Current Directions in Networking and Cloud Computing

P. Brighten Godfrey
University of Illinois at Urbana-Champaign
pbg@illinois.edu

Imam University Futures in Information Technology Program
July 10, 2012

Cloud Computing

Cloud Computing: Computing as a utility

- Purchase however much you need, whenever you need it
- Service ranges from access to raw (virtual) machines, to higher level: distributed storage, web services

Implications

- Reduces barrier to entry to building large service
  - No need for up-front capital investment
- No need to plan ahead
- Reduces cost
- Compute and storage becomes more centralized

"The Cloud": Data Centers

Facebook data center, North Carolina

National Petascale Computing Facility,
UIUC

Key advantage: economy of scale

One technician for each 15,000 servers [Facebook]

Facility / power infrastructure operated in bulk

Ability to custom-design equipment

- Facebook (servers), Google (servers & networking gear)

Statistical multiplexing

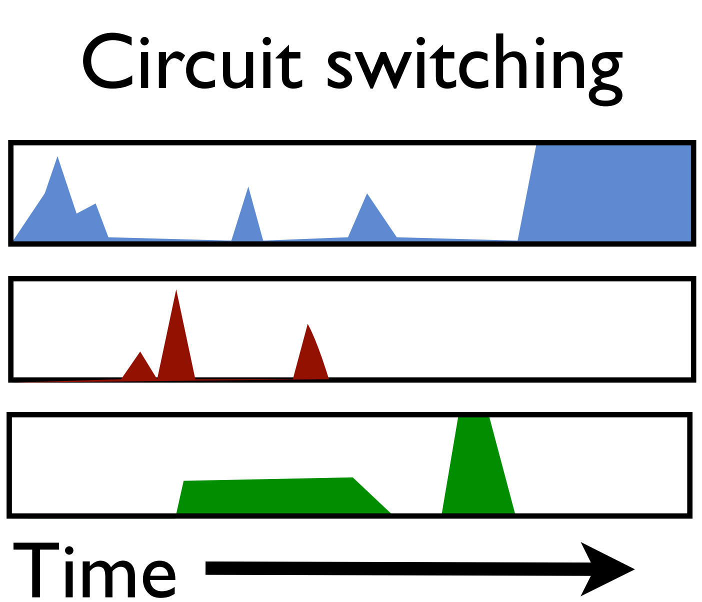
- Must provision for peak load
- Many users sharing a resource are unlikely to have their peaks all at the same time

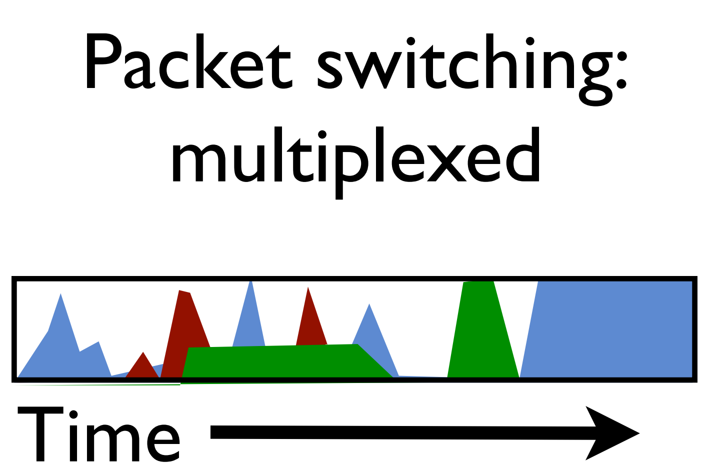
1961-64: Packet switching

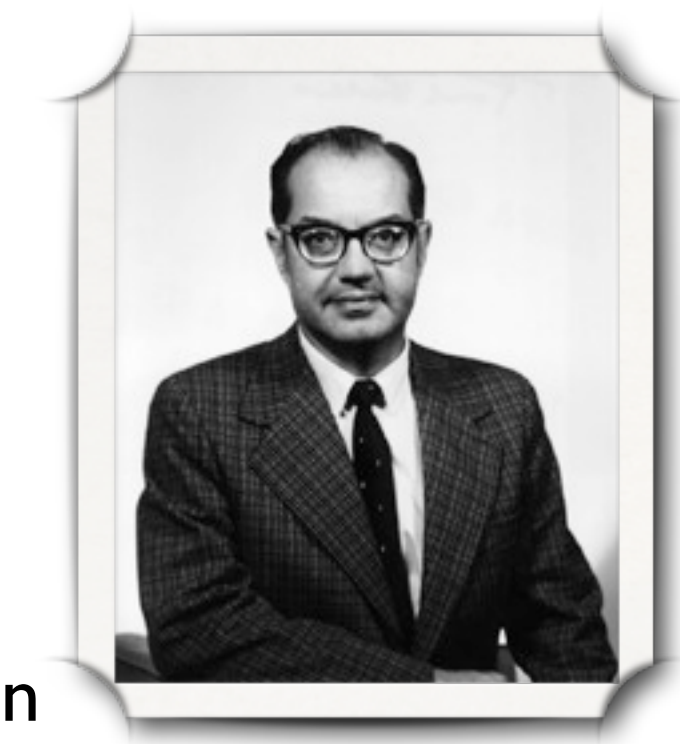
Leonard Kleinrock: queueing-theoretic analysis of packet switching in MIT Ph.D. thesis (1961-63) demonstrated value of statistical multiplexing

Concurrent work from Paul Baran (RAND), Donald Davies (National Physical Labratories, UK)

Kleinrock







Baran

Challenges for Cloud Computing

Confidentiality of data and computation

Integration with existing systems

Robustness

Latency

Bandwidth

Programmability

Outline

Importance of low latency to the cloud

High bandwidth within the cloud

Programmable networks

Closing thoughts: Networking Research

Low Latency to the Cloud

Low latency to the cloud

Cloud implies data and computation outsourced and partially centralized

- i.e., physically more distant from users

Fundamental Challenge:
How do we make the net feel like it is *right here* even when it is distant?

even when it is distant?

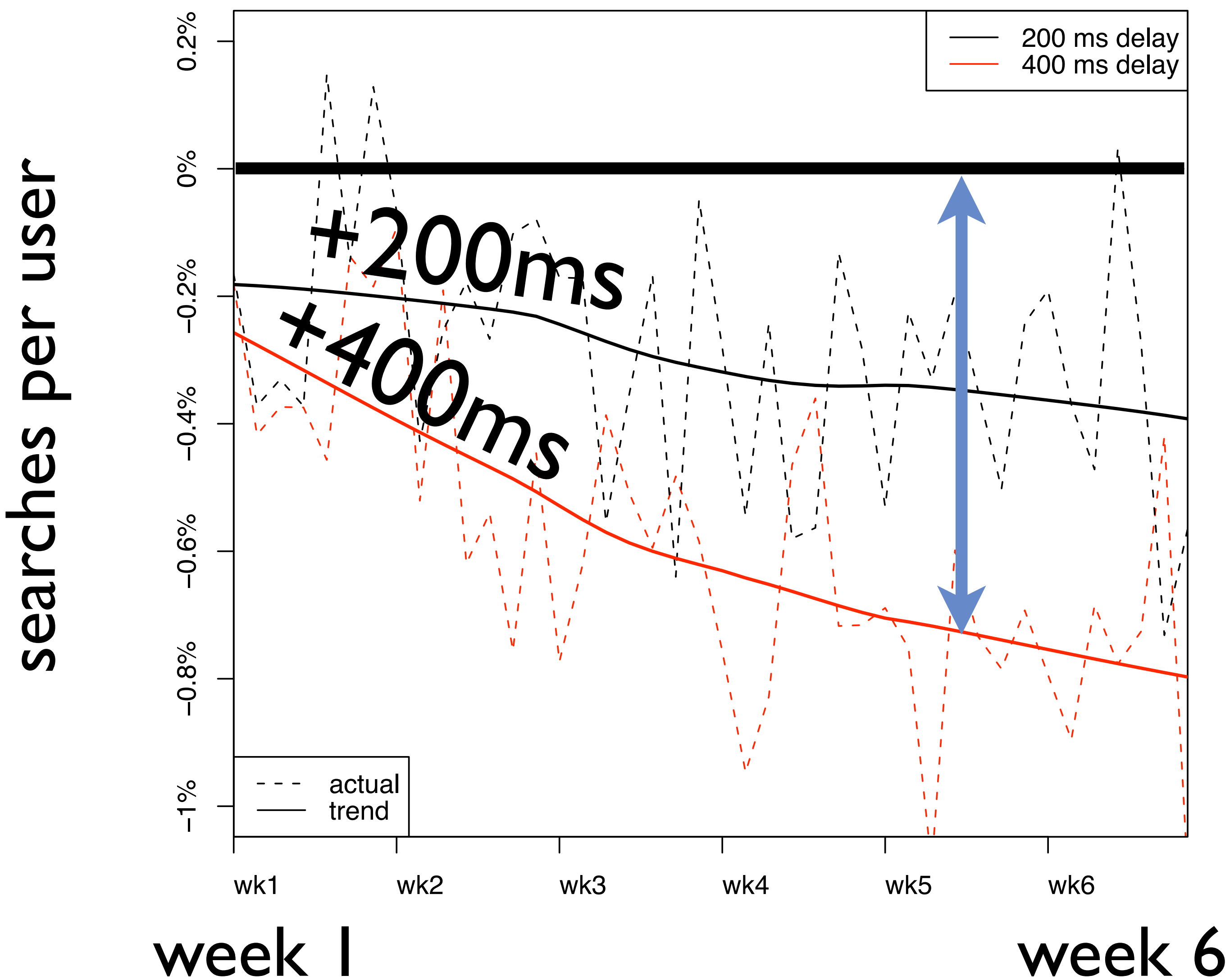
Aside: How much does latency matter to humans?

Milliseconds matter

Hiromi Uehara
“Kung Fu World Champion”

88 msec per note

Milliseconds matter



[Jake Brutlag, Google, 2009]

Low latency to the cloud

Cloud implies data and computation outsourced and partially centralized

- i.e., physically more distant from users

Fundamental Challenge:
How do we make the net feel like it is *right here* even when it is distant?

Possible solutions:

- Bring the cloud closer: "micro-clouds"
- Reduce network latency: better protocols
  - Lots of room for improvement!

High Bandwidth
Within the Cloud

Costs in a data center

Servers are expensive!

Amortized Cost	Component	Sub-Components
$\sim$45%	Servers	CPU, memory, storage systems
$\sim$25%	Infrastructure	Power distribution and cooling
$\sim$15%	Power draw	Electrical utility costs
$\sim$15%	Network	Links, transit, equipment

[Greenberg, CCR Jan. 2009]

Goal: Agility

Agility: Use any server for any service at any time

- Increase utilization of servers
- Reduce costs, increase reliability

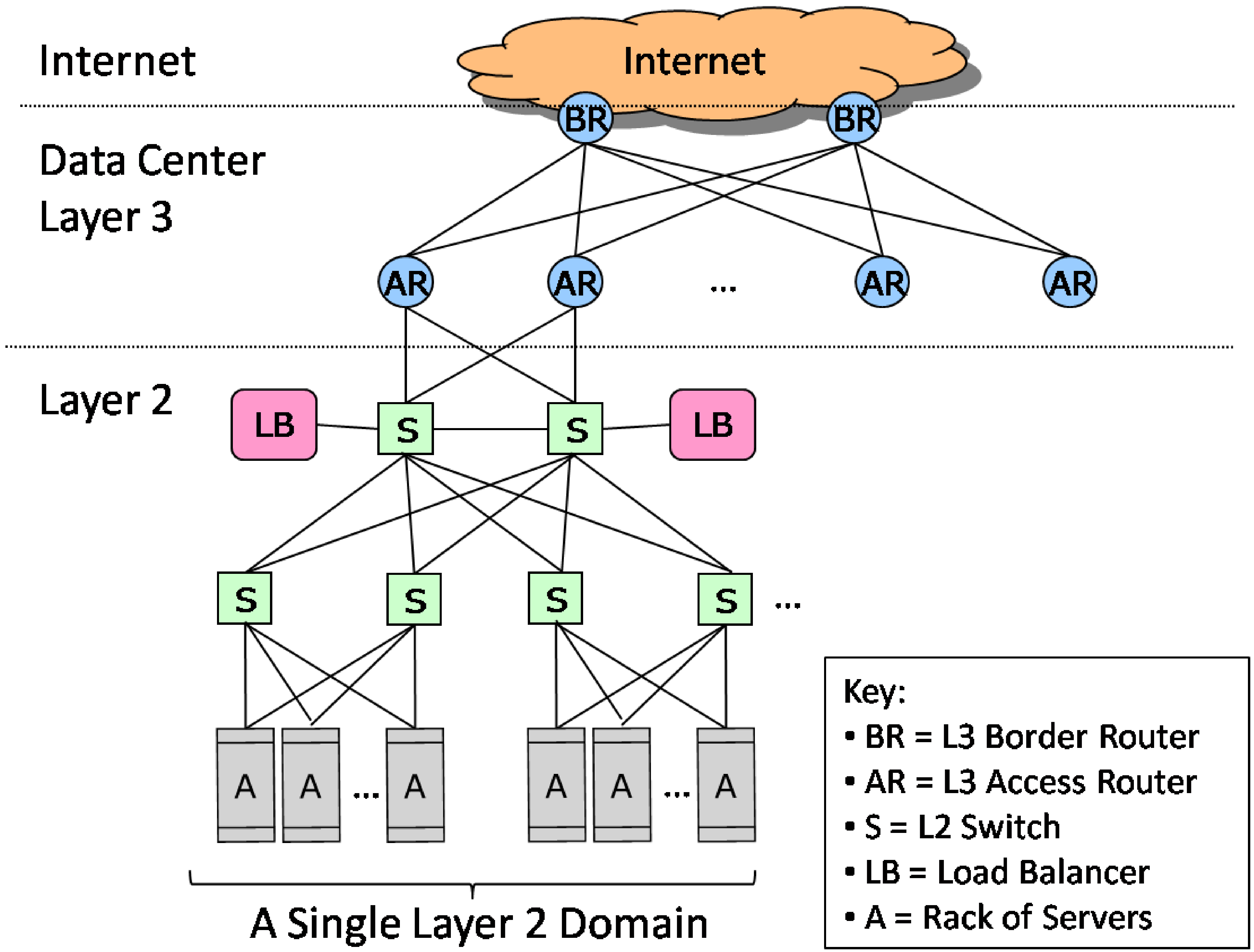
What we need: [Greenberg, ICDCS'09]

- Rapid installation of service's code
  - Solution: virtual machines

- Access to data from anywhere
  - Solution: distributed filesystems

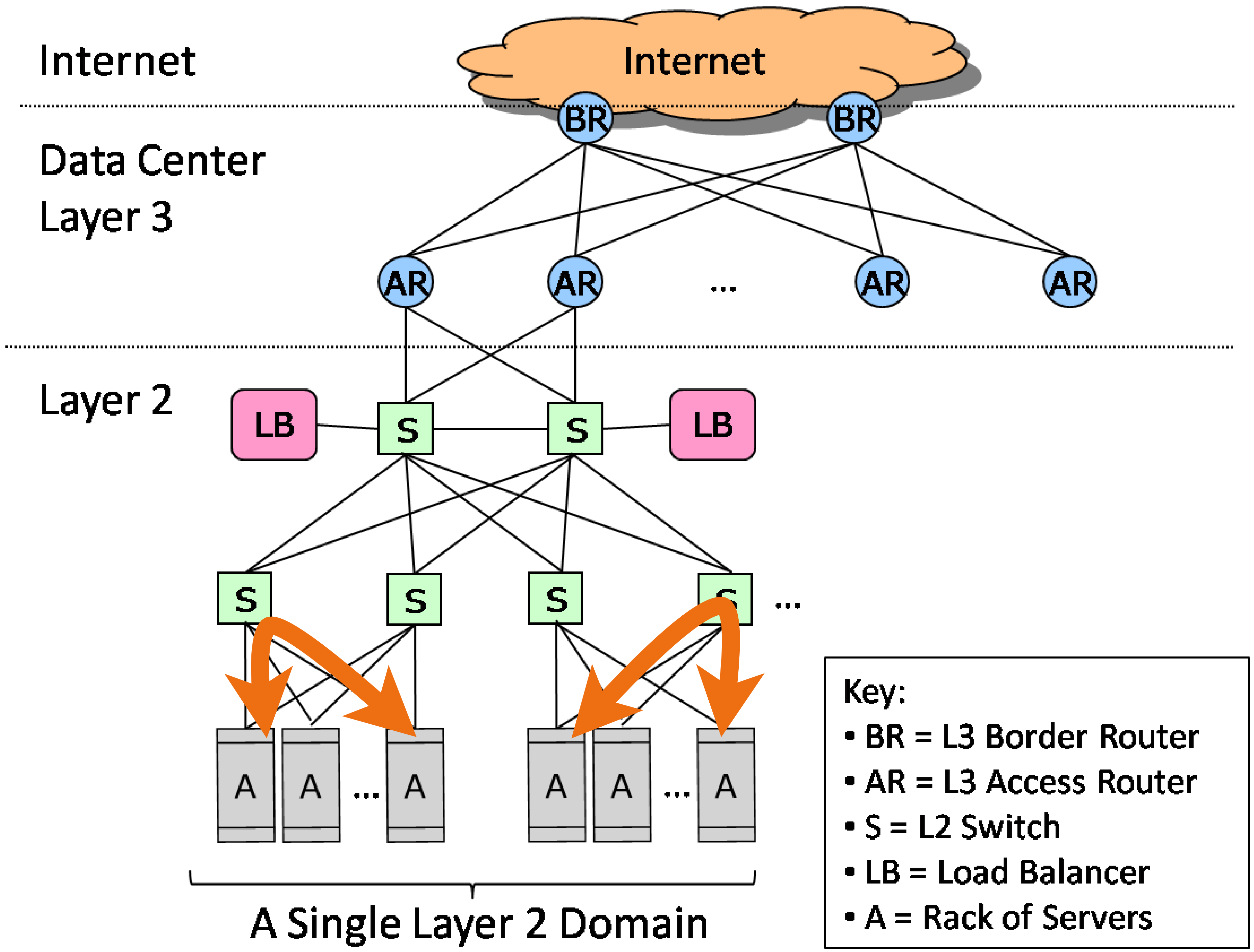
- Ability to communicate between servers quickly, regardless of where they are in the data center

