Routing in Networks

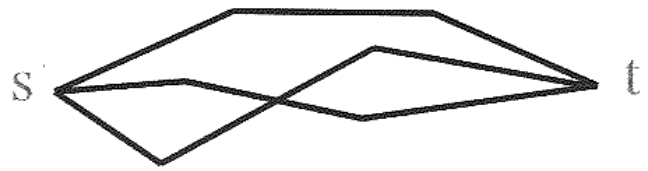
Éva Tardos
Cornell University

Talk Outline

- Good approximation for high capacities.
  - off-line and on-line
    - Raghavan/Thompson
    - Klein/Stein/Plotkin/Tardos
    - Awerbuch/Azar/Plotkin
- Results for disjoint paths.
    - Garg/Vazirani/Yannakakis
    - Kleinberg/Tardos
- Packet routing along paths.

Large capacity + off-line
⟶ provably good routing

- Consider fractional routing



  - fractional routing is a linear program
  - ⟶ can be solved efficiently
- Use fractions as probabilities to select one of the paths at random
- Theorem (Raghavan-Thompson'87)
  - if $u(e) > c_e \log n$ then result is a $1+\epsilon$ approximation

More on large capacity networks

- Klein-Plotkin-Stein-Tardos
  - combinatorial method to solve LP
  - leads directly to integral solution
  - 1+e approximation
- Azar-Aspnes-Fiat-Plotkin-Waarts; Awerbuch-Azar-Plotkin-Waarts
  - O(log n)-competitive algorithm on-line

    - k-competitive = k-approximation algorithm

Klein-Plotkin-Stein-Tardos

- Routing off-line = rerouting allowed
  - start with any routing
  - reroute to improve gradually
    - edge costs: exponential in current congestion:
      - $c(e) = a^{f(e)/u(e)}$ f(e)= number of paths on edge e
  - Inner loop:
    - pick an s & t pair
    - reroute onto a cheaper path
    - update edge-costs
- Theorem: after O(log n) reroutings /paths on the average an 1+e optimal solution is found

On-line version: Awerbuch-Azar-Plotkin

- Route immediately: no rerouting allowed
- Assume $k \sim \log n$ paths may share any edge
  - use cost $c(e)=2^{f(e)}$ (assume uses unit. BW edges)
  - compute $cost(s,t)$ = min-cost of a path from s to to t
  - reject request if $cost(s,t) \geq 2^k$
  - otherwise accept and route on shortest path
- routing feasible:
  - if s & t pair accepted then $cost(s,t) < 2^k$
  - hence each edge e used has at most $k-1$ paths so far

(for proof → admission control so don't route very expensive things)

Why is solution competitive?

- Idea: use $c(e)$ as LP dual solution to bound optimum

real background: alg. can solve LP's arbitrarily

- $\sum_{s,t \text{ in Opt}} cost(s,t) \leq k \sum_{e} c(e)$

every edge used at most k times

$k \sum_{e} c(e)$ dual objective

in this particular solution

- $2^k \geq$ # paths accepted $\geq \sum_e (c(e)-1)$

  cost = exp. # of paths on a edge

proof by induction

new path from s to t accepted

$\Rightarrow \sum_e c(e)$ increases by $\text{cost}(s,t)$

So far: large capacity

$\geq c \log m$ paths may share an edge

Summary: easy to approximate

	congestion	admission
off-line	$1+\varepsilon$ *	$1+\varepsilon$
on-line	$O(\log m)$	$O(\log m)$ *

$\varepsilon \geq 0$

* best possible on directed graphs (because on line, not because of complexity)

Next: • disjoint paths

or.... at most 2 paths share any edge

Disjoint paths

Recall: **NP-hard & max-SNP-hard**

- admission on trees
- congestion on mesh

only few + results sense of hopelessness : neg. result : no approx. for edge-disjoint paths up to n^{ε} (meshes) for an $\varepsilon > 0$ ↑ bound

