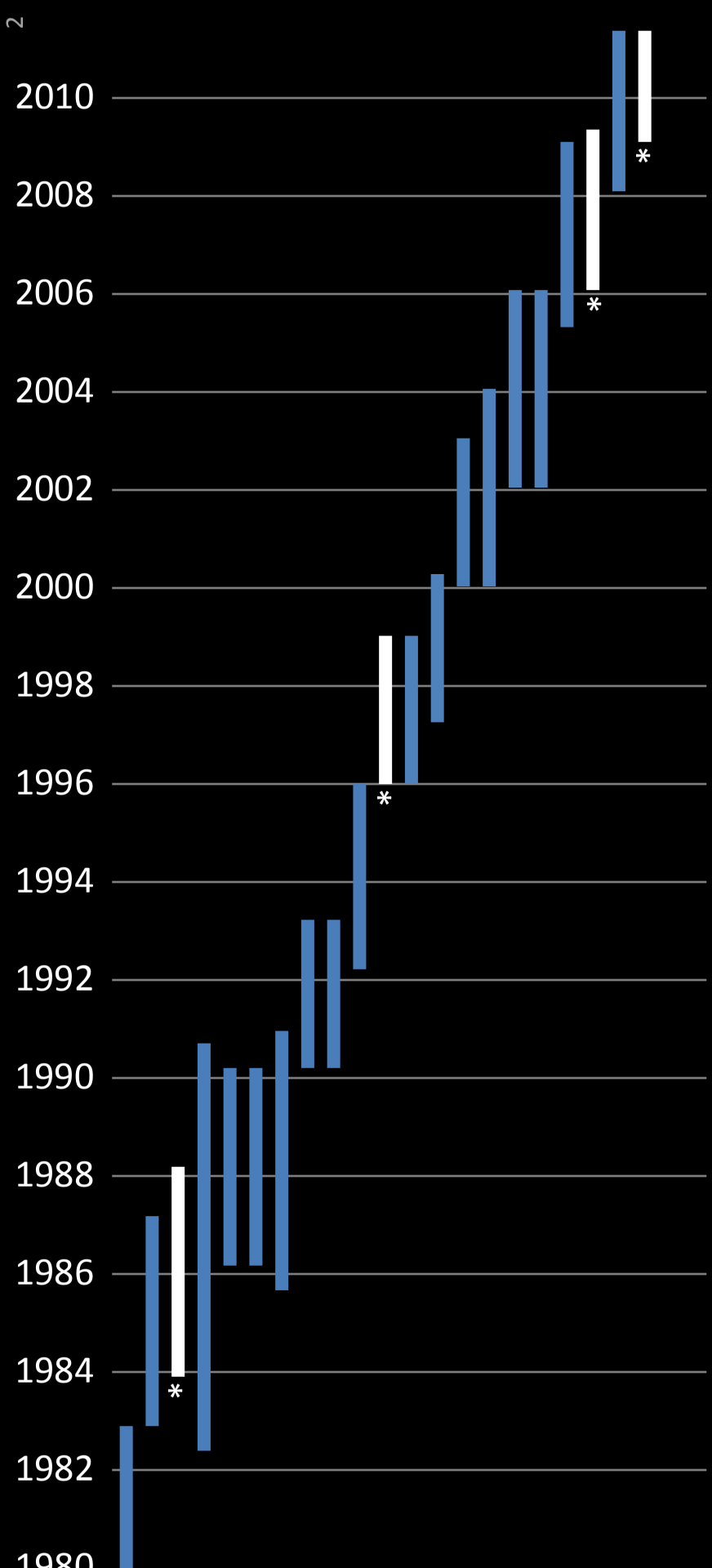
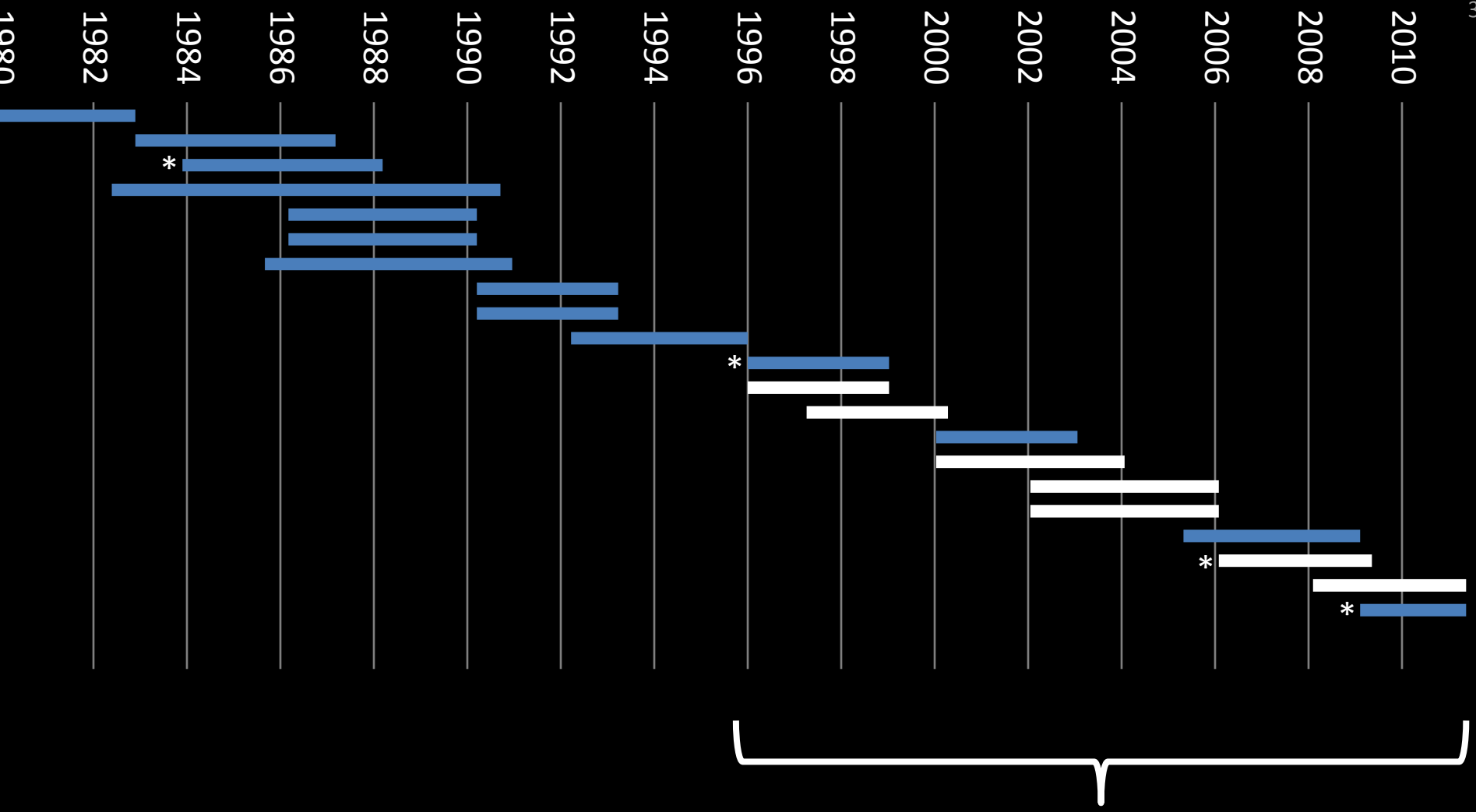
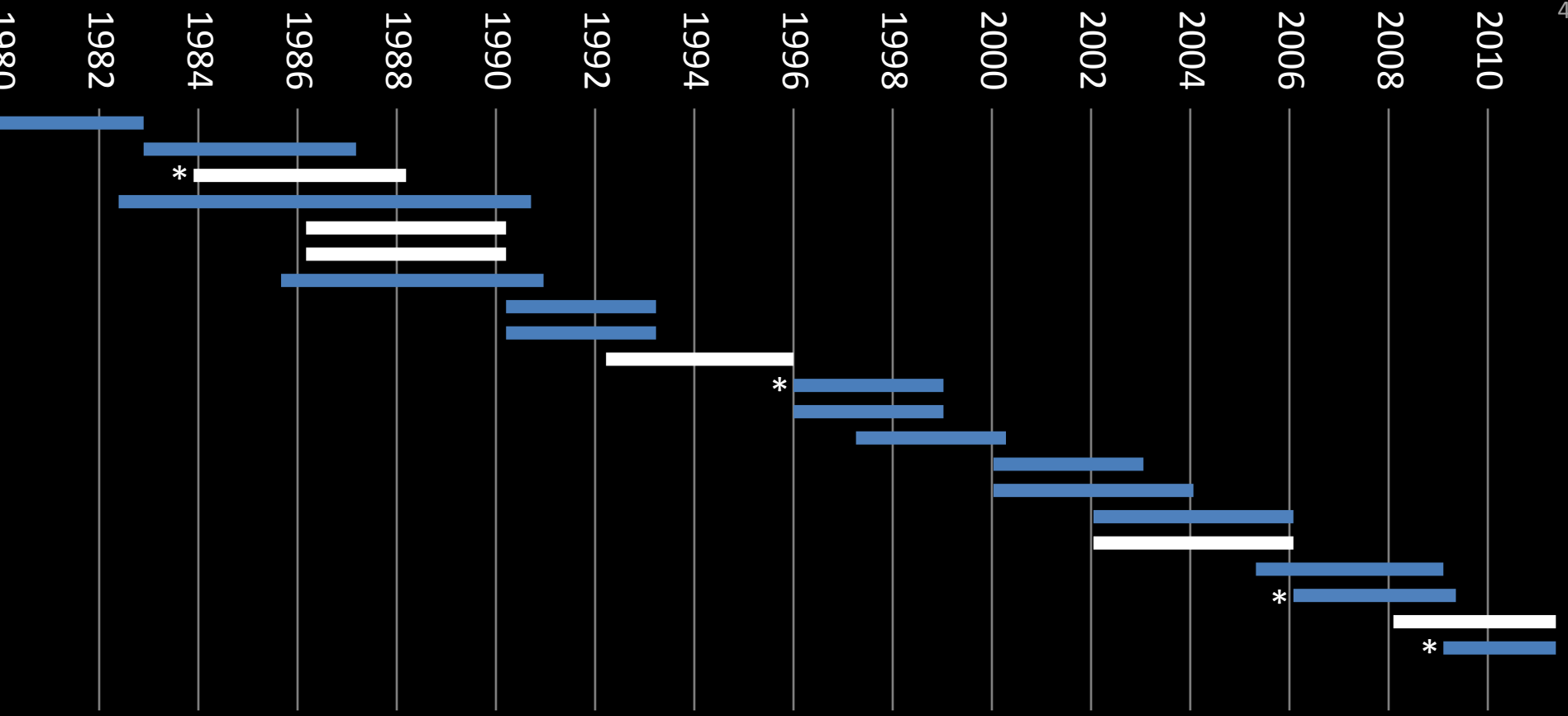
Developing Multipath TCP

Damon Wischik, Mark Handley, Costin Raiciu


2
2010
2008
2006
2004
2002
2000
1998
1996
1994
1992
1990
1988
1986
1984
1982
1980


1980
1982
1984
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004
2006
2008
2010
*
*
*
congestion control
PhD topics



multipath
PhD topics

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

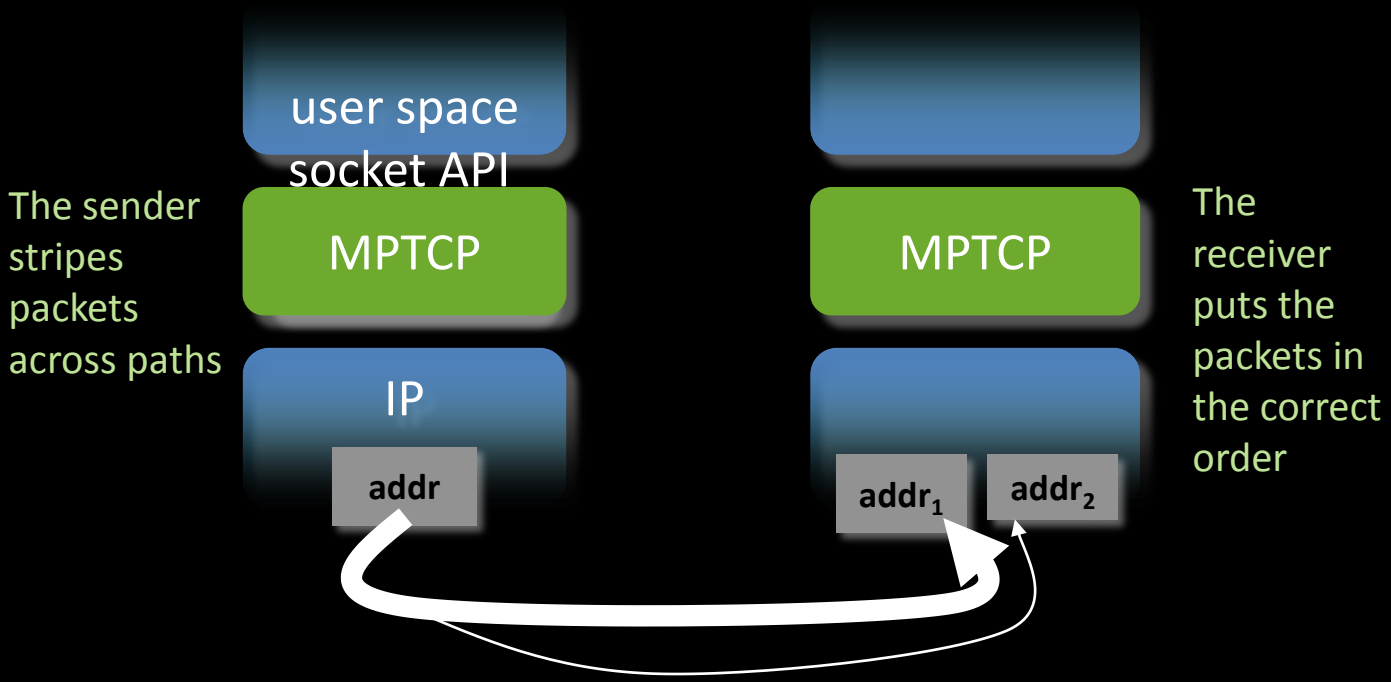
Kelly and Voice, *Computer Communication Review* 2005

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.