Traditional data center network



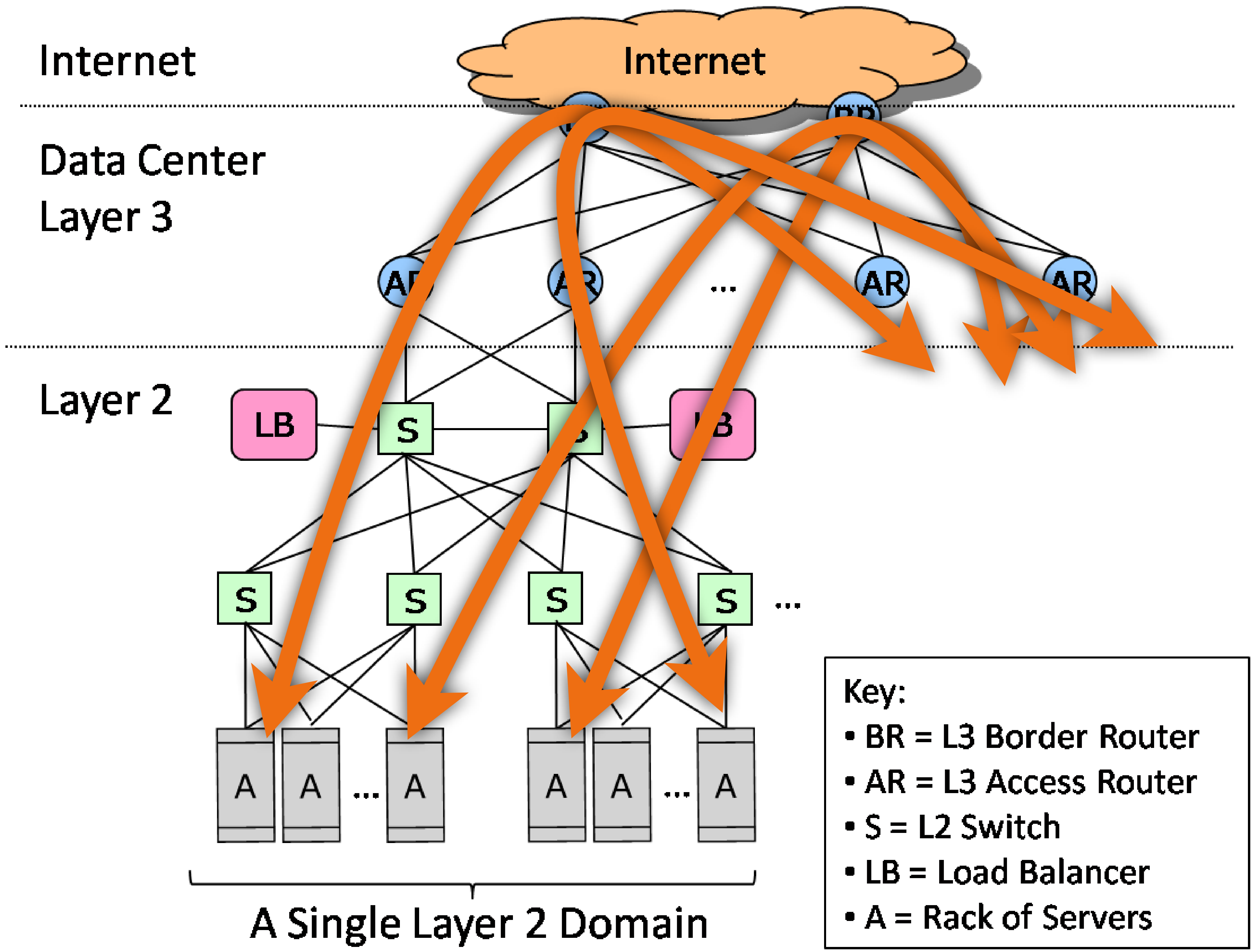
[Greenberg et al, CCR Jan. 2009]

Traditional data center network



[Greenberg et al, CCR Jan. 2009]

Traditional data center network

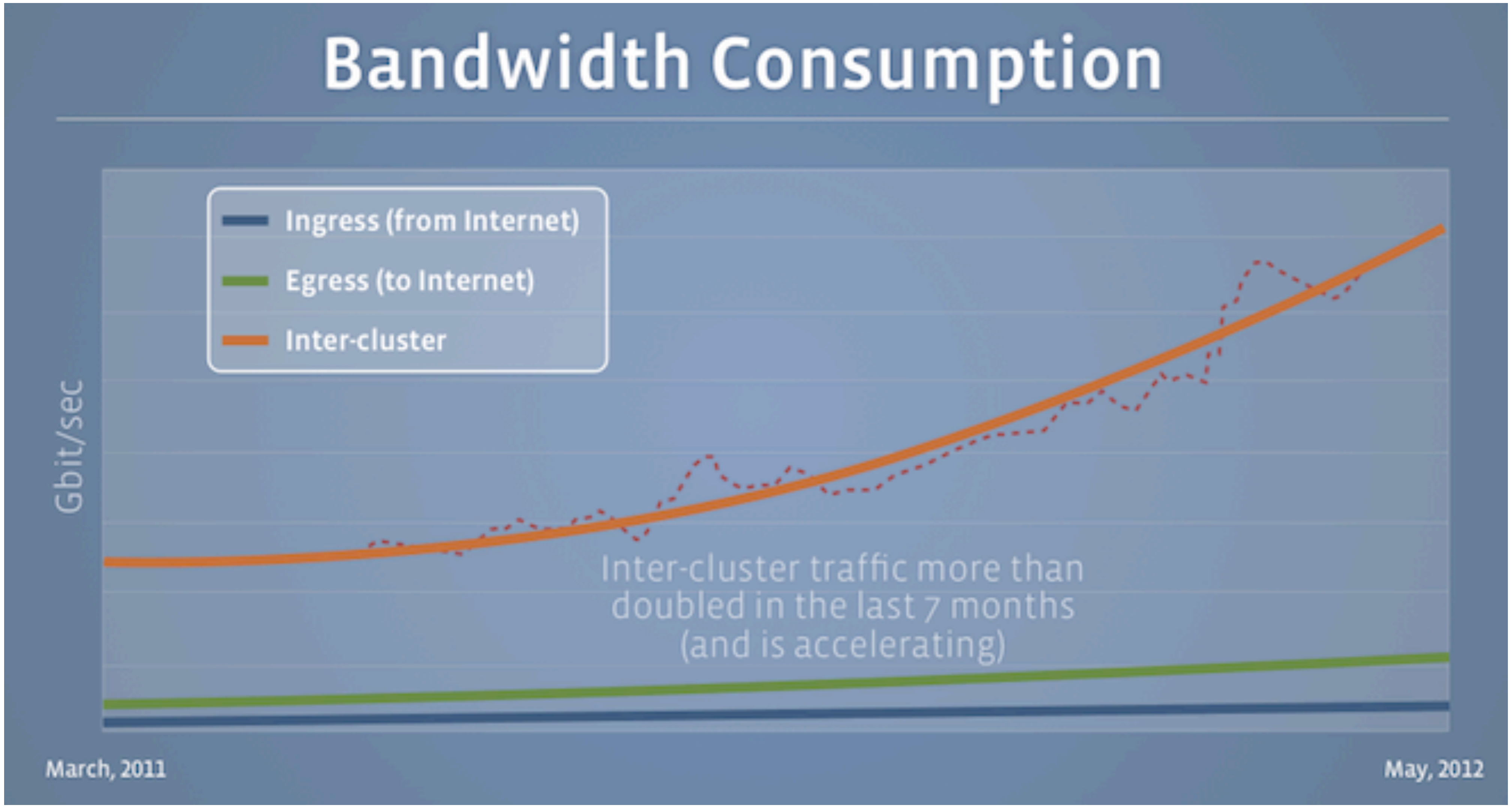


[Greenberg et al, CCR Jan. 2009]

Need for high bandwidth increasing

Big data processing tasks becoming more common

- Web indexing, machine learning, storage replication, ...



[Facebook]

VL2: A Scalable and Flexible Data Center Network

[Greenberg, Hamilton, Jain, Kandula, Kim, Lahiri, Maltz, Patel, Sengupta, SIGCOMM 2009]

Key features:

- Flat addressing
  - Ethernet-style (layer 2) addresses to forward data, rather than IP addresses
  - Separates names from locations
- Randomized load balancing
  - Makes better use of network resources
- High bandwidth network
  - Folded Clos network
  - Special case: fat tree

"Fat tree" network

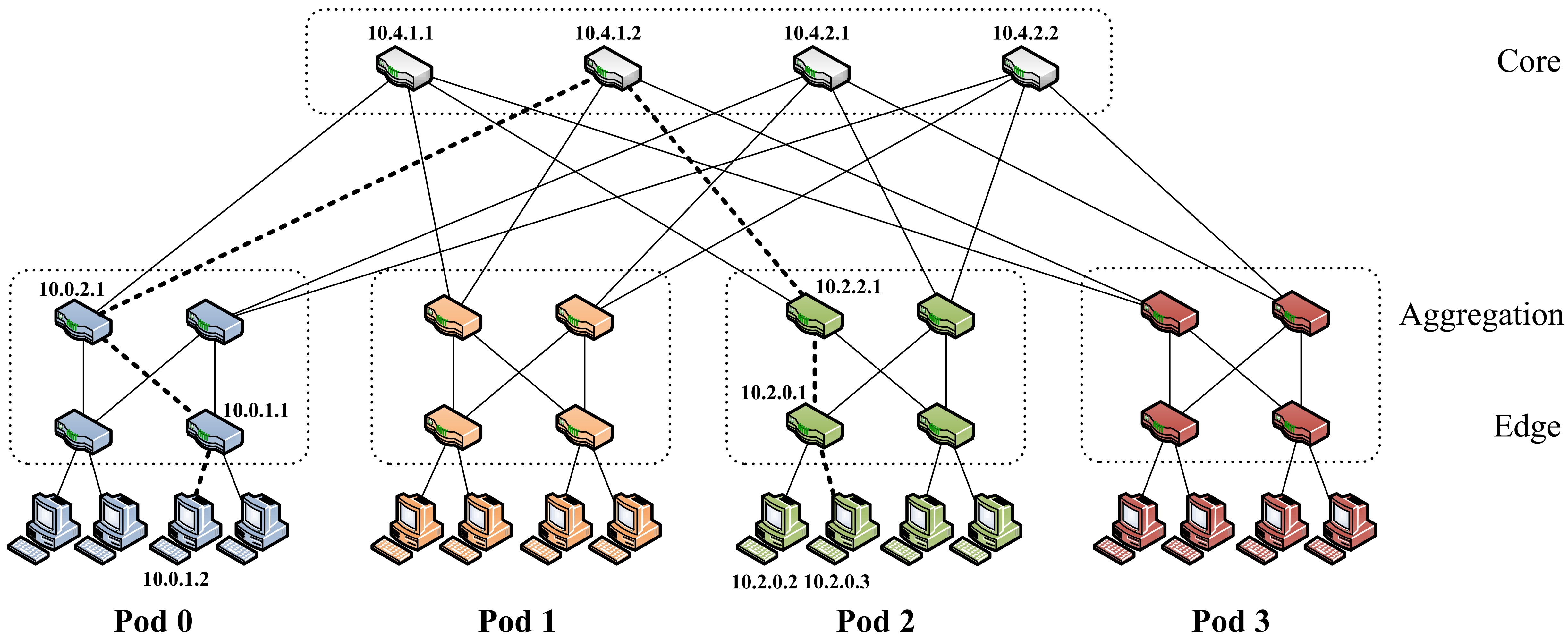
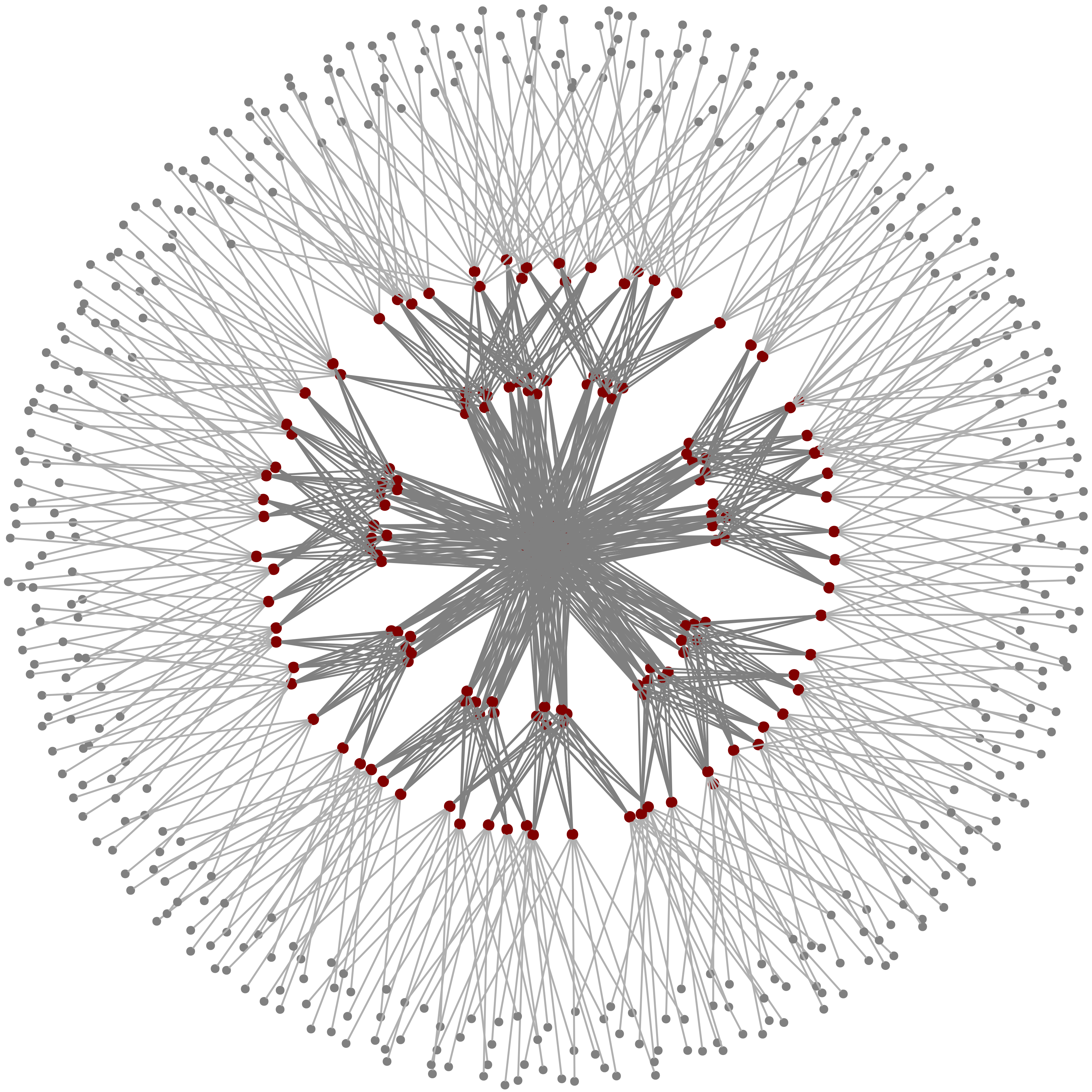


Figure from [Al Fares et al, SIGCOMM 2008]

Nonblocking: servers limited only by their network card's speed, regardless of communication pattern between servers

“Fat tree” network

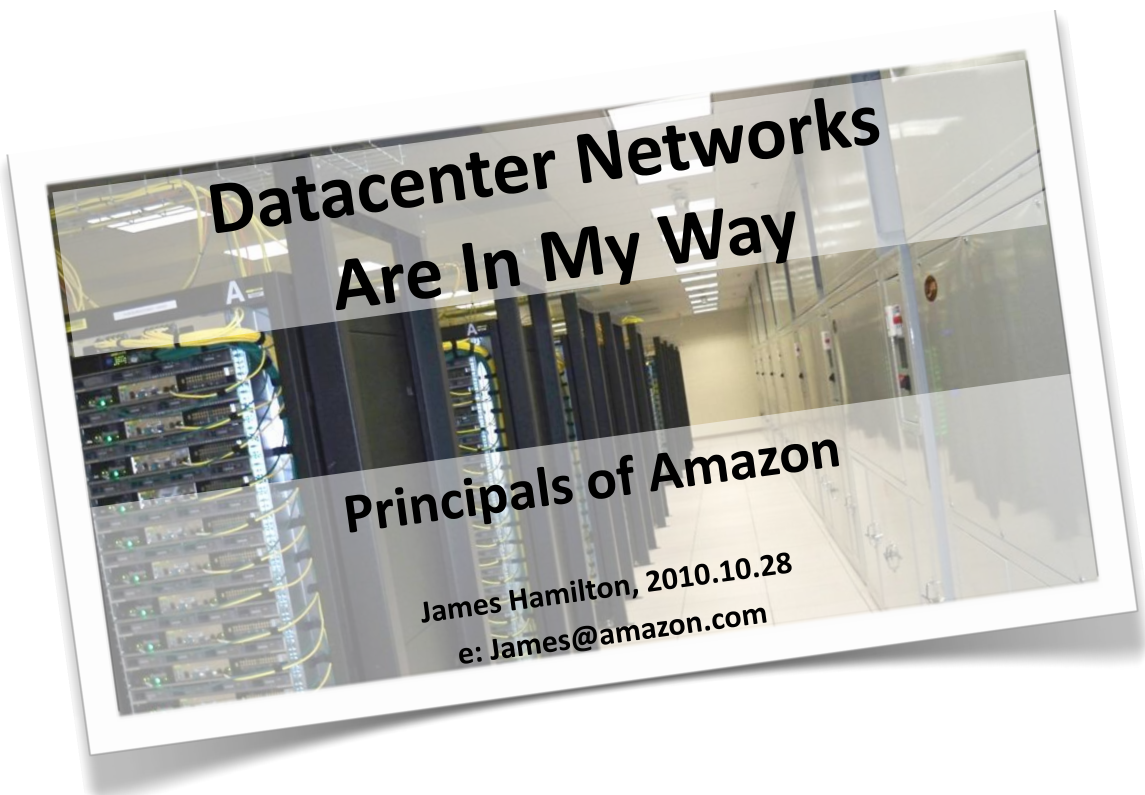


Our work: Jellyfish

[Singla, Hong, Popa, Godfrey, NSDI'12]