Approximation results

- on trees

  Garg - Vazirani - Yannakakis 2-approx

  Awerbuch - Gawlick - Leighton - Rabani

  $O(\log u)$ - competitive on-line

- mesh

  Kleinberg - Tardos * $O(1)$ - approx

  Awerbuck - Gawlick - Leighton - Rabani

  $O(\log u \log\log u)$ - competitive on-line

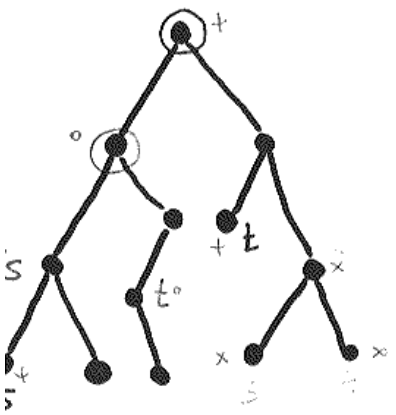
- expander

  Broader - Frieze - Peleg - Suen - Upfal

* extends to densely embedded graphs

2-approximation on trees

Garg-Vazirani-Yannakakis



- order requests with lower turning point (LCA) first. E.g.:
- route & accept greedily in above order

choice of root arbitrary!

Theorem

pairs routed $\geq \frac{1}{2}$ optimum

only for trees

Disjoint paths: summary

- at least $\log n$ path may share

  $\Longrightarrow$ close to optimum routing

  - **off-line** $1+\varepsilon$ approx
  - **on-line** $O(\log n)$-competitive

    but $1+\varepsilon$ approx in average case

    Kamath-Palmon-Plotkin

    (any graph, but requests from a specific distribution : essentially online → offline : Congestion any moment in time, (paths come & go) not adversarial

  - analysis extends to routing
    - short paths
    - trees

    etc

disjoint paths: no sharing

- $O(1)$ - approximation
  on large class of graphs

- uses polynomial solvable special cases as building blocks

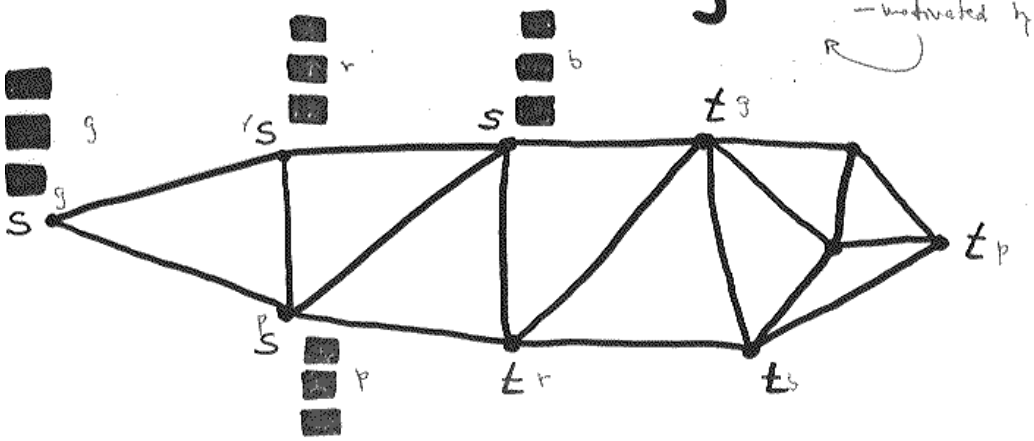
– can trees, short paths, etc be routed??

believes can be done in existing networks
: clusters naturally exist in graphs
other than meshes (e.g. real computer networks)