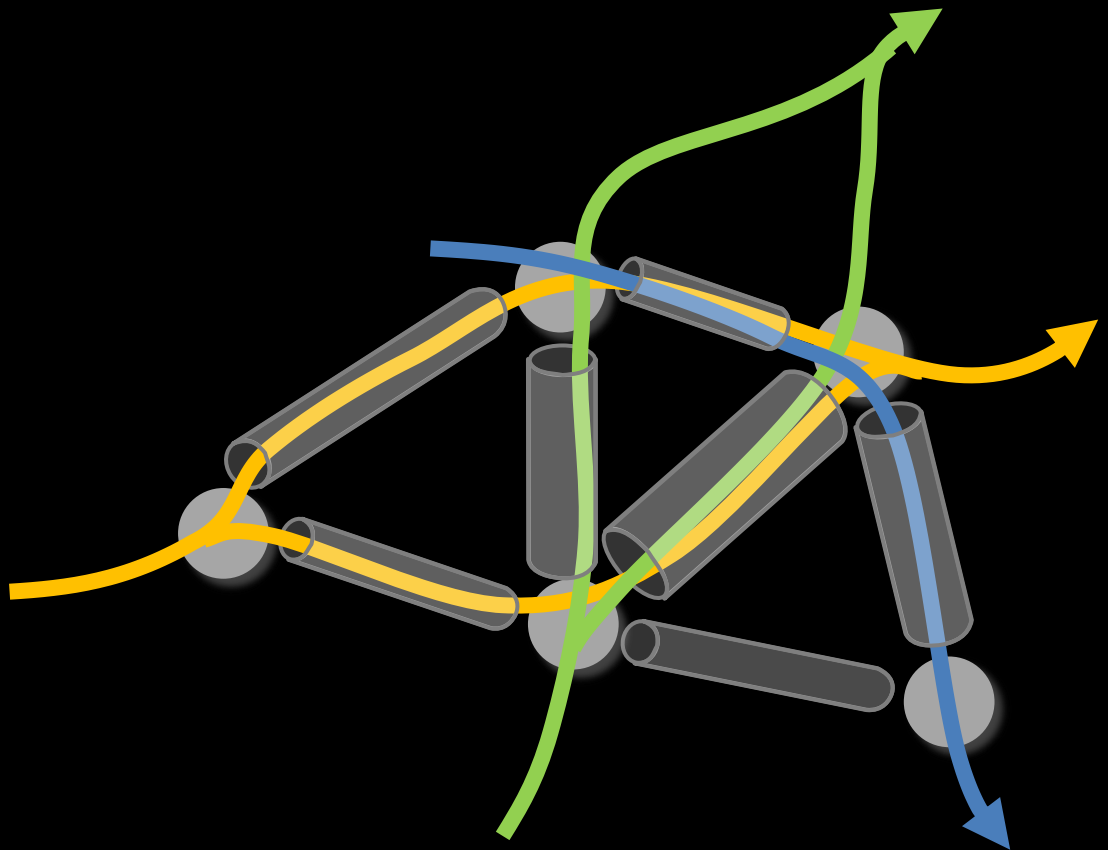
We present a sufficient condition for the local stability of end-to-end algorithms for joint routing and rate control. The network model considered allows an arbitrary interconnection of sources and resources, and heterogeneous propagation delays. The sufficient condition we present is decentralized: 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.

MPTCP is a replacement for TCP, which lets you use multiple paths simultaneously. The mptcp working group at the IETF has just standardized it as an Experimental RFC.

Thanks to trilogy



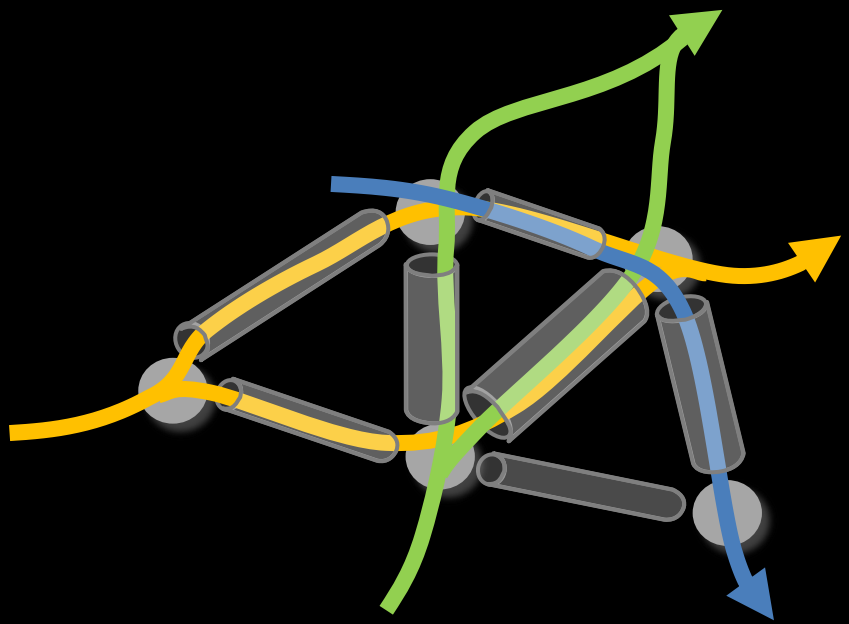
What problem is being solved by joint routing and rate control?



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*,

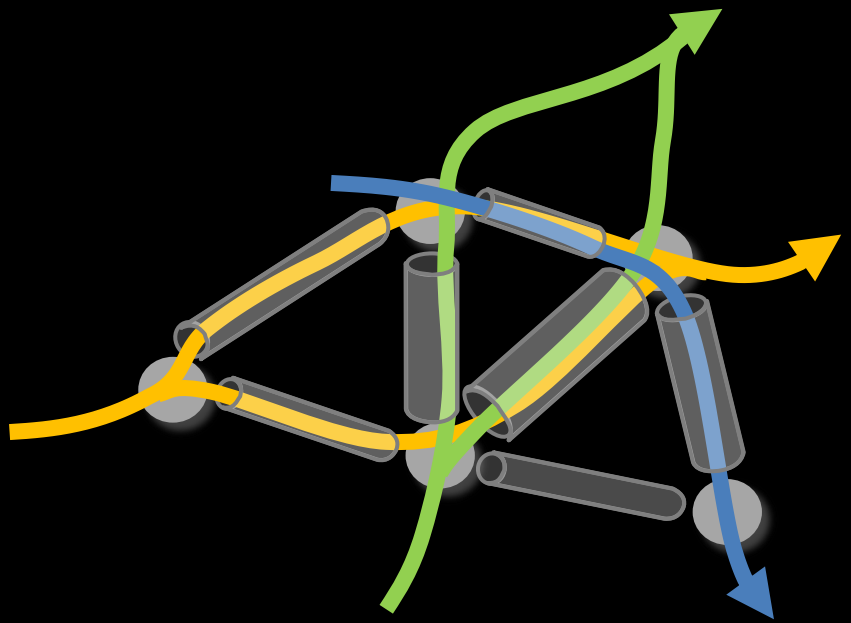
What problem is being solved by end-to-end algorithms for joint routing and rate control?



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.

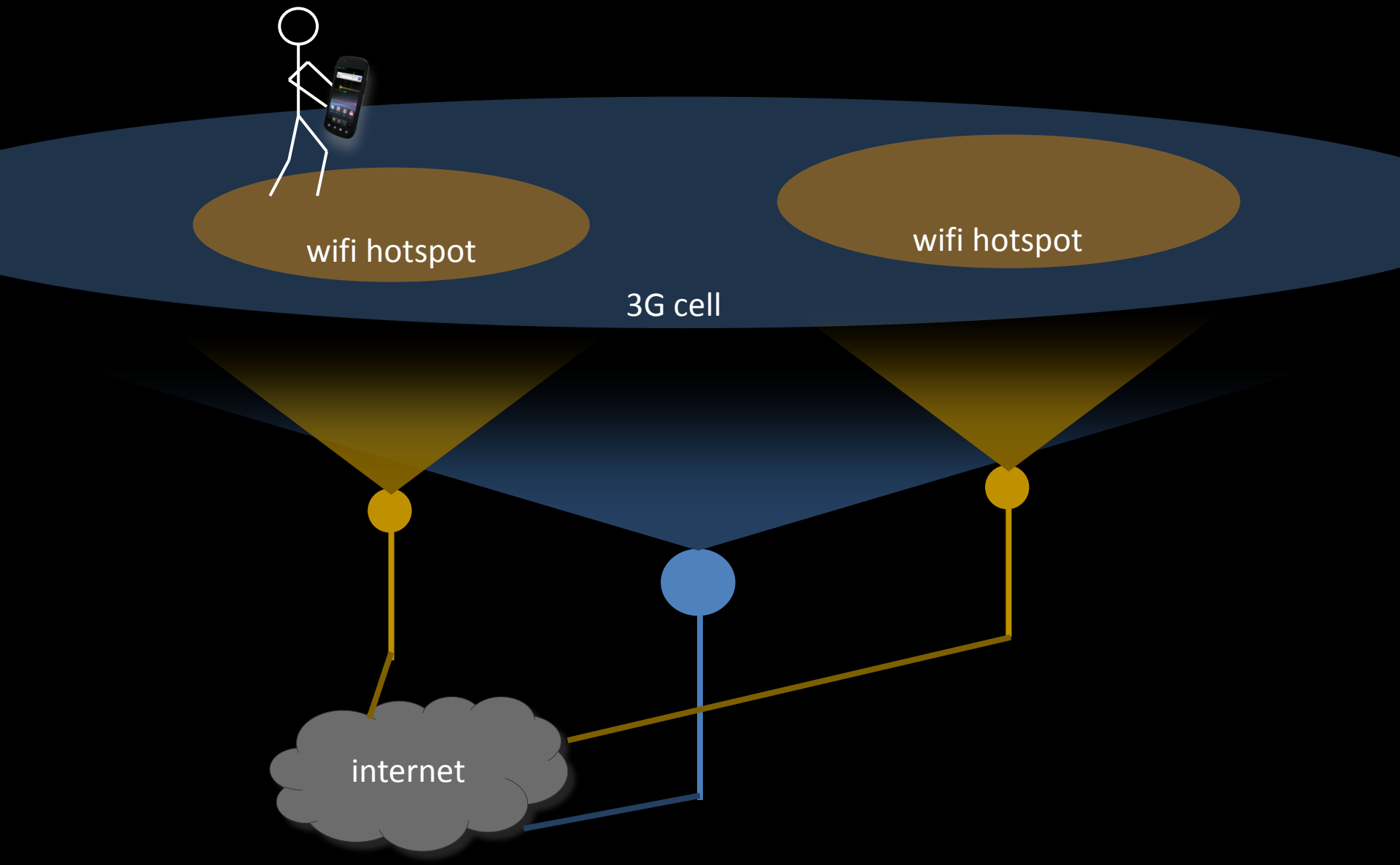
3. Network systems people don’t see the point in utility maximization.

“You have to be careful talking to mathematicians. You tell them a problem, then they restate it in their own language, and straight away it means something completely different.”

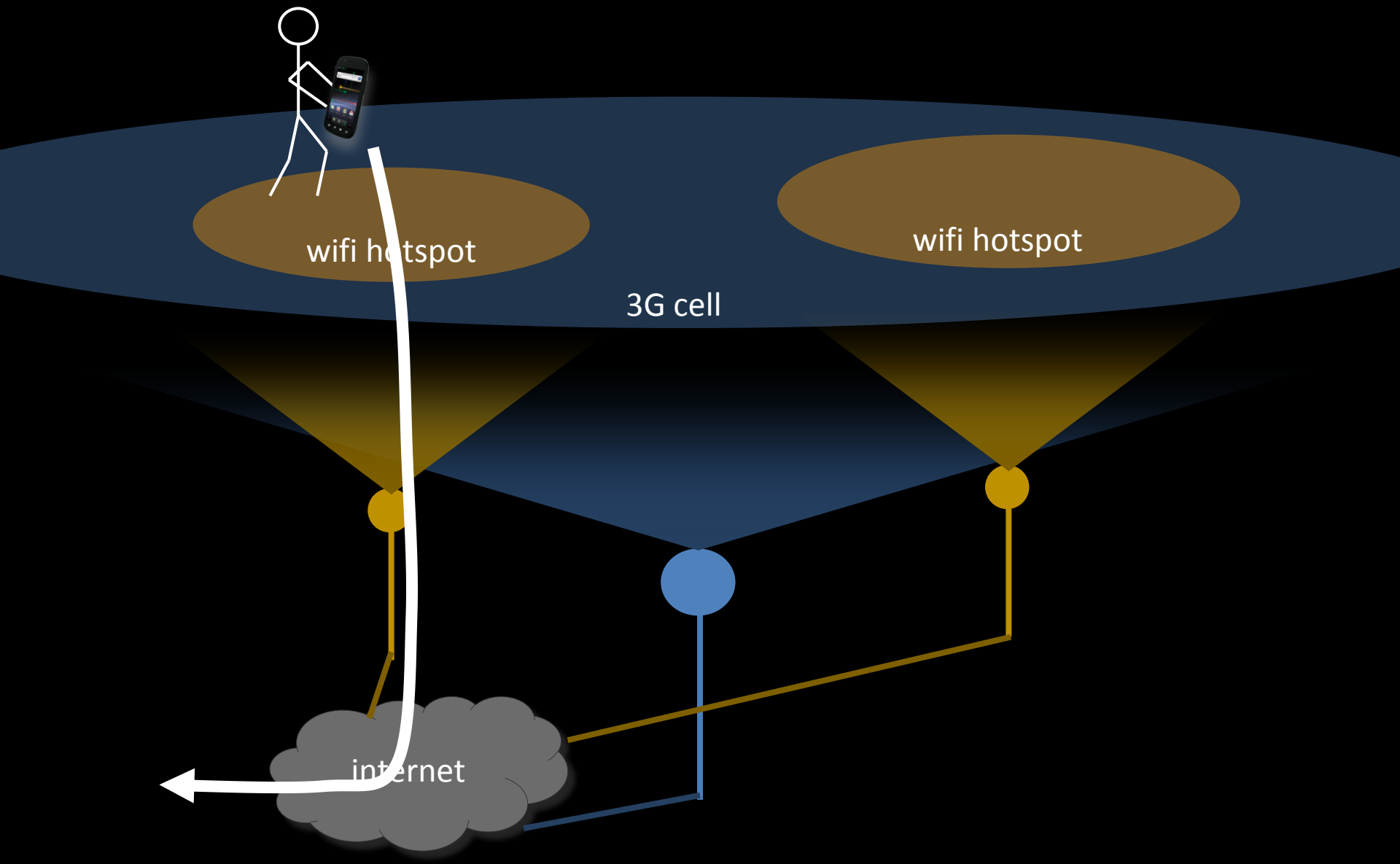
What are the current multipath technologies?

What problems are they solving?

