What mathematicians should

know about the Internet:

a case study

Overlay TCP for multi-path routing and congestion control

Han, Shakkottai, Hollot, Srikant, Towsley,

ENS-INRIA ARC-TCP Workshop 2003

ABSTRACT. We consider the problem of multi-path routing in the Internet. Currently, Internet routing protocols select only a single path between a source and a destination. However, due to many policy routing decisions, single-path routing may limit the achievable throughput. In this paper, we envision a scenario where application-level routers are overlaid on the Internet to allow multi-path routing. Using minimal congestion feedback signals from the overlay routers, we present an algorithm that can be implemented at the sources to stably and optimally split the flow between each source-destination pair. We then show that the connection-level throughput region of such a multi-path routing/congestion control scheme can be larger than that of a single-path congestion control scheme.

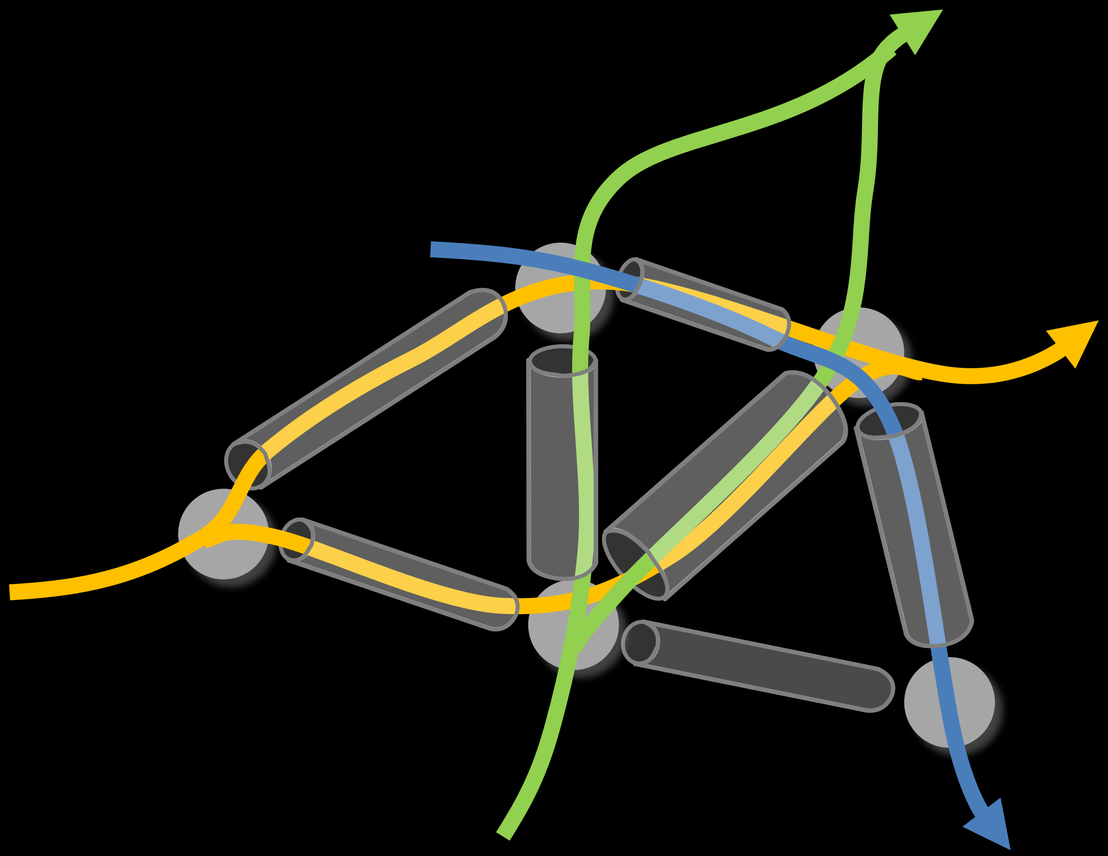
Stability of end-to-end algorithms for joint routing and rate control

Kelly and Voice, *Computer Communication Review* 2005

ABSTRACT. Dynamic multi-path routing has the potential to improve the reliability and performance of a communication network, but carries a risk. Routing needs to respond quickly to achieve the potential benefits, but not so quickly that the network is destabilized. This paper studies how rapidly routing can respond, without compromising stability.

We present a sufficient condition for the local stability of end-to-end algorithms for joint routing and rate control. The network model considered allows an arbitrary interconnection of sources and resources, and heterogeneous propagation delays. The sufficient condition we present is decentralized: the responsiveness of each route is restricted by the round-trip time of that route alone, and not by the roundtrip times of other routes. Our results suggest that stable, scalable load-sharing across paths, based on end-to-end measurements, can be achieved on the same rapid time-scale as rate control, namely the time-scale of round-trip times.

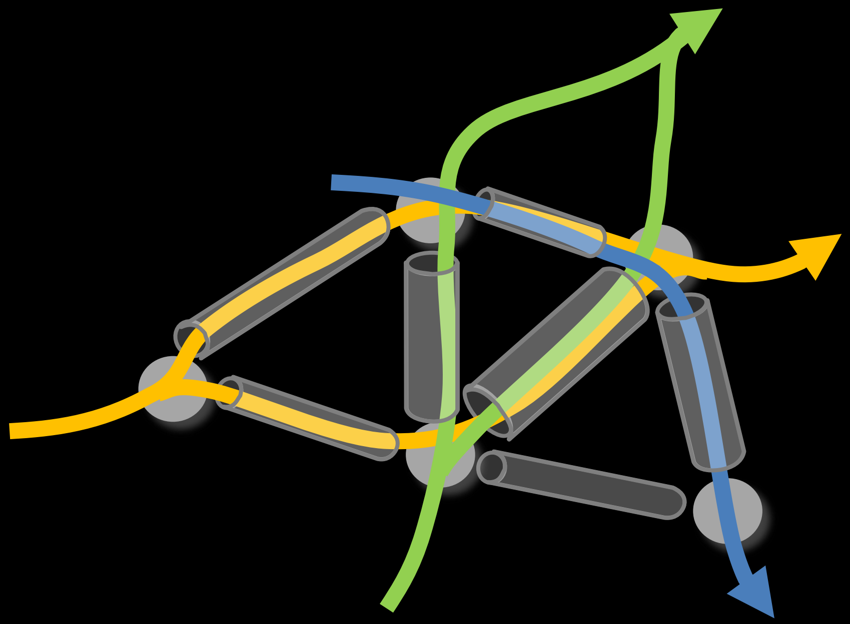
What problem is being solved by joint routing and rate control?



Given a network consisting of

- a set of links indexed by *j*, each with its own penalty function C_j
- a set of users indexed by *s*, each with his/her own utility function U_s
- and a set of routes indexed by *r*,

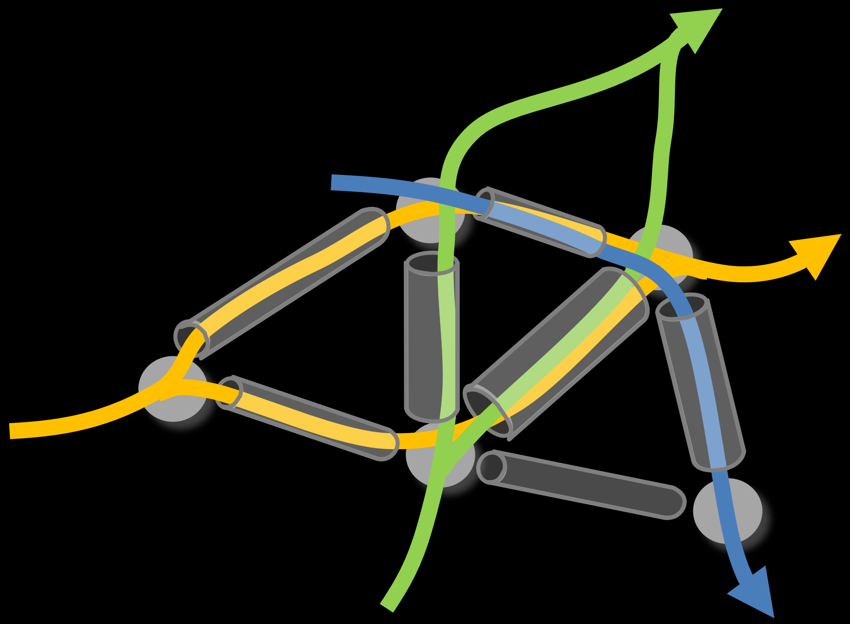
What problem is being solved by end-to-end algorithms for joint routing and rate control?



Find a distributed algorithm for solving the network utility maximization problem. That is,

where the fixed point of this system of equations solves the utility maximization problem.

What problem is being solved by stable end-to-end algorithms for joint routing and rate control?



Find a distributed algorithm for solving the network utility maximization problem.

The system of differential equations should be stable (or at least locally stable, in *y* and *z*) about the equilibrium.

What are the limits of this solution?

1. It is tricky to evolve TCP to implement the principles of the Kelly+Voice algorithm.

2. Network systems people don’t see the applicability of fluid stability results.

3. Network systems people don’t see the point in utility maximization.

"Mathematicians are like Frenchmen: whatever you say to them they translate into their own language and forthwith it is something entirely different."

Goethe, 1829

“All mathematical models are wrong.

Some are good wrong, some are bad wrong.”

Han, Shakkottai, Hollot, Srikant, Towsley, 2003

Kelly+Voice, 2005

UCL / Trilogy project, 2008

IETF working group, 2009

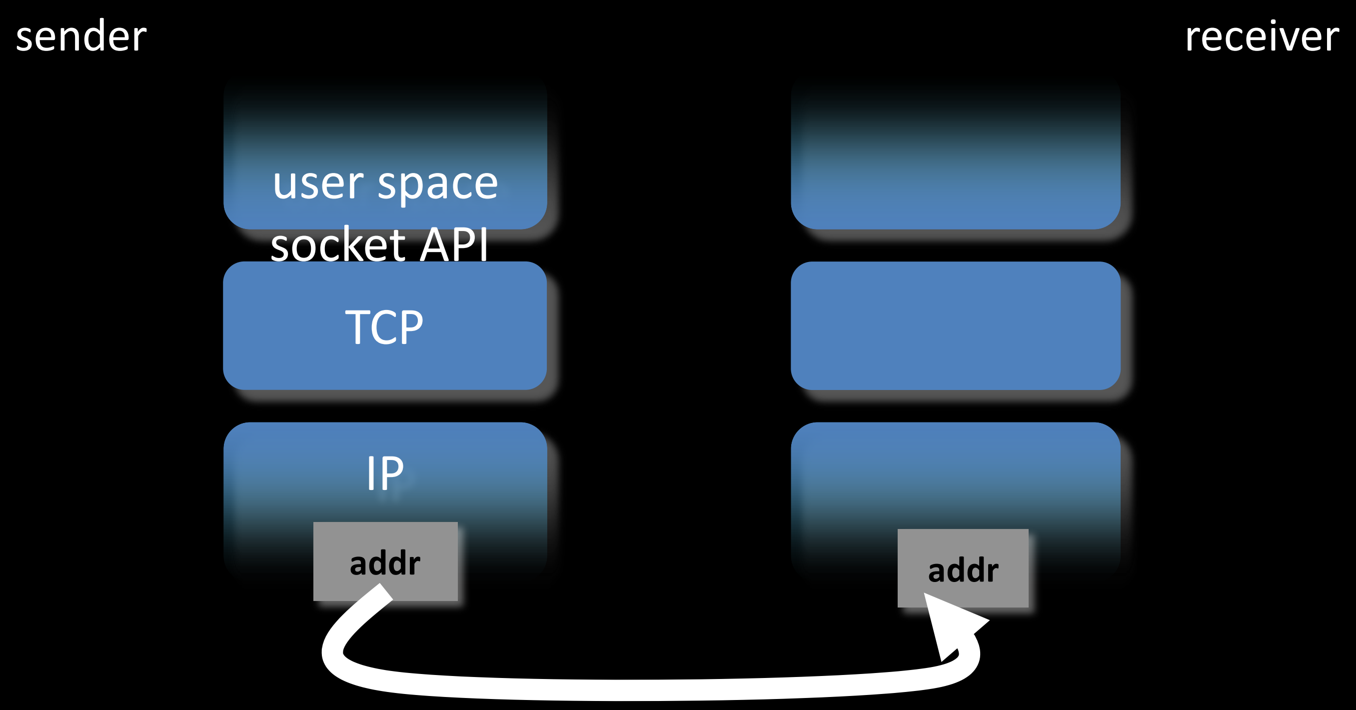
IETF experimental standard, 2011?

widespread interest, 2011

What does multipath look like to an Internet systems person?

Why does the Internet need multipath?

If the Internet needs multipath, why don’t we have it already?