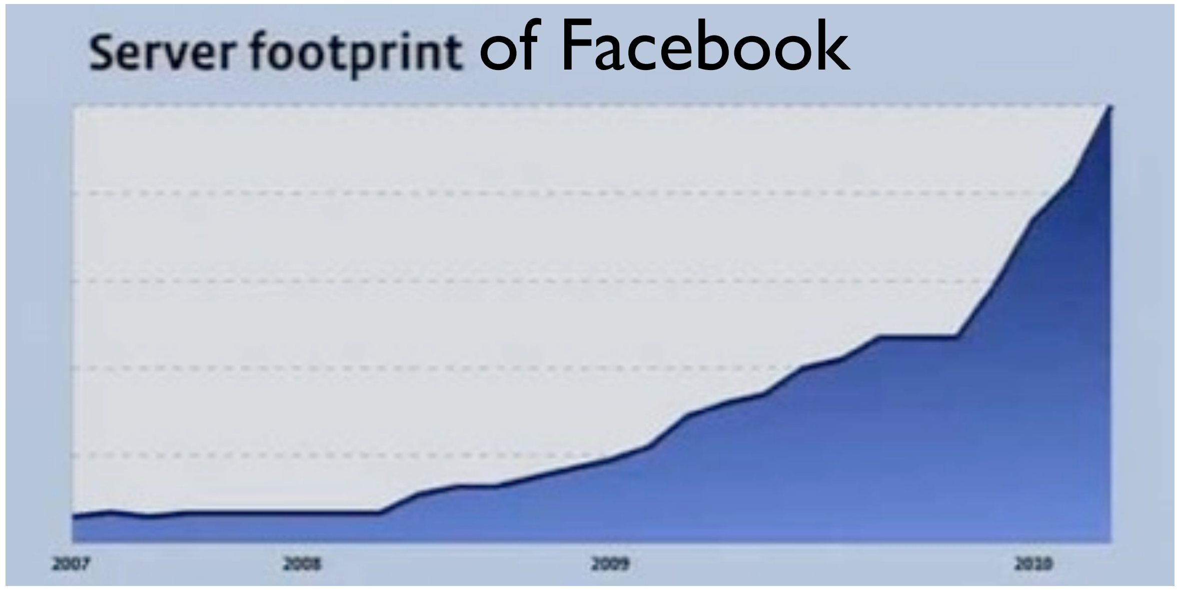
High throughput

Eliminate bottlenecks
"Agile" network



Incremental expandability

Easily add/replace
servers & switches



2007 08 09 10

Structure constrains expansion

Coarse design points

- Hypercube: 2^k switches
- de Bruijn-like: 3^k switches
- 3-level fat tree: $5k^2/4$ switches
    - 3456 servers, 8192 servers, 27648 servers with common switch port-counts

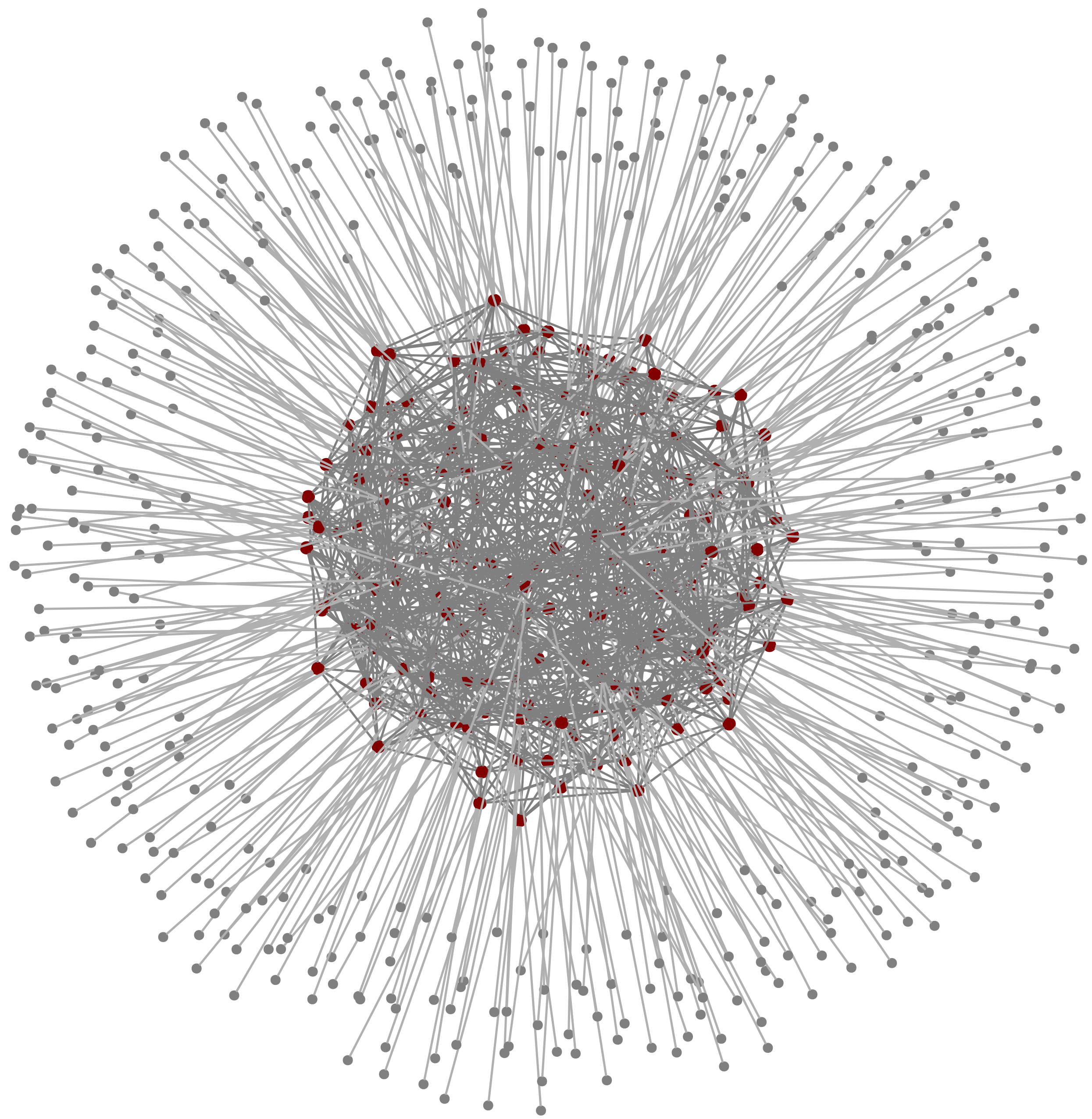
Unclear how to maintain structure incrementally

- Overutilize switches? Uneven / constrained bandwidth
- Leave ports free for later? Wasted investment

Our Solution

Forget about structure –
let's have no structure at all!

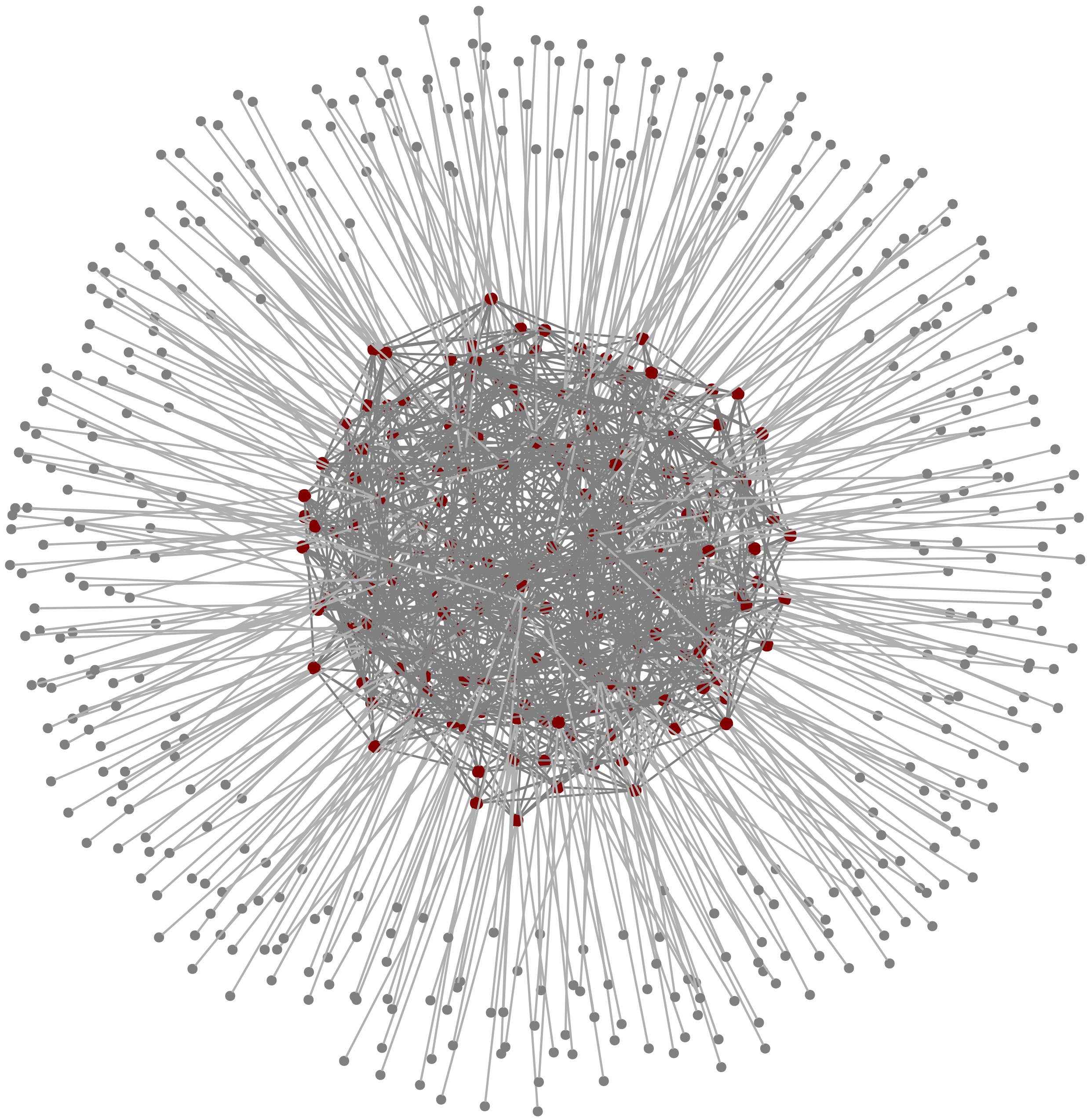
Capacity as a fluid



Jellyfish random graph

432 servers, 180 switches, degree 12

Capacity as a fluid



Jellyfish random graph

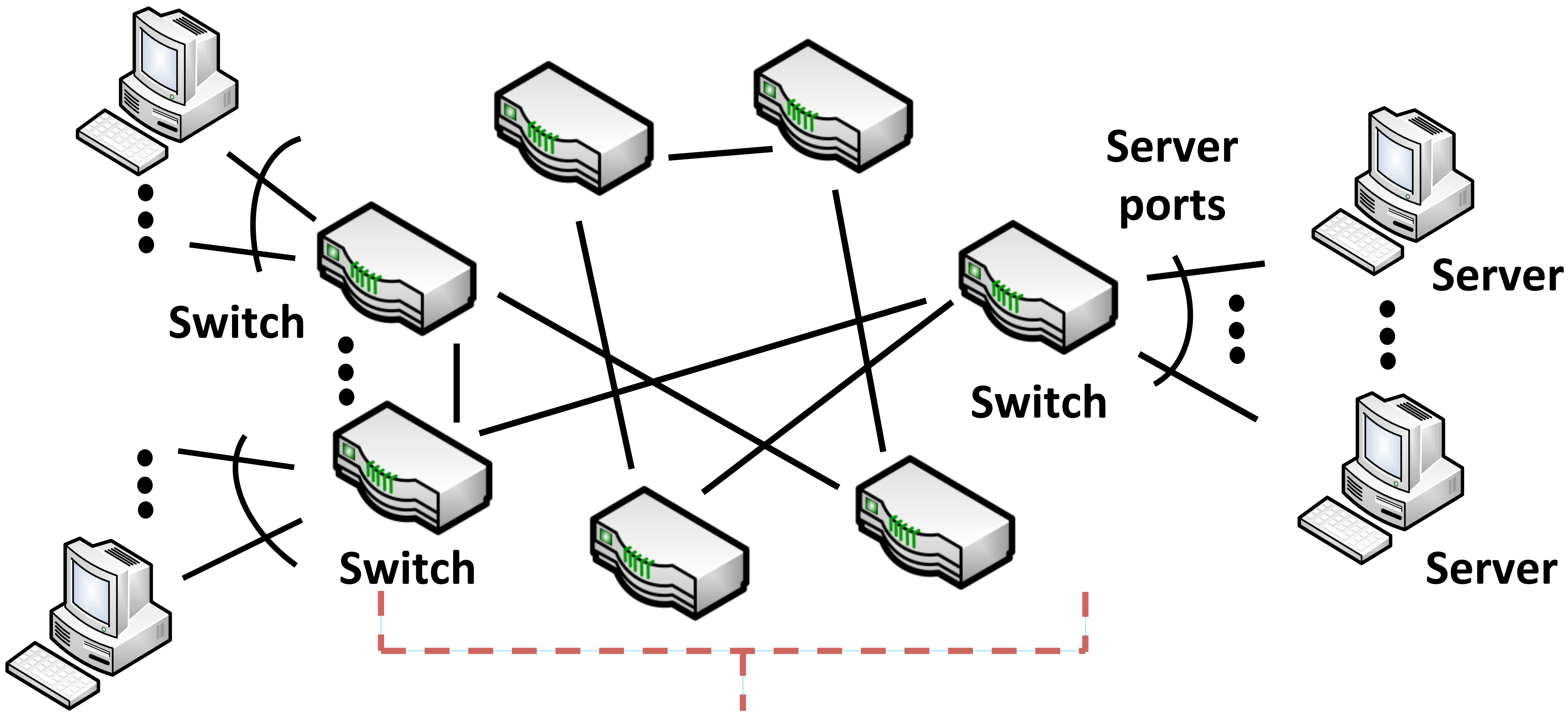
432 servers, 180 switches, degree 12

Jellyfish

Arctapodema (http://goo.gl/KoAC3)

[Photo: Bill Curtsinger, National Geographic]

