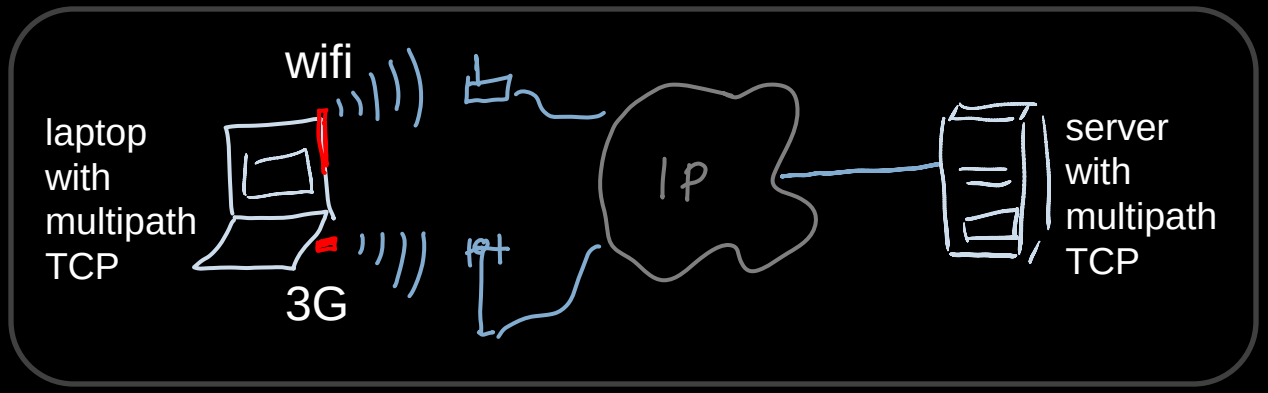
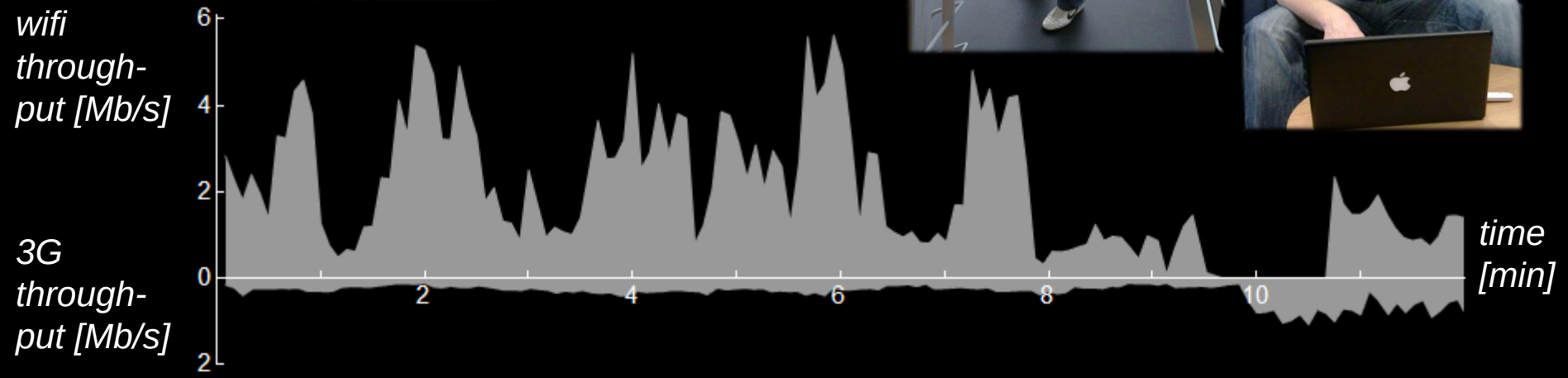
# Resource Pooling

Damon Wischik, UCL

# We have a working implementation of multipath transport



This user clearly benefits from multipath. But is it safe for the network and other users? Or does it cause instability, route flap, unfairness, disaster?

This user clearly benefits from multipath. But is it safe for the network and other users? Or does it cause instability, route flap, unfairness, disaster?

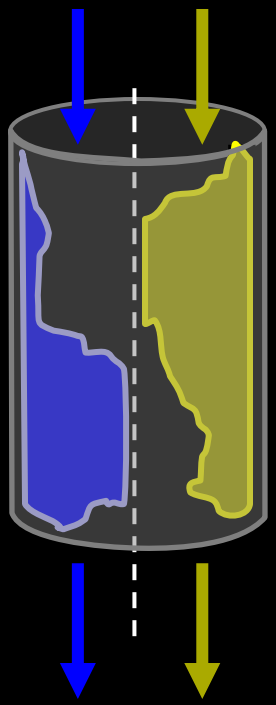
I. **Resource pooling as a design principle**
   The earliest design goal of the Internet aimed to achieve “resource pooling”. Multipath transport is a natural extension. It can improve resilience to traffic surges and link failures.

II. **A metric for resource pooling (WP1+WP2)**
   If multipath load balancing is done right, then the network achieves some degree of resource pooling. We have a metric for measuring how much, given the topology and traffic matrix.

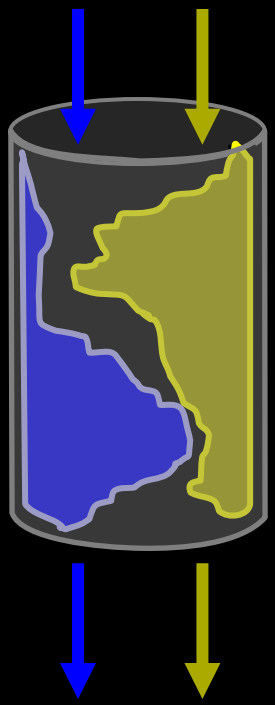
III. **A coupled congestion control algorithm (WP2)**
   We have designed and implemented a multipath congestion control algorithm that balances load, and we can guarantee it’s safe to deploy (but it’s harder than you’d think to do it right)

# I. Resource pooling as a design principle

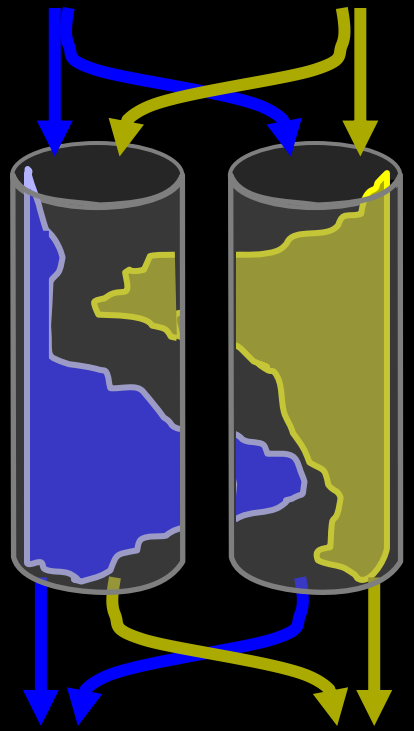
Resource pooling means “making a collection of resources behave like a single pooled resource”. It has been a design goal of the Internet from the beginning.