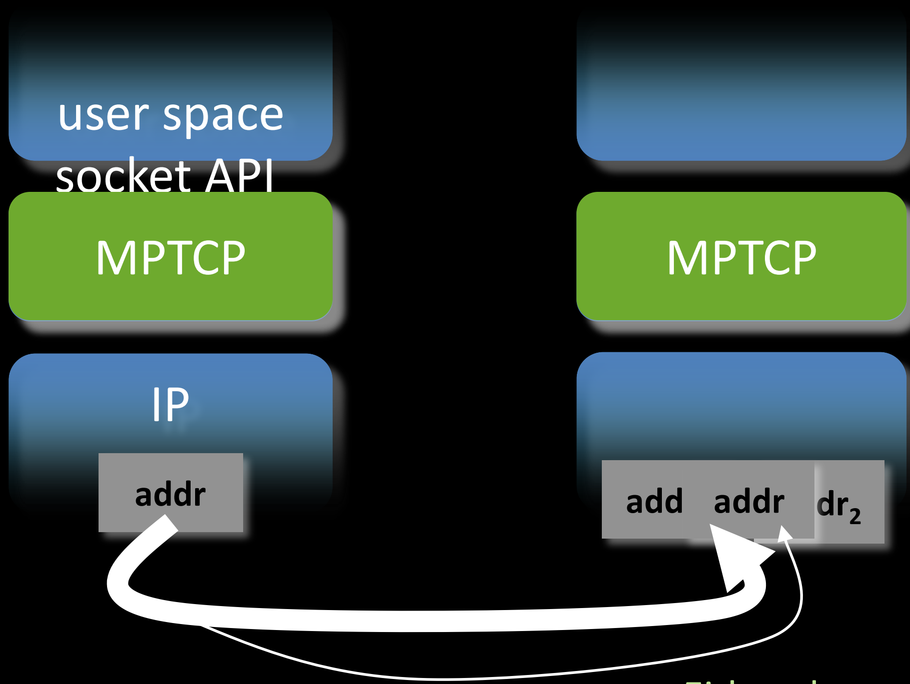

sender
receiver
user space
socket API
TCP
IP
addr
addr

We designed MPTCP to be a drop-in replacement for TCP.

sender

receiver

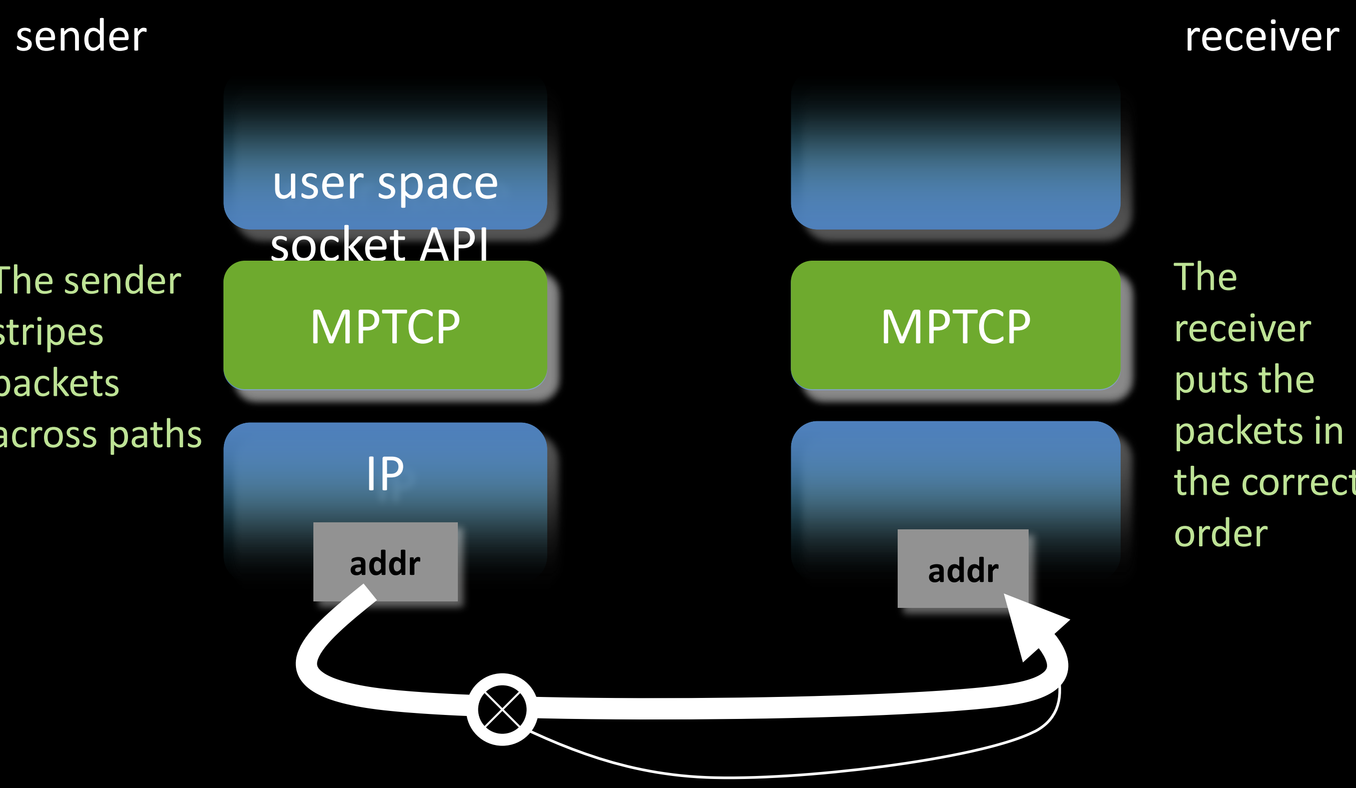
The sender stripes packets across paths



The receiver puts the packets in the correct order

Either the receiver or the sender has multiple addresses, so the sender can use multiple paths

We designed MPTCP to be a drop-in replacement for TCP.

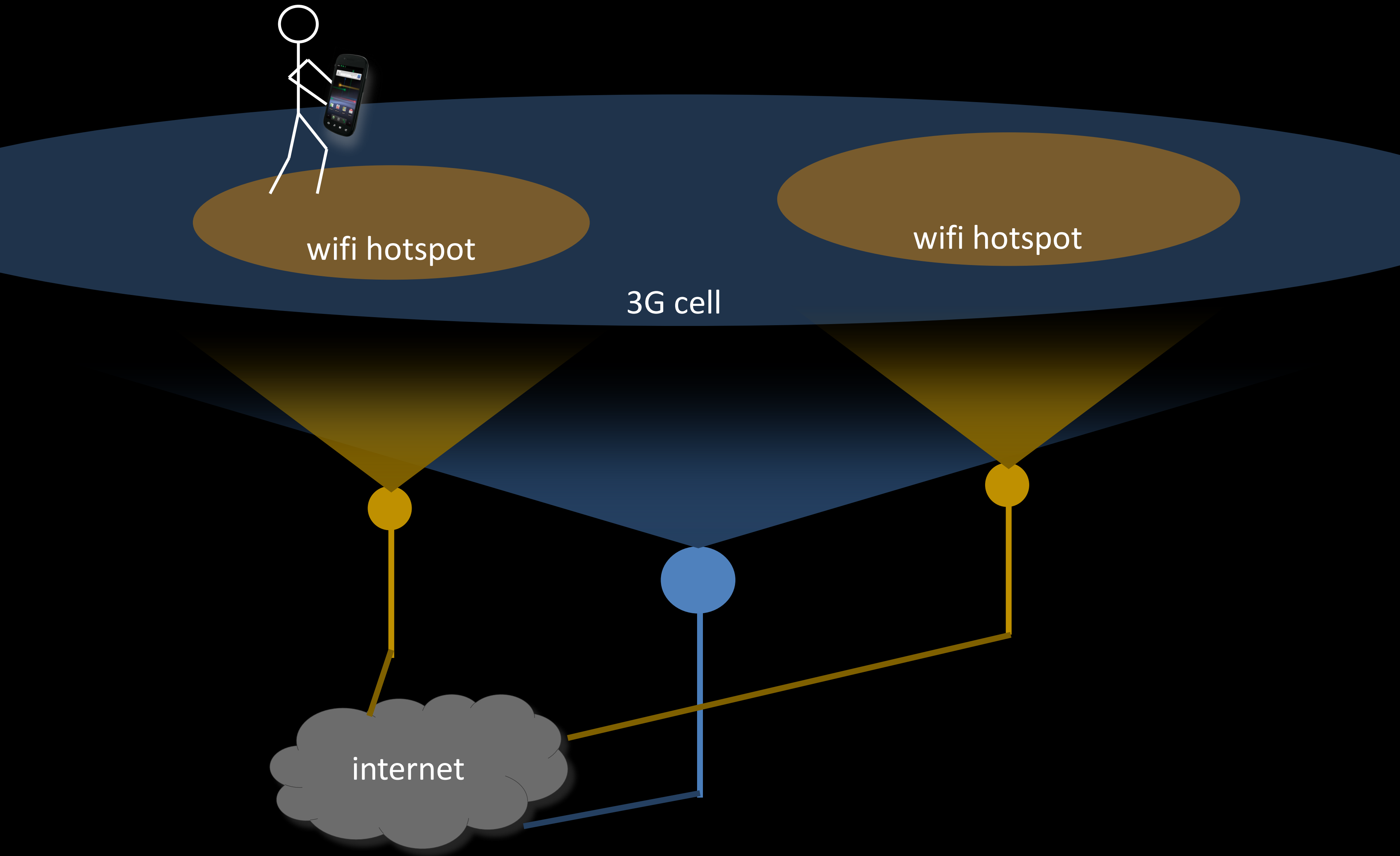


The sender stripes packets across paths

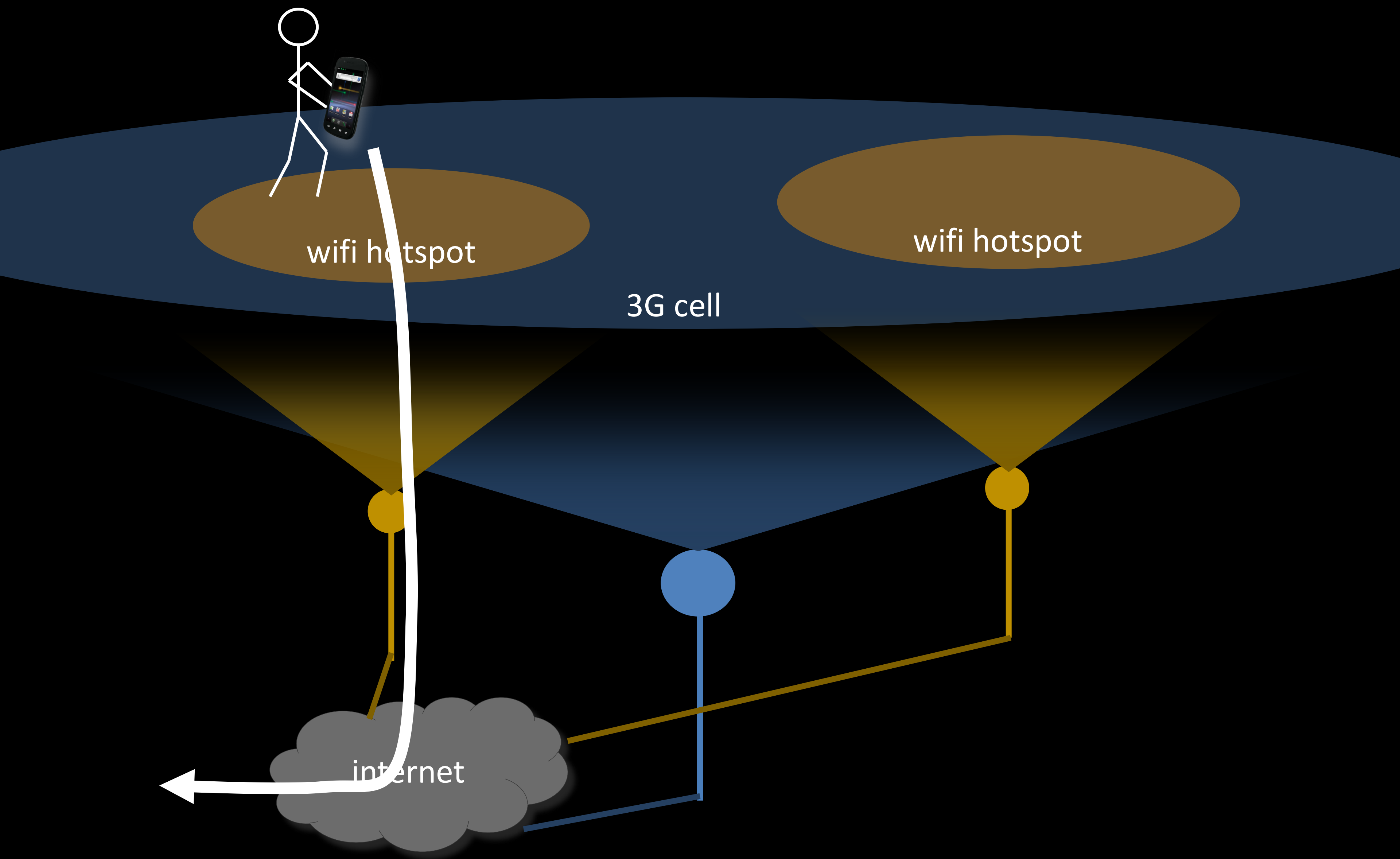
The receiver puts the packets in the correct order

Or, the Internet has some direct mechanisms for giving you 'lucky dip' access to multiple paths

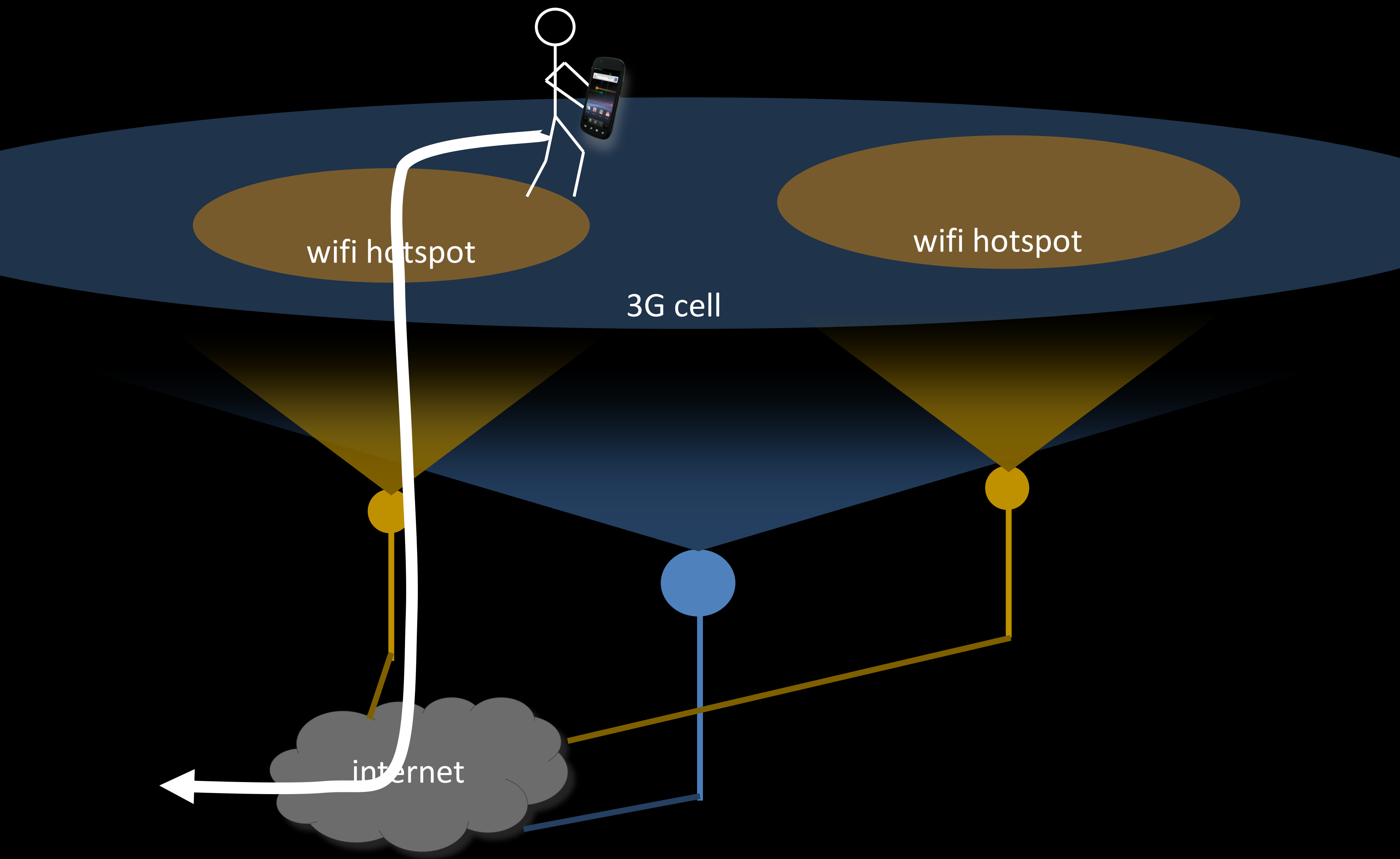
Can multipath help with mobile hand-offs?