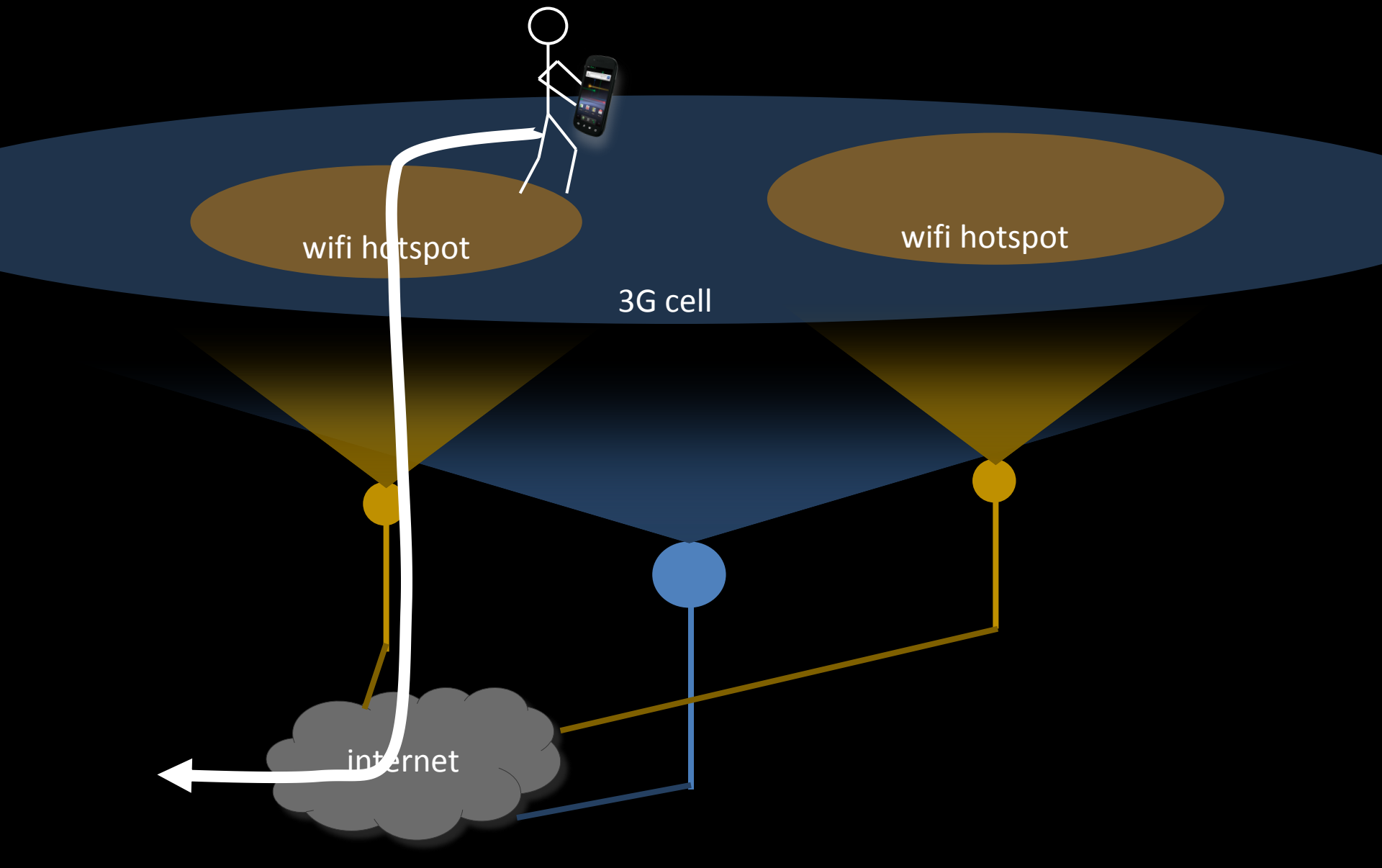
Mobile hand-offs



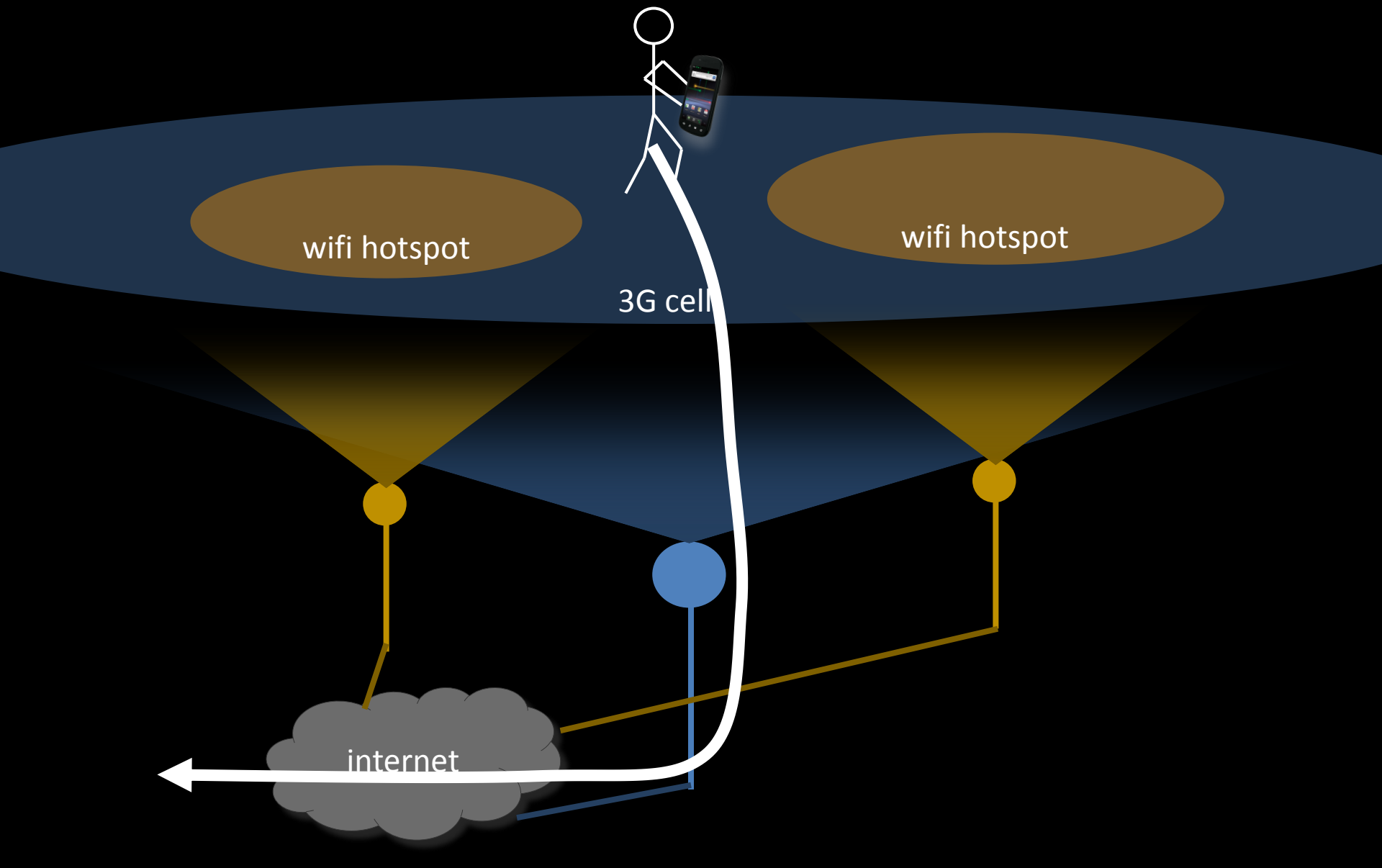
Mobile hand-offs



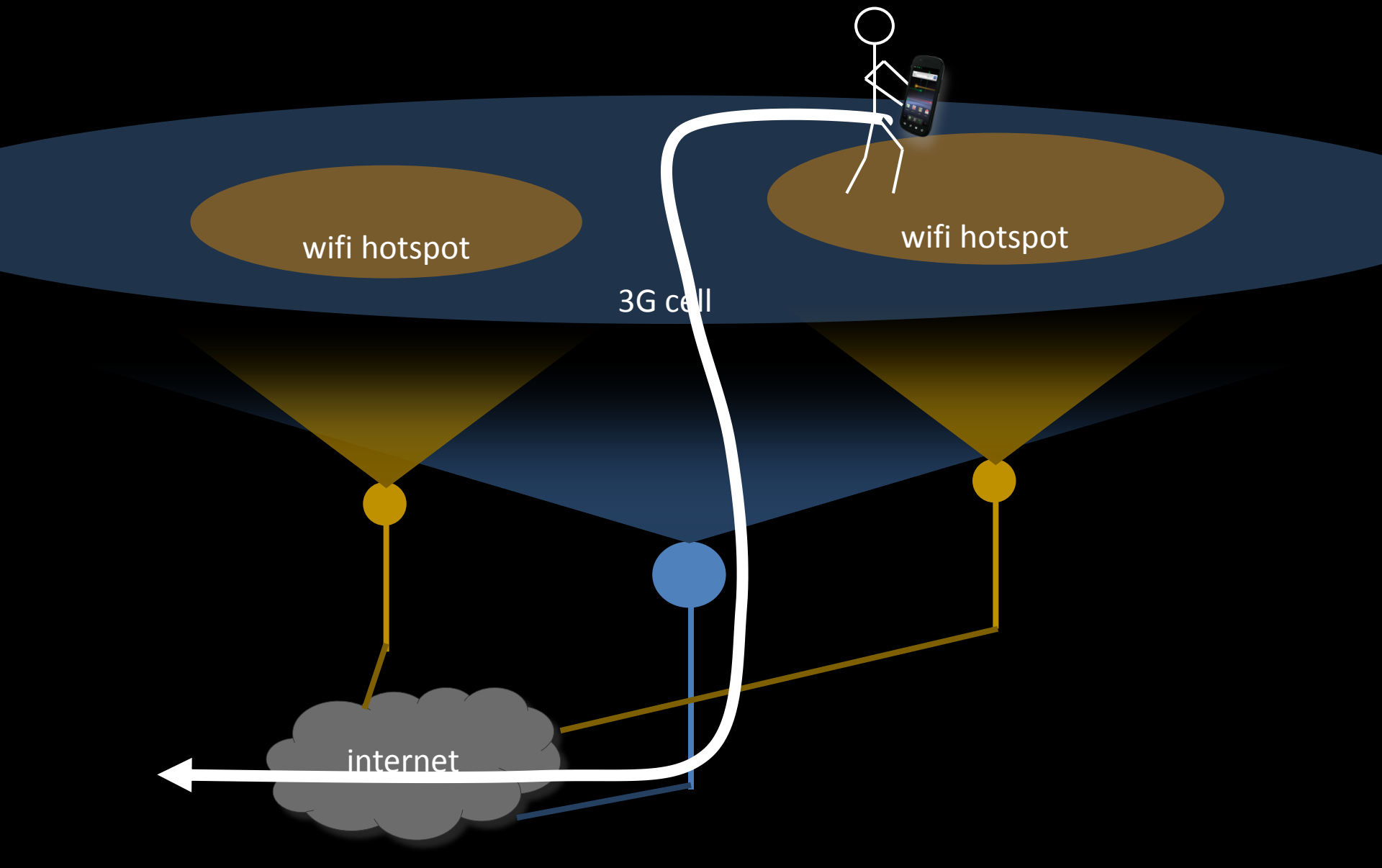
Mobile hand-offs



Mobile hand-offs

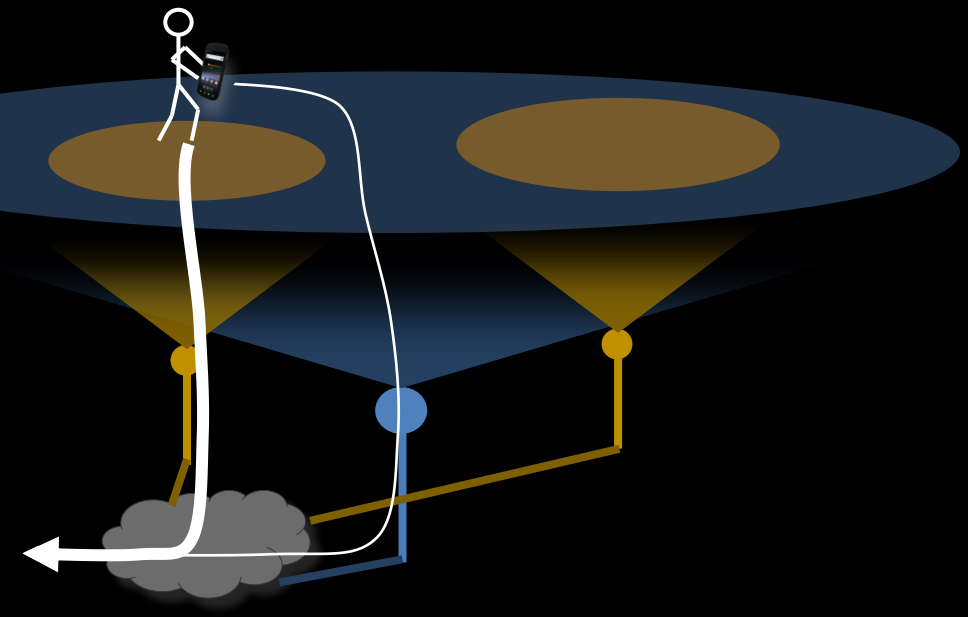


Mobile hand-offs



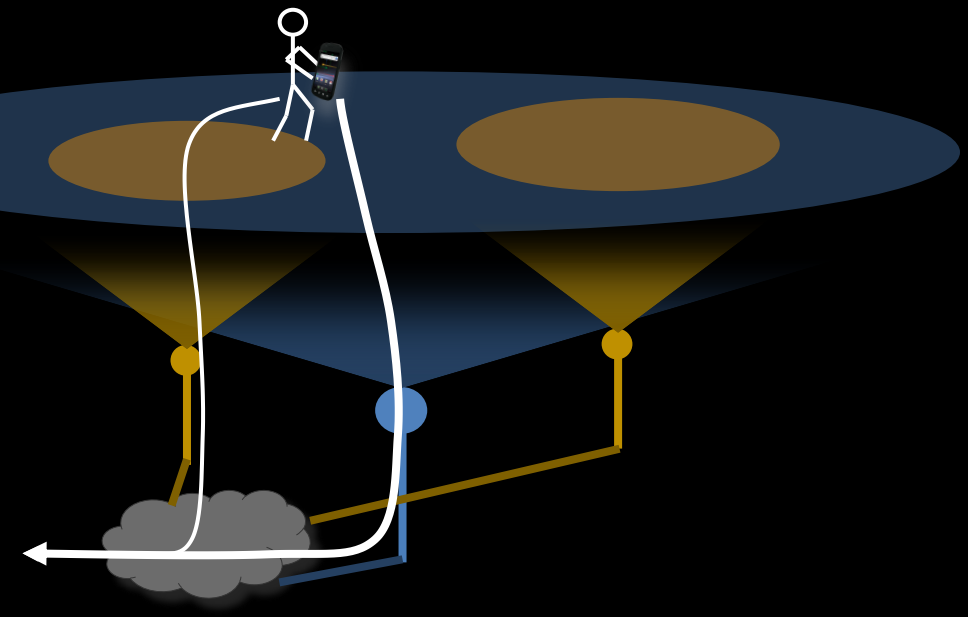
Mobile hand-offs

With multipath, your phone could use both radios at the same time, so you experience no interruption.