*A single link, split into two circuits*

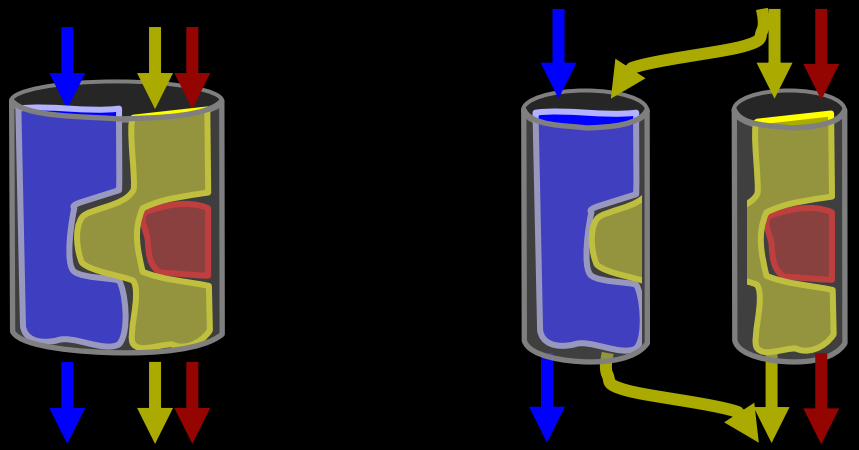


*Packet switching “pools” the two circuits*

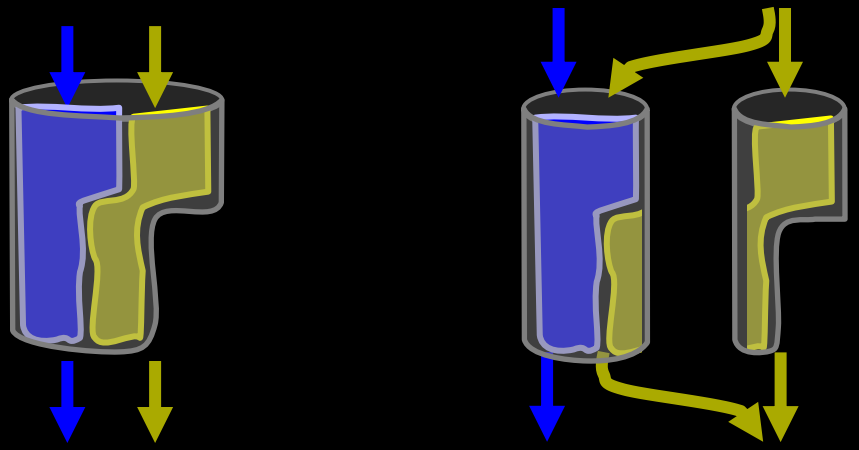


*Multipath “pools” the two links*

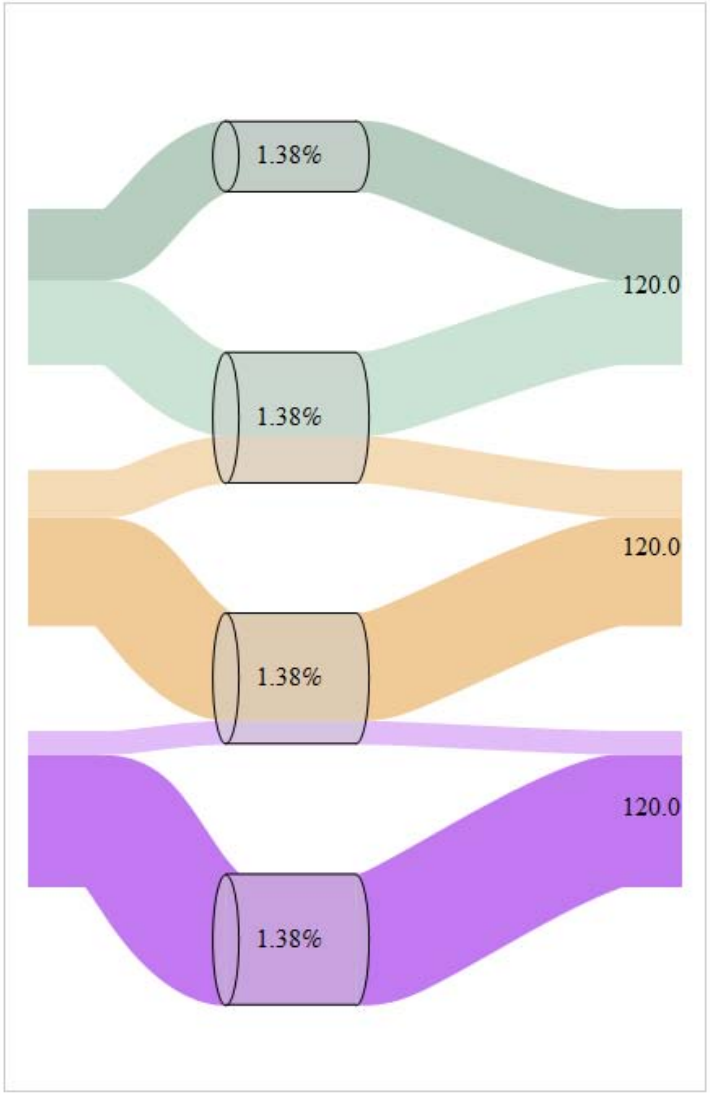
Resource pooling means the network is better able to accommodate a surge in traffic



or a loss of capacity



by shifting traffic and thereby “diffusing” congestion across the network.




coupled TCP/2

$\phi$ 0.0

$C_1$ 54 pkt/s

$C_2$ 100 pkt/s

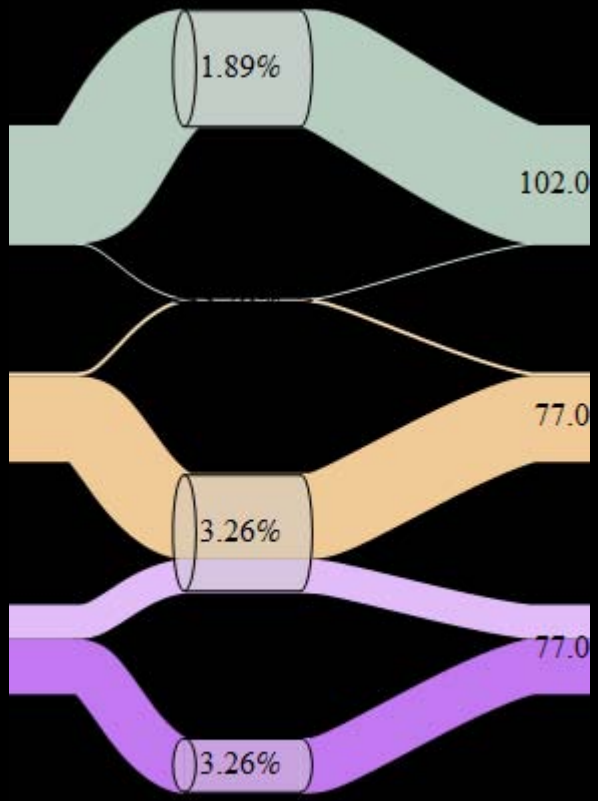
$C_3$ 100 pkt/s

$C_4$ 100 pkt/s

Resource pooling means:

Congestion due to reduced capacity etc. can be shared across the entire network

Resource pooling relies on there being enough path choices, and enough traffic that can make a choice

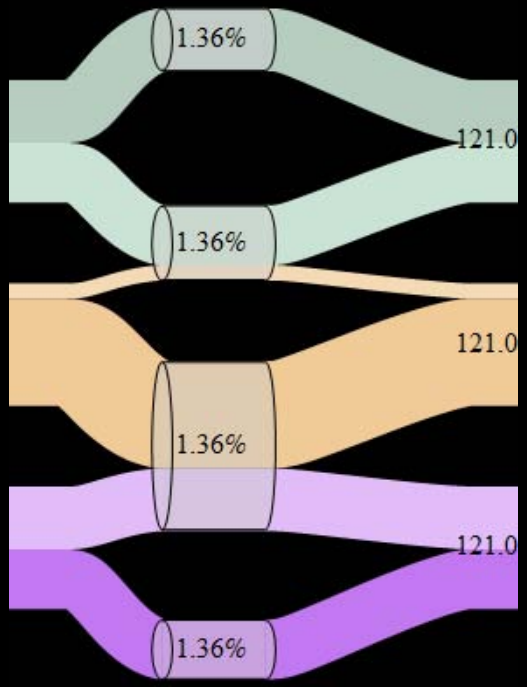



*The network has split into two resource pools, because neither of the bottom two flows can access the top resource pool.*