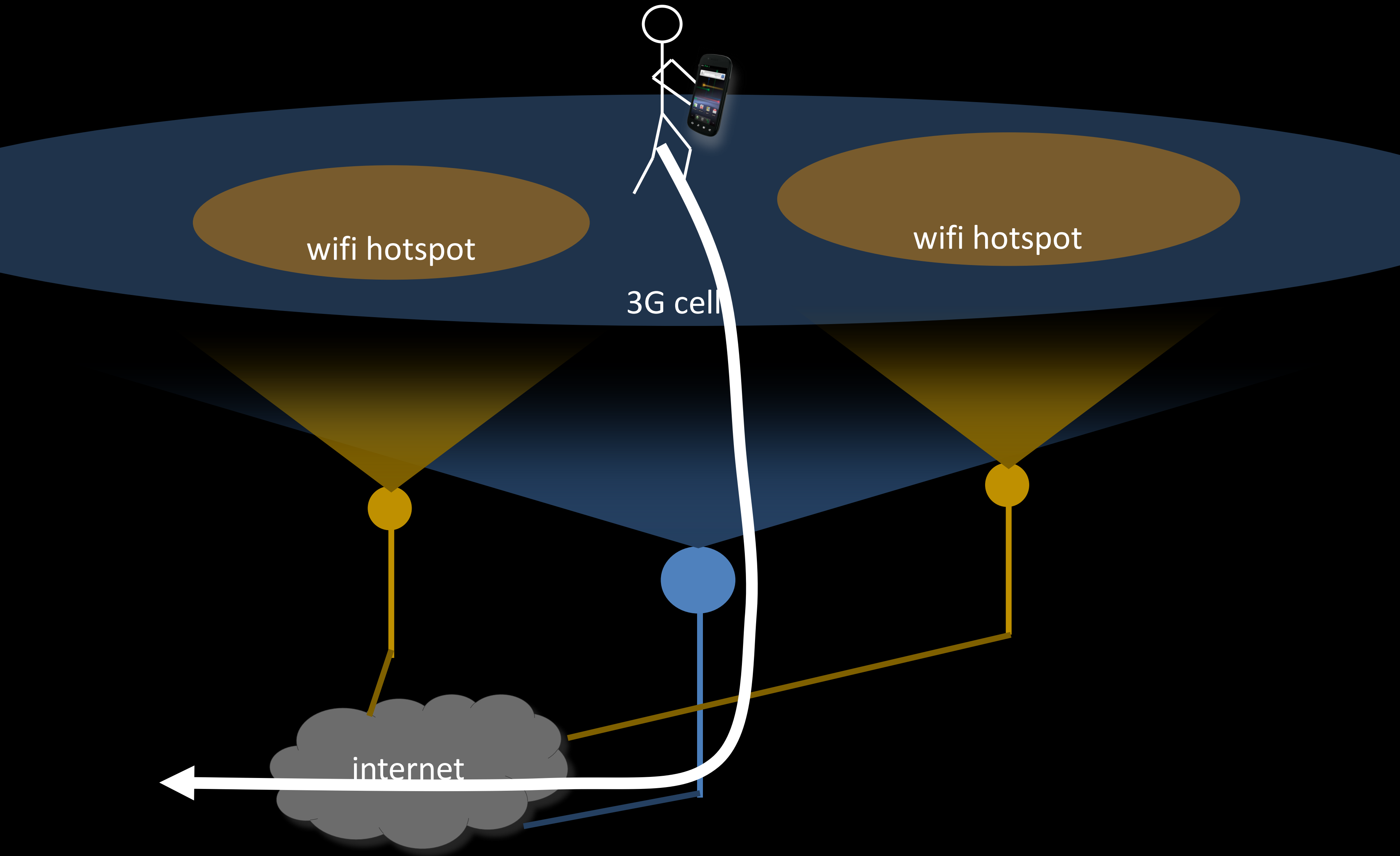
Can multipath help with mobile hand-offs?



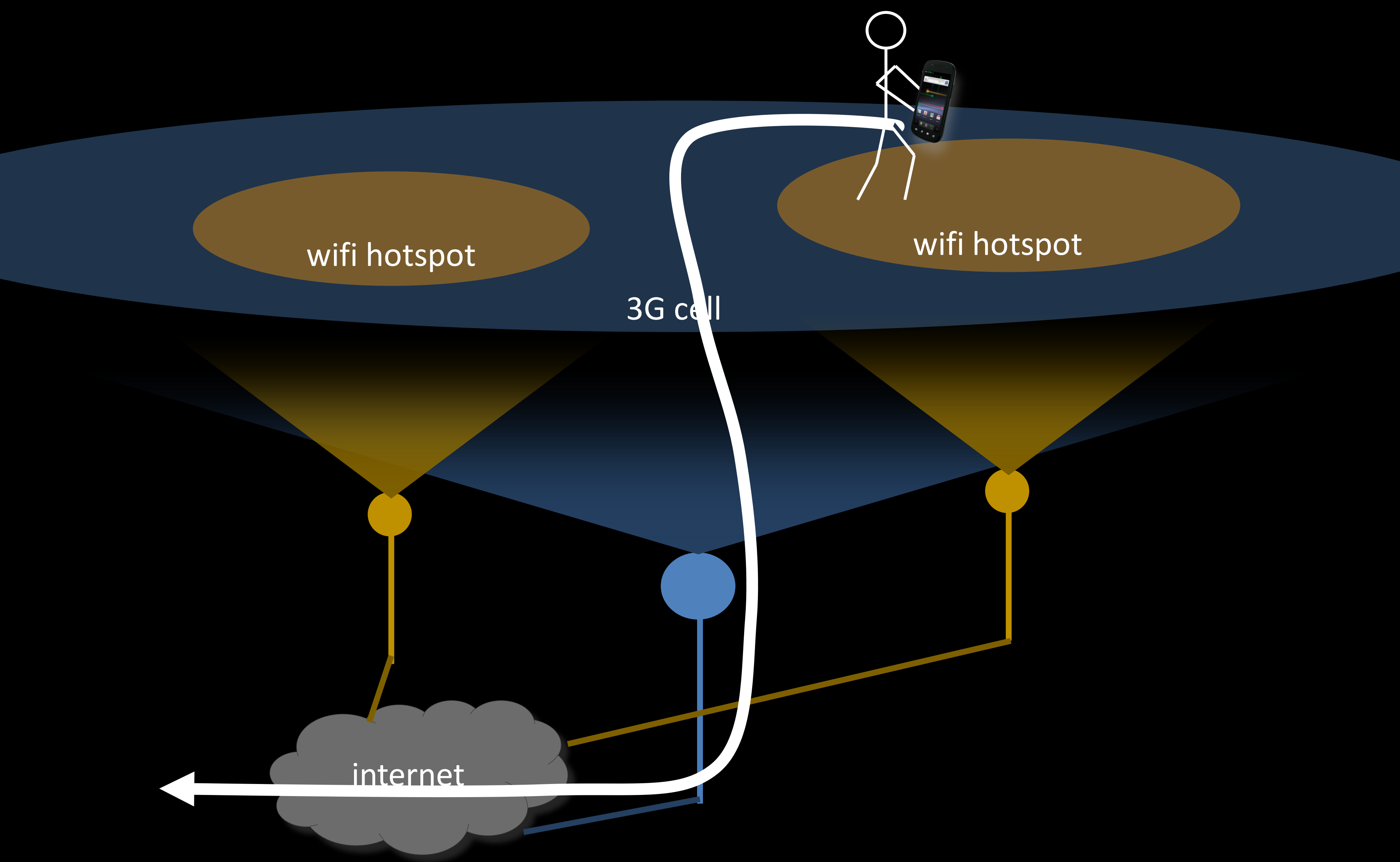
Can multipath help with mobile hand-offs?



Can multipath help with mobile hand-offs?

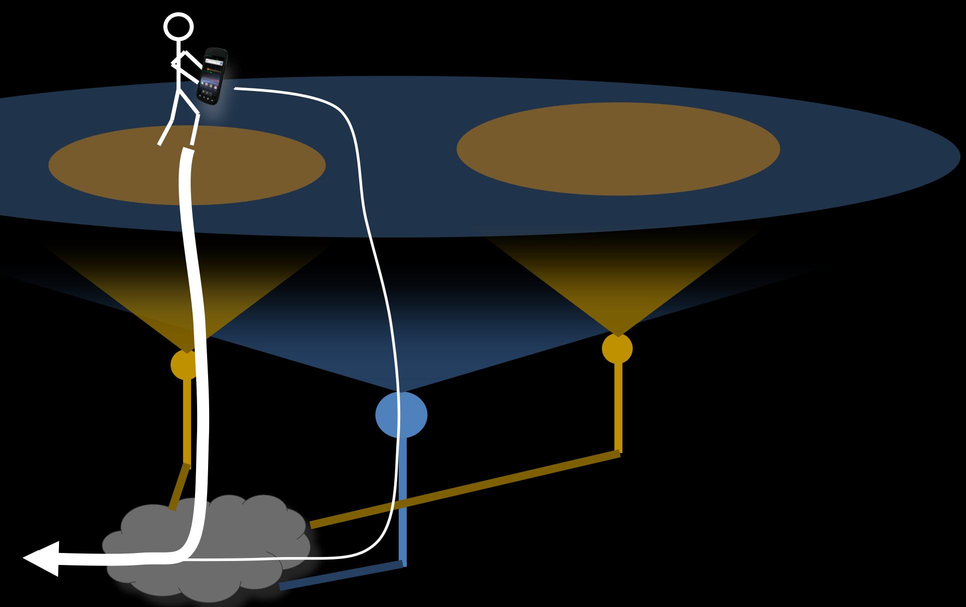


Can multipath help with mobile hand-offs?



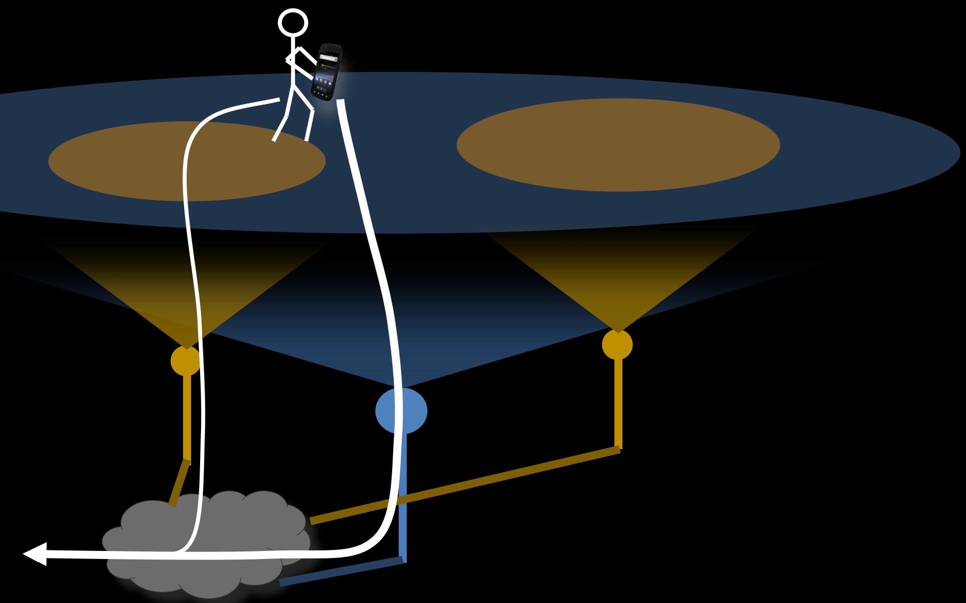
Can multipath help with mobile hand-offs?

If your phone uses both radios simultaneously, you needn't experience any interruption.



