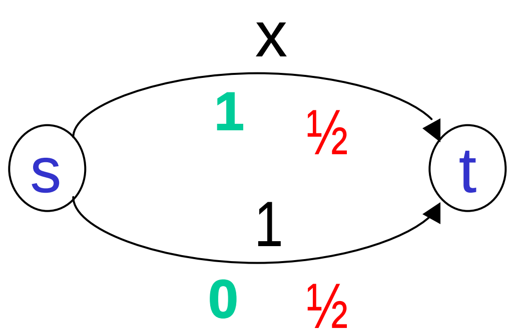
network w/linear latency fns Þ

cost of Nash flow = 4/3 × cost of opt flow

Sources of Inefficiency

Corollary of main Theorem:

- For linear latency fns, worst Nash/OPT ratio is realized in a two-link network!



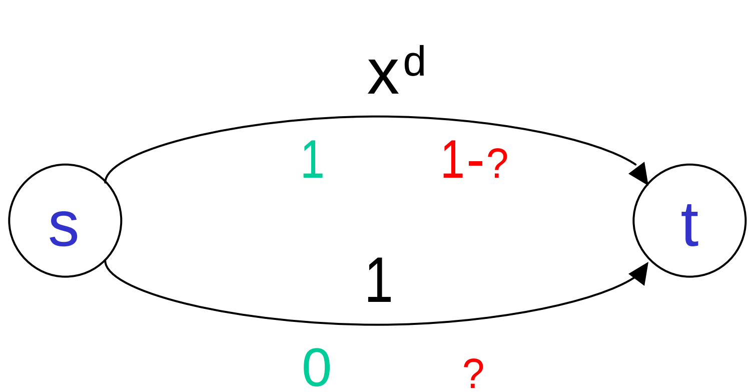
- Cost of Nash = 1
- Cost of OPT = ¾

- one source of inefficiency:
  - confronted w/ two routes, selfish users overcongest one of them
- Corollary $\Rightarrow$ that's all, folks!
  - network topology plays no role

No Dependence on Network Topology

Theorem: [Roughgarden 02] for (almost) any class of latency fns including the constant fns, worst Nash/OPT ratio occurs in a two-link network.

Corollary: worst-case for bounded-degree polynomials is:



Coping with Selfishness

Motivation:

- Nash flows inefficient
- centralized routing often infeasible

Goal: design/manage networks s.t. selfish routing "not too bad"

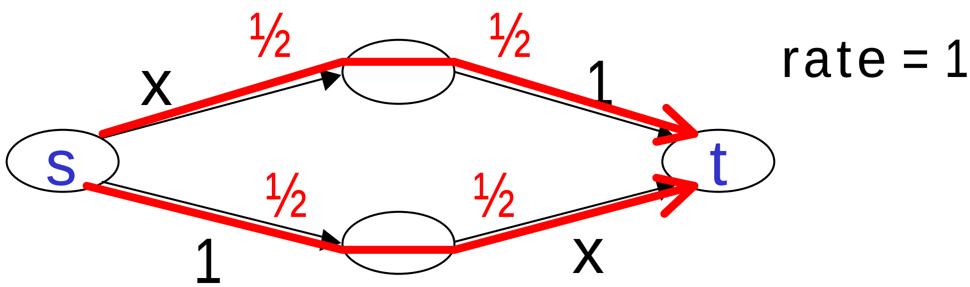
$\Rightarrow$ adds new algorithmic dimension

Two Approaches:

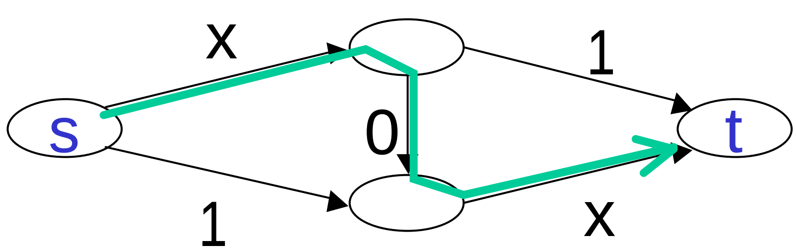
- network design (next)
- Stackelberg routing (see thesis)

Braess's Paradox

Better network, worse Nash flow:



Cost of Nash flow = 1.5



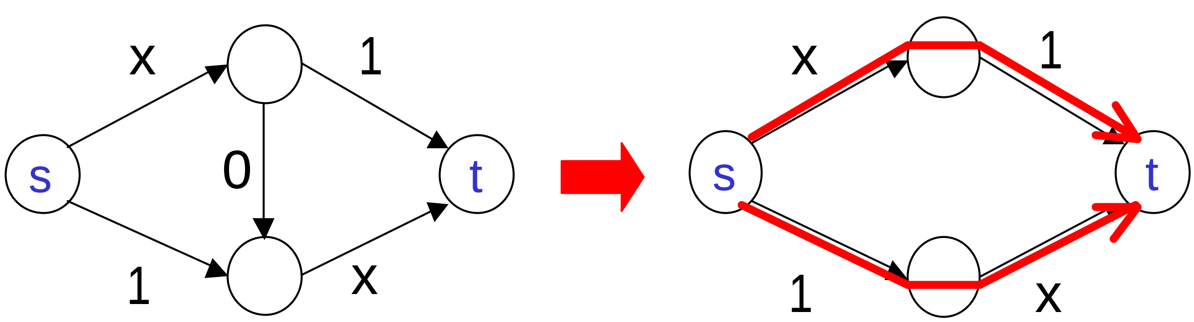
Cost of Nash flow = 2

All traffic experiences additional latency! [Braess 68]