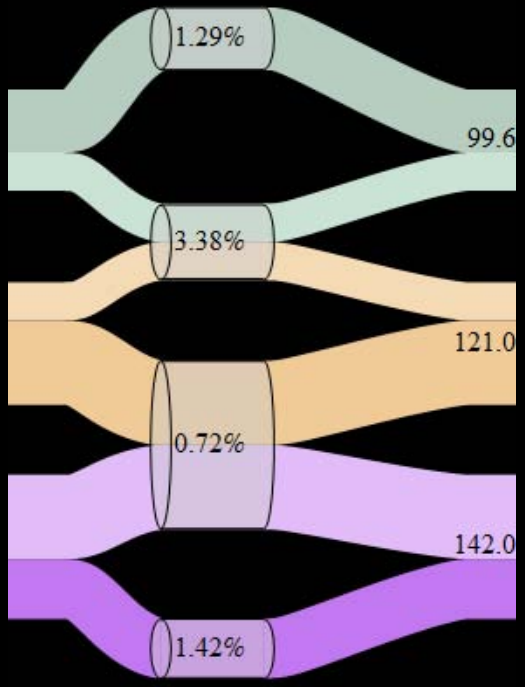
**Topic II. How much resource pooling can be achieved, given a set of multipath routes and a traffic matrix?**

Will there be one big pool, or many small pools?

# Resource pooling relies on proper load-balancing by the end-systems




*Using an idealized coupled congestion controller, there is resource pooling*




*Using separate TCP controllers for each path, congestion is **not** equalized and capacity is **not** shared*

**Topic III. Can we design a congestion controller such that users react in the right way to achieve resource pooling?**

If they don't, there may be a single pool but it won't be shared properly.

# Topic II. How much resource pooling can be achieved, given a set of multipath routes and a traffic matrix?

For the purposes of network-wide resource pooling,

- Is it sufficient to use end-host addressing?
- How much path diversity is enough, and what sort of diversity is useful?

To answer this, we first need a metric for the amount of resource pooling that a network achieves.

How should we measure resource pooling? It means

*"making a collection of resources behave like a single pooled resource".*

To measure resource pooling, we need to decide what we mean by "behave" and "like a single resource".

## "Behave"

We've seen that resource pooling has the effect that congestion hotspots can be diffused across the network. So the behaviour I shall examine is "what is the change in congestion at a link, in response to a change in the capacity at that link?"

## "Like a single resource"

Suppose for example that

- at an isolated link with capacity 100Mb/s, the loss of 50Mb/s increases packet loss by a factor of 20
- at an isolated link with capacity 1Gb/s, the loss of 50Mb/s increases packet loss by a factor of 1.03
- at a resource-pooling link with capacity 100Mb/s, the loss of 50Mb/s increases packet loss by a factor of 1.03

Then we'll say that the "effective pooled capacity at that link" is 1Gb/s.

# Theorem

In a network with idealized multipath congestion control, the change in loss rate at link $j$ in response to a given drop in capacity is the same as would be experienced by an isolated link whose capacity is $C_j/(1-\Psi_{jj})$ where $C_j$ is the actual link's capacity.

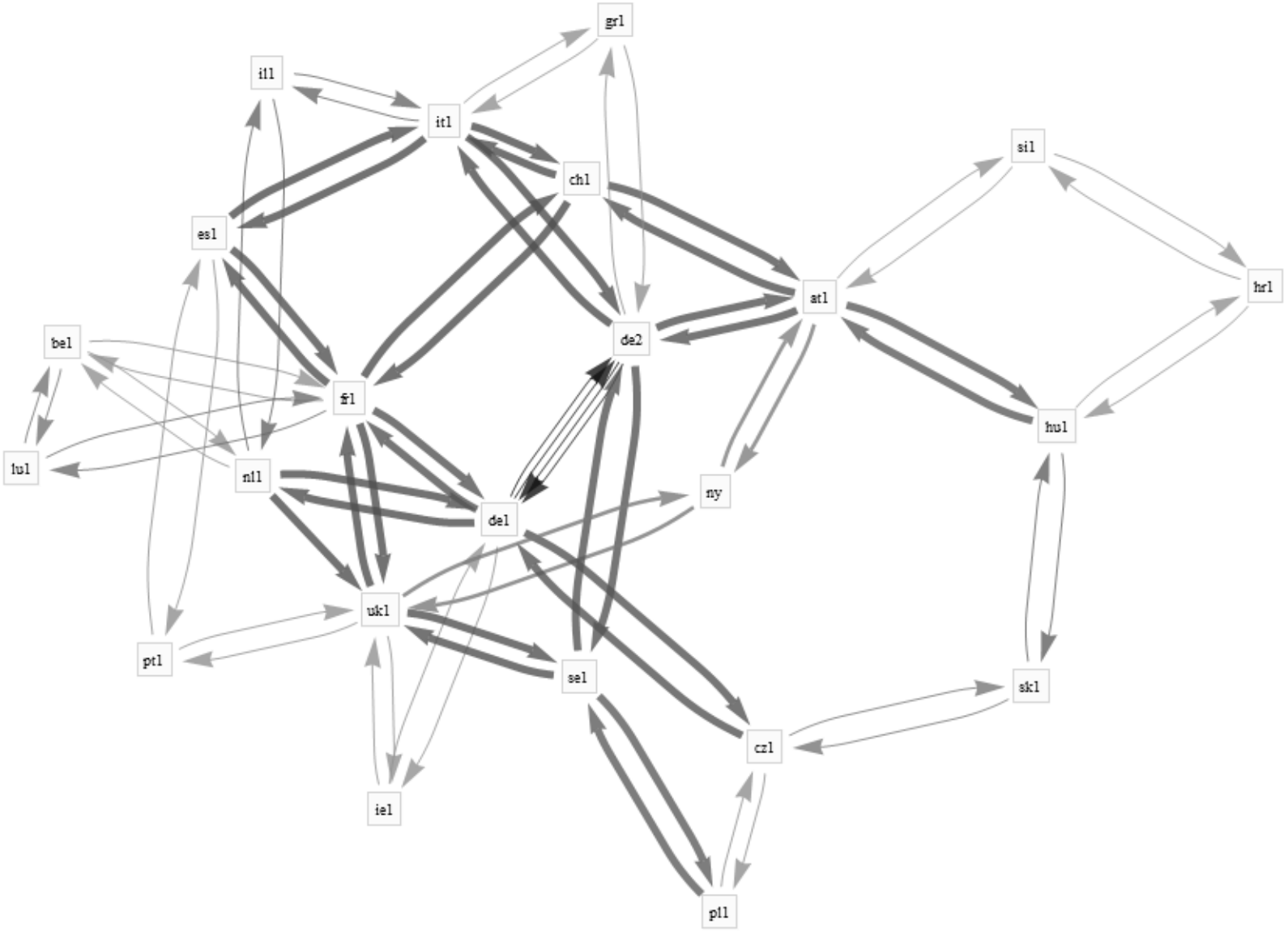
I call $\Psi_{jj}$ the "poolability score", and $C_j/(1-\Psi_{jj})$ the "effective pooled capacity".