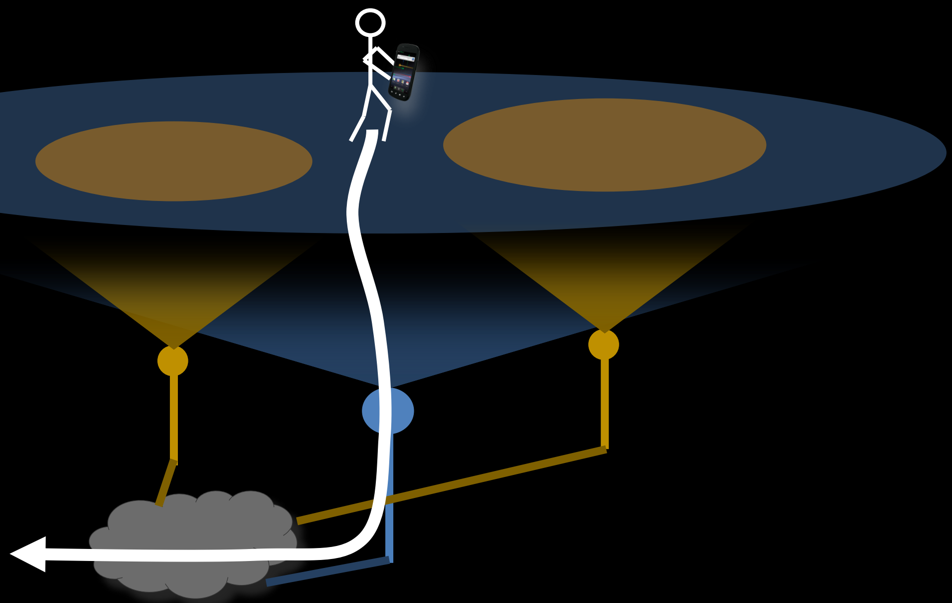
Can multipath help with mobile hand-offs?

If your phone uses both radios simultaneously, you needn’t experience any interruption.



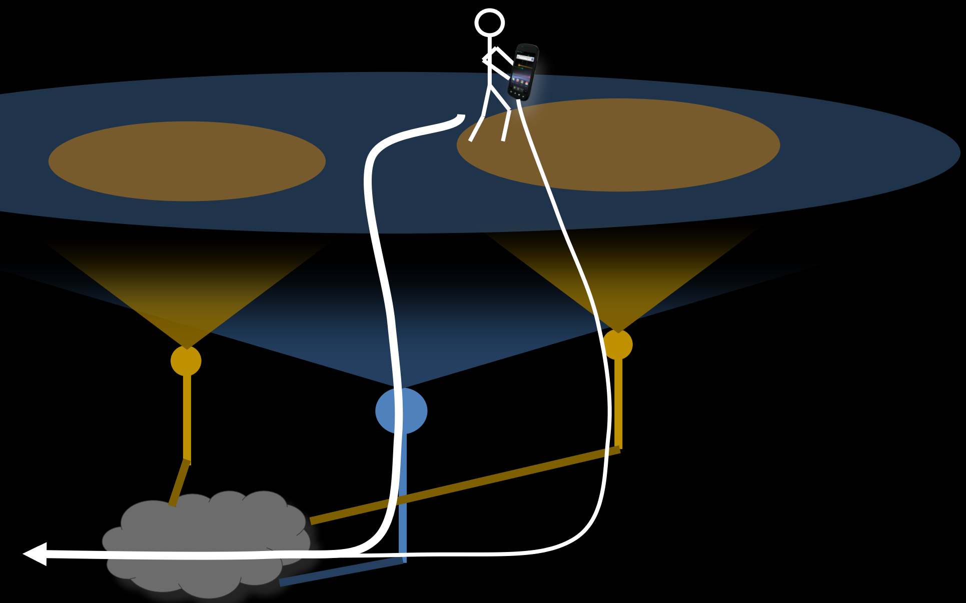
Can multipath help with mobile hand-offs?

If your phone uses both radios simultaneously, you needn't experience any interruption.



Can multipath help with mobile hand-offs?

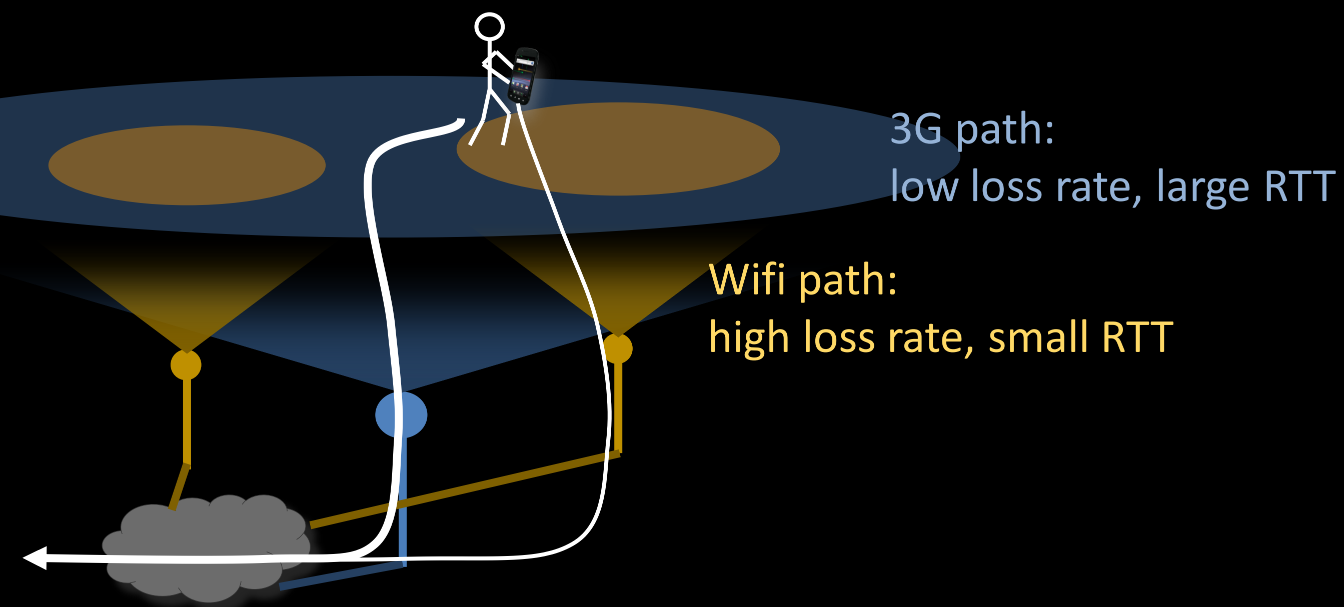
If your phone uses both radios simultaneously, you needn't experience any interruption.



Can multipath help with mobile hand-offs?

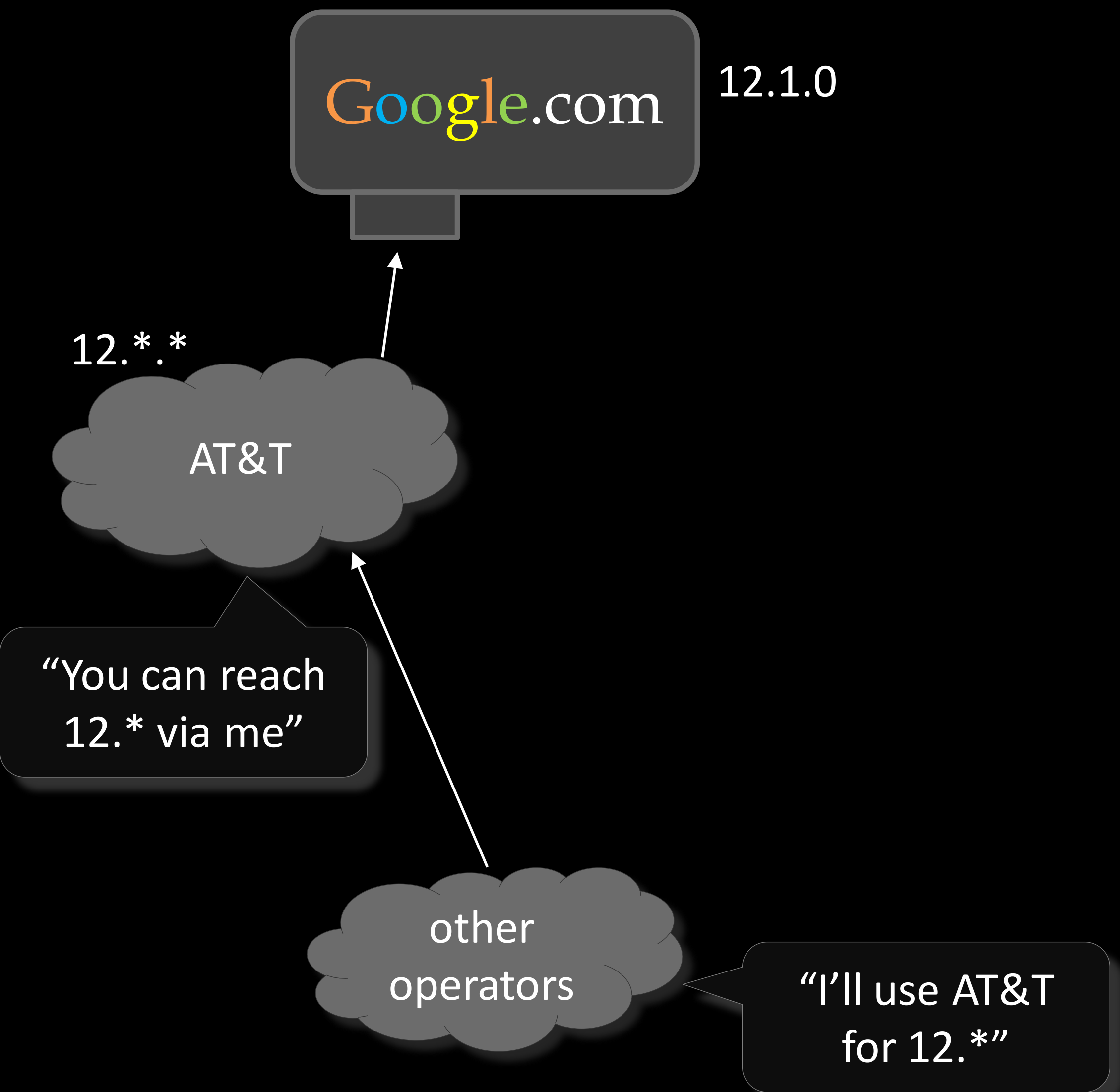
If your phone uses both radios simultaneously, you needn't experience any interruption.

How should it balance traffic across dissimilar paths?