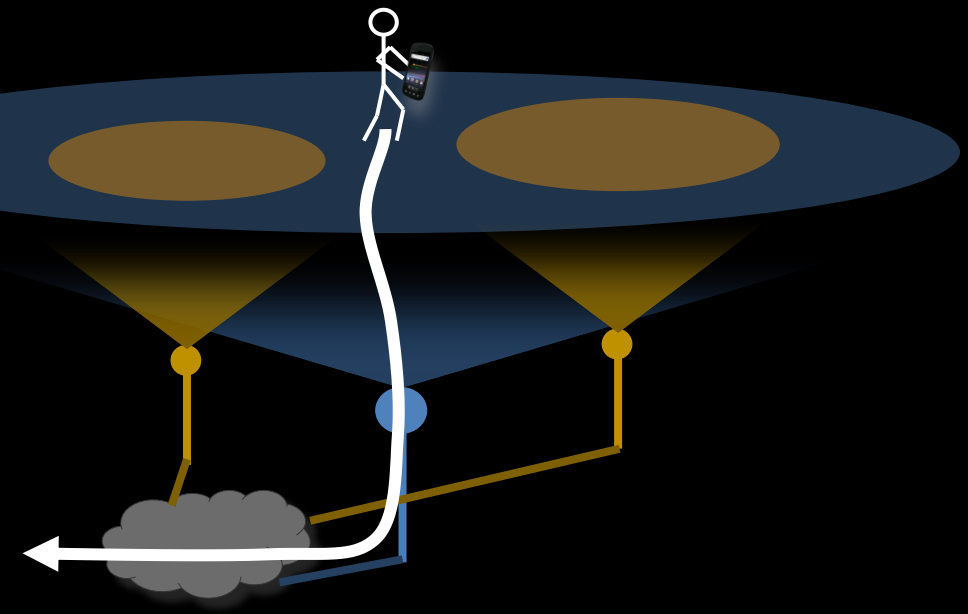
Mobile hand-offs

With multipath, your phone could use both radios at the same time, so you experience no interruption.



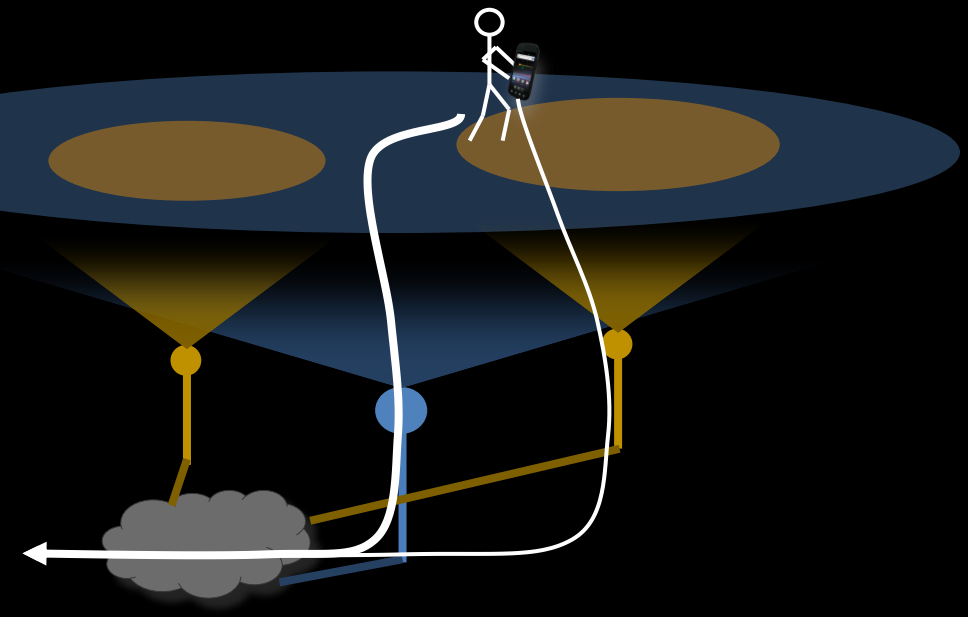
Mobile hand-offs

With multipath, your phone could use both radios at the same time, so you experience no interruption.



Mobile hand-offs

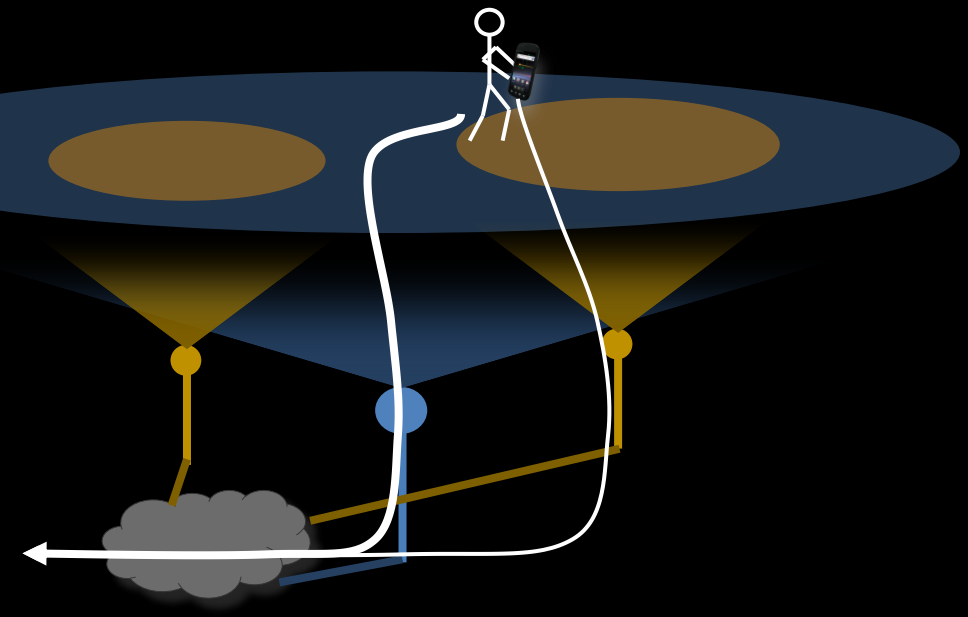
With multipath, your phone could use both radios at the same time, so you experience no interruption.



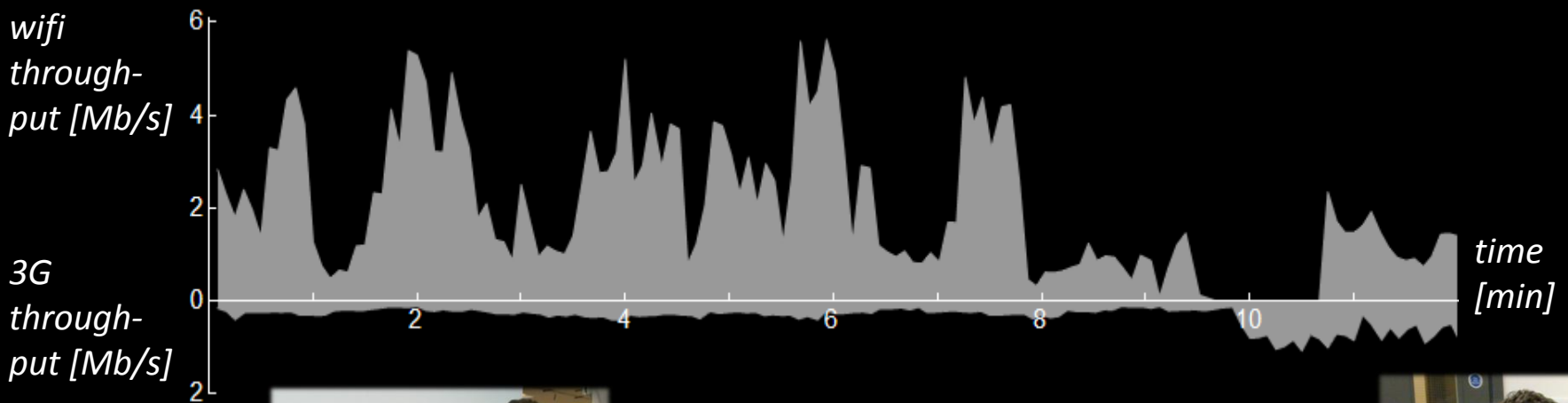
Mobile hand-offs

With multipath, your phone could use both radios at the same time, so you experience no interruption.

(Is there any maths here, or would any sensible rate control be good enough?)