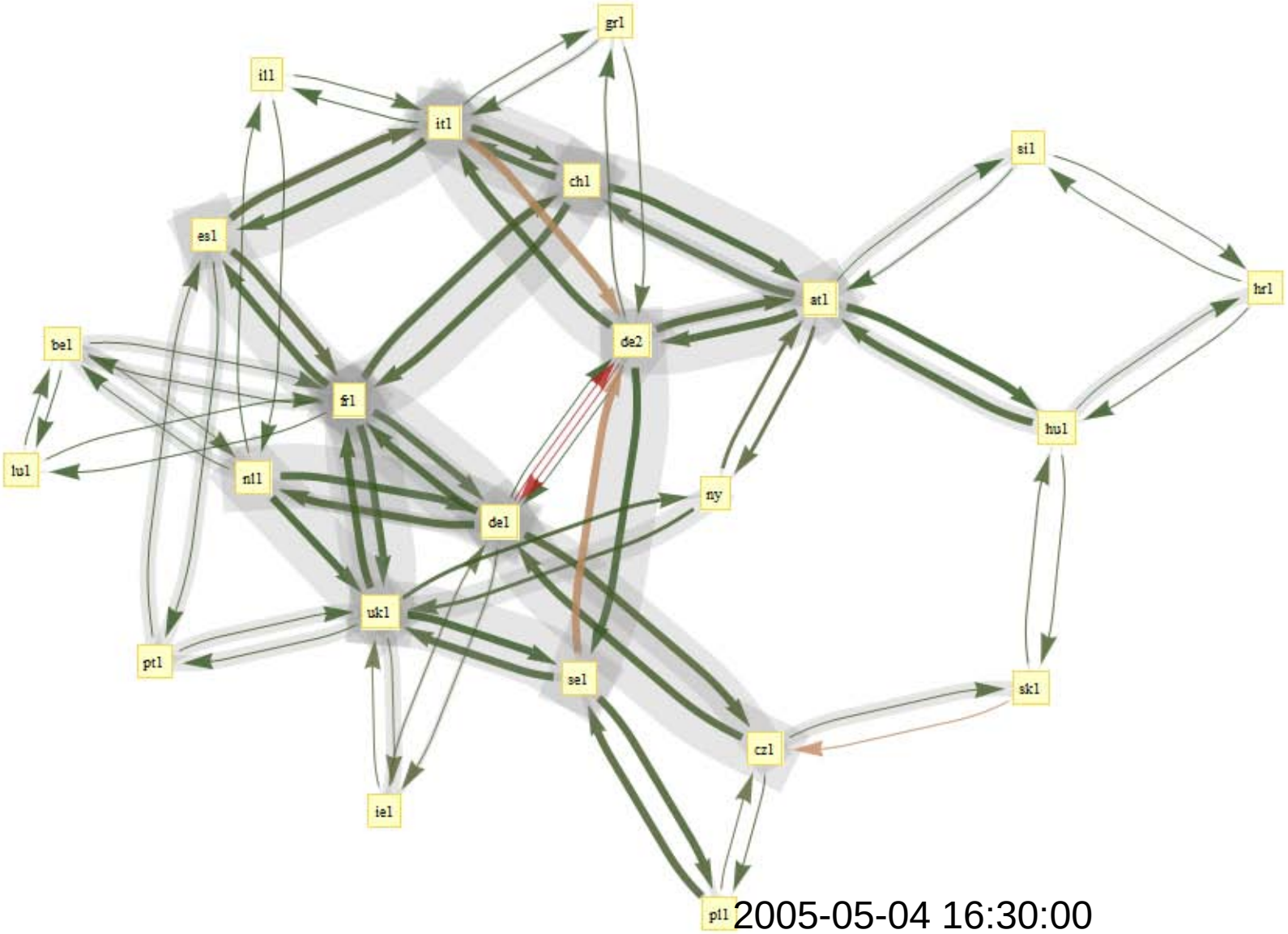
Here, $\begin{bmatrix} \Psi \\ \Phi \end{bmatrix} = \begin{bmatrix} \bar{A} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \bar{A}^T \tilde{C}^{-1} \bar{A} & -\bar{H}^T \\ \bar{H} & 0 \end{bmatrix} \begin{bmatrix} \bar{A}^T \tilde{C}^{-1} \\ 0 \end{bmatrix}$

and $\tilde{C} = \begin{bmatrix} C_1/L_1''(p_1) & & 0 \\ & \ddots & \\ 0 & & C_n/L_n''(p_n) \end{bmatrix}$

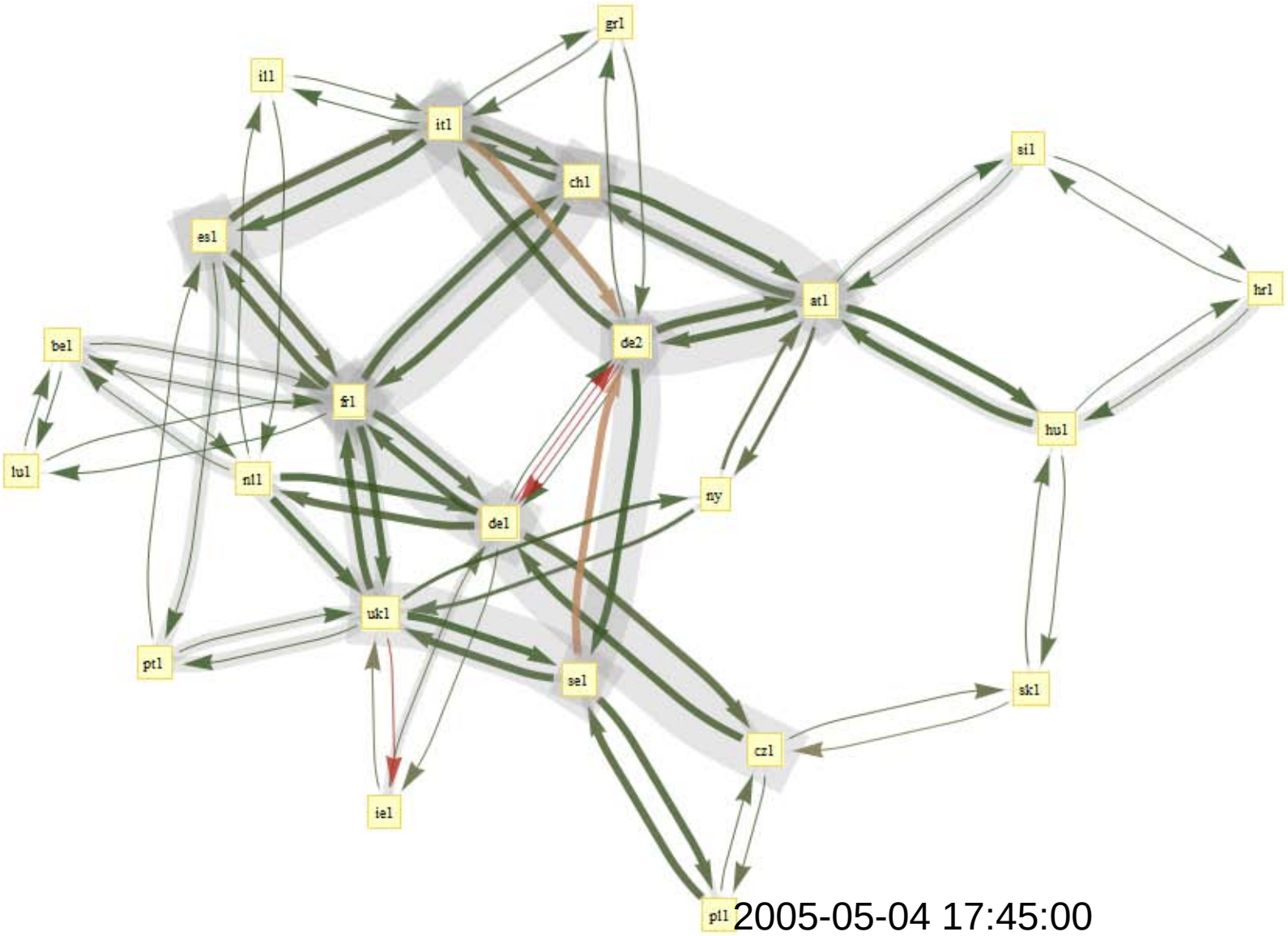
and $\bar{A}, \bar{H}$ are the adjacency matrix and the source/path matrix, restricted to paths with non-zero traffic.