in progress but no quantifiable results yet

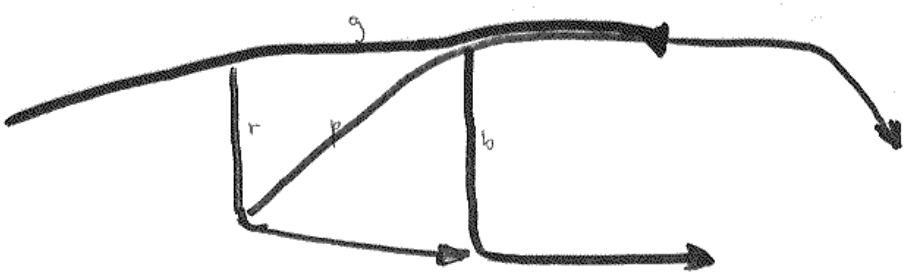
Packet scheduling

get short paths?
— motivated by



schedule the motion of packets

•



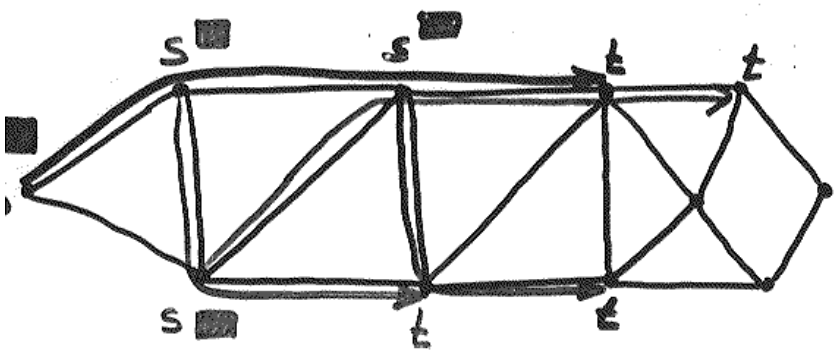
along pre selected paths

all packets between one s & t pair must follow the same paths

natural: want paths short
don't want too much congestion

- Rules for moving packets
  - store & forward
  - synchronized network
    at most one packet per edge
    at each time step

How fast can packets be delivered?



for now:
1 packet
per paths

D = dialation

max # edges / path

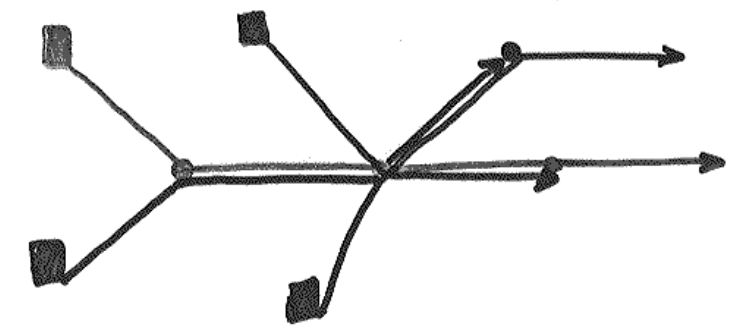
C = congestion

max # paths through an edge

Both D & C are lower bounds
on schedule length

a simple schedule

Leighton, Maggs & Rao



Use random delays:

- each path selets a random delay $\leq C/\log n$

C = max congestion

- wait delay & then follow path
  ignoring all other packets

M = max conflict
= max # packets on an edge at a time

- slow down schedule by a factor of M

to eliminate conflicts

Length of resulting schedule

$$\left[\frac{C}{\log n} + D\right] \cdot M$$

$\frac{C}{\log n}$: random delay

M: slow-down factor

Theorem Leighton-Maggs-Rao

$M \in O(\log n)$ with high probability

$\Rightarrow$ schedule length:

$$O(C + D \log n)$$

Improved versions

- **simple:** $O(C + D \log n)$

Leighton & Maggs & Rao

there exists schedule $O(C+D)$

uses Lovász local lemma

Leighton & Maggs

algorithmic version

using Beck's algorithmic LLL version

Ostrovsky & Rabani & Tardos $O(C+D+\log n)$

distributed version

without LLL

use random delays recursively

extension to many packets / path

Andrews & Fernandez & Harchol-Bartel & Leighton, Zhang

version dependent on local delays

but using LLL

delivery time depends on its own congestion, dilation

? distributively ?

Conclusion

two related problems

- routing
- packet scheduling

good routing should have

- limited sharing of edges
  small congestion C
- short paths
  small dilation D

$\Rightarrow$ important problem:
routing short paths

✓ • in high capacity networks

? • disjoint short paths

We have

- $\sum_{s,t \text{ in Opt}} \text{cost}(s,t) \leq k \sum_{e} c(e)$
- 2^k # paths accepted $\geq \sum_{e} (c(e)-1)$
- rejected s & t pairs have

$$\text{cost}(s,t) \geq 2^k$$

$\Rightarrow$ algorithm $O(k)$-competitive

Recall $k \sim \log m$