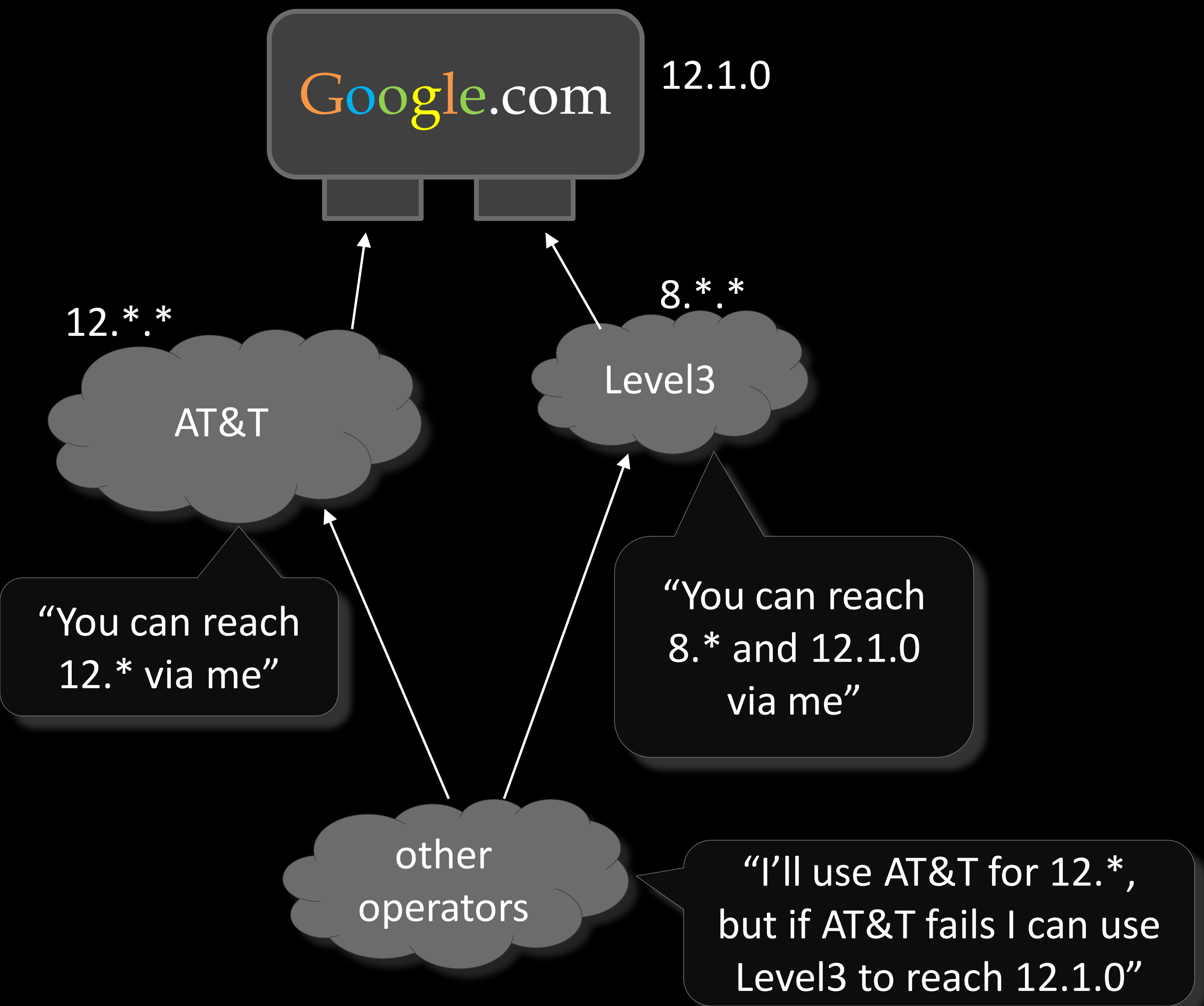
Multi-homed web sites

(a) with classic hierarchical routing



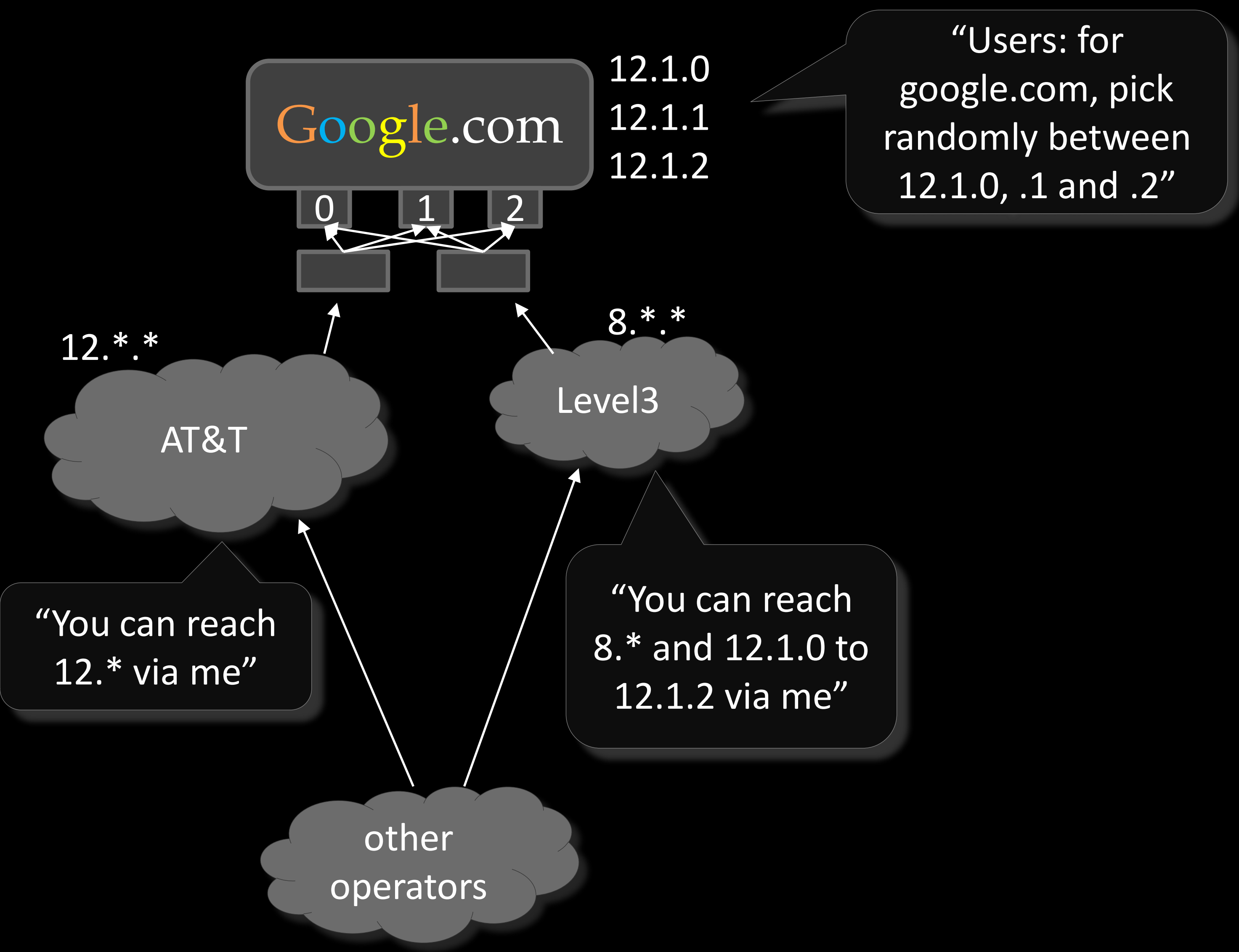
Multi-homed web sites

(b) with redundancy, in case links fail



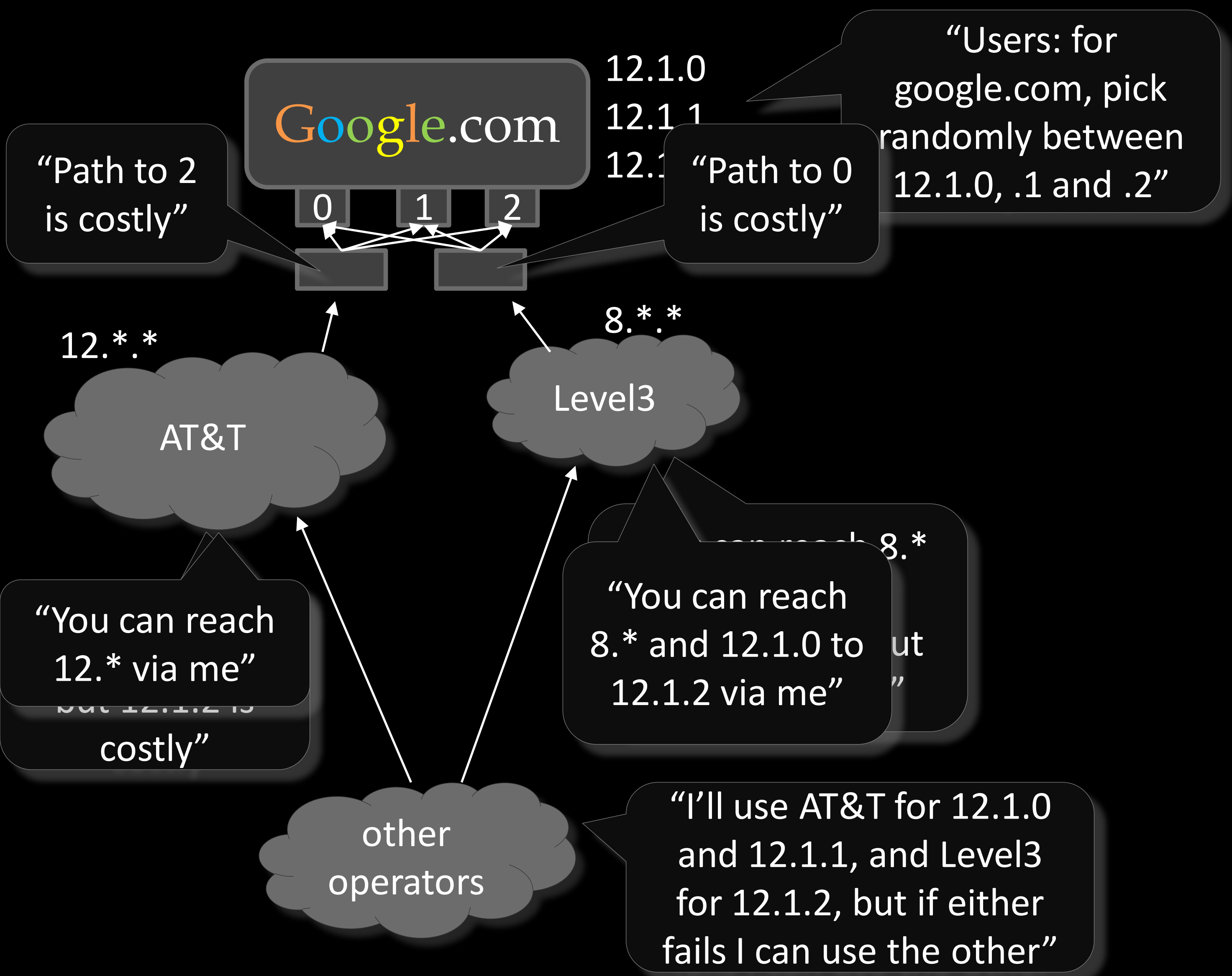
Multi-homed web sites

(c) with load balancing across the gateway machines

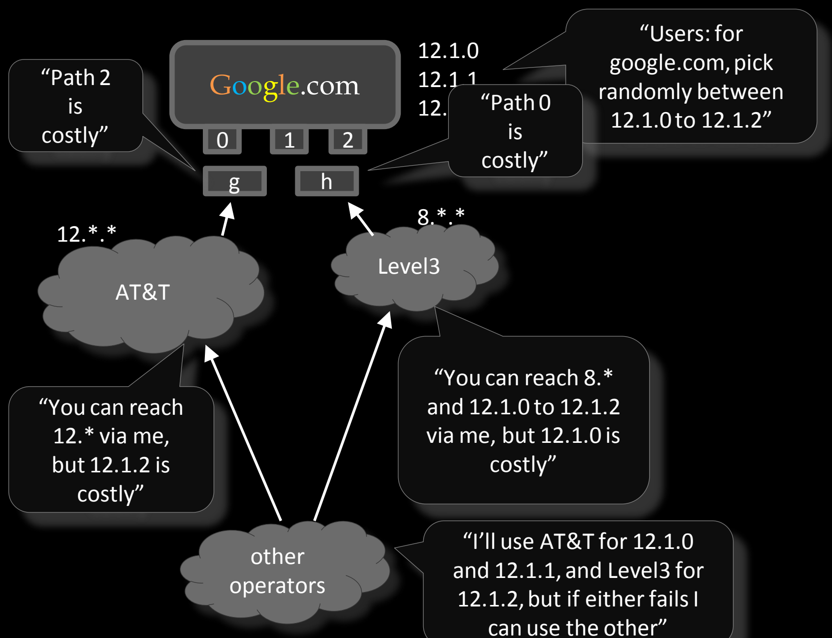


Multi-homed web sites

(c) with load balancing across the gateway machines



Multi-homed web sites

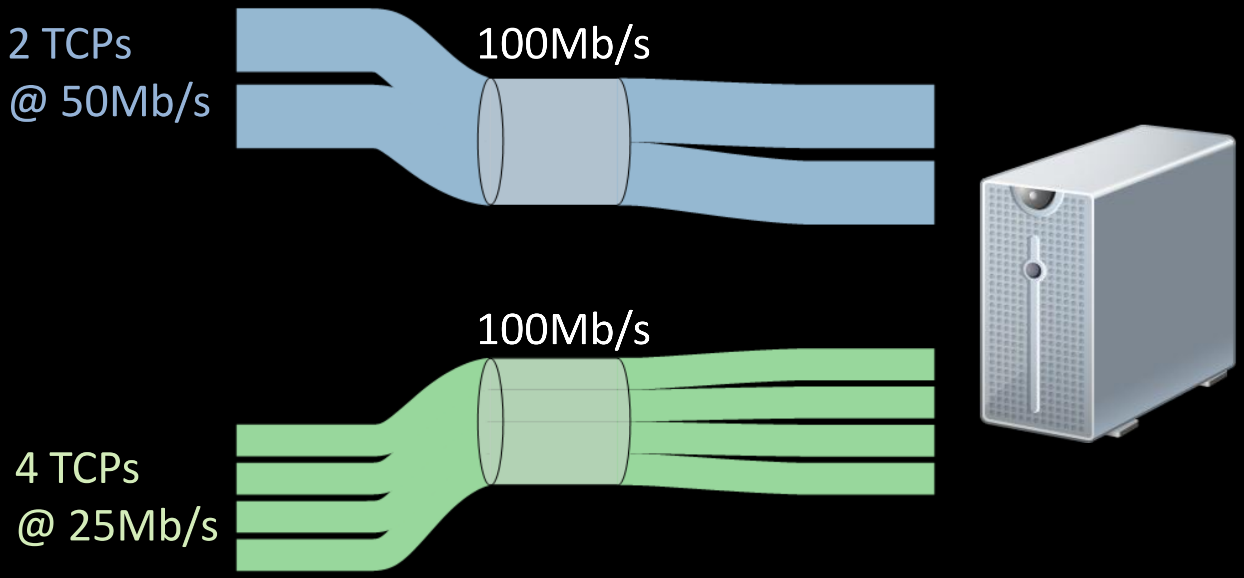


The limited resources are the memory and CPU needed by routers to remember specific paths and costs.

It can take hours or days for path choices to stabilize.