GEANT data provided by WP1—
multipath routes, link capacities, and traffic matrices

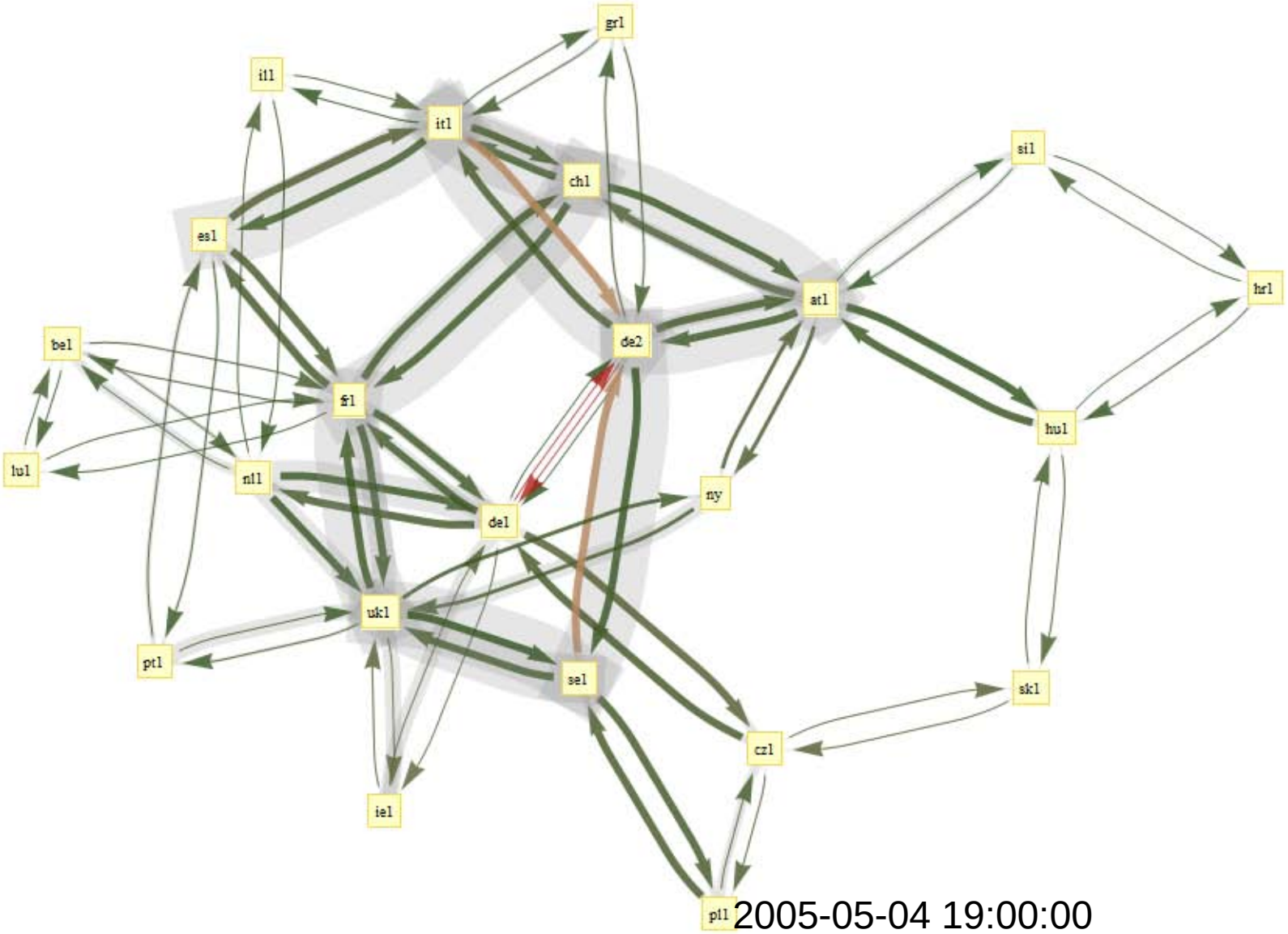




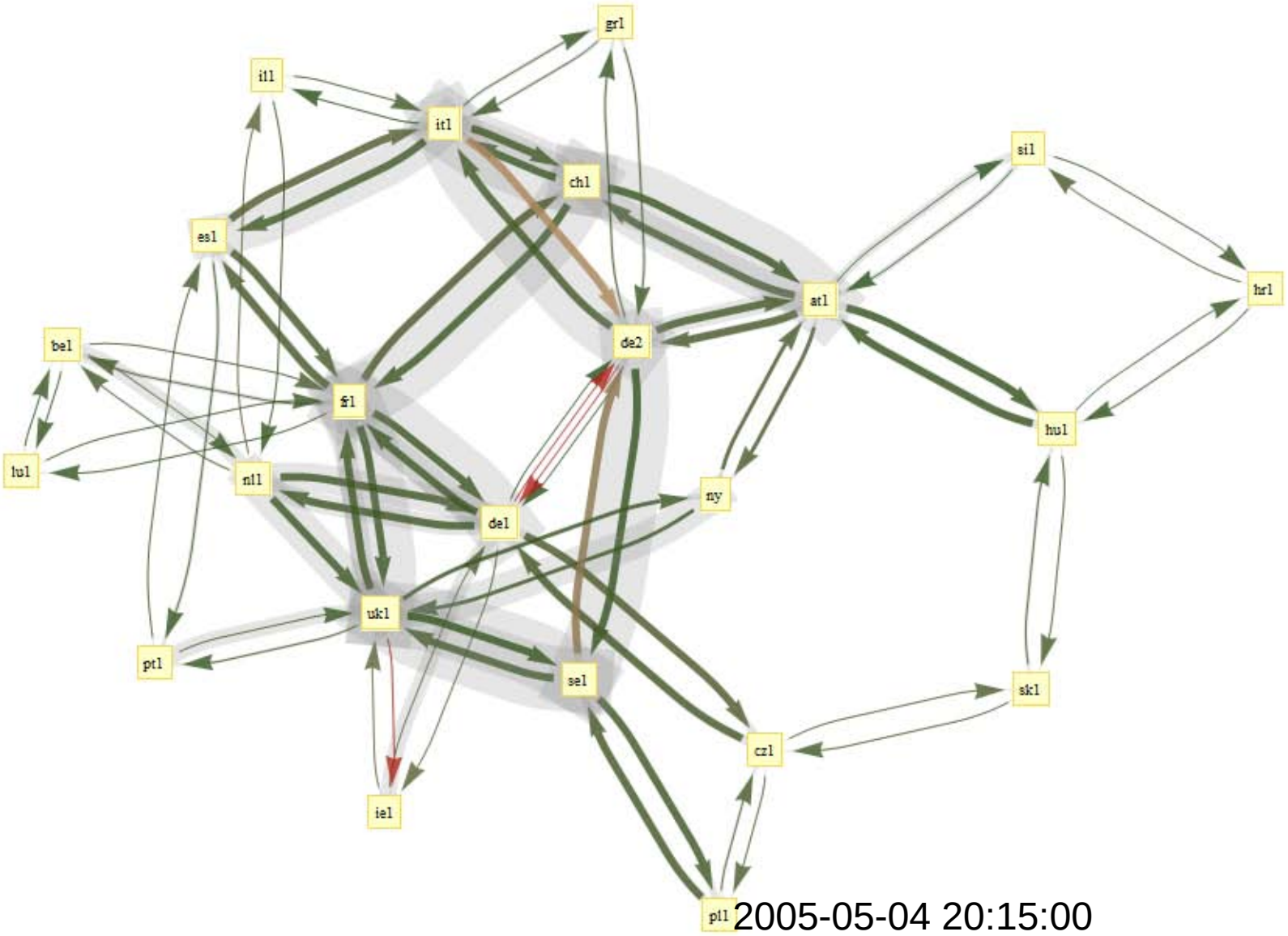

2005-05-04 16:30:00
Colours show utilization
Grey shows effective pooled capacity