Designing Networks for Selfish Users

The Problem:

- given network $G = (V,E,l)$
  - assume single-commodity
- find subnetwork minimizing latency experienced by all selfish users in a Nash flow

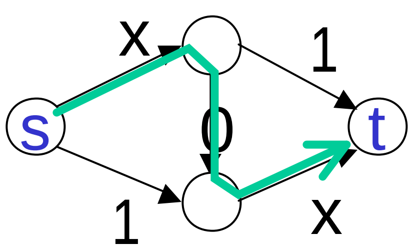


$\Rightarrow$ want to avoid Braess's Paradox

Generalizing Braess's Paradox

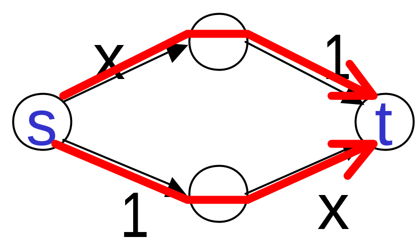
Question: is Braess's Paradox more severe in bigger networks?

Fact: with linear latency fns, worst case is



cost = 2

vs.



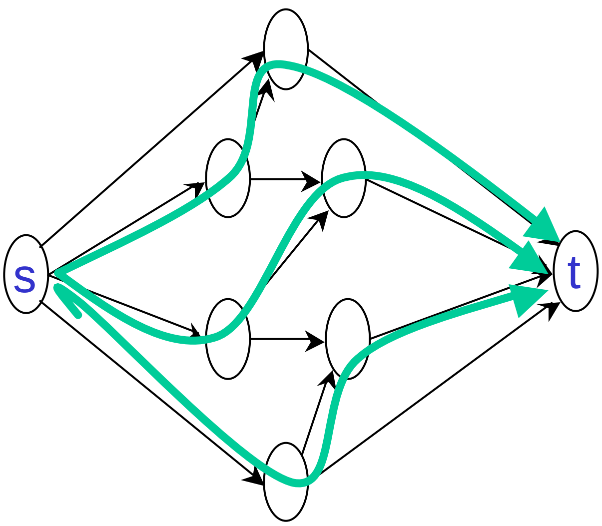
cost = 3/2

Reason: with linear latency fns,

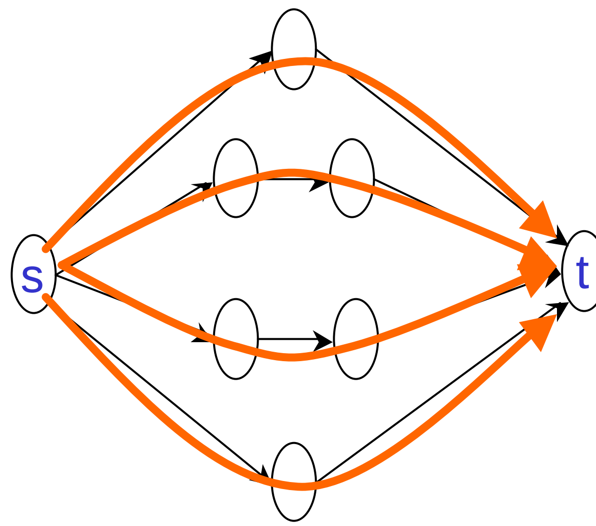
average latency of Nash flow = 4/3 × average latency of any other flow

Braess's Paradox with General Latency Fns

A Bigger Braess Paradox:



Nash in whole graph
common latency = 4



Nash in opt subgraph
common latency = 1

$\Rightarrow$ removing edges can improve
Nash by a $n/2$ factor ($n=|V|$)

Thm: [R 01] this is worst possible.

The Trivial Algorithm

Def: The trivial algorithm is to build the entire network.

We know: the trivial algorithm is

- a $4/3$-approx alg with linear latency fns
- an $n/2$-approx alg with general latency fns

Question: what about more sophisticated algorithms?

Designing Networks for Selfish Users is Hard

Thm: [R 01] For ? > 0, no (n/2 - ?)-approximation algorithm exists (unless P=NP).

Thm: [R 01] For linear latency functions, no (4/3 - ?)-approx algorithm exists (unless P=NP).

Remark: similar results hold for other classes of latency fns.

Corollary: Braess's Paradox eludes efficient algorithms.

Directions for Further Research

Selfish Routing: many open questions, see thesis

Other Games: e.g., flow control, competitive facility location, auctions

Paradigm for studying selfishness:

- what is worst Nash/OPT objective fn value ratio?
- are other meaningful bounds (e.g., bicriteria) possible?
- sources of inefficiency?
- design/management strategies for coping with selfishness?