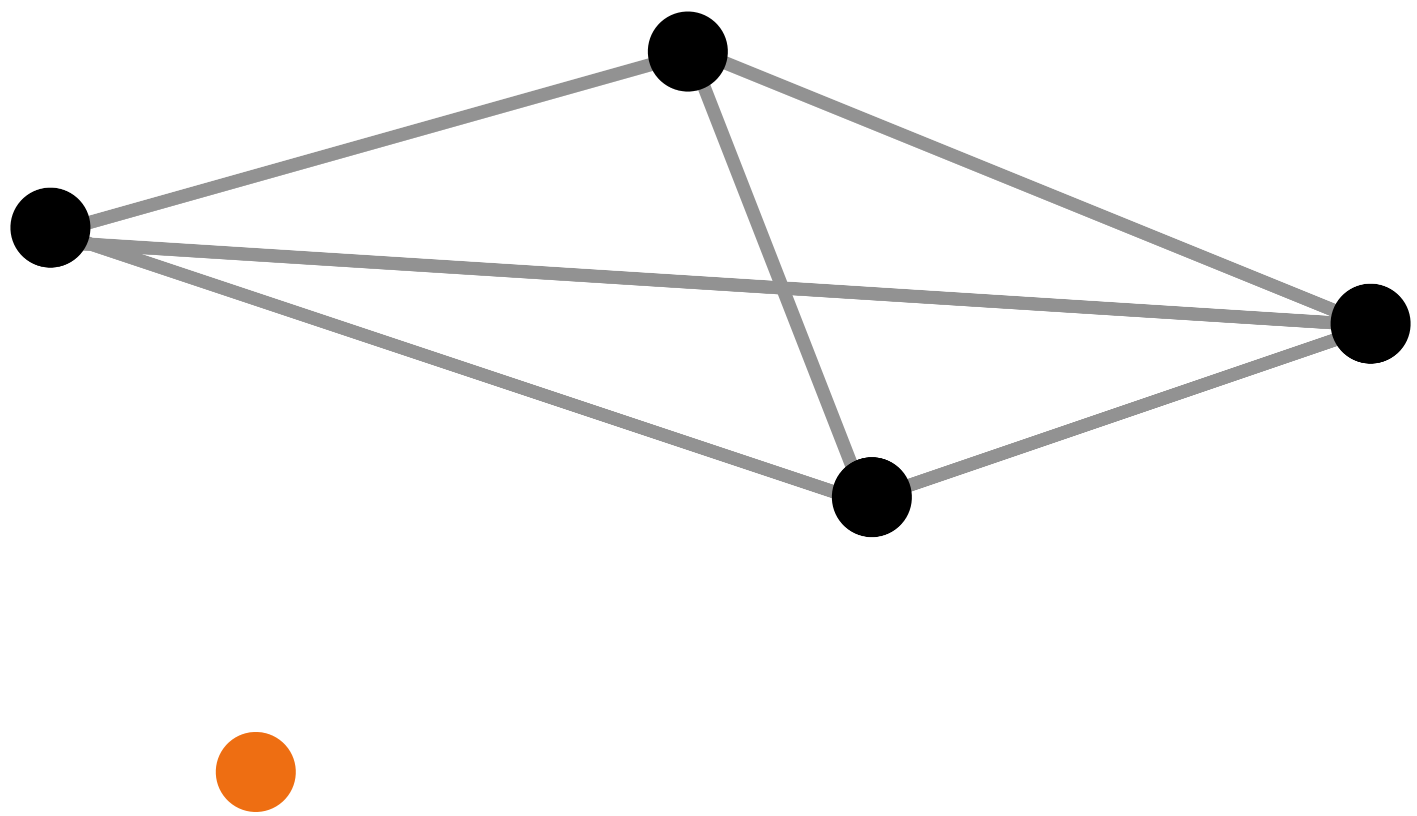
Jellyfish: The Topology



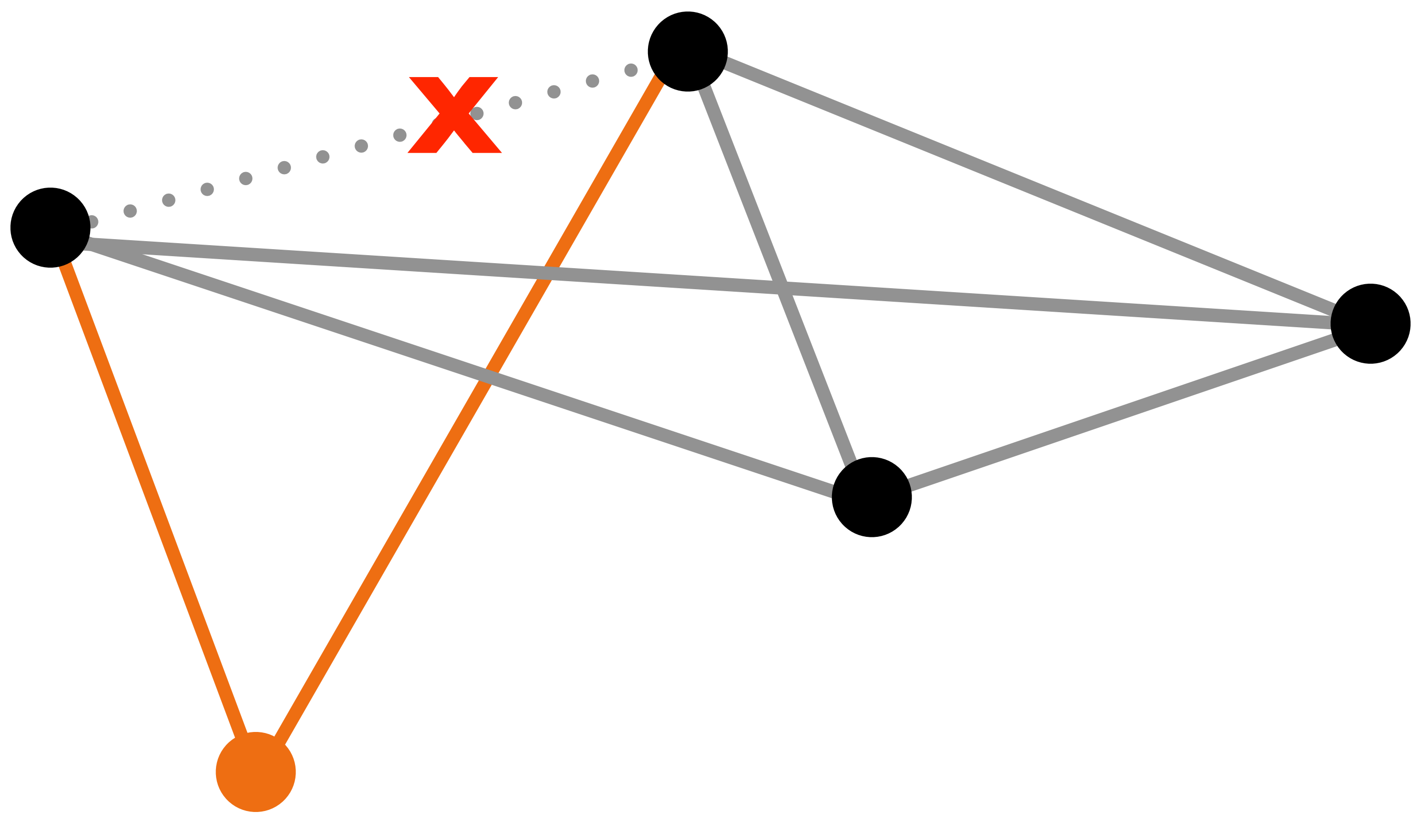
(Approximately) uniform-randomly selected from all valid graphs

Switches are nodes

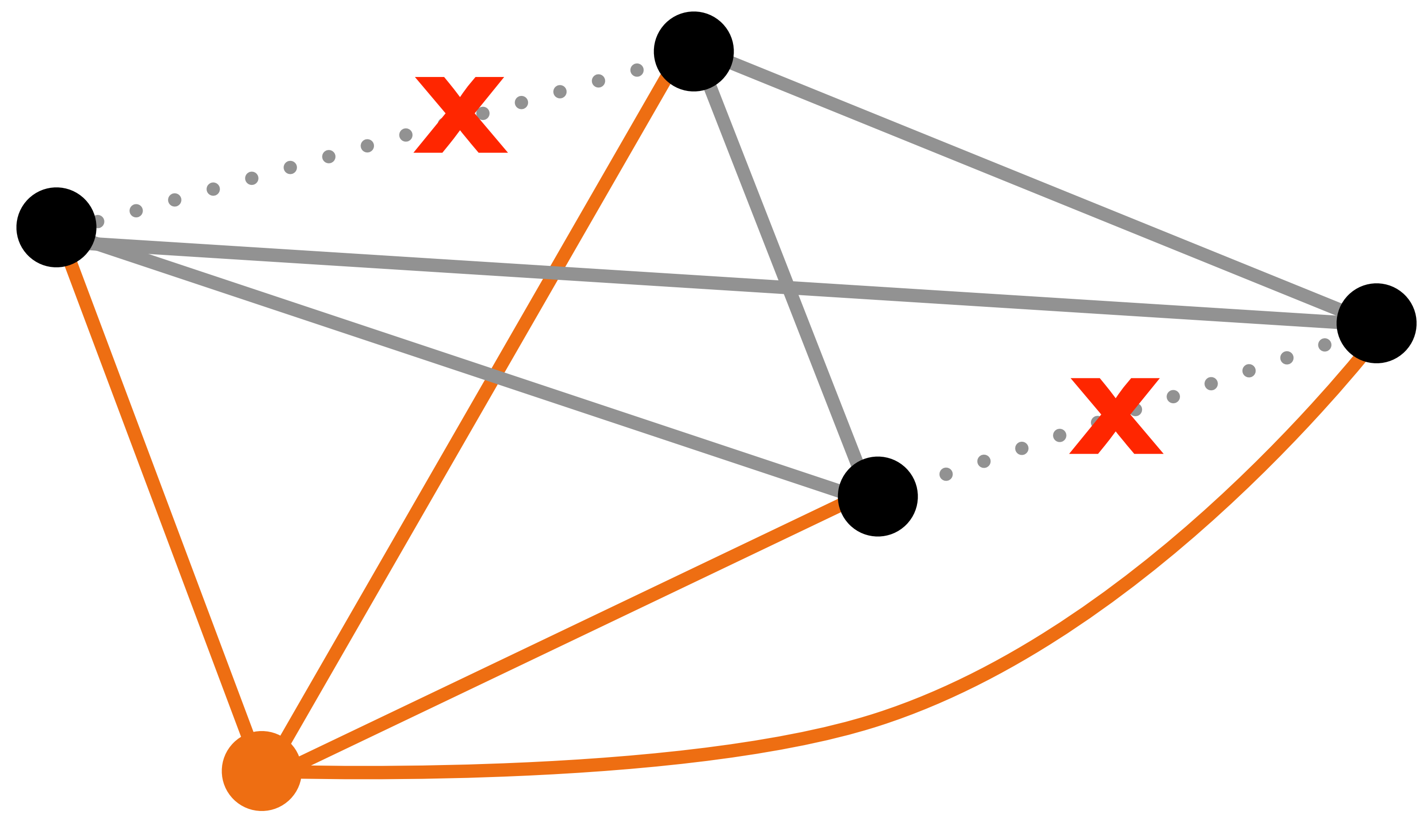
Building Jellyfish



Building Jellyfish



Building Jellyfish



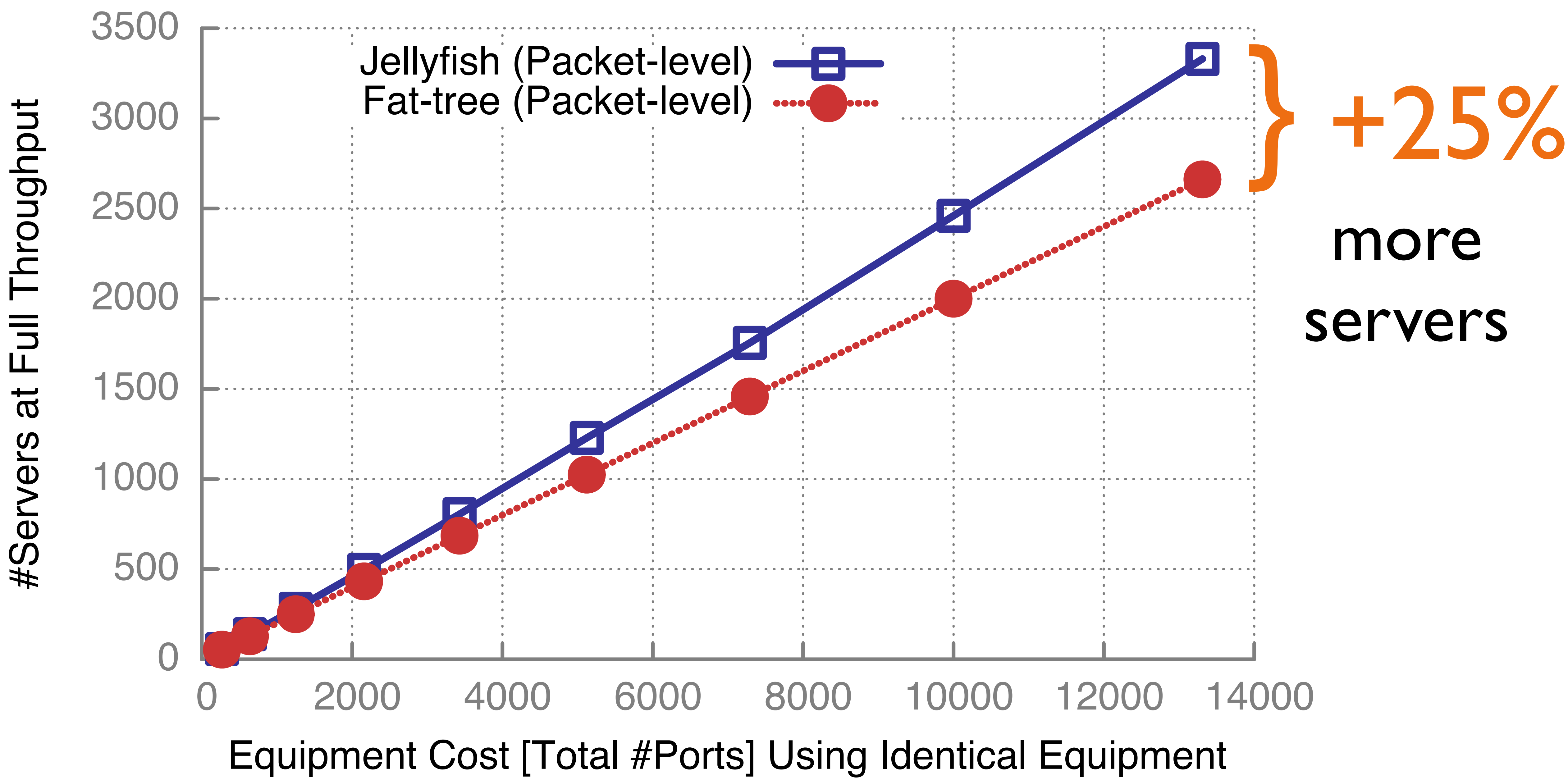
Building Jellyfish

60% cheaper incremental expansion compared with past technique for traditional networks

LEGUP: [Curtis, Keshav, Lopez-Ortiz, CoNEXT'10]

"OK, but... does it really work?"

Throughput: Jellyfish vs. fat tree



Intuition

if we fully utilize all available capacity ...

$$\#\ 1\ \text{Gbps flows} = \frac{\text{total capacity}}{\text{used capacity per flow}}$$

Intuition

if we fully utilize all available capacity ...

$$\# \text{ 1 Gbps flows} = \frac{\sum_{\text{links}} \text{capacity}(\textit{link})}{\text{used capacity per flow}}$$

Intuition

if we fully utilize all available capacity …

$$\#\ 1\ \text{Gbps flows} = \frac{\sum_{\text{links}} \text{capacity}(\textit{link})}{1\ \text{Gbps} \bullet \text{mean path length}}$$

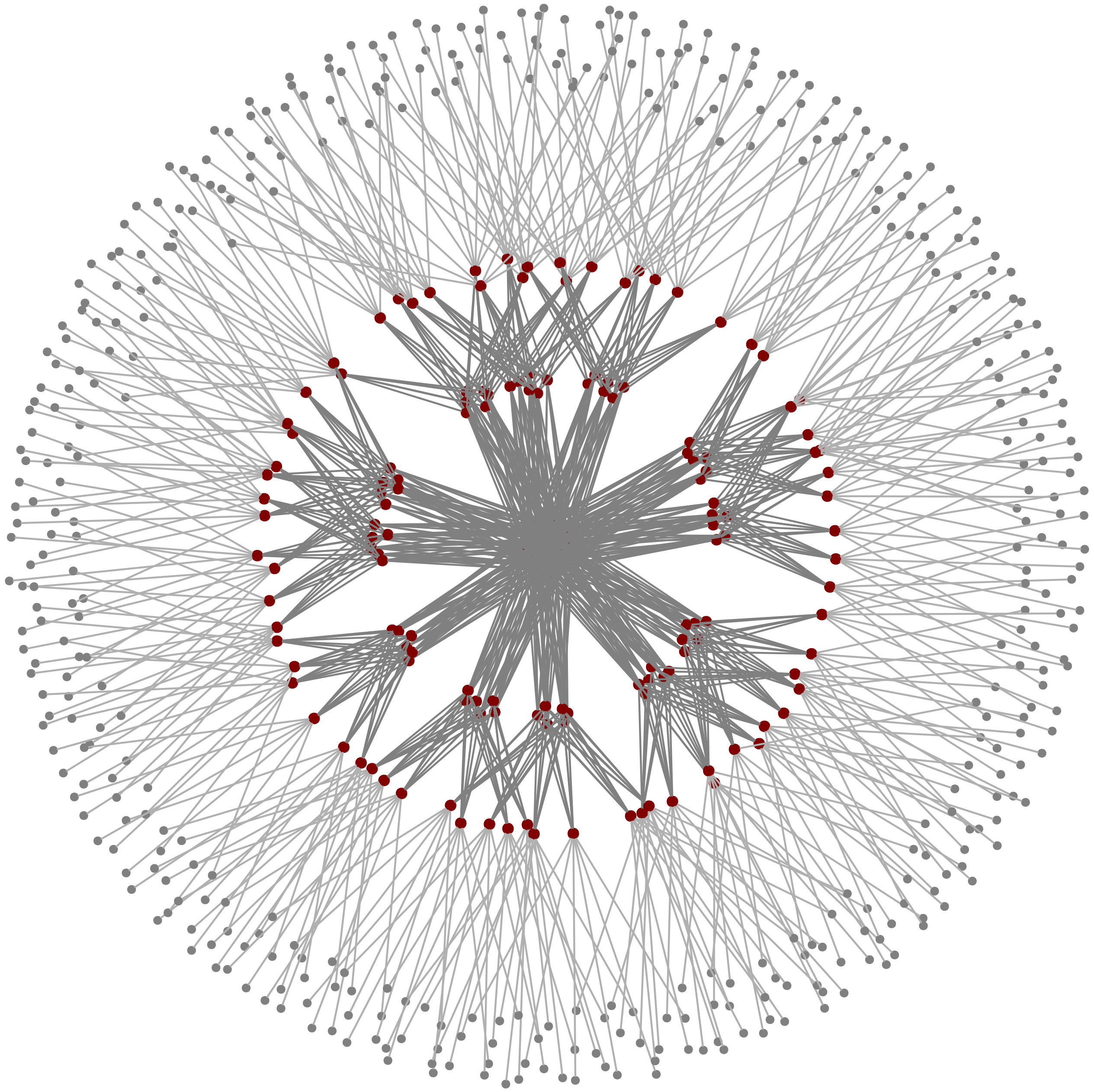
Intuition

if we fully utilize all available capacity ...

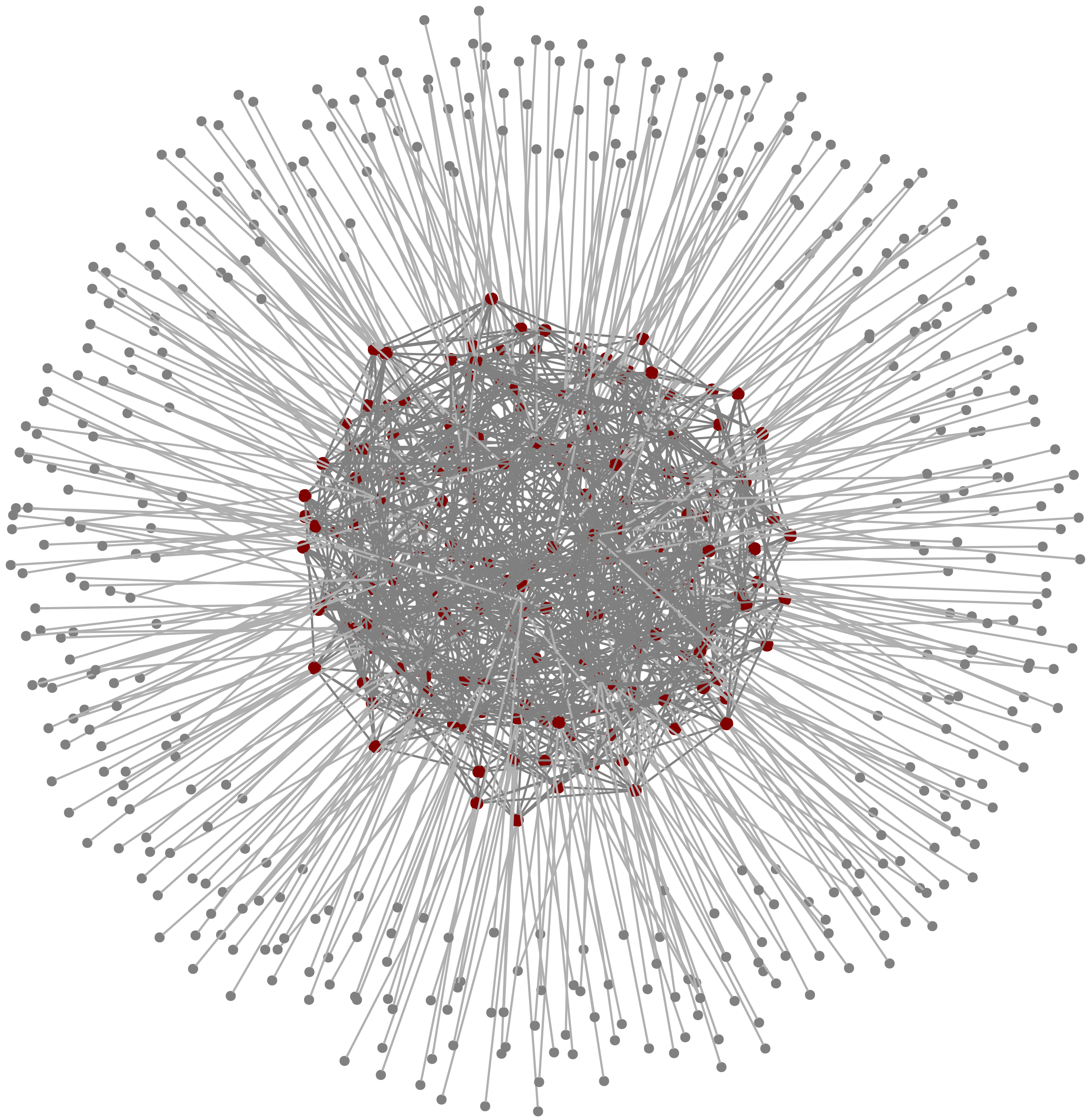
$$\#\ 1\text{ Gbps flows} = \frac{\sum_{\text{links}} \text{capacity}(\textit{link})}{1\text{ Gbps} \cdot \text{mean path length}}$$