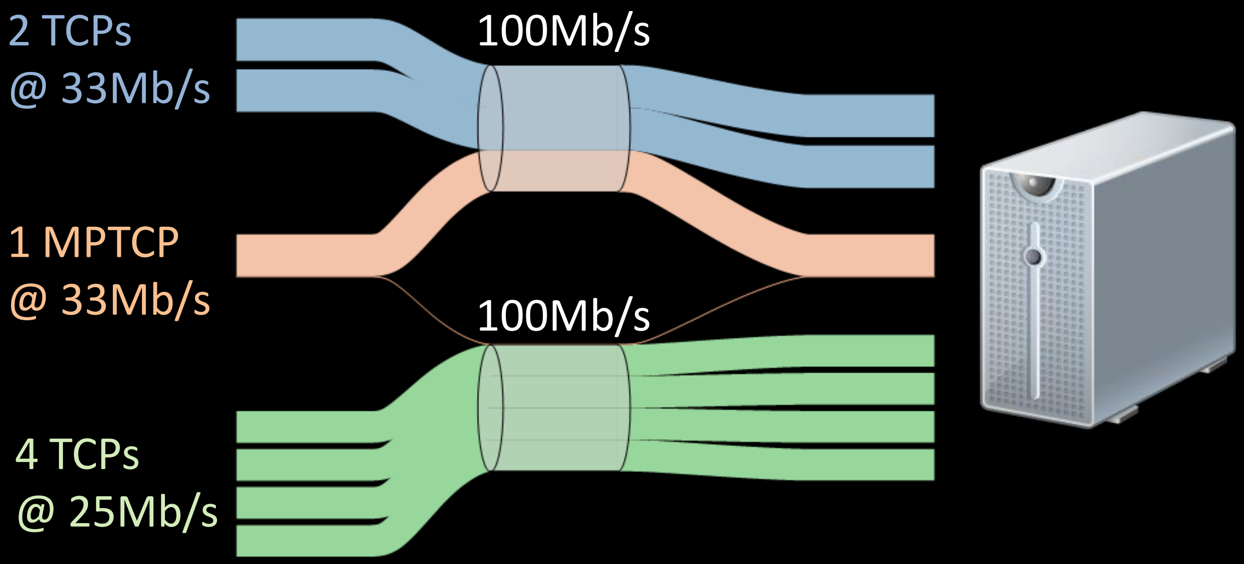
Multi-homed web sites

(d) with Kelly+Voice multipath TCP to balance traffic



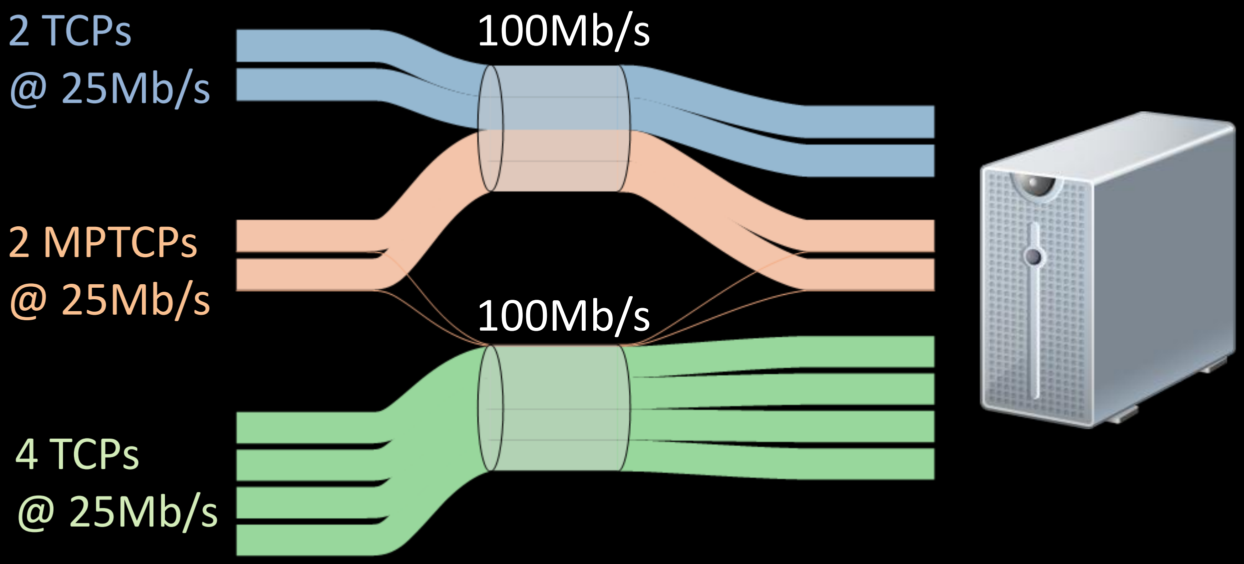
Multi-homed web sites

(d) with Kelly+Voice multipath TCP to balance traffic



Multi-homed web sites

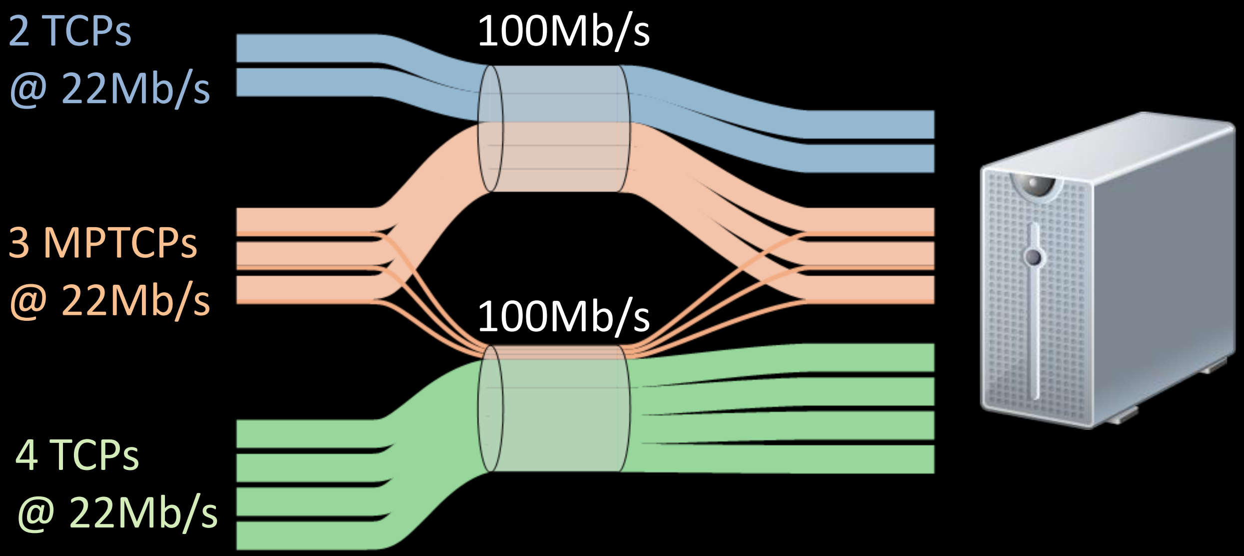
(d) with Kelly+Voice multipath TCP to balance traffic



The total capacity, 200Mb/s, is shared out evenly between all 8 flows.

Multi-homed web sites

(d) with Kelly+Voice multipath TCP to balance traffic

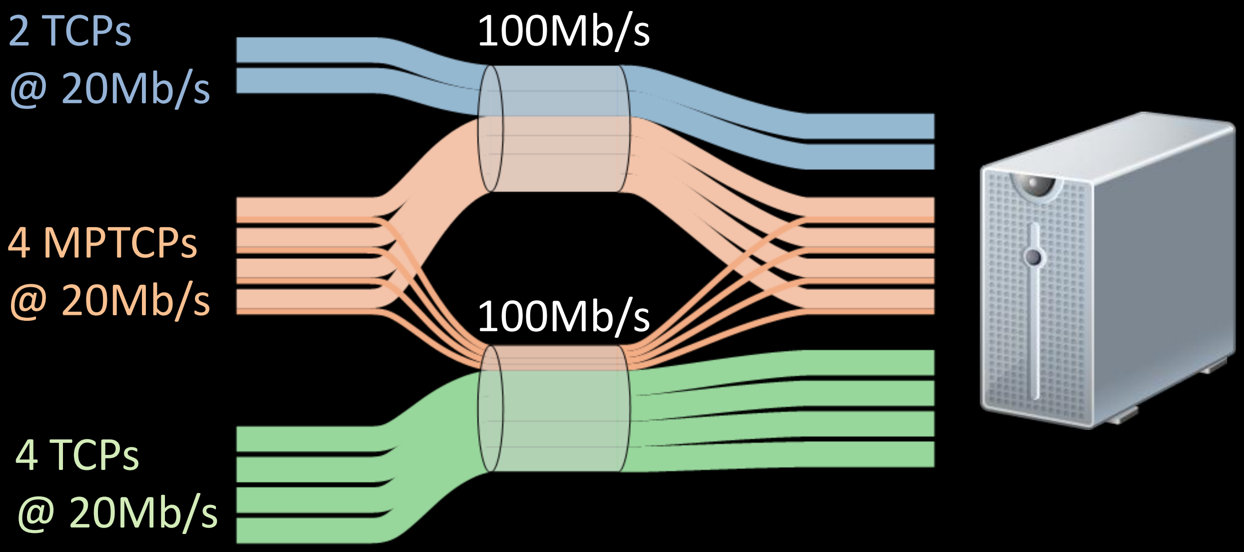


The total capacity, 200Mb/s, is shared out evenly between all 9 flows.

It’s as if they were all sharing a single 200Mb/s link. The two links can be said to form a 200Mb/s pool.

Multi-homed web sites

(d) with Kelly+Voice multipath TCP to balance traffic

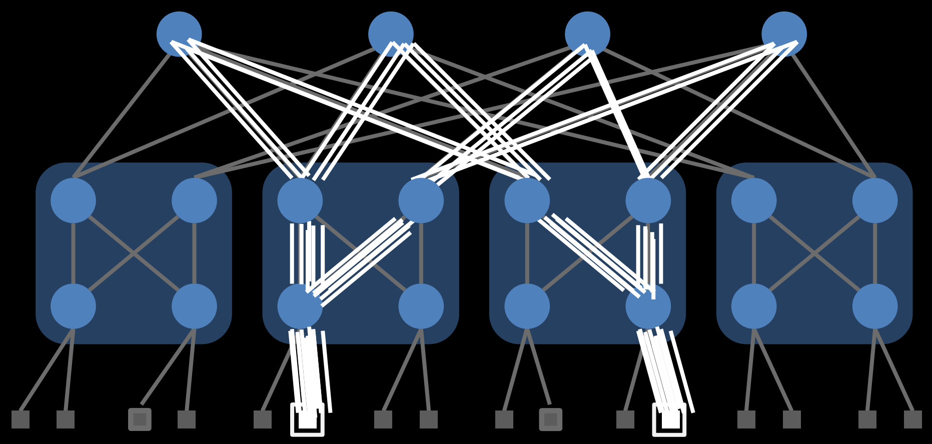


The total capacity, 200Mb/s, is shared out evenly between all 10 flows.

It’s as if they were all sharing a single 200Mb/s link. The two links can be said to form a 200Mb/s pool.

Load-balancing in data centers

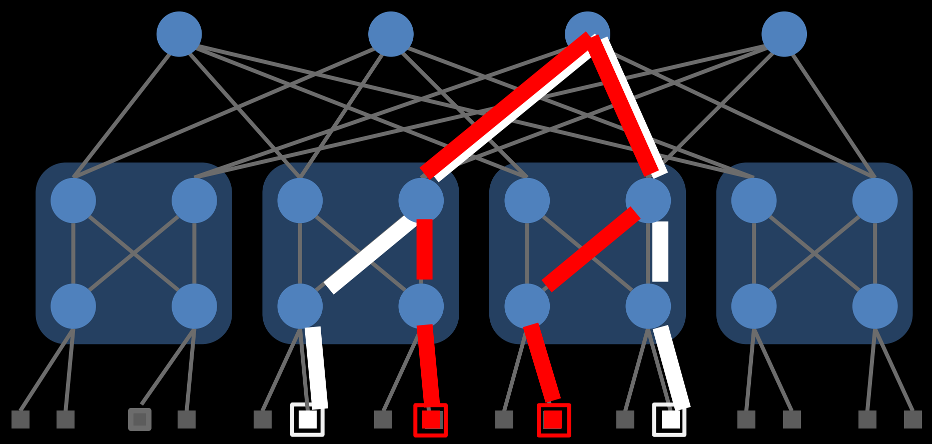
An obvious way to balance load is to pick randomly from available paths for each TCP flow.



This balances traffic nicely, as long as there are enough flows.

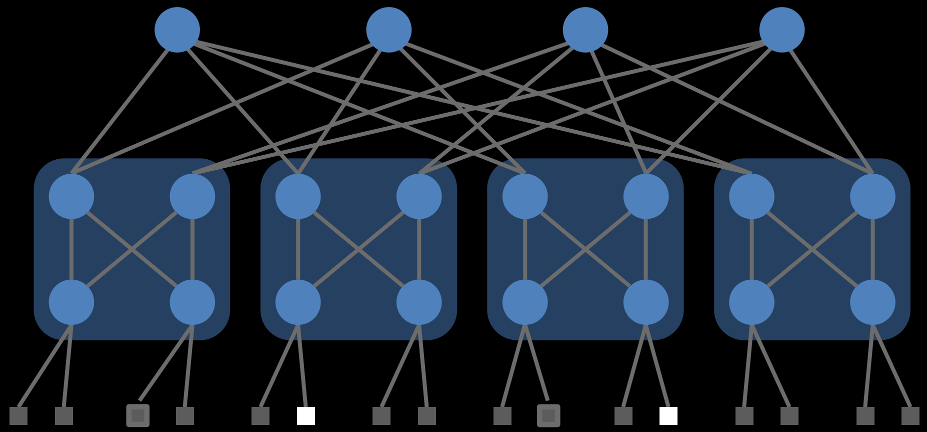
Load-balancing in data centers

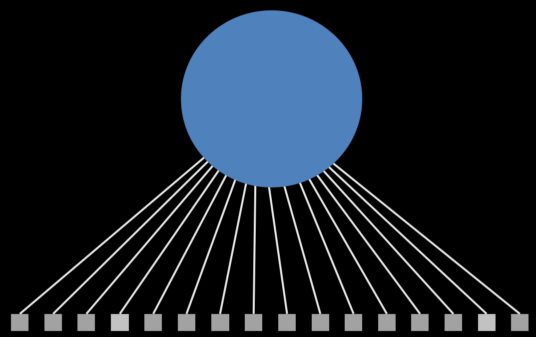
An obvious way to balance load is to pick randomly from available paths for each TCP flow.



This balances traffic nicely, as long as there are enough flows. But if there are fewer flows, there may be collisions and wasted capacity.

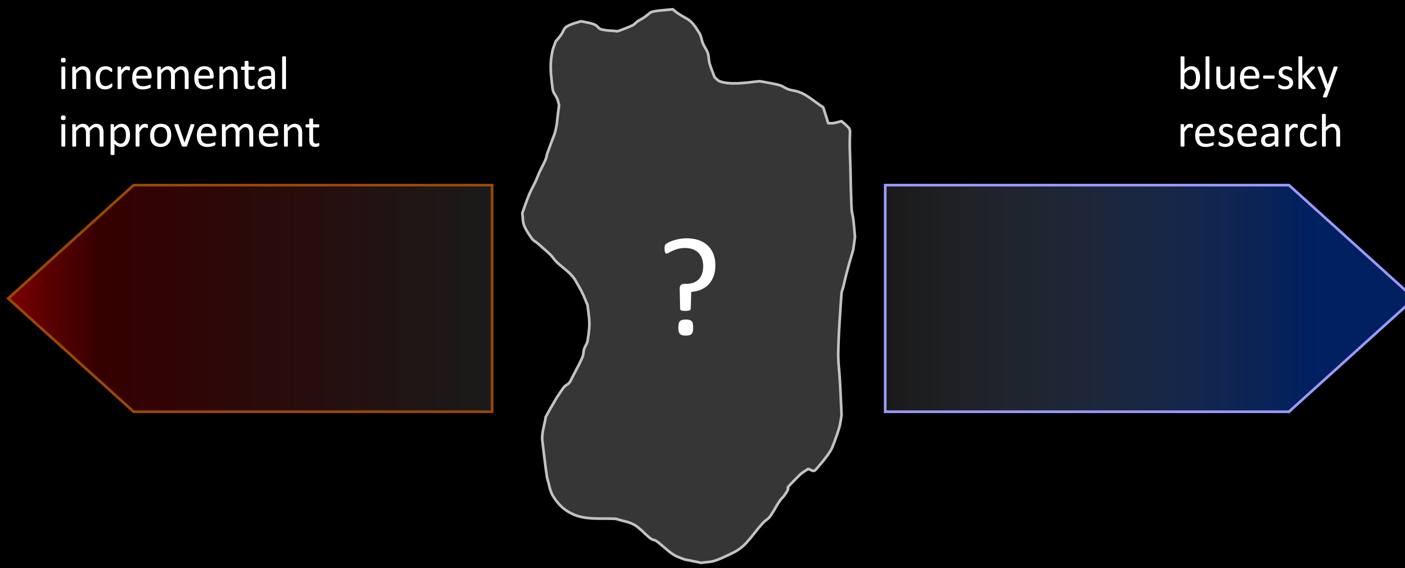
Can a data center be made to behave like a simple easily-managed resource pool?