Multipath TCP in practice



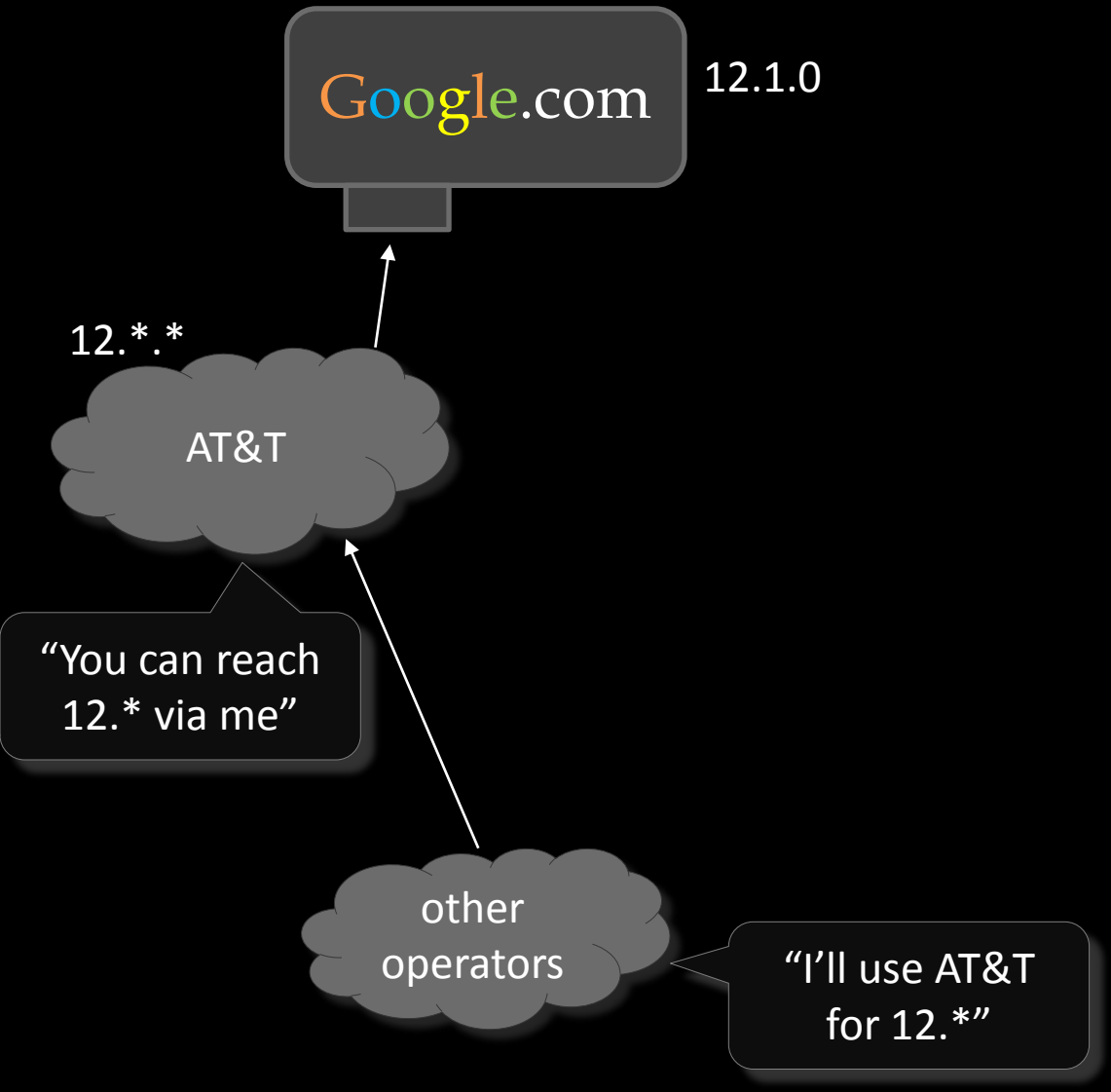
User in his office, using wifi and 3G

Going downstairs

In the kitchen

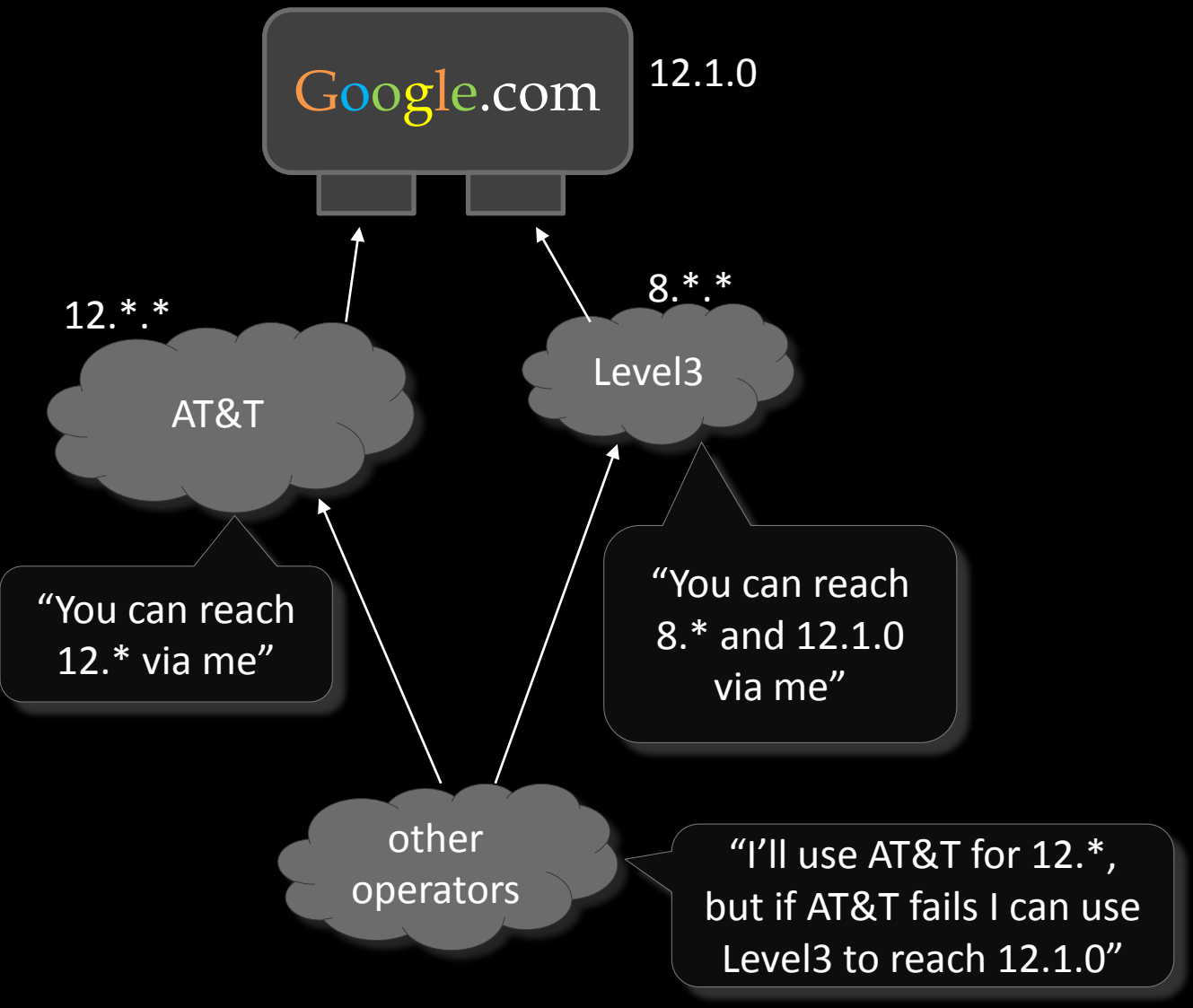
Multi-homed web sites

(a) with classic hierarchical routing



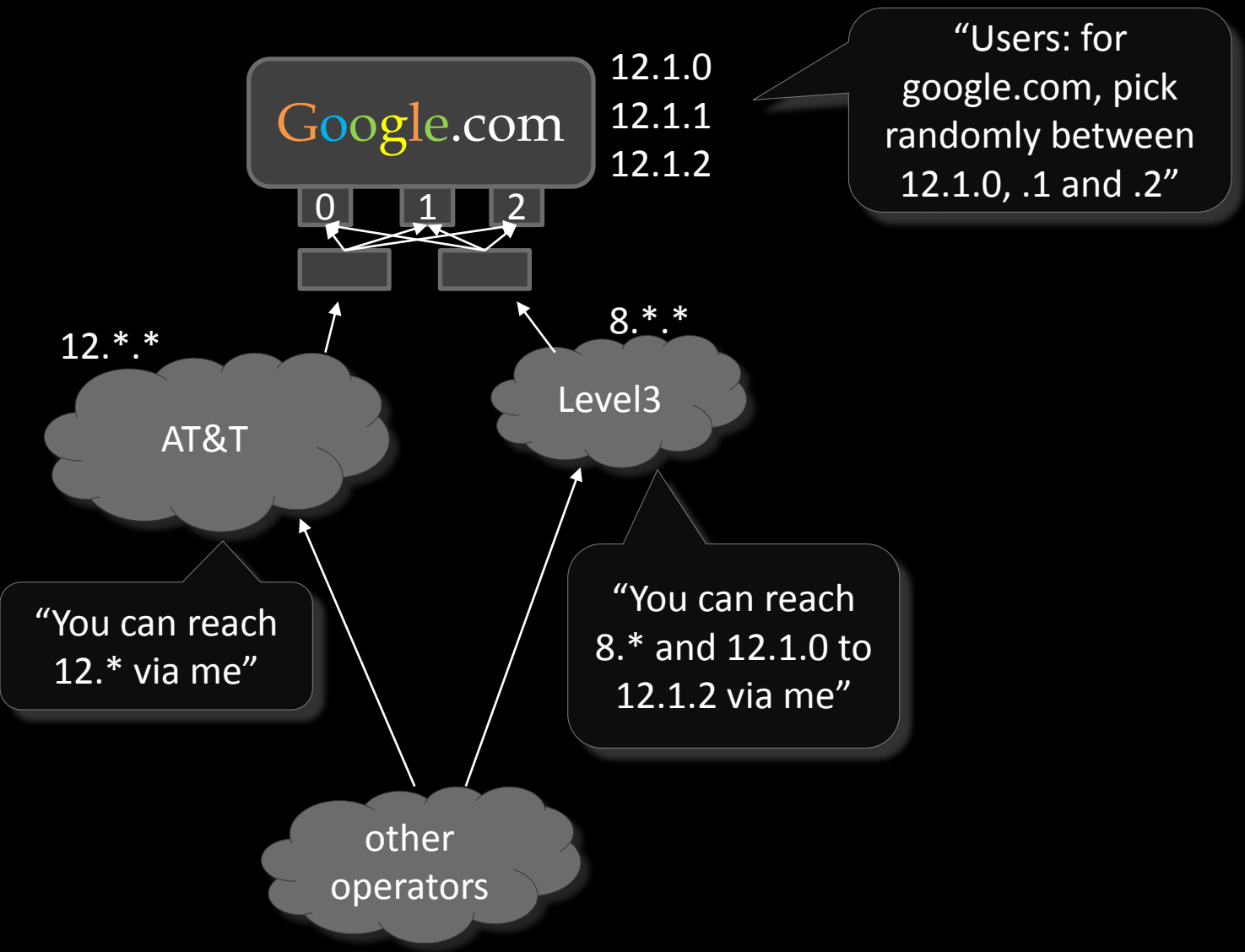
Multi-homed web sites

(b) with redundancy, in case links fail



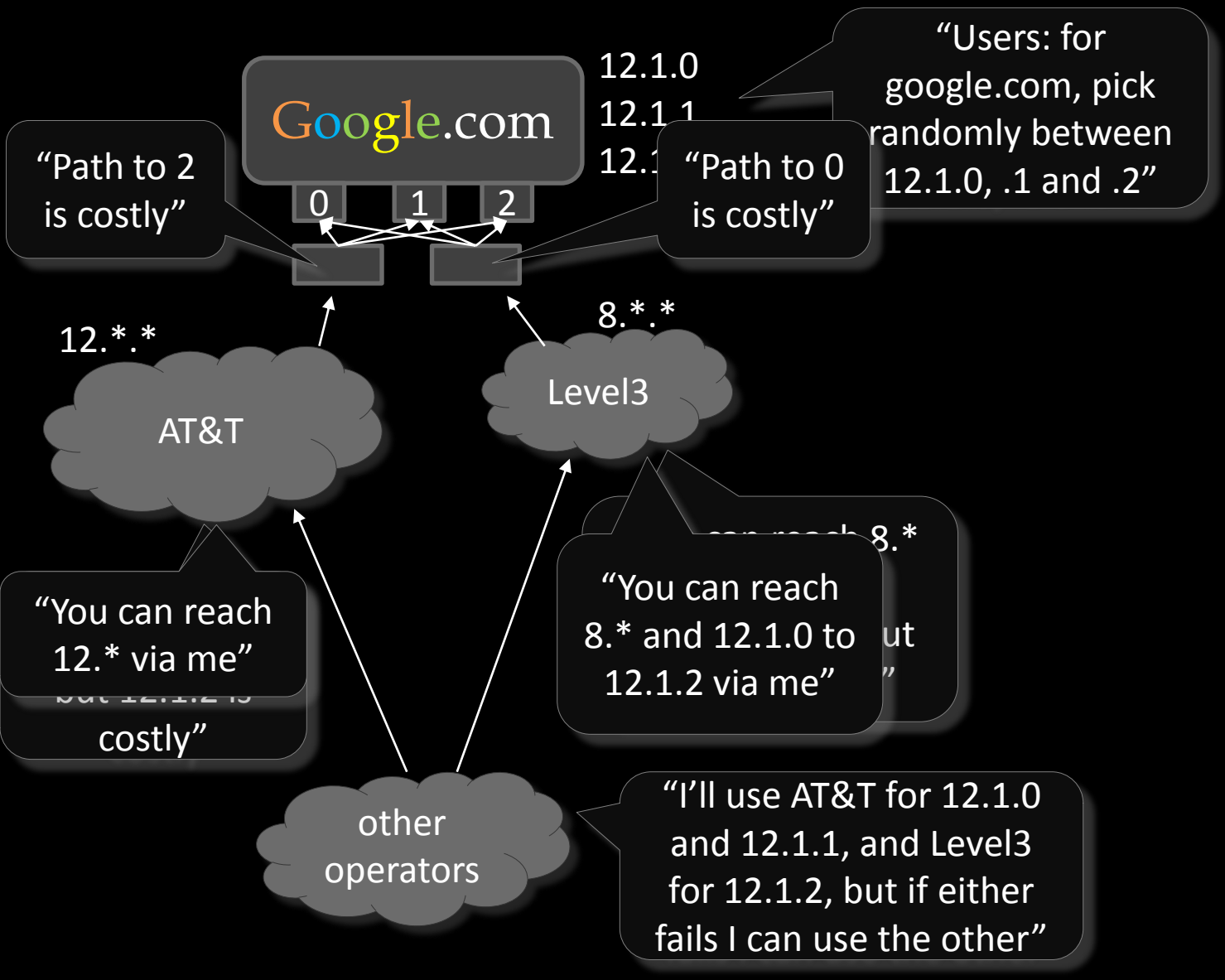
Multi-homed web sites

(c) with load balancing across the gateway machines

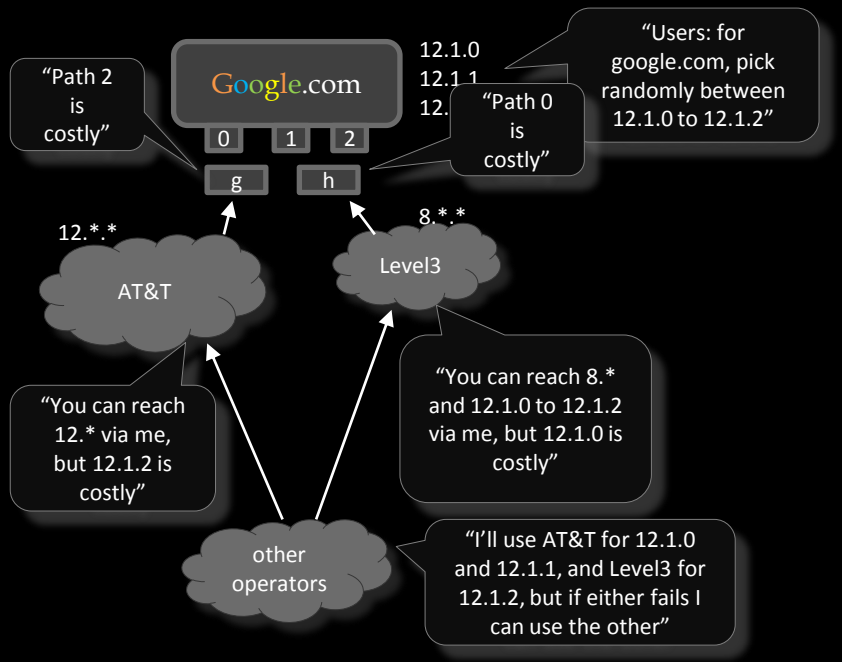


Multi-homed web sites

(c) with load balancing across the gateway machines



Multi-homed web sites

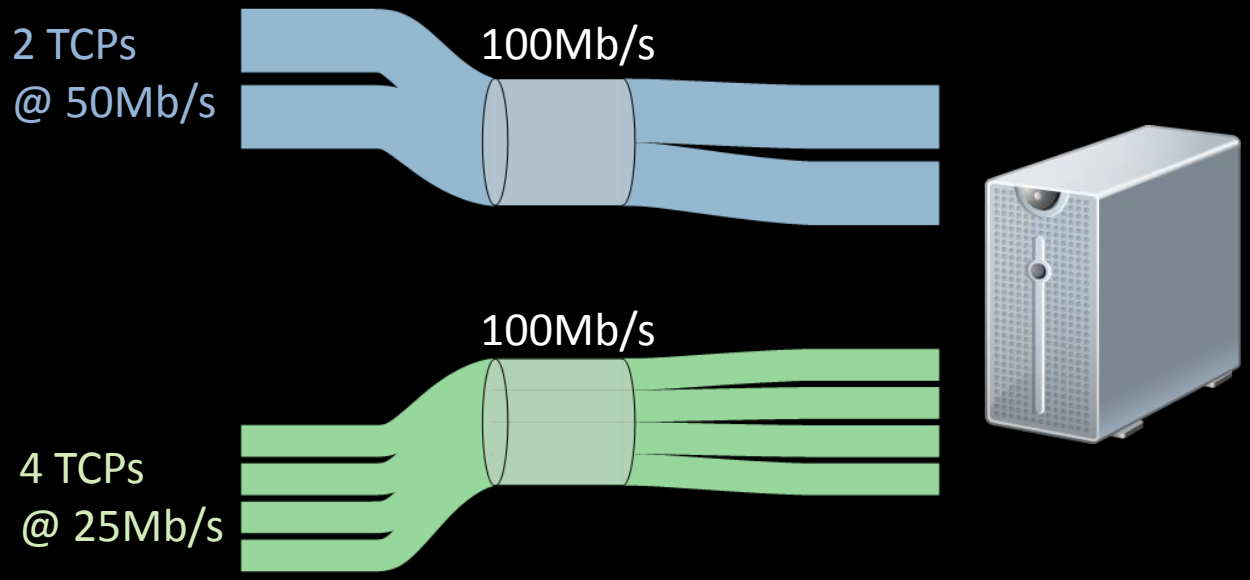


The limited resources are the memory and CPU needed by routers to remember specific paths and costs.