Mission:
minimize average path length

Example

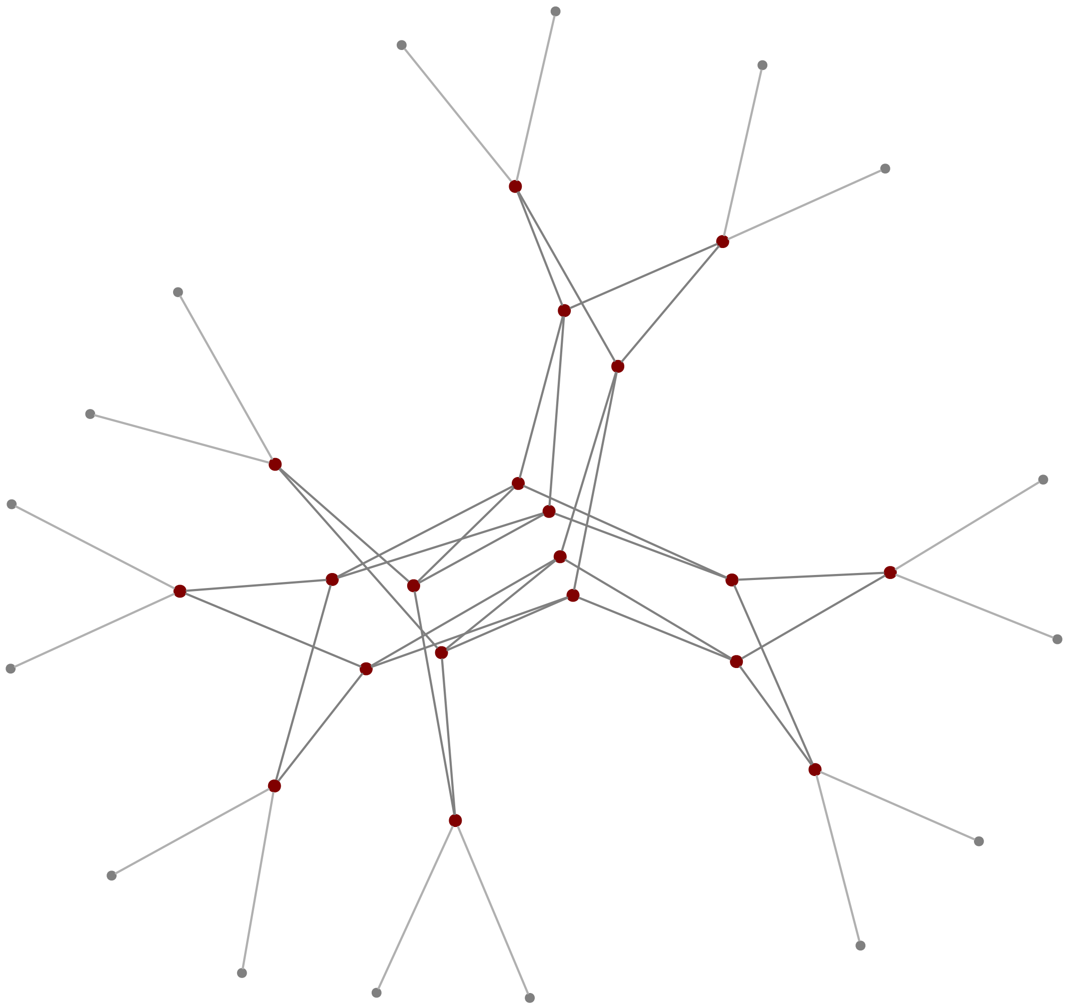


Fat tree
432 servers, 180 switches, degree 12



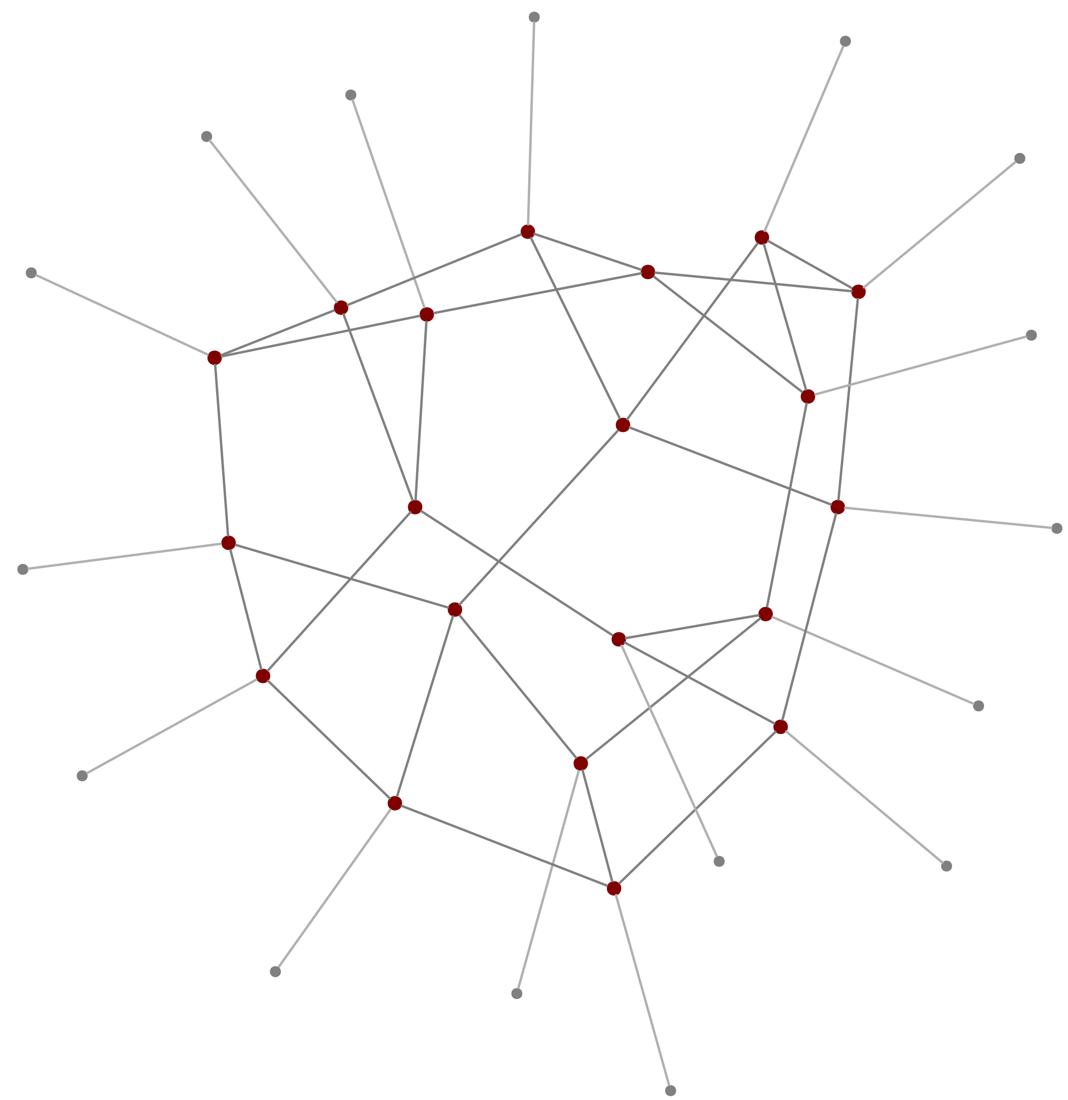
Jellyfish random graph
432 servers, 180 switches, degree 12