2005-05-04 17:45:00
Colours show utilization
Grey shows effective pooled capacity




2005-05-04 19:00:00
Colours show utilization
Grey shows effective pooled capacity




2005-05-04 20:15:00
Colours show utilization
Grey shows effective pooled capacity

# Topic III. Can we design a congestion controller such that users react in the right way to achieve resource pooling?

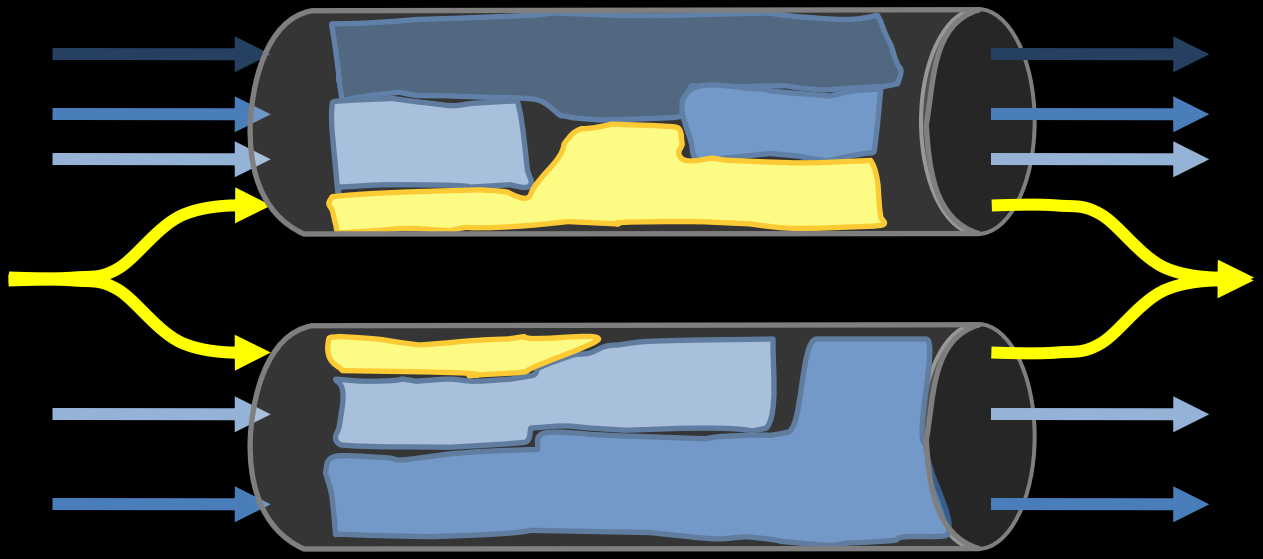
In the analysis of resource pooling, I assumed an idealized congestion controller: one which knows exactly the level of congestion on each path, and shifts its traffic onto the least congested.

To achieve this, we thought it would be a simple matter of taking a published "fluid model" of a load-balancing congestion controller, and implementing it. [Kelly+Voice, 2005]

$$\frac{d}{dt}x_r(t) = \frac{x_r(t-\tau_r)}{\tau_r}\left(a\left(1-\lambda_r(t)\right) - b_r\, y_{s(r)}(t)\,\lambda_r(t)\right)^{+[x_r(t)=0]}$$

We were wrong.

The idealized congestion control algorithm puts *all* its traffic on the least congested path. This can a failure of load balancing, when congestion levels vary.

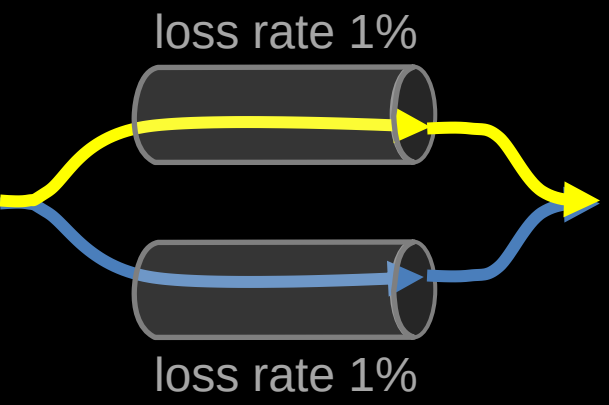


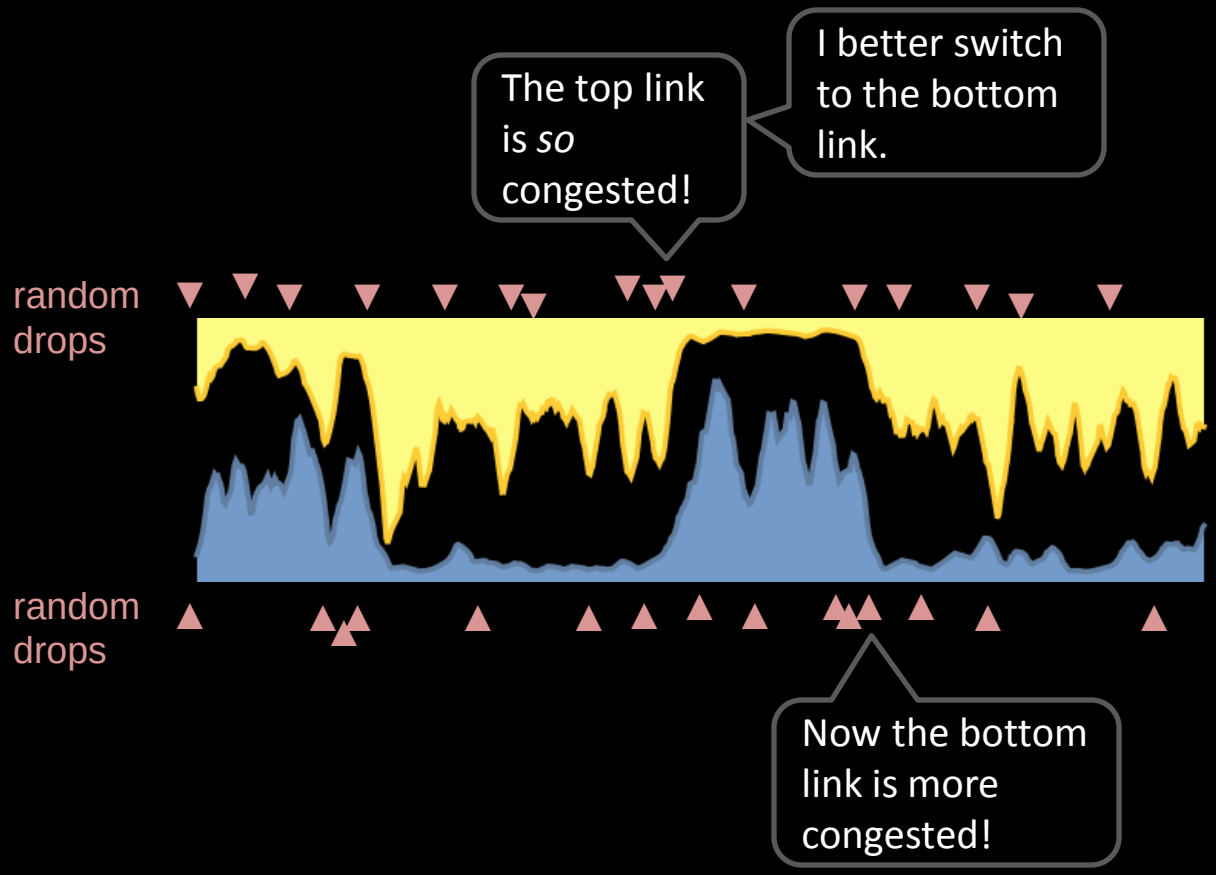
Each flow should get 1/5 of the pool.