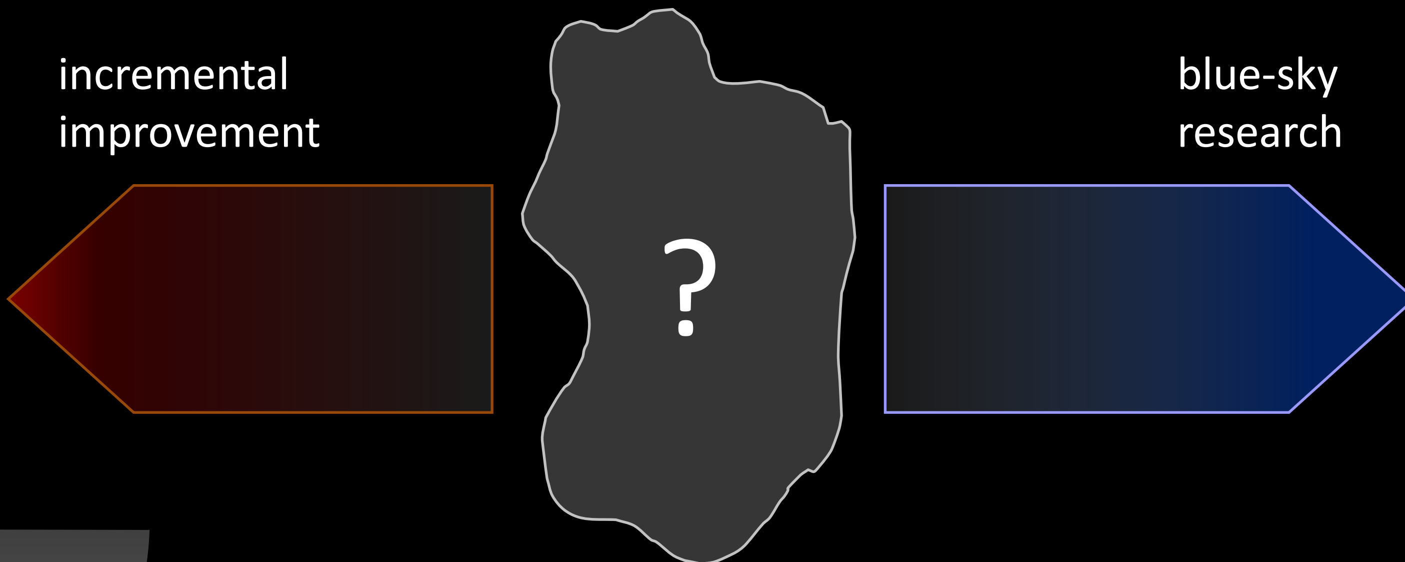


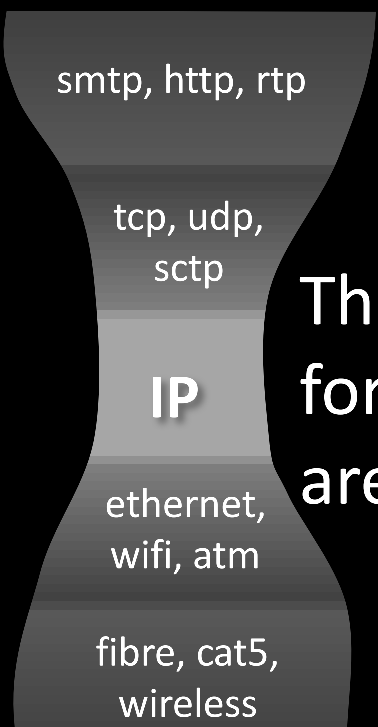
Can this be achieved across a range of traffic patterns? For data centers of different sizes?

The spectrum of Internet research



The spectrum of Internet research

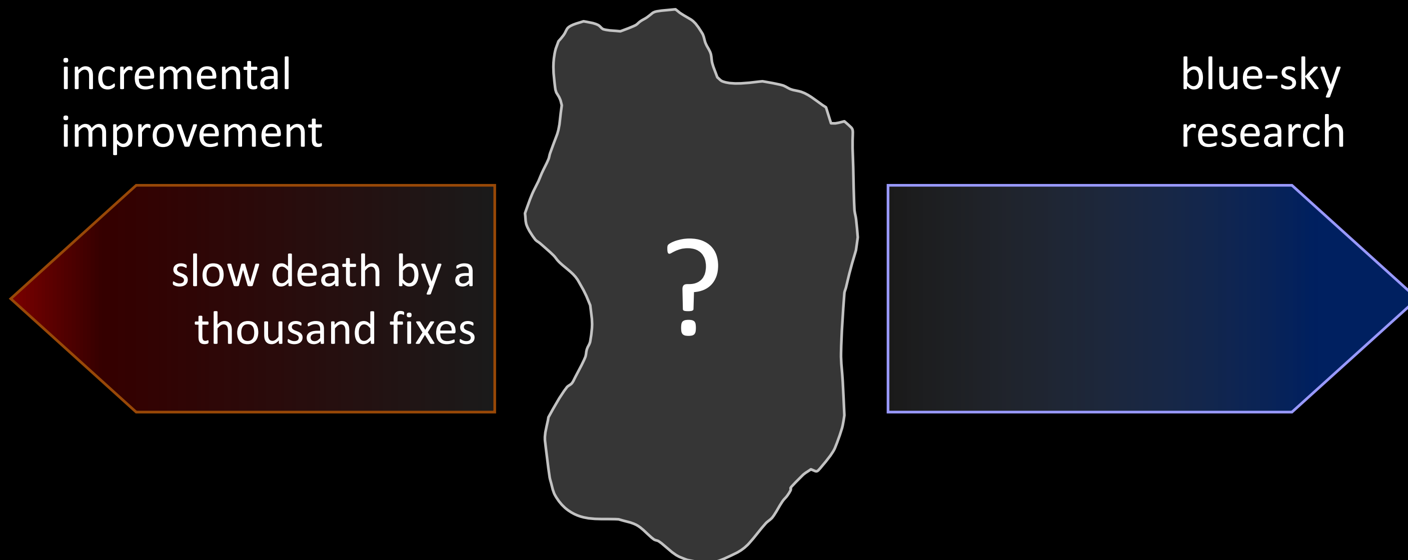


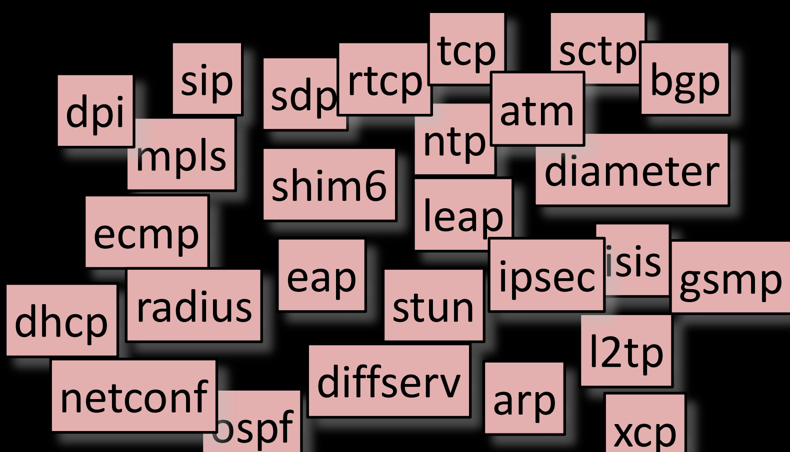


The Internet's standards
for carrying data
are an hourglass.

This has made it easy to build
new new things.

The spectrum of Internet research





The Internet's standards for control are an accumulation of fixes to specific problems.

The spectrum of Internet research

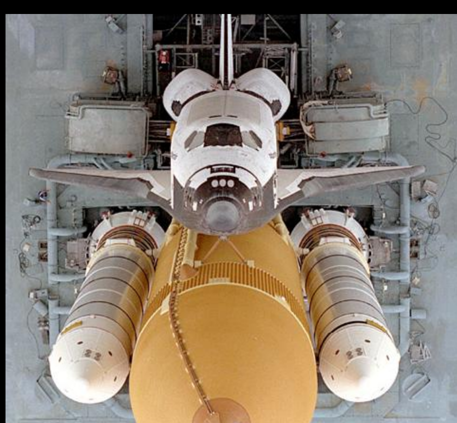
incremental
improvement

slow death by a
thousand fixes

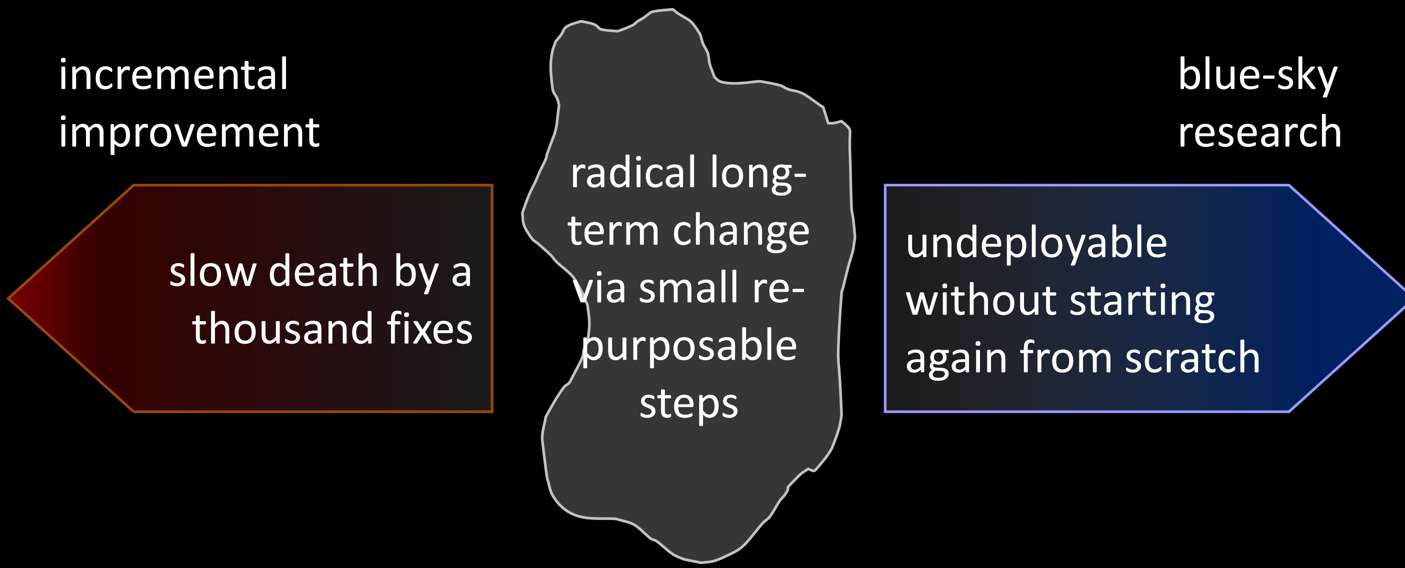
?

blue-sky
research

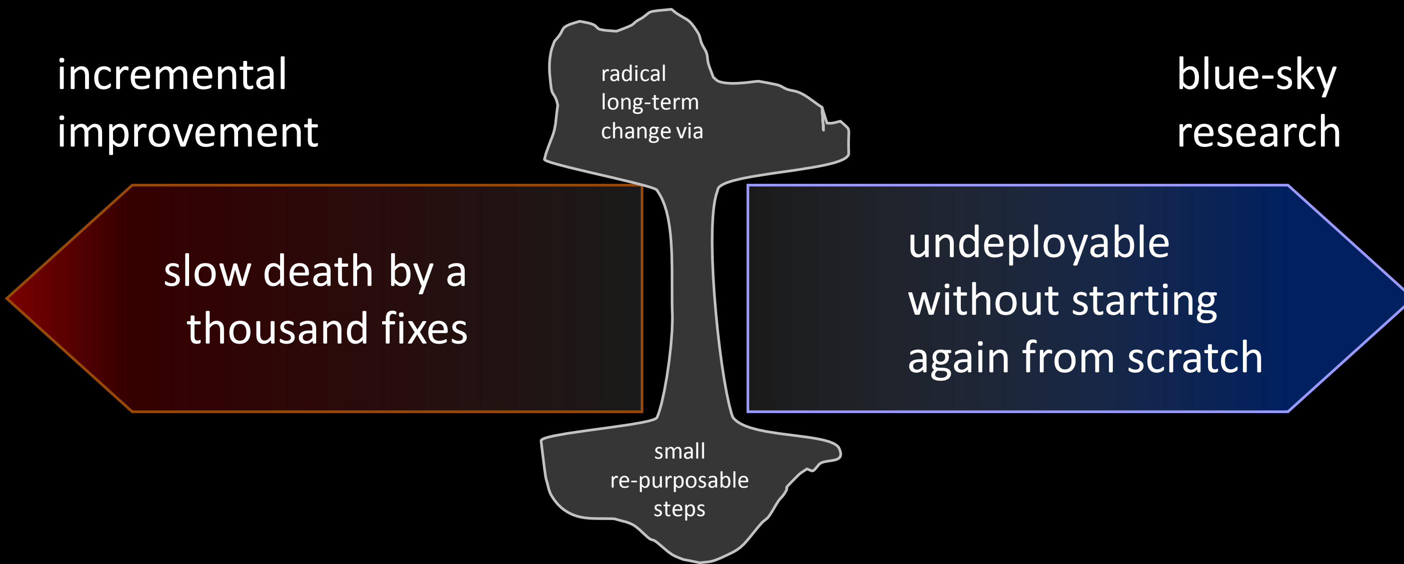
undeployable
without starting
again from scratch