Example



Fat tree

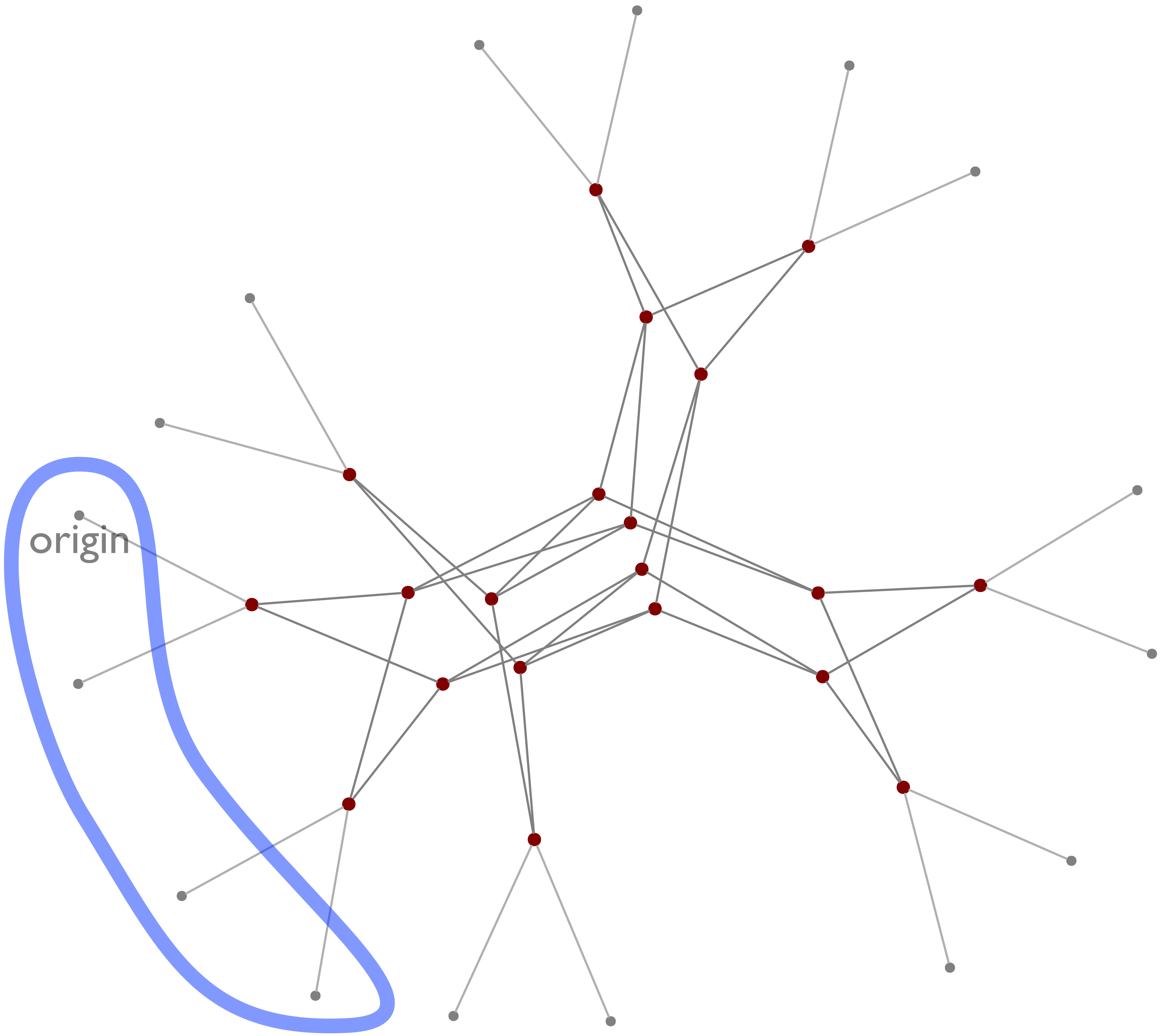
16 servers, 20 switches, degree 4



Jellyfish random graph

16 servers, 20 switches, degree 4

Example: Fat Tree

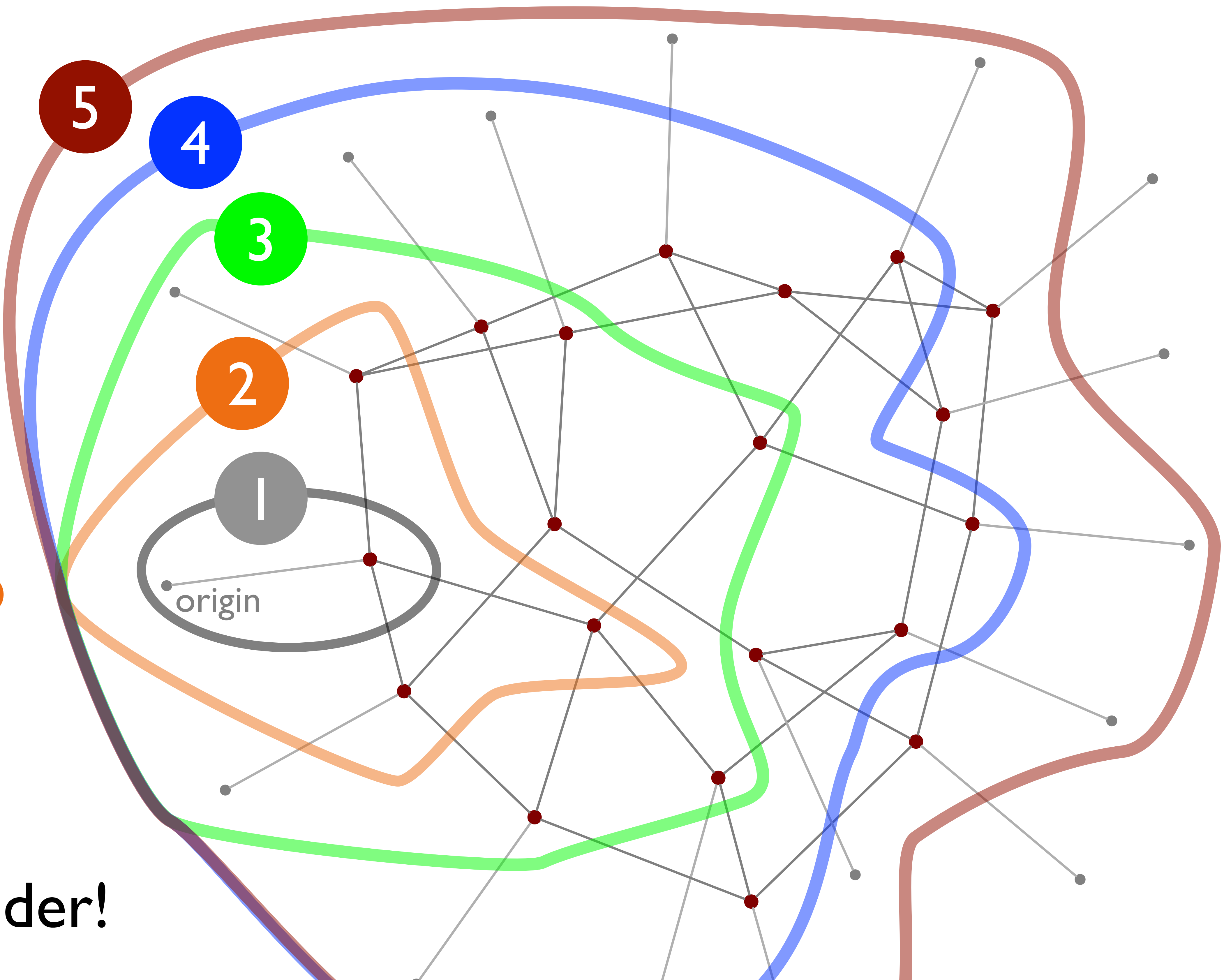


4 of 16

reachable

in < 6 hops

Example: Jellyfish



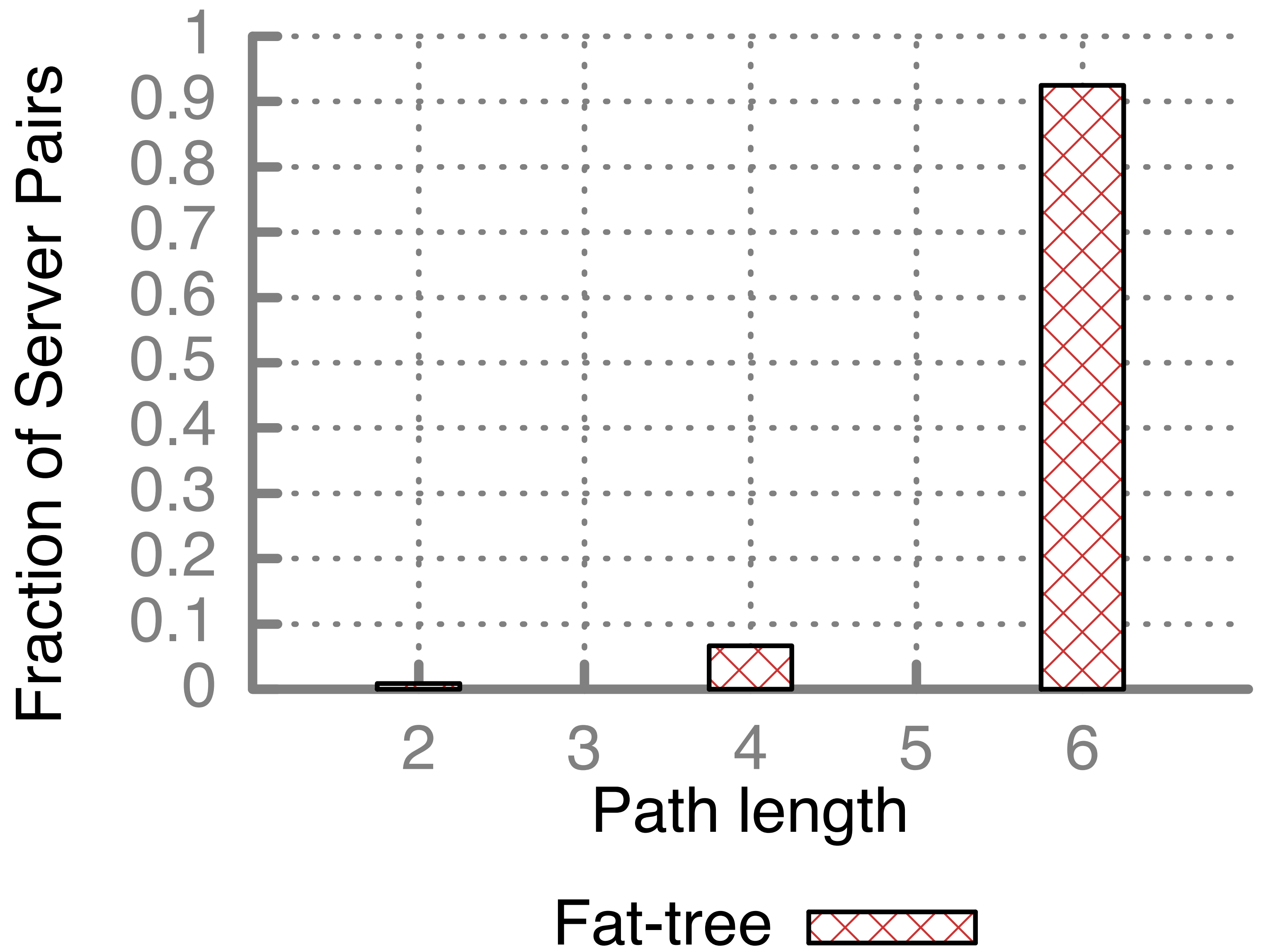
13 of 16

reachable in

< 6 hops

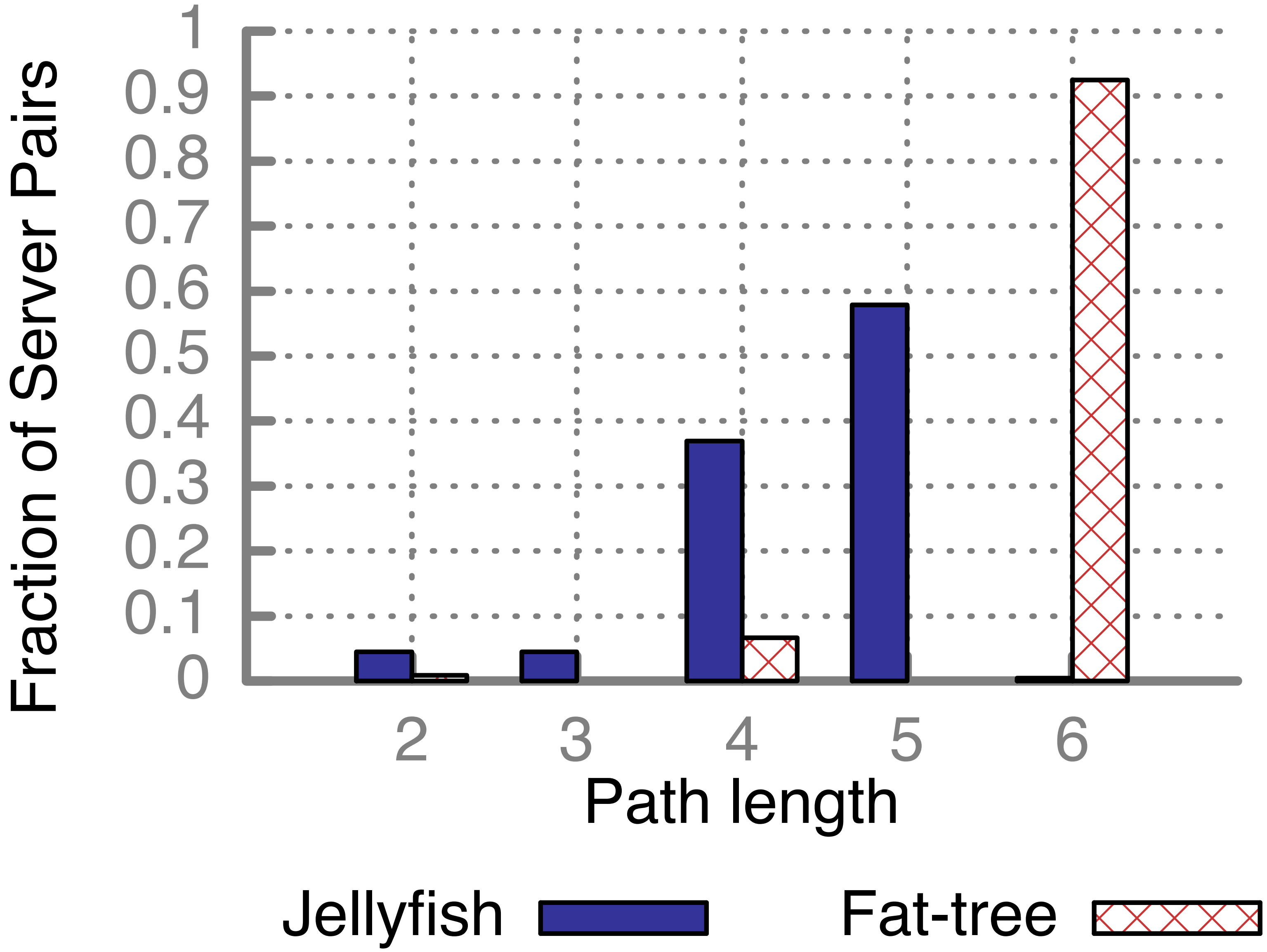
Good expander!

Jellyfish has short paths



Fat-tree with 686 servers

Jellyfish has short paths

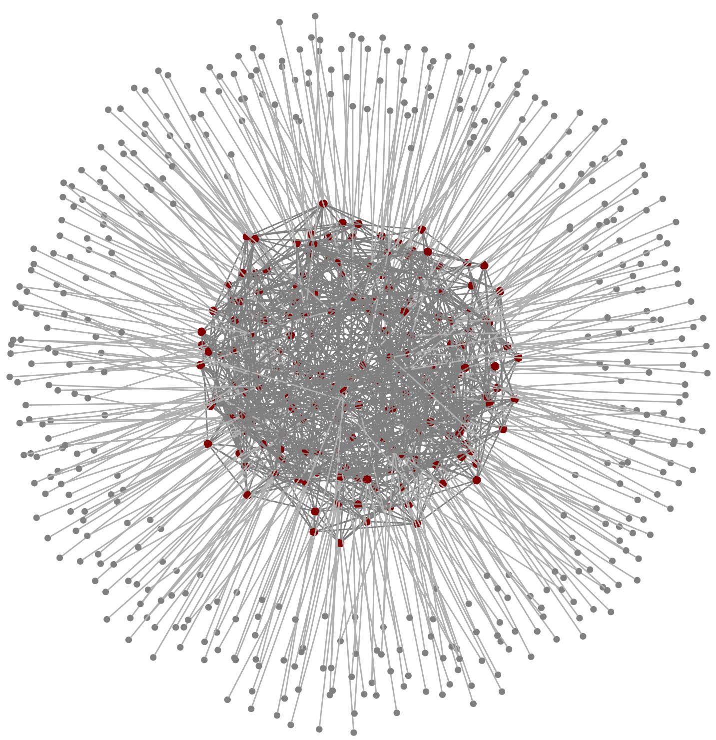


Same-equipment Jellyfish

What I'm not telling you

How do you route in a network without structure?

How do you cable it?



≈

?

Is there anything better than random?

Software Defined Networking

The Problem

Networks are complicated

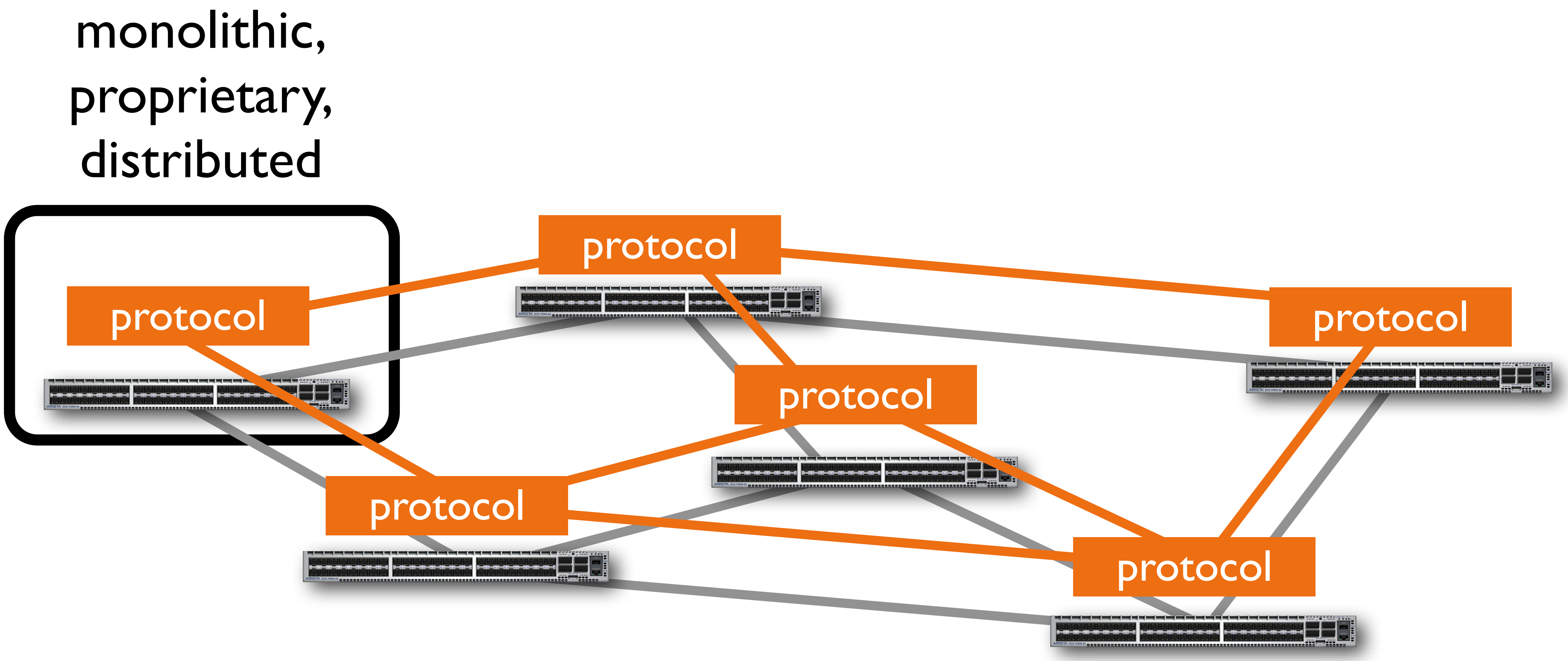
- Just like any computer system
- Worse: it's distributed

Network equipment is proprietary

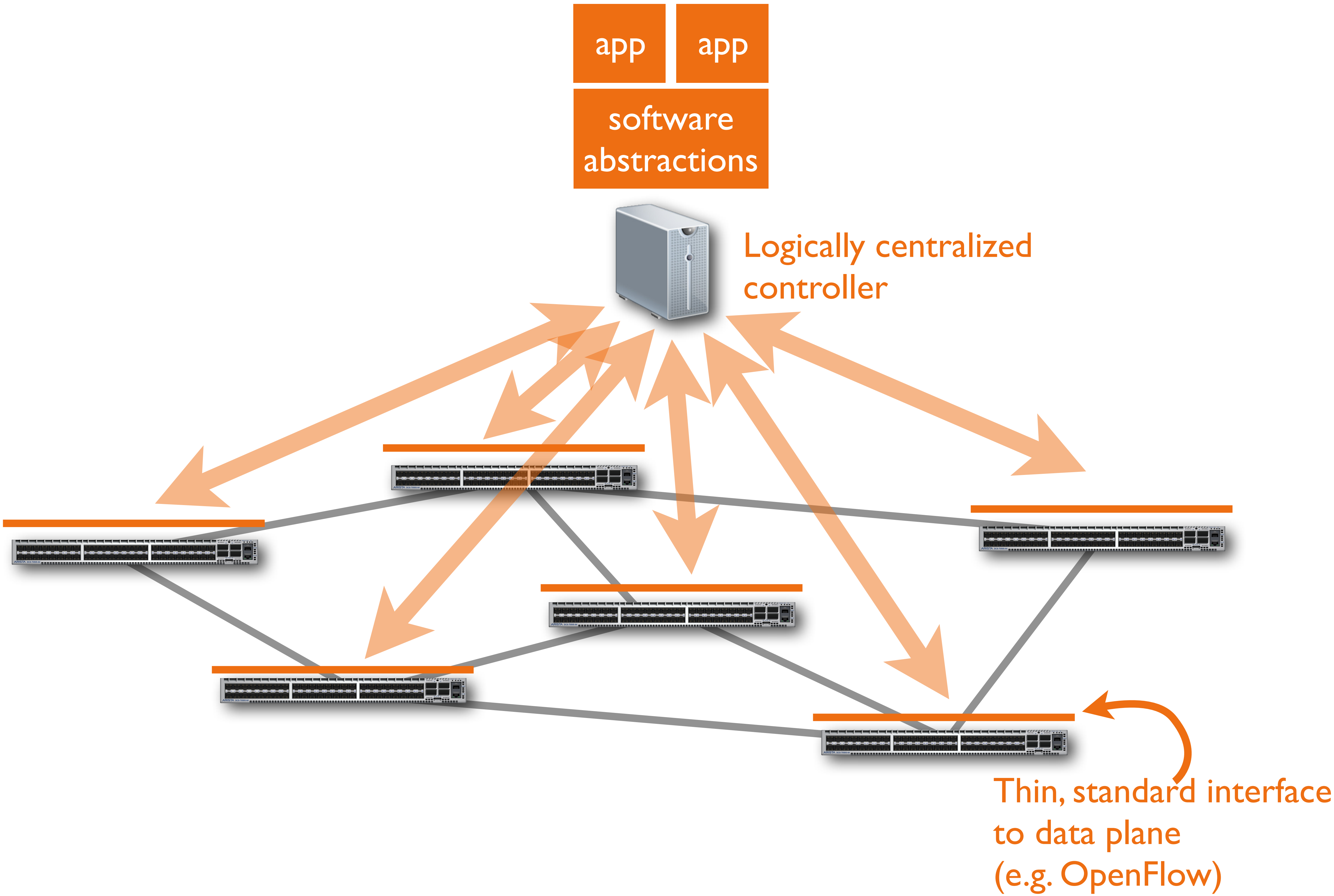
- Integrated solutions (software, configuration, protocol implementations, hardware) from major vendors (Cisco, Juniper)

Result: Hard to innovate and modify networks

Traditional networking



Software defined networking



Making networks programmable

Standard interface to data plane

- Enables innovation in hardware and software

Centralized controller

- Handles state collection and distribution
- Network appears as one big switch

Programming abstractions

- Don’t want to think about each switch
- Like moving from assembly language to Python / Java

All active areas of current research

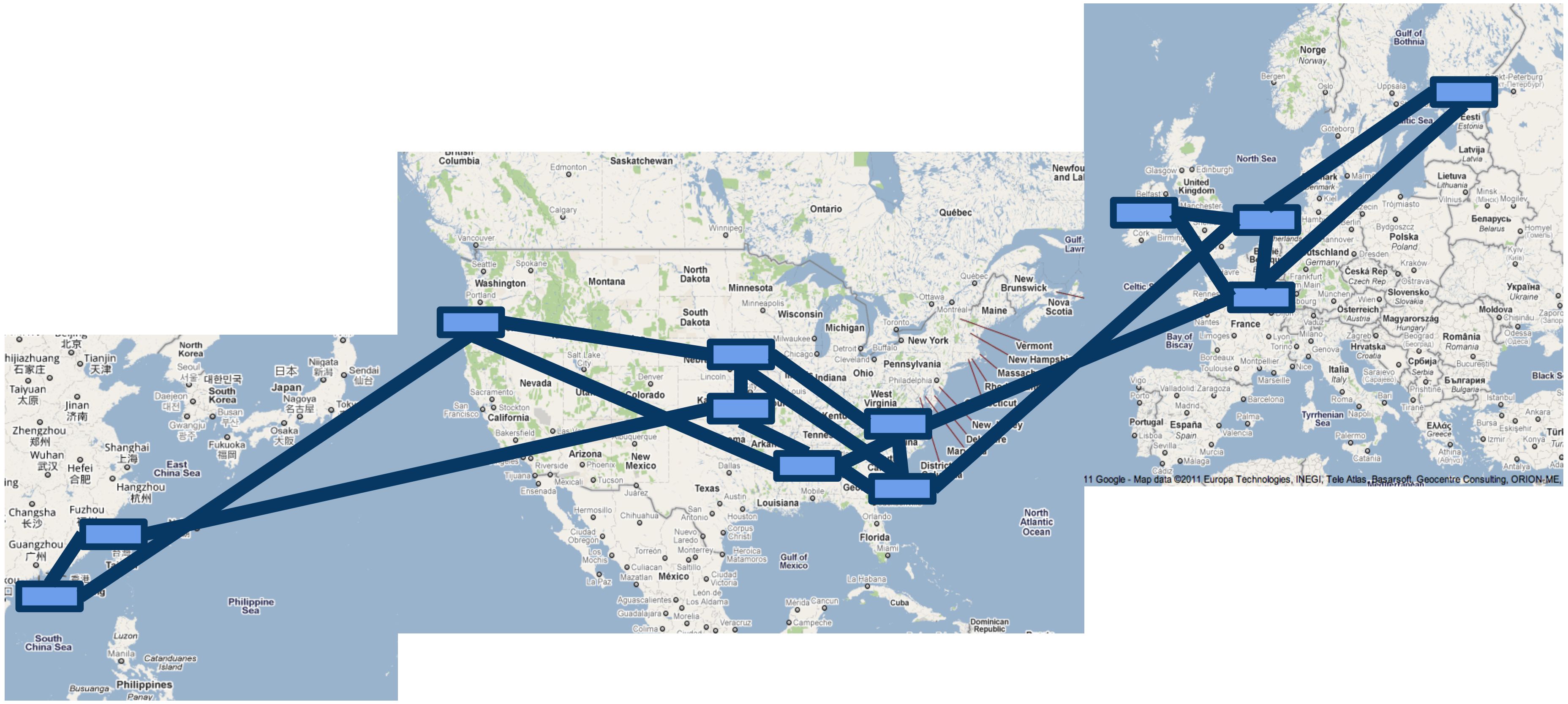
From research to reality

Original papers: 2007, 2008

Now:

- Offerings from major vendors and startups (NEC, IBM, Nicira, ...)
- Deployment in production networks

SDN Deployment at Google, 2012



Advantages

- Faster reaction to dynamic environment
- Fine-grained control of traffic
    - High priority / low priority
- Test before deploying
    - Run real new software on top of simulated hardware
    - Only need to simulate the thin interface (OpenFlow)

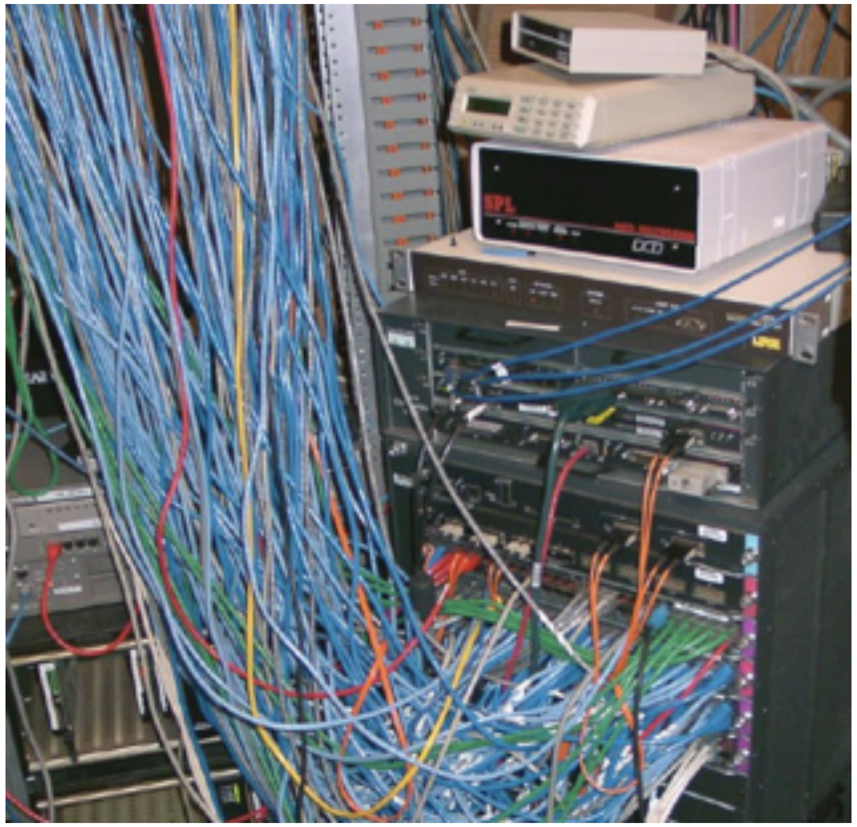
Our work: debugging the data plane

[Work with Ahmed Khurshid, Haohui Mai, Wenxuan Zhou, Rachit Agarwal, Matthew Caesar, and Sam King]

Network debugging is challenging

Production networks are complex

- Security policies
- Traffic engineering
- Legacy devices
- Protocol inter-dependencies
- …



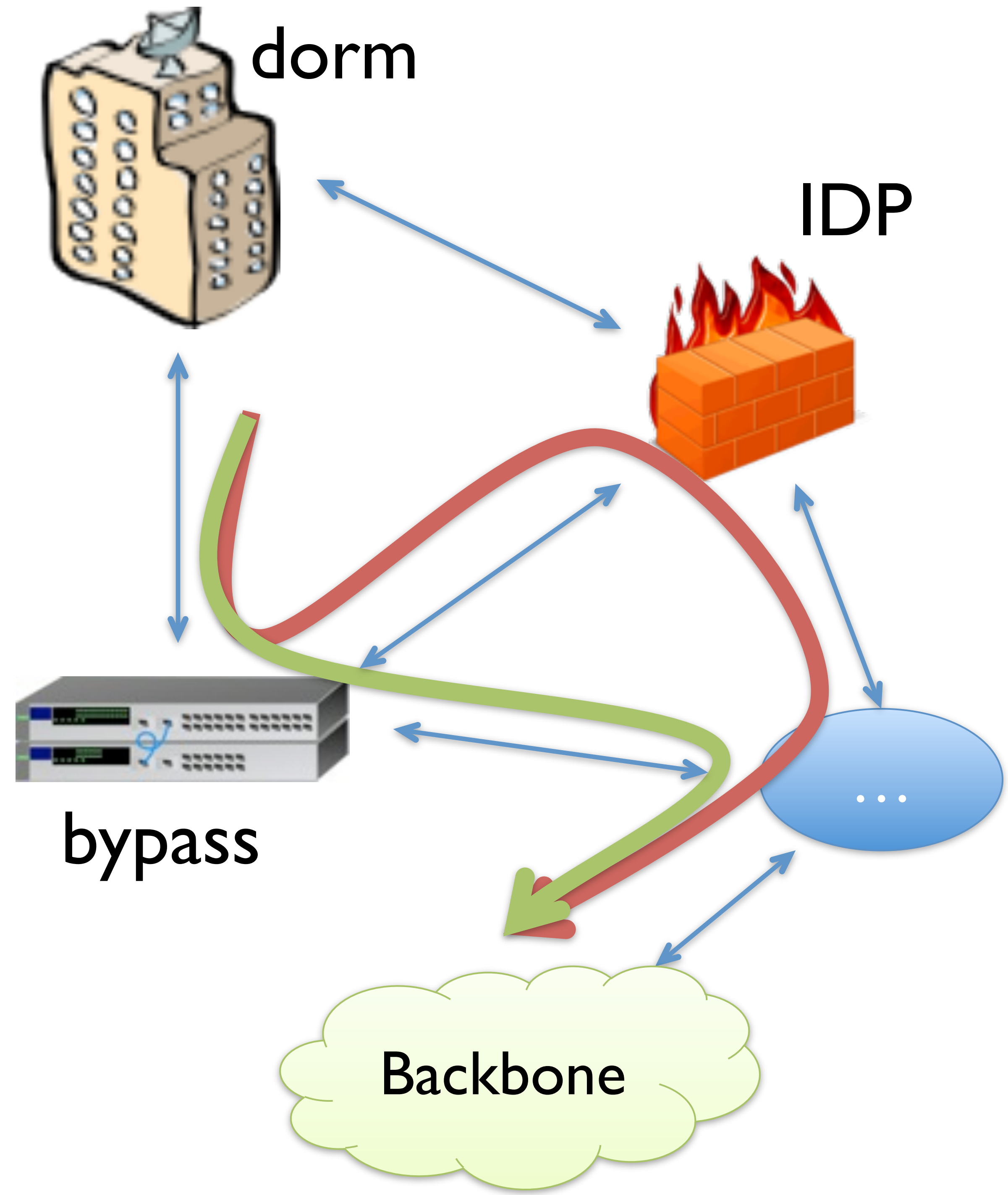
- Even well-managed networks can go down
- Few good tools to ensure all networking components working together correctly

A real example from UIUC's network

Previously, an intrusion detection and prevention (IDP) device inspected all traffic to/from dorms

IDP couldn't handle load; added bypass

- IDP only inspected traffic between dorm and campus
- Seemingly simple changes

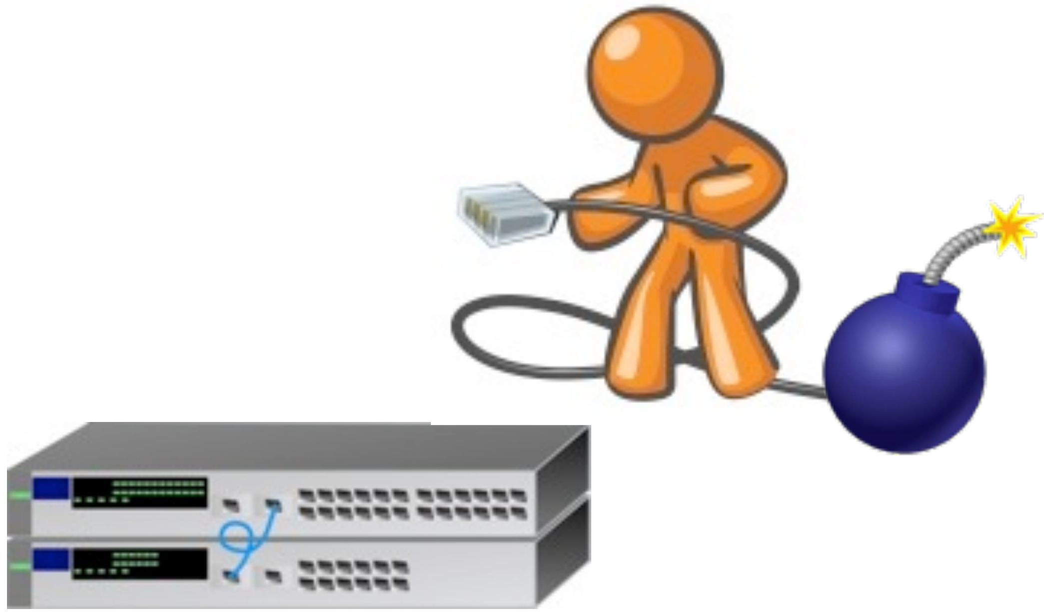


Challenge: Did it work correctly?

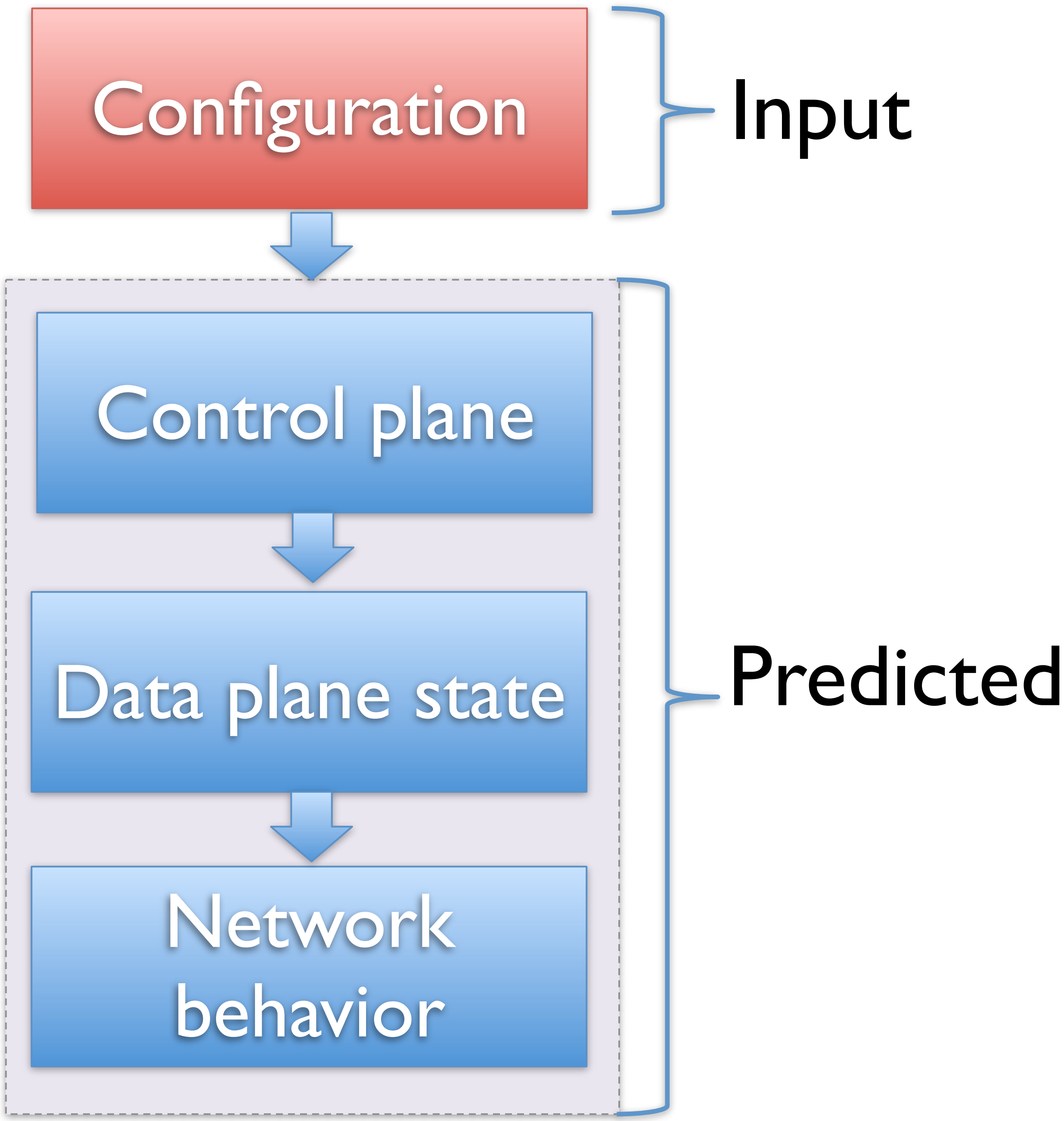
Ping and traceroute provide limited testing of exponentially large space

- 2^{32} destination IPs * 2^{16} destination ports * …

Bugs not triggered during testing might plague the system in production runs



Previous approach: Configuration analysis



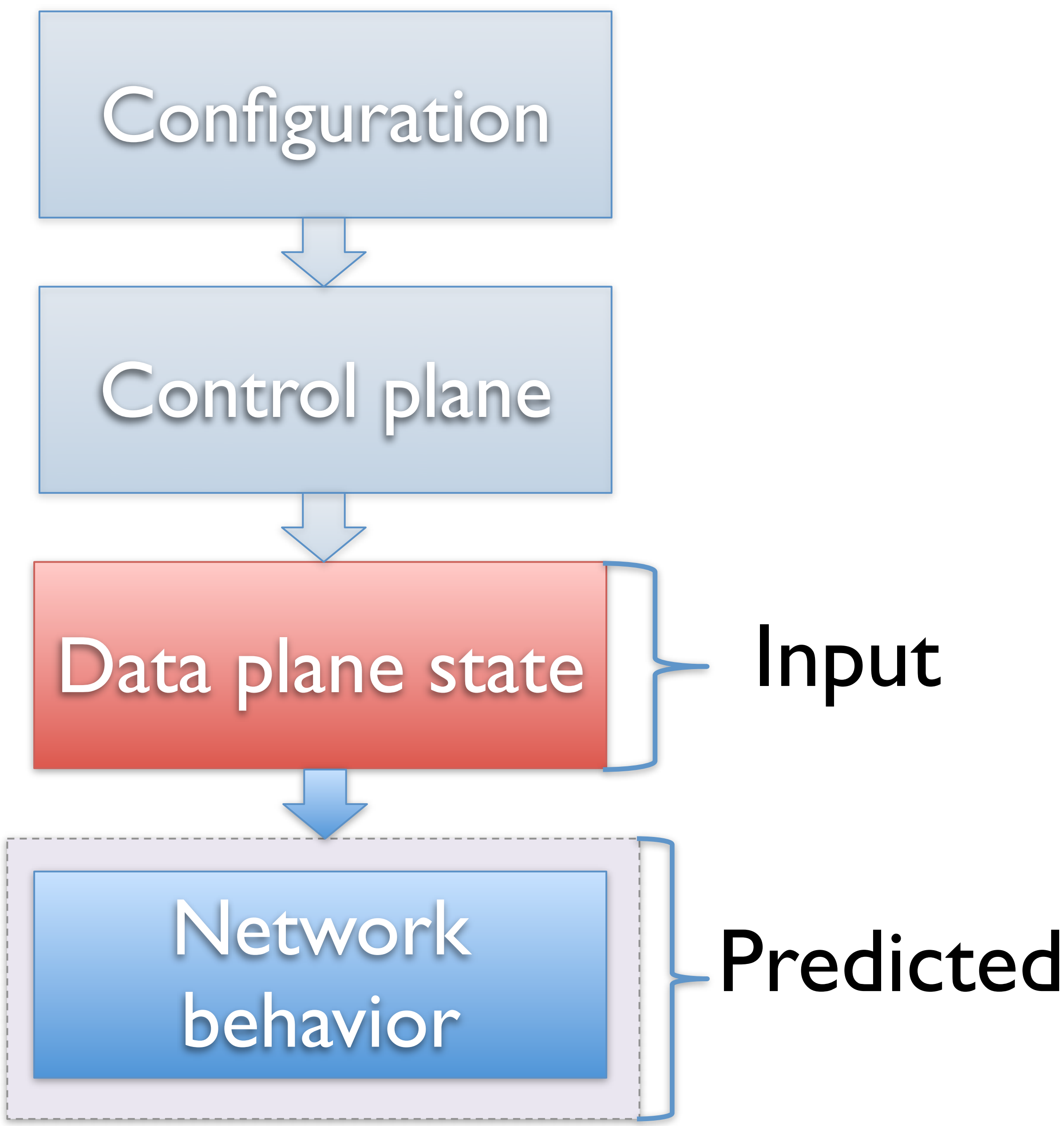
+ Test before deployment

– Prediction is difficult
  - Various configuration languages
  - Dynamic distributed protocols

– Prediction misses implementation bugs in control plane

Our approach: Debugging the data plane

diagnose problems as close as possible to actual network behavior



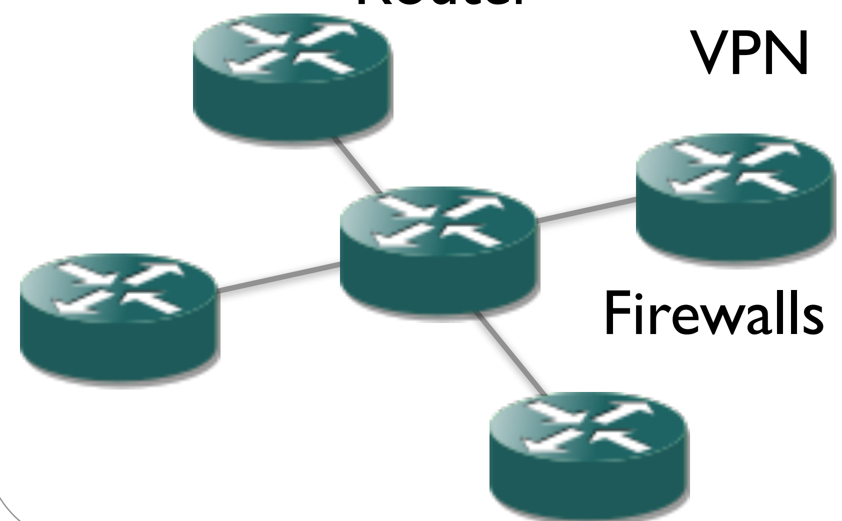
+ Less prediction
+ Data plane is a "narrower waist" than configuration
  + Unified analysis for multiple control plane protocols
+ Can catch implementation bugs in control plane

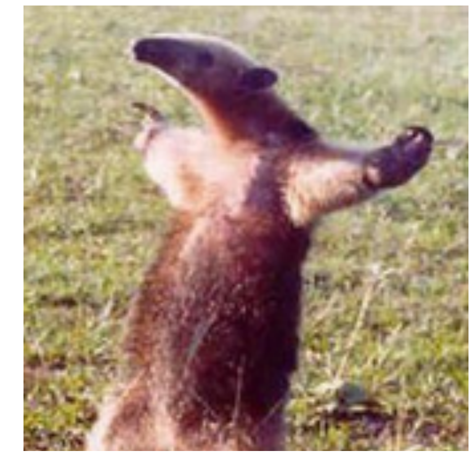
– Checks one snapshot

Anteater from 30,000 feet

Operator

Anteater





Data plane state

Invariants

∃Loops?