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

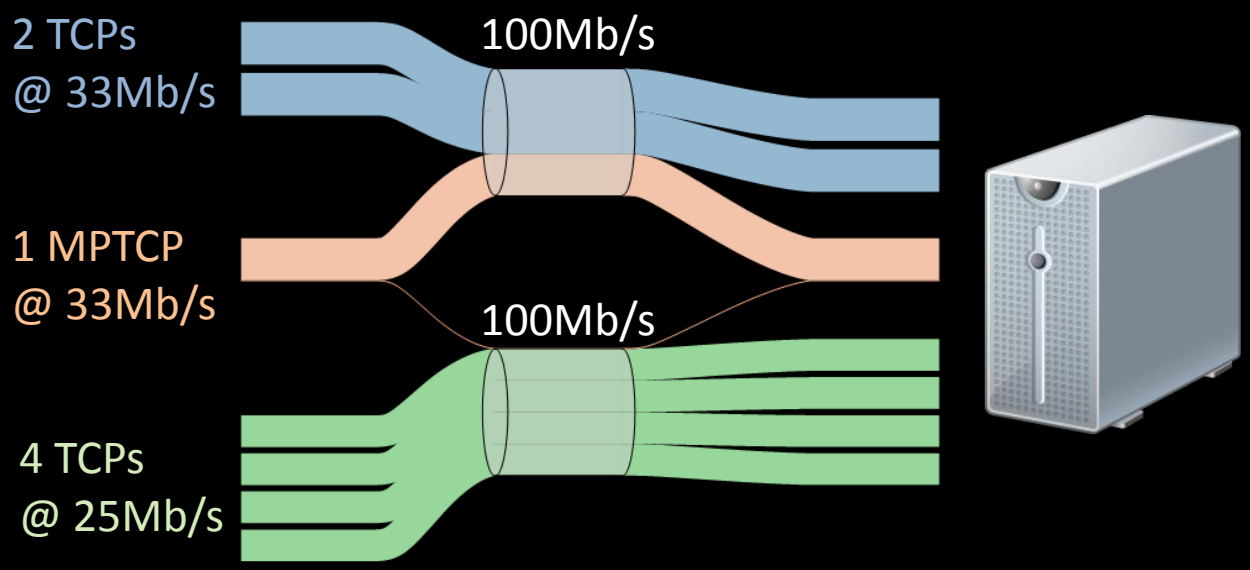
Multi-homed web sites

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



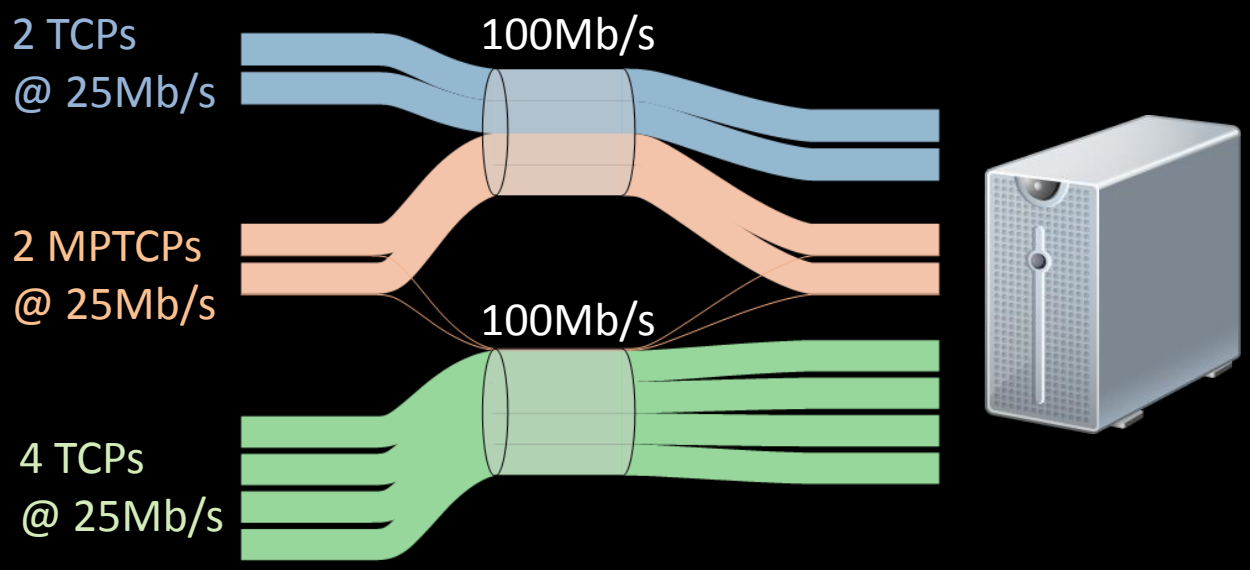
Multi-homed web sites

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



Multi-homed web sites

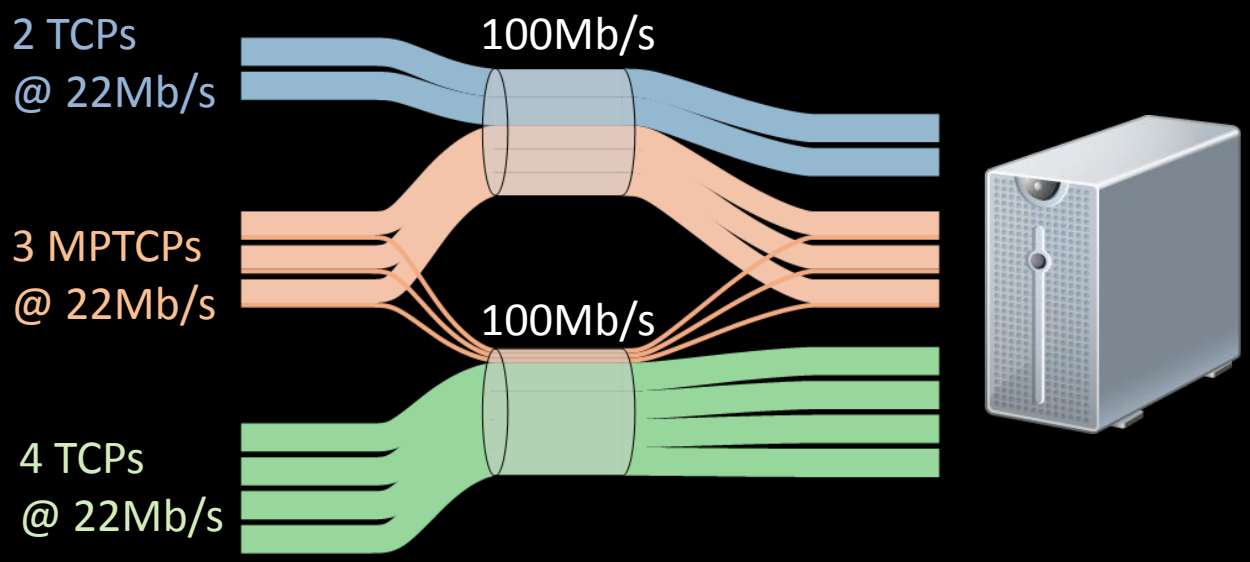
(d) with Kelly+Voice multipath TCP to balance traffic



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

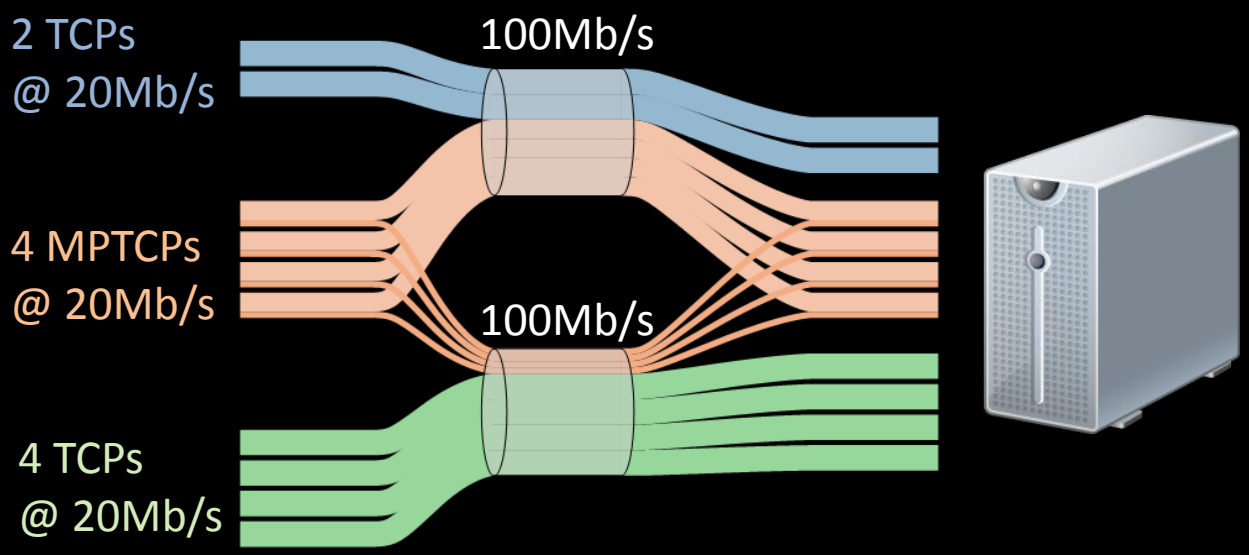


The total capacity, 200Mb/s, is shared out evenly between all 9 flows.

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

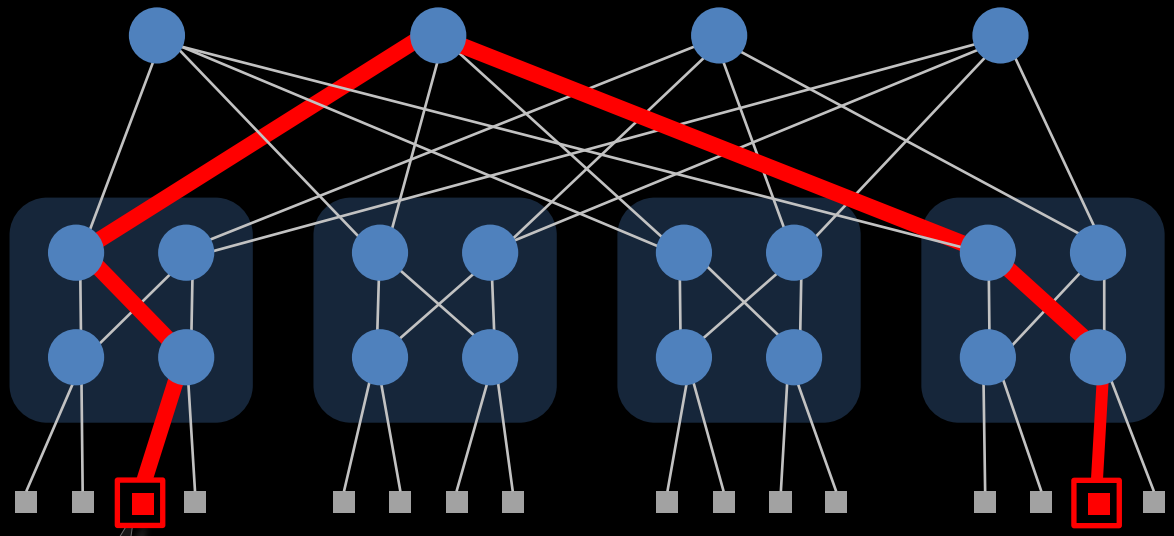
(d) with Kelly+Voice multipath TCP to balance traffic



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

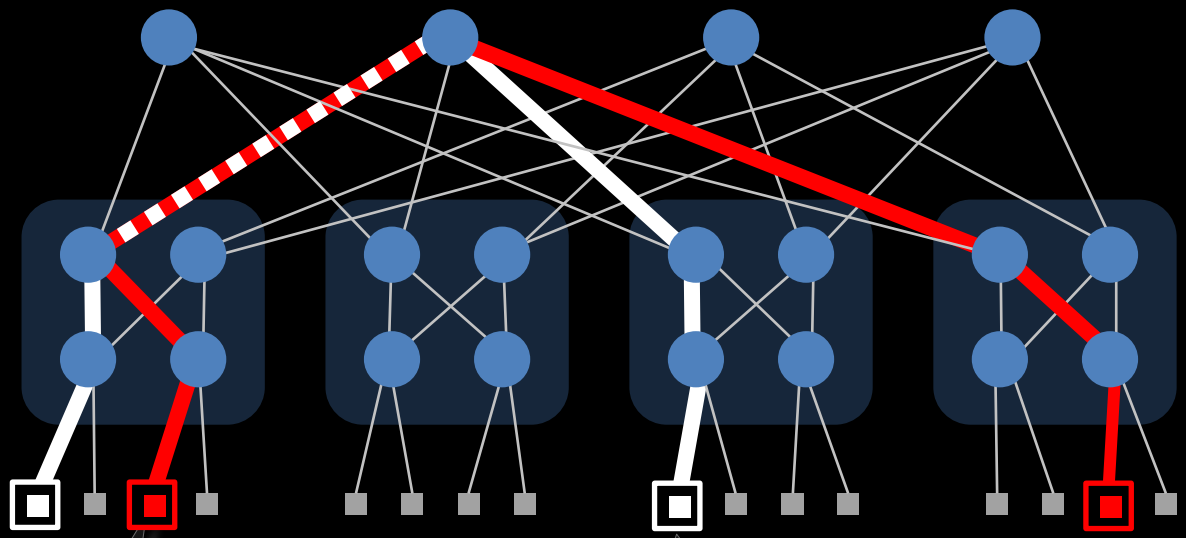
It's as if they were all sharing a single 200Mb/s link. The two links can be said to form a 200Mb/s pool.

Efficient use of data centres



Initially, there is one flow.

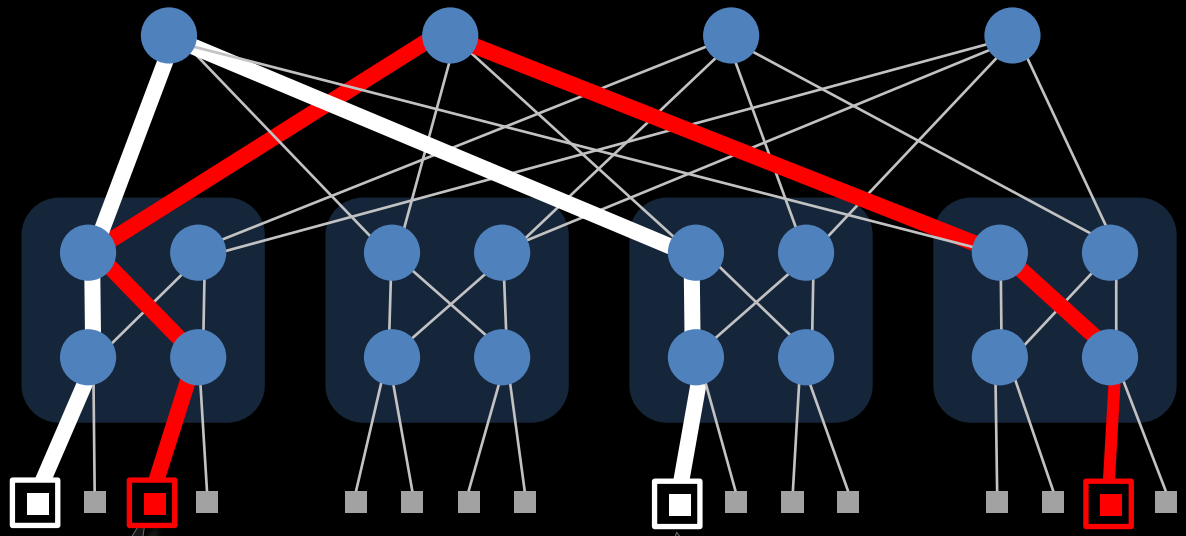
Efficient use of data centres



Initially, there is one flow.