The spectrum of Internet research



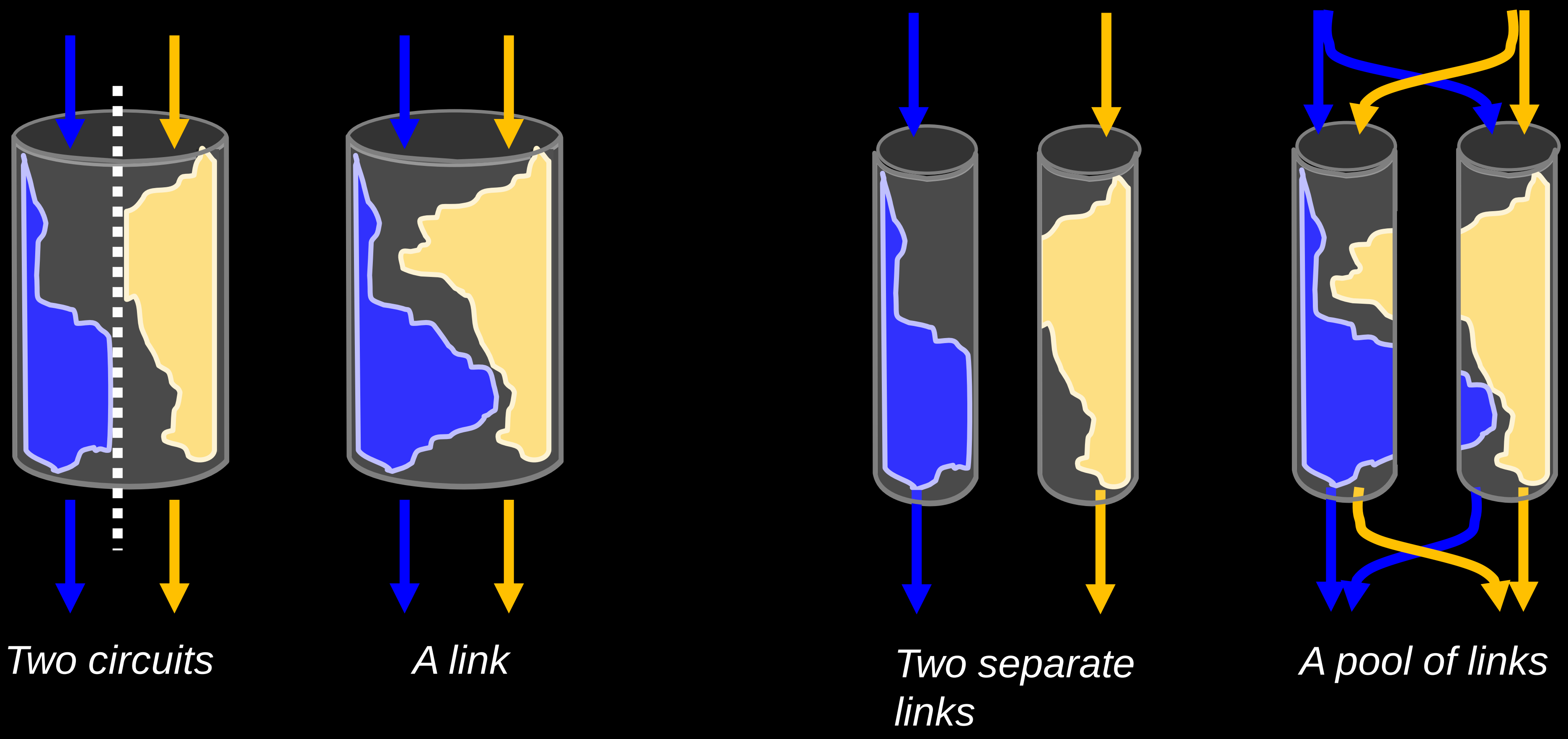
The spectrum of Internet research



End-system multipath congestion control will succeed because it is a re-purposable interface for solving many different problems — so it can become the 'narrow waist' of the Internet's control architecture.

"Network utility maximization" is mathematician's shorthand for this.

Multipath is Packet Switching 2.0, and multipath congestion control is TCP 2.0.



How did the theoretical results of Han et al. and Kelly+Voice help?

(beyond the fundamental idea of distributed network utility maximization—the idea that end-systems can by themselves manage to allocate the Internet's resources sensibly, in many different multipath settings)

What extra work did we need to do, to sell multipath?

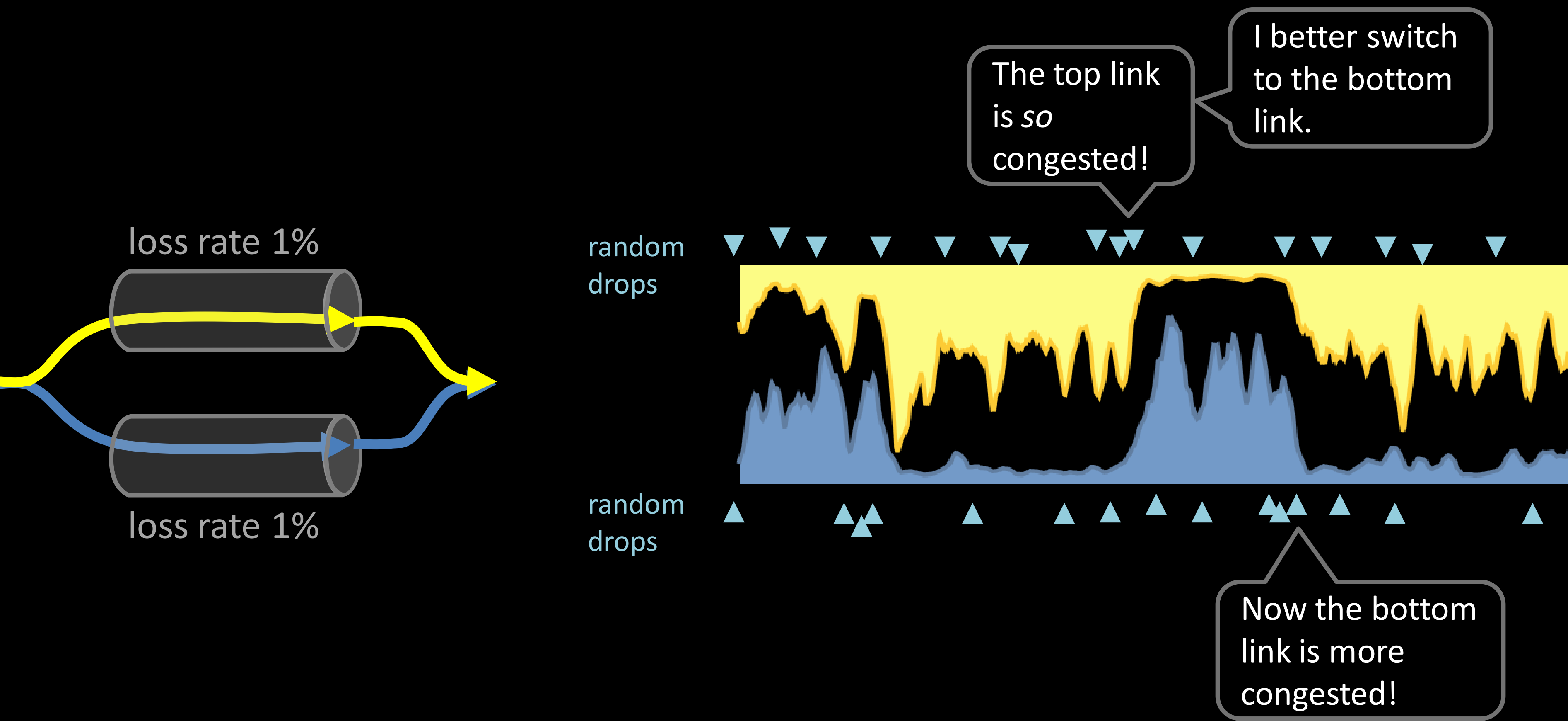
The big theoretical result is 'local stability of a fluid model of multipath congestion control'.

- Internet engineers have tried load-sensitive routing before, and observed route flap, and decided it's unsafe.
- We can point to the theory and say "The maths guarantees our multipath TCP is safe".

This theoretical result is unhelpful, on two counts.

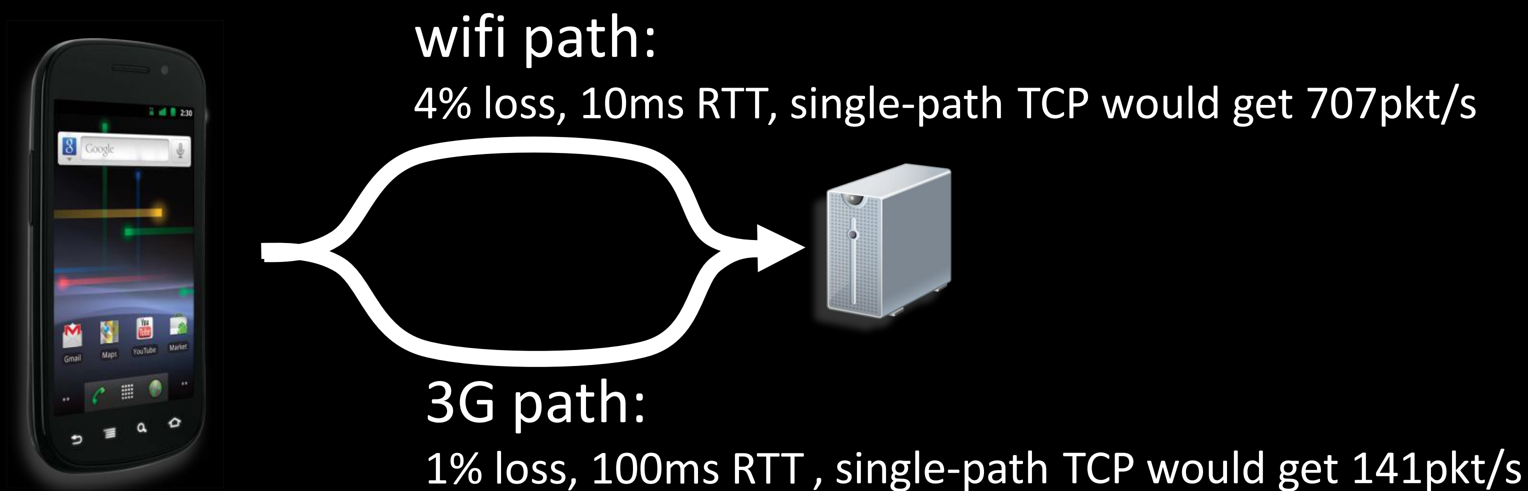
- The same theory says the current Internet is unstable, and engineers do not believe this.
- At low levels of aggregation (e.g. access links, which are the most congested part of today's Internet), the fluid model is misleading.

Why does the fluid model fail?



The Kelly+Voice algorithm puts *all* its traffic on the least congested path. The noisy nature of congestion feedback makes it difficult to estimate congestion levels, leading to bistability. But the fluid limit (the average of many bistable flows) is stable!

What changes did we need, to make it incrementally deployable?



The Kelly+Voice algorithm makes you shift all your traffic onto the least-congested path, in this case 3G.

Do you end up with 141pkt/s (fair to other 3G users)?
Or with 707pkt/s (what you would get without multipath)?

Conclusion

- End-system multipath congestion control
  will be the biggest change to the architecture of the Internet since packet switching and TCP.

- Network utility maximization
  is a goal which in itself is moot. But network algorithms which can maximize arbitrary utilities are necessarily rich enough to solve any control problem.

- Interesting new mathematical models may arise
  when you try to make theory work.

What should mathematicians know about the Internet?

Cool stuff is better than correctness. *What sort of maths makes the cool stuff work?*

- *Under the hood of BitTorrent,* Bram Cohen, 2005

The Internet is barely manageable, and it barely works. *What sort of maths helps us make it autonomic?*

- *End-to-end arguments in system design*, Saltzer, Reed, Clark, 1981
- *Why the Internet only just works*, Handley, 2006

No one can tell you what the Internet is for. *What sort of maths gives us tools, not solutions?*

How does TCP congestion control work?

Maintain a congestion window w.

- Increase w for each ACK, by $1/w$

- Decrease w for each drop, by $w/2$

How does MPTCP congestion control work?

Maintain a congestion window w_r, one window for each path, where $r \in R$ ranges over the set of available paths.

- Increase w_r for each ACK on path r, by

$$\min_{S \subseteq R : r \in S} \frac{\max_{s \in S} w_s / \mathsf{RTT}_s^2}{\left(\sum_{s \in S} w_s / \mathsf{RTT}_s\right)^2}$$

- Decrease w_r for each drop on path r, by $w_r/2$

How does MPTCP congestion control work?

Maintain a congestion window w_r, **one window for each path**, where $r \in R$ ranges over the set of available paths.

We want to shift traffic away from congestion.

To achieve this, we increase windows in proportion to their size.

- Increase w_r for **each ACK on path** r, by

$$\min_{S \subseteq R : r \in S} \frac{\max_{s \in S} w_s / \mathrm{RTT}_s^2}{\left(\sum_{s \in S} w_s / \mathrm{RTT}_s\right)^2}$$

- Decrease w_r for **each drop on path** r, by $w_r/2$

How does MPTCP congestion control work?

Maintain a congestion window w_r, **one window for each path**, where $r \in R$ ranges over the set of available paths.

MPTCP puts an amount of flow on path *r* proportional to $1/p_r$

(whereas Kelly+Voice put all the flow on the least-congested paths).

We do this so that we send more probe traffic, so we react faster to changes.

- Increase w_r for **each ACK on path** r, by

$$\min_{S \subseteq R : r \in S} \frac{\max_{s \in S} w_s / \mathrm{RTT}_s^2}{\left(\sum_{s \in S} w_s / \mathrm{RTT}_s\right)^2}$$

- Decrease w_r for **each drop on path** r, by $w_r/2$

How does MPTCP congestion control work?

Maintain a congestion window w_r, one window for each path, where $r \in R$ ranges over the set of available paths.

- Increase w_r for each ACK on path r, by

$$\min_{S \subseteq R : r \in S} \frac{\max_{s \in S} w_s / \mathrm{RTT}_s^2}{\left(\sum_{s \in S} w_s / \mathrm{RTT}_s\right)^2}$$

Take no more than TCP would, at any potential bottleneck *S*:
look at the best that a single-path TCP could get,
and compare to what I'm getting.

- Decrease w_r for each drop on path r, by $w_r/2$