The multipath flow should shift to using the top link. Then each flow gets 1/4 of the pool.

The multipath flow is not using the lower link, so it never learns it should shift back.

# The noisy nature of congestion feedback makes it difficult to estimate congestion levels.



The information feedback stream (packet drops, delays) is noisy. To get a good measure of the true state of the link, we have to average the signal.

But congestion is not static. To react promptly to changes in congestion, we have to look only at recent data about congestion, and we should constantly probe all paths.

**The Zen of resource pooling**

To pool resources effectively, the end-system should not try too hard to pool resources.

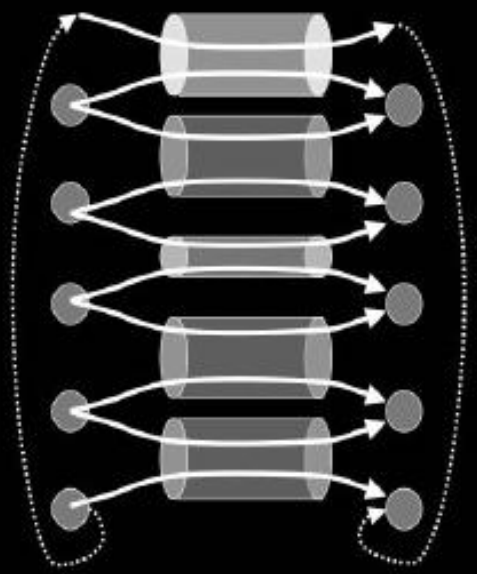
Instead, it should maintain **equipoise**, i.e. balance its traffic rate across its paths, *to the extent necessary* to achieve resource pooling.

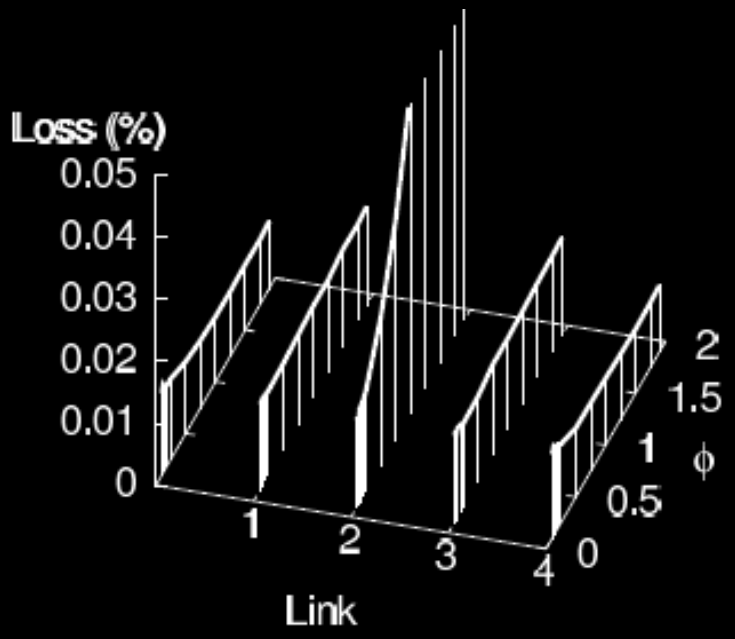
We devised a parameterized family of multipath congestion control algorithms, indexed by $\varphi \in [0,2]$, to investigate the tradeoff between load balancing and equipoise.


the idealized congestion controller, inspired by Kelly+Voice

run independent TCP control on each path

# How good is this congestion controller at achieving resource pooling, in a static network?



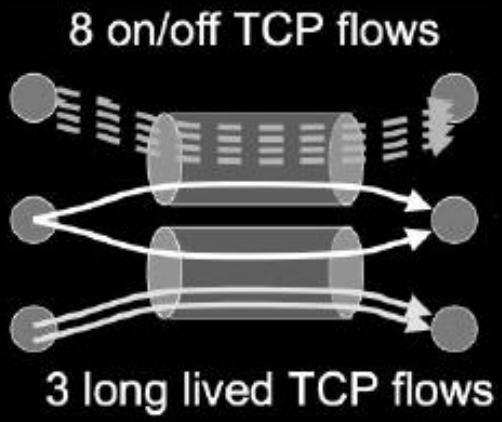


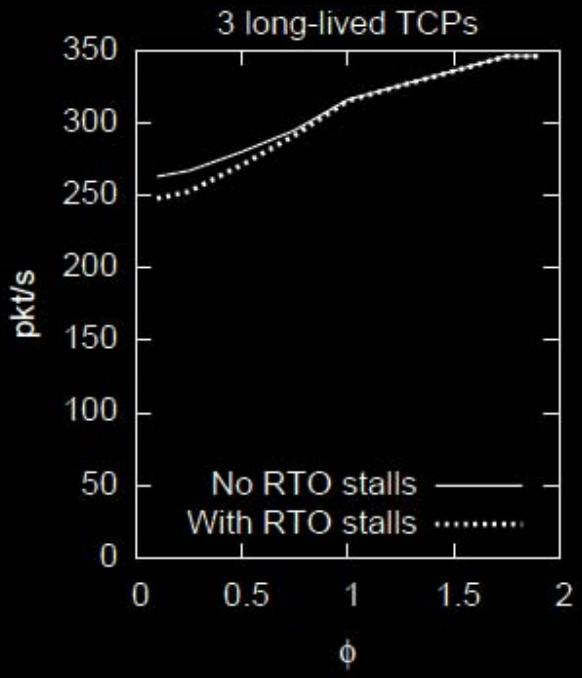

$\varphi$=0 $\varphi$=2

**good at resource pooling:**
even though the links have unequal capacities, congestion is balanced perfectly

**bad at resource pooling:**
the low-capacity link is highly congested

# How good is this congestion controller at achieving resource pooling, in a dynamic network?




$\varphi$=0 $\varphi$=2

**bad at resource pooling:**
shifts too enthusiastically to the less loaded link, and is slow to learn when the other link improves