A new flow starts. Its default route collides with the first flow.

Efficient use of data centres

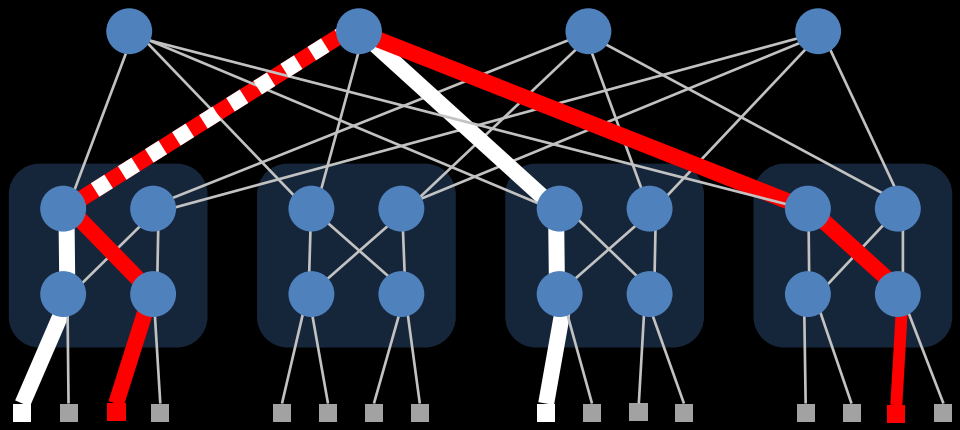


Initially, there is one flow.

A new flow starts. Its default route collides with the first flow.

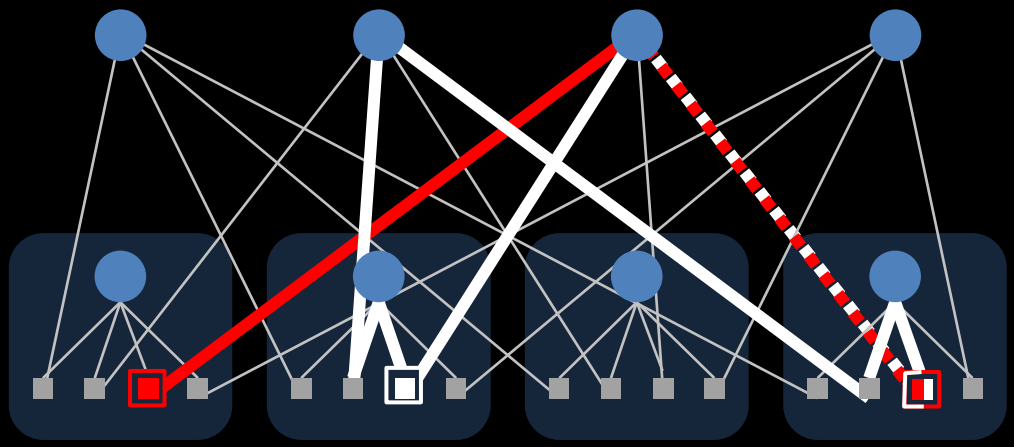
But it has other routes available, which do not collide.

Efficient use of data centres



The current solution is *Equal Cost Multipath* (ECMP). It says: for each flow, independently, pick one of the shortest-hop paths, chosen at random.

Efficient use of data centres

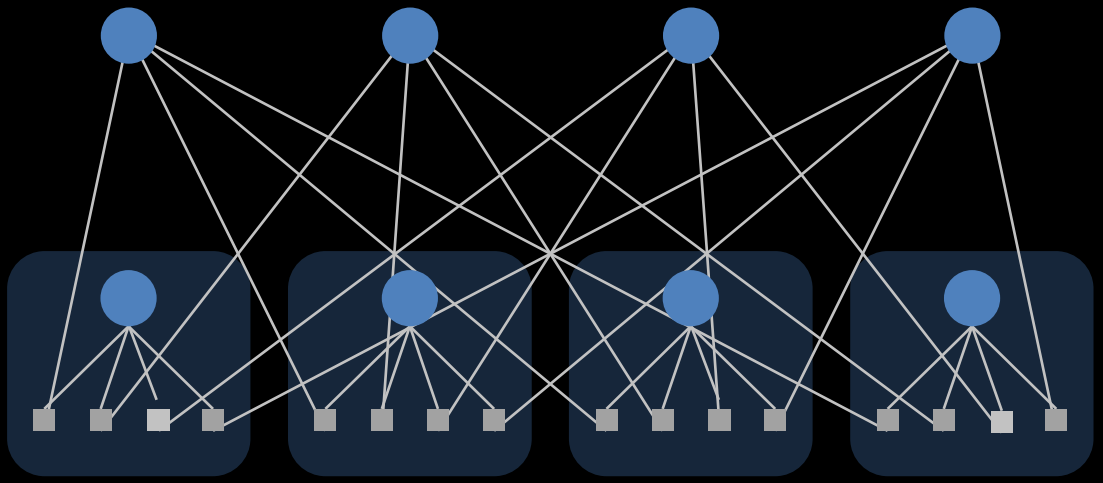


The current solution is *Equal Cost Multipath* (ECMP). It says: for each flow, independently, pick one of the shortest-hop paths, chosen at random.

But ECMP doesn't help, if the optimal answer is to shift traffic onto a longer path.

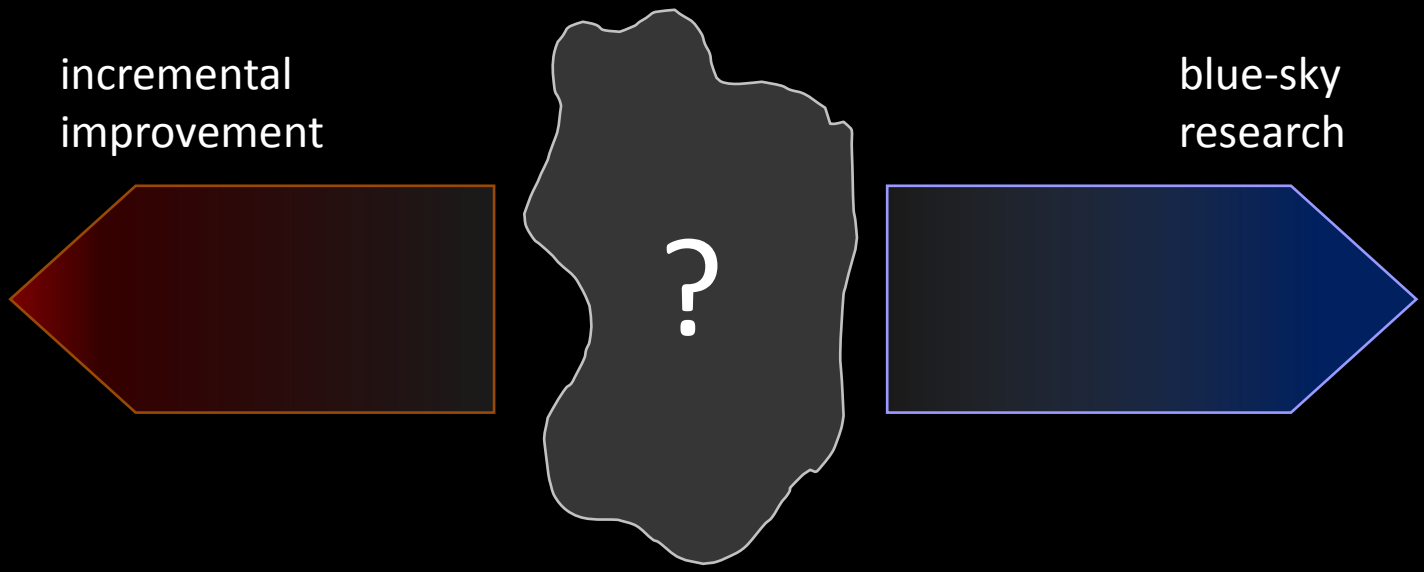
Open question:

Can the Kelly+Voice algorithm make a data center behave like a simple easy-to-manage capacity pool?

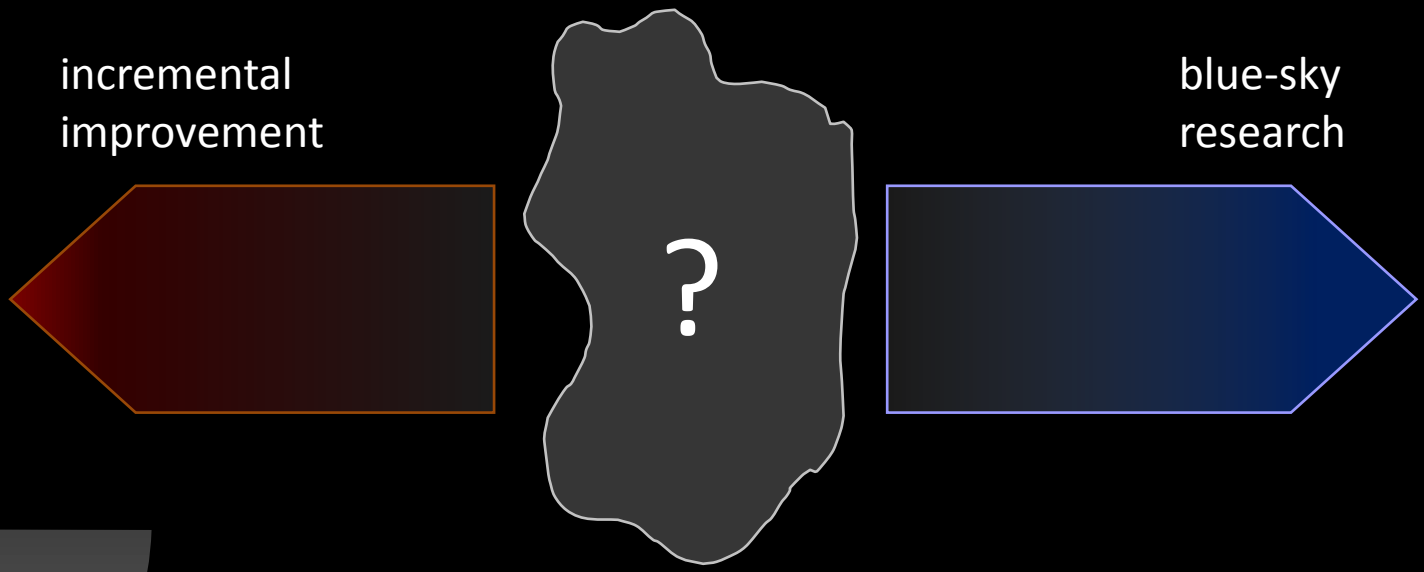


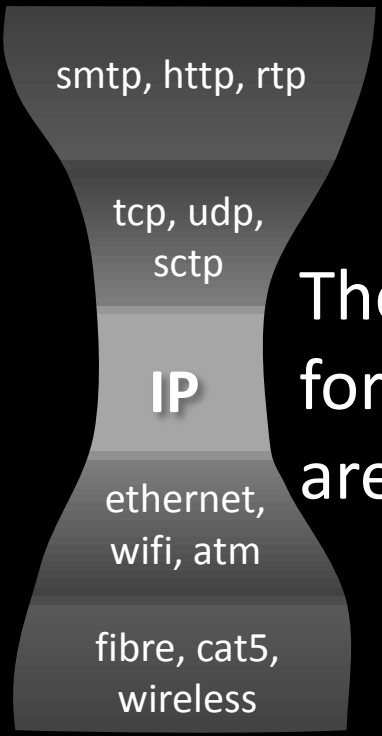
What topologies and path choices might achieve pooling, for what range of traffic patterns?

The spectrum of Internet research



The spectrum of Internet research

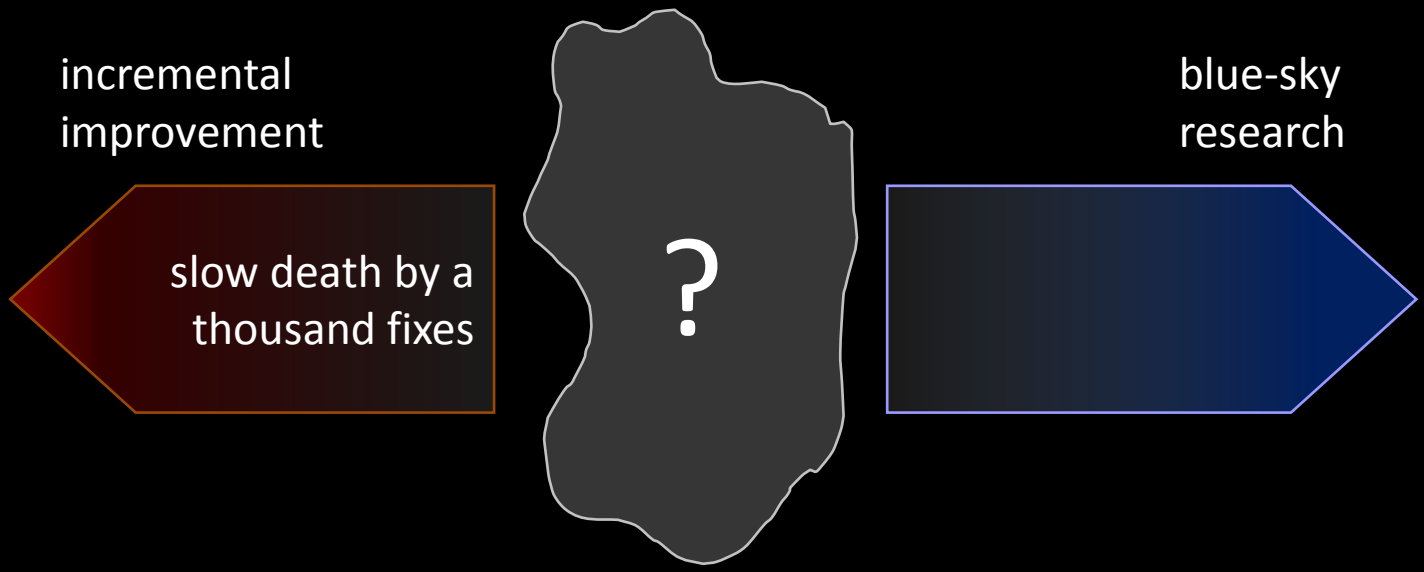


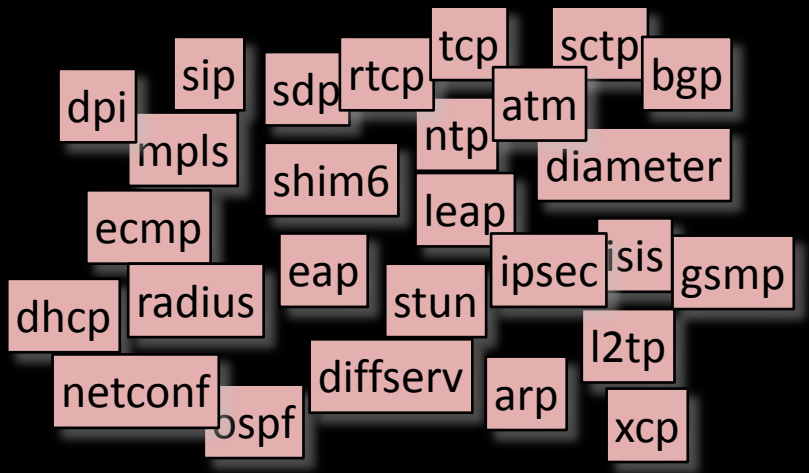


The Internet's standards
for carrying data
are an hourglass.