∃Security policy violation?

...

SAT formulas

Results of SAT solving

Diagnosis report

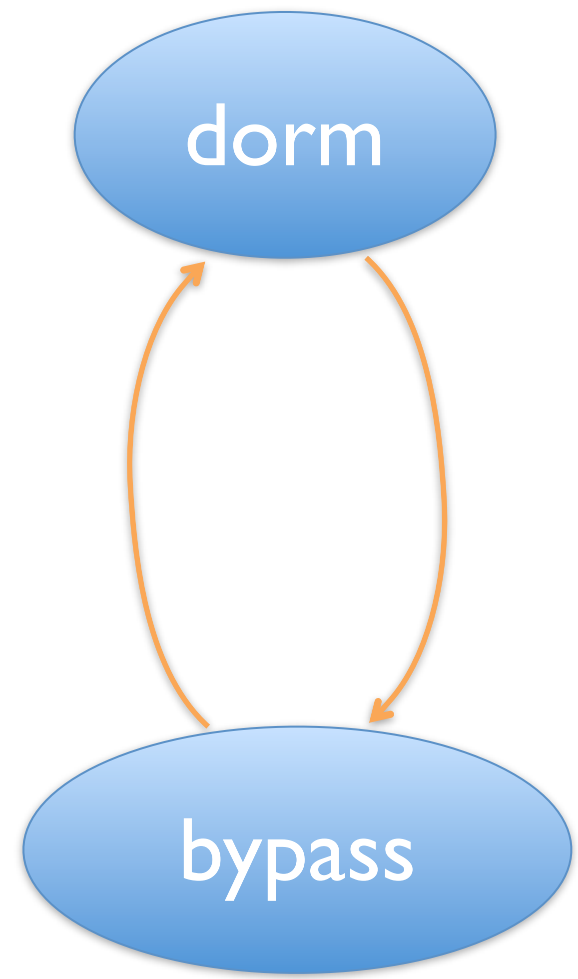
Experiences with UIUC network

- Evaluated Anteater with UIUC campus network
  - ～178 routers
  - Predominantly OSPF, also uses BGP and static routing
  - 1,627 FIB entries per router (mean)
  - State collected using operator's SNMP scripts
- Revealed 23 bugs with 3 invariants in 2 hours

	Loop	Packet loss	Consistency
Being fixed	9	0	0
Stale config.	0	13	1
False pos.	0	4	1
Total alerts	9	17	2

Forwarding loops

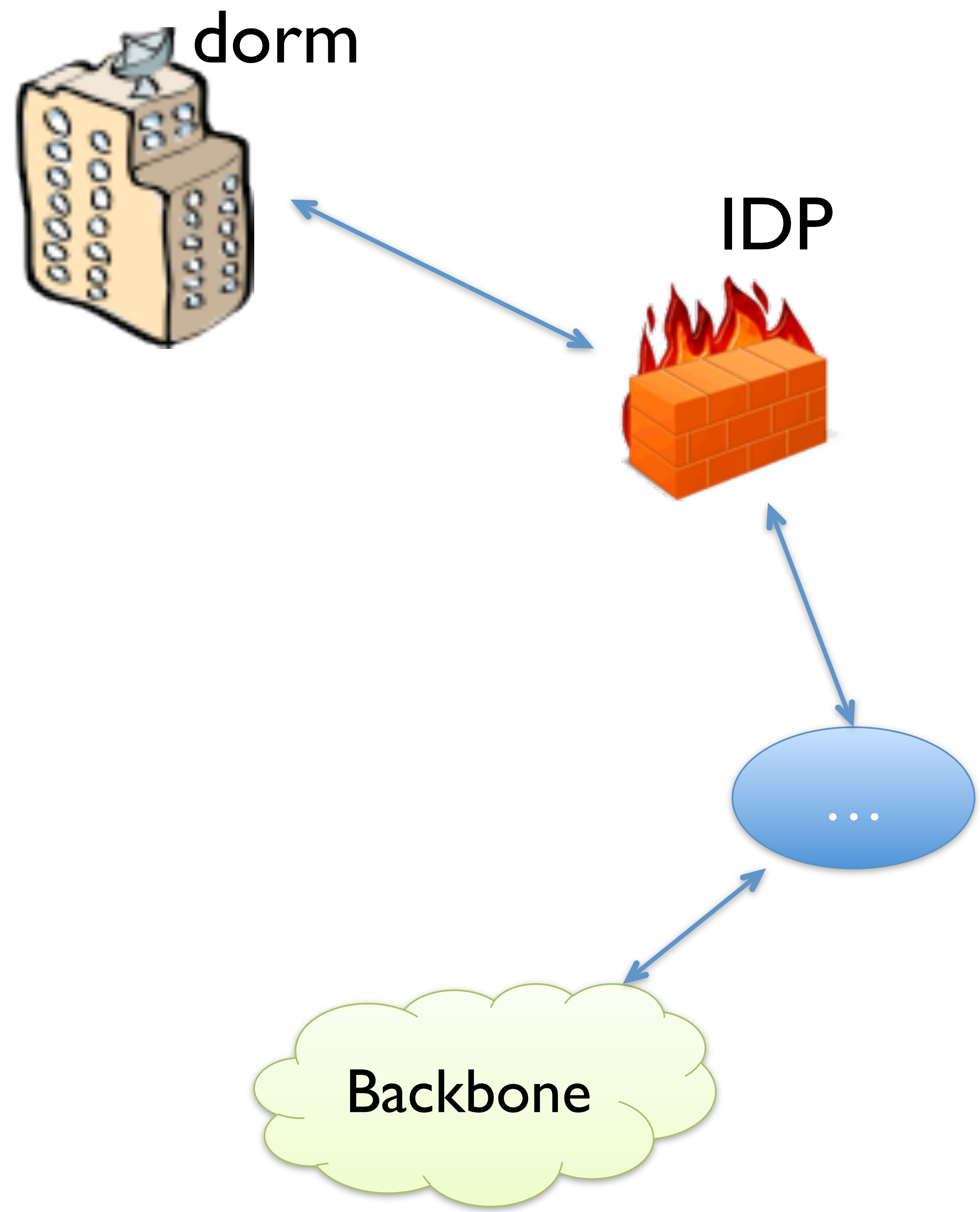
- 9 loops between router dorm and bypass
- Existed for more than a month
- Anteater gives one concrete example of forwarding loop
  - Given this example, relatively easy for operators to fix




```
$ anteater
Loop:
128.163.250.30@bypass
```

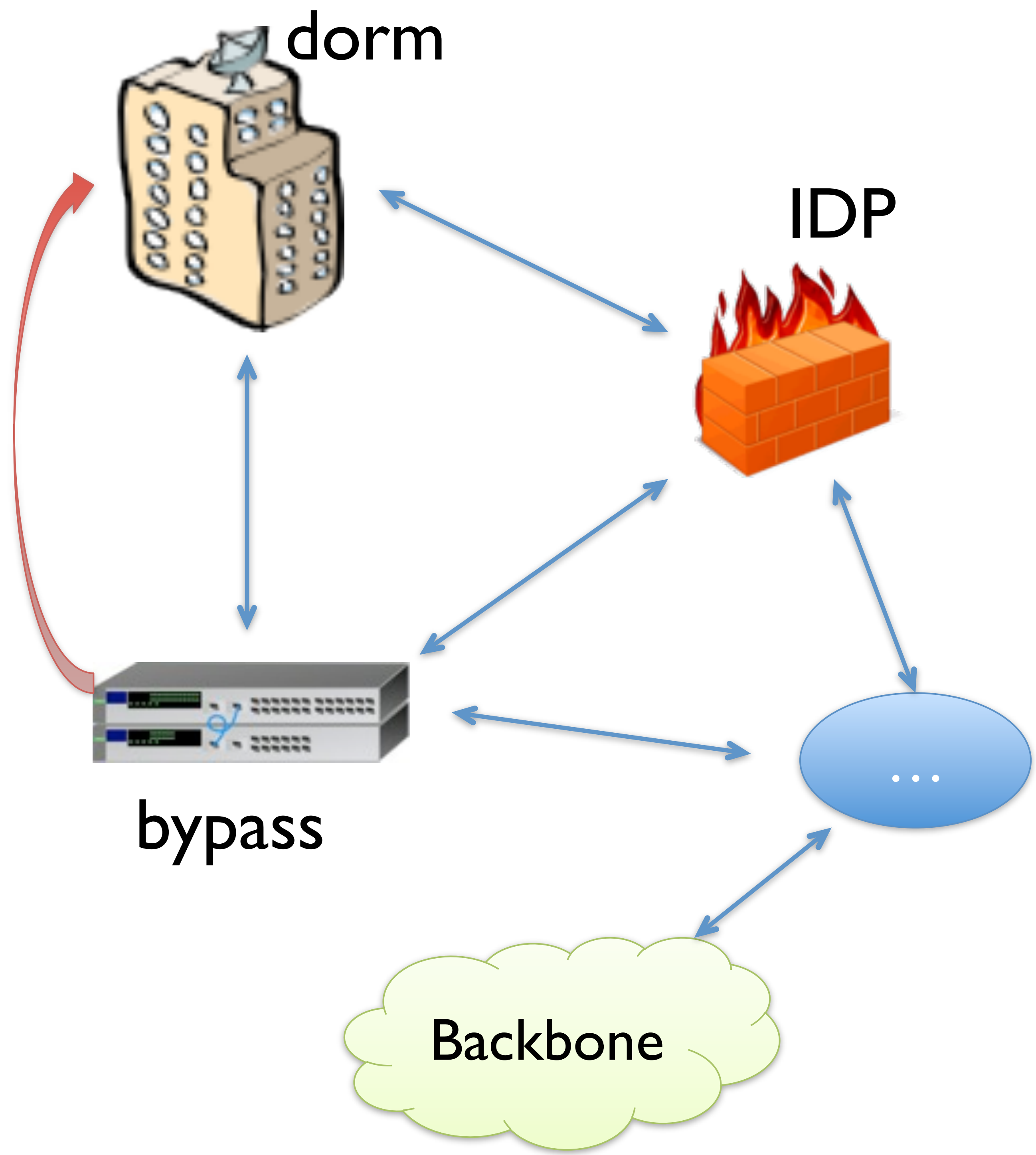
Forwarding loops

- Previously, dorm connected to IDP directly
- IDP inspected all traffic to/from dorms



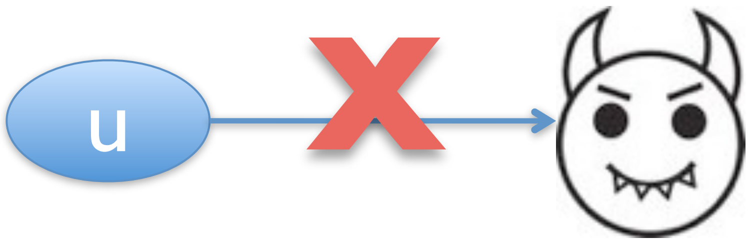
Forwarding loops

- IDP was overloaded, operator introduced bypass
  - IDP only inspected traffic for campus
- bypass routed campus traffic to IDP through static routes
- Introduced loops



Bugs found by other invariants

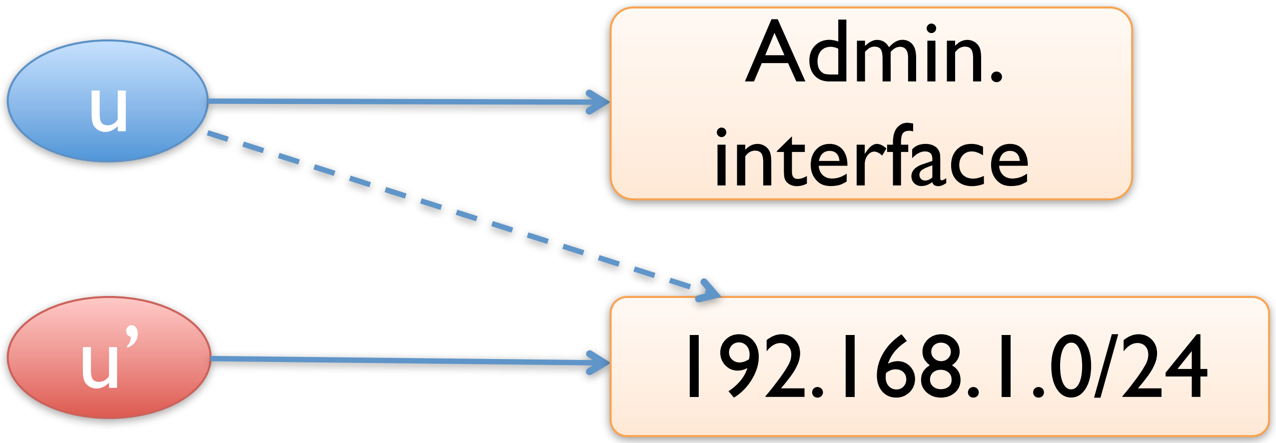
Packet loss



- Blocking compromised machines at IP level
- Stale configuration

From Sep, 2008

Consistency



- One router exposed web admin interface in FIB
- Different policy on private IP address range

Maintaining compatibility

VeriFlow

Goal: Verify network-wide invariants in real time

- Verify correctness continually as network state changes
- ~ one millisecond or less per verification
- Can provide immediate warning, or block dangerous modifications

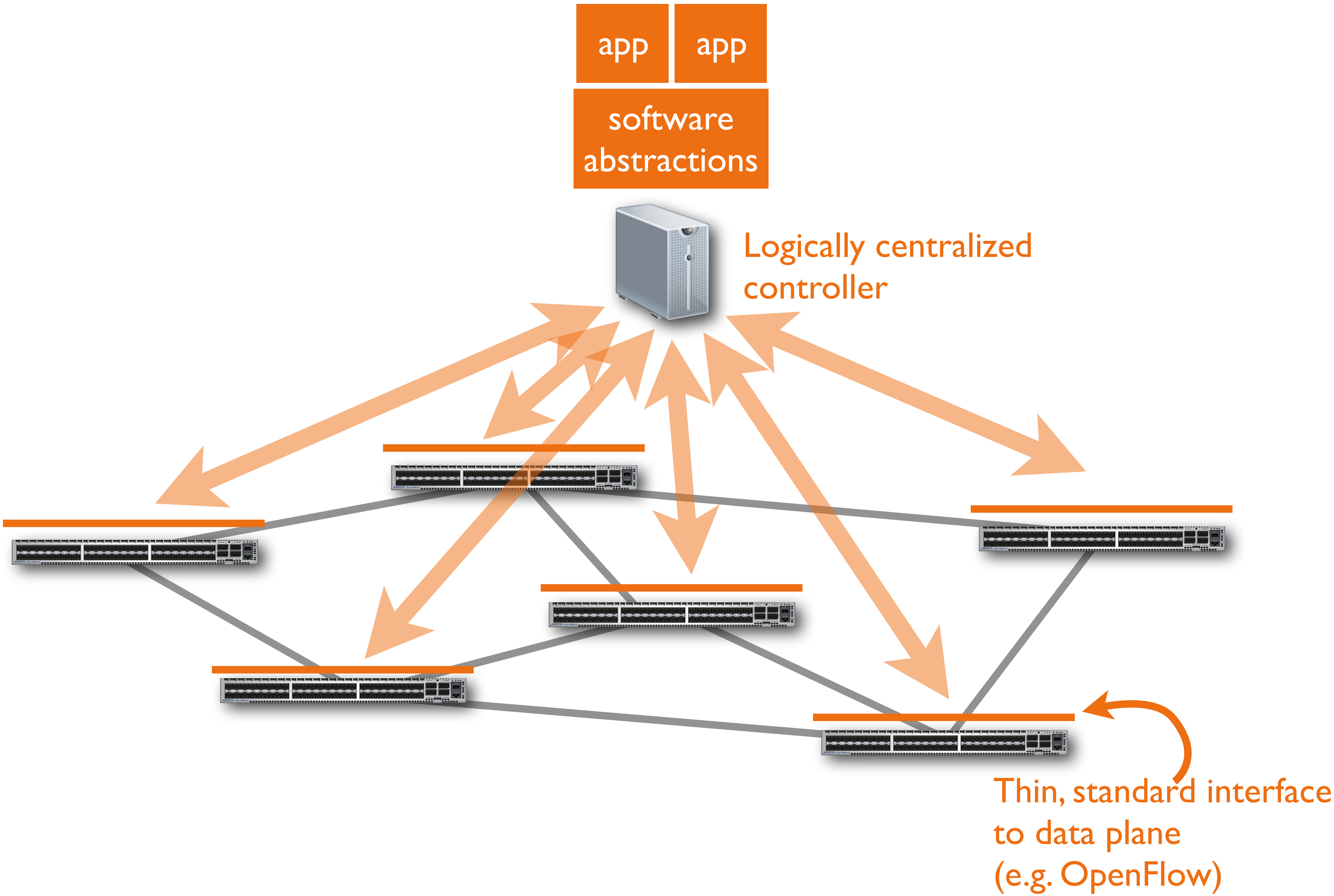
Challenge #1: Obtaining real time view of network

- Solution: interpose between Software Defined Networking (SDN) controller and routers/switches