**good at resource pooling:**
constantly probes both links, so learns quickly when congestion levels change

our choice

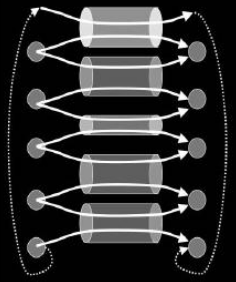
the naïve coupled congestion controller, inspired by Kelly+Voice

run independent TCP control on each path

$\varphi$=0

$\varphi$=2

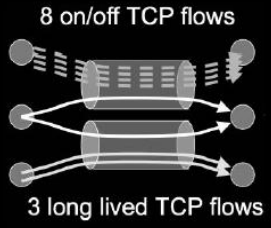
static network



**good at resource pooling:**
even though the links have unequal capacities, congestion is balanced perfectly

**bad at resource pooling:**
the low-capacity link is highly congested

dynamic network



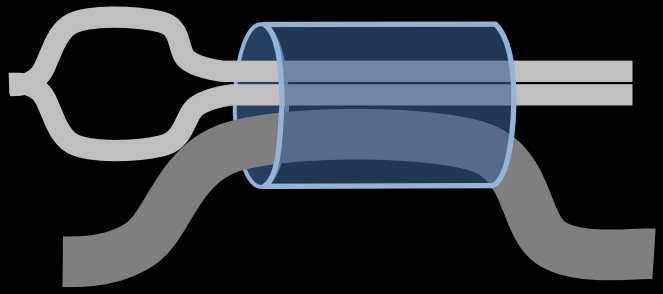

**bad at resource pooling:**
shifts too enthusiastically to the less loaded link, and is slow to learn when the other link improves

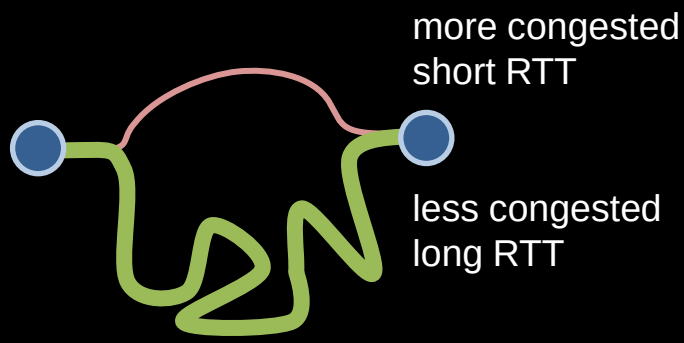
**good at resource pooling:**
constantly probes both links, so learns quickly when congestion levels change

# We tweaked the $\varphi$ algorithm, to ensure fairness with TCP.

We assign a weight to each link, and run a weighted version of the $\varphi$-algorithm. We have an adaptive algorithm for choosing the weights, to guarantee that

- the multipath user gets as least as much throughput as if he/she used the best single path
- the multipath user takes no more bandwidth on any link than a single-path TCP would.

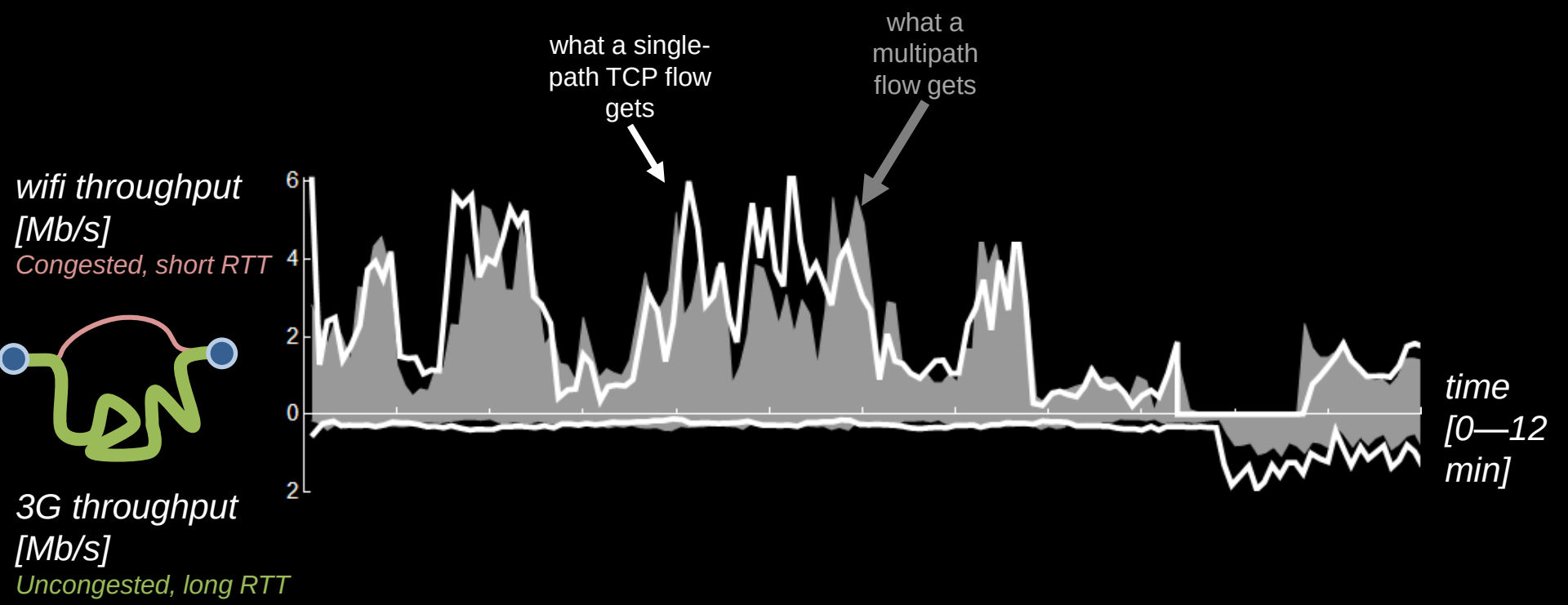




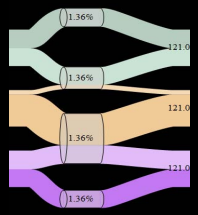
The 3G link has lower drop probability. We'd prefer to use the 3G link, to get resource pooling.

But the 3G link has a long RTT, so single-path TCP gets low throughput. We shouldn't take any more than single-path TCP would.

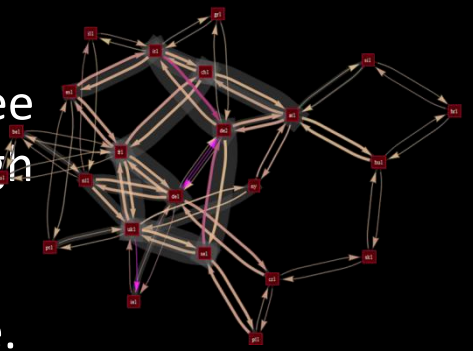
Therefore we need to keep some traffic on the wifi link, so that the multipath user gets as good throughput as if he used single-path TCP.



# We have a working implementation of multipath transport.

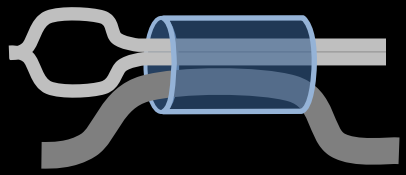



It achieves a reasonable degree of load balancing.



This means that the network achieves some degree of resource pooling (subject to having good enough routes).

It maintains a reasonable degree of equipoise. This means it adapts sensibly to fluctuating congestion.



It is guaranteed to be fair compared to TCP.

I E T F

The algorithm is ready for deployment. It is an experimental RFC in the mptcp working group at the IETF.

The work has been submitted to a top conference.

# Ongoing research topics

How can we use poolability scores to help design a multipath routing algorithm? Is it sufficient to rely on end-host addressing?

The poolability analysis assumed idealized congestion control, which shifts all its traffic onto the least congested paths. But equipoise and fairness mean that such behaviour is not good. How does this impact resource pooling?

What is a principled way to choose the tradeoff between load balancing and equipoise?

What is the impact of resource pooling on competition and pricing?