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

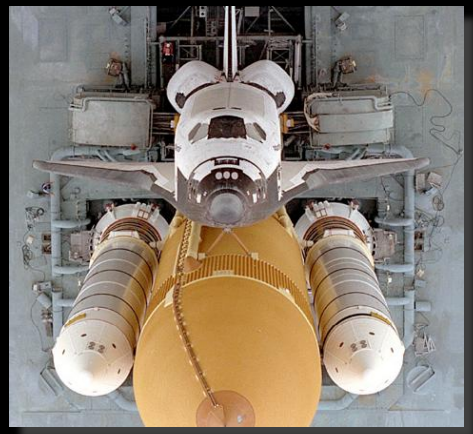
incremental improvement

slow death by a thousand fixes

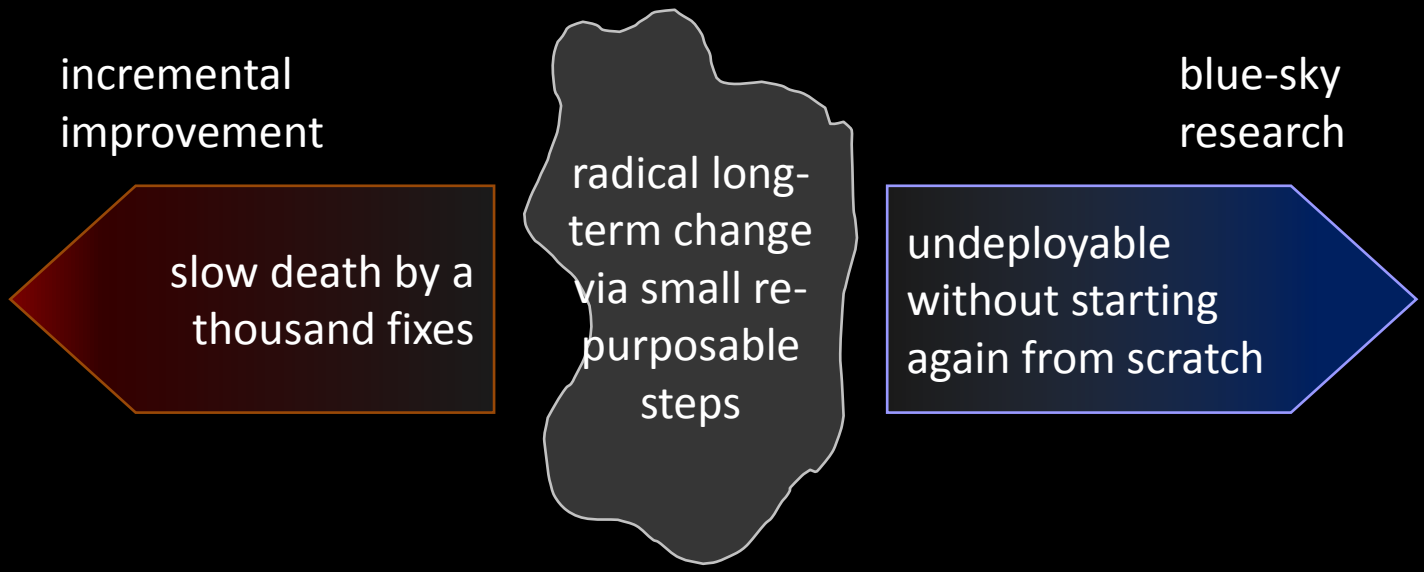
?

blue-sky research

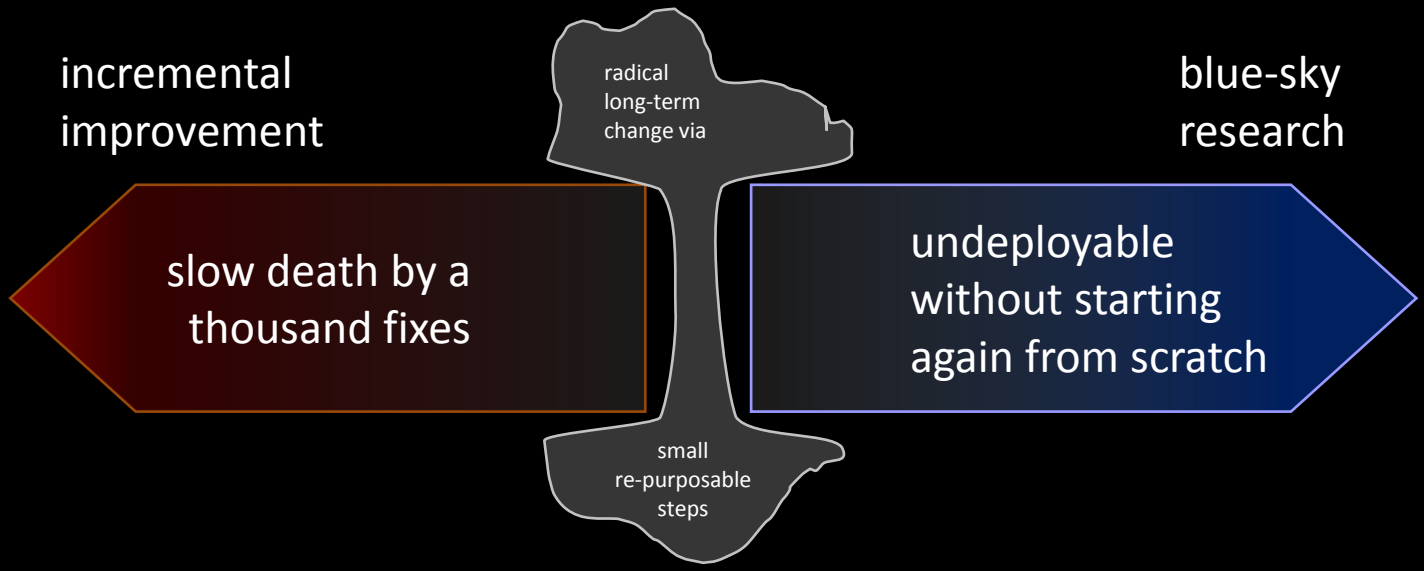
undeployable without starting again from scratch



The spectrum of Internet research



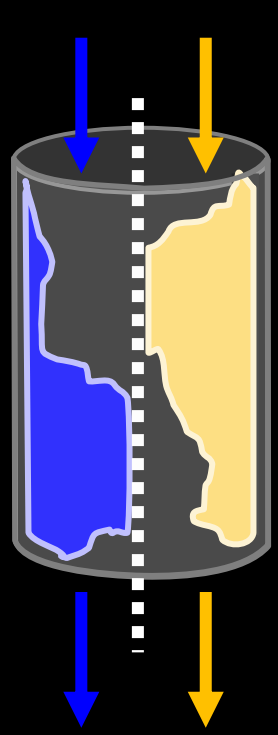
The spectrum of Internet research



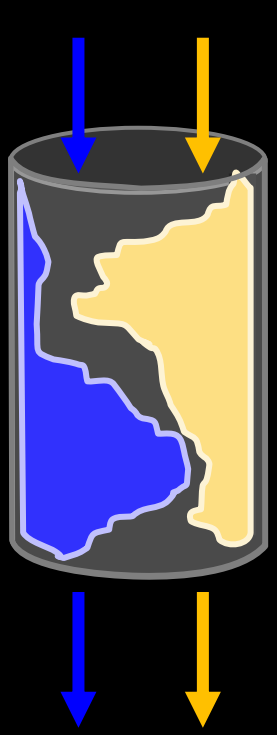
End-system multipath congestion control will succeed because it is a re-purposable interface for solving many different problems — so it can become the ‘narrow waist’ of the Internet’s control architecture.

“Network utility maximization” is mathematician’s shorthand for this.

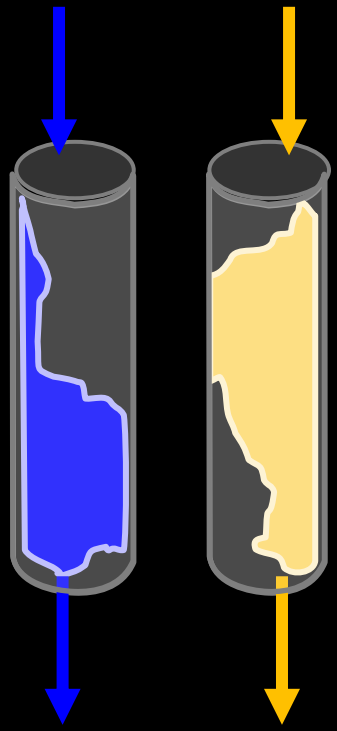
Multipath is Packet Switching 2.0, and multipath congestion control is TCP 2.0.



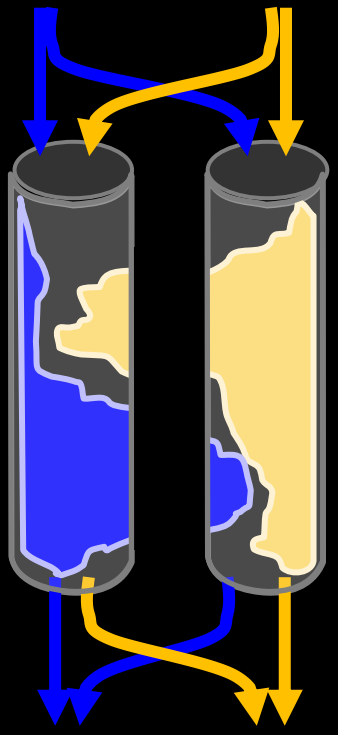
Two circuits



A link



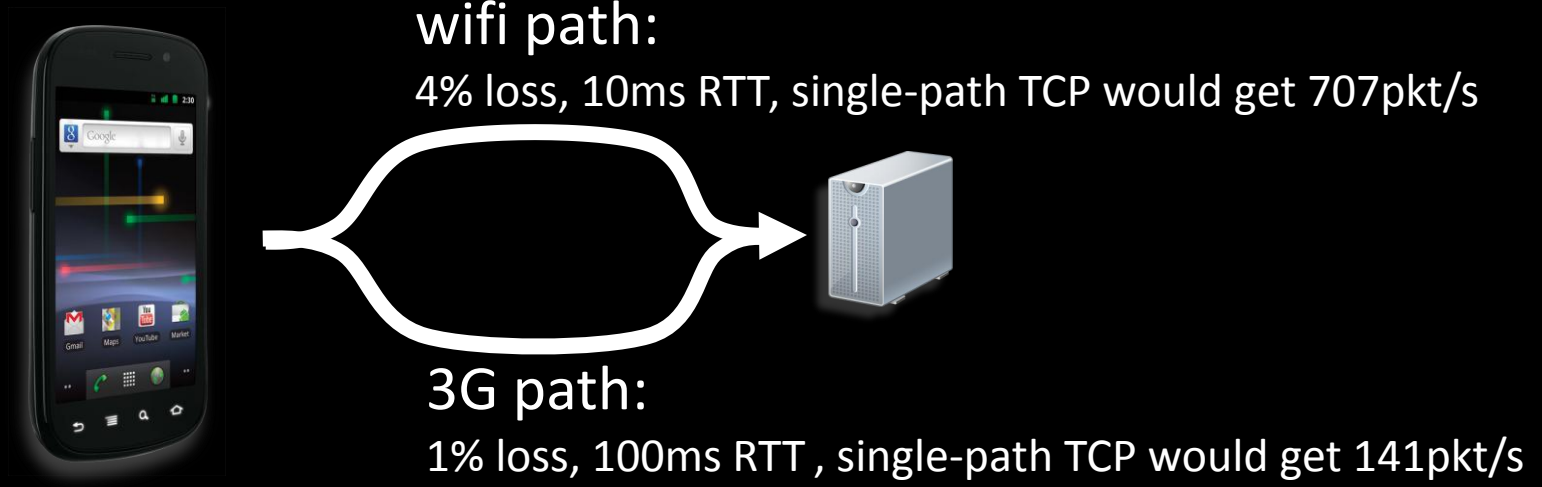
Two separate links



A pool of links

We have made some changes to Kelly+Voice multipath congestion control, to make it an incremental step from the current Internet.

1. Be fair to single-path TCP
2. Change its bandwidth probing, so it works well for individual flows and not just for ‘fluid’ aggregates



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

So should a multipath user be fair to other 3G users and take only 141pkt/s, or should he/she be competitive with wifi and take 707pkt/s?

The fluid model for congestion control is no good for describing individual flows.

- With the obvious packet-level implementation of Kelly+Voice, a single flow is bistable.
  Network systems people will test the algorithm on a small scale, see it doesn't work, and not look any further.
- Most congestion today is at access links, where there is little aggregation, so no one cares about fluid model stability (yet).
  e.g. CompoundTCP solves the problem of bloated buffers at access links, with no regard for the core

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.

- Stability of fluid models
  will be a big issue once the US core is congested. But it's not the pressing need now.

Further questions for mathematicians

In what sense can a data centre be made to behave like a simple capacity pool?

For static traffic patterns, the answer comes from simple linear algebra.
What about dynamic flow-level traffic models?

How will resource pooling affect the pricing models of network operators?

Will it force congestion exposure and congestion pricing?

What is a principled way to model probing and adaptation by a single flow?

Idea: dynamic programming

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

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

- 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.

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 $w_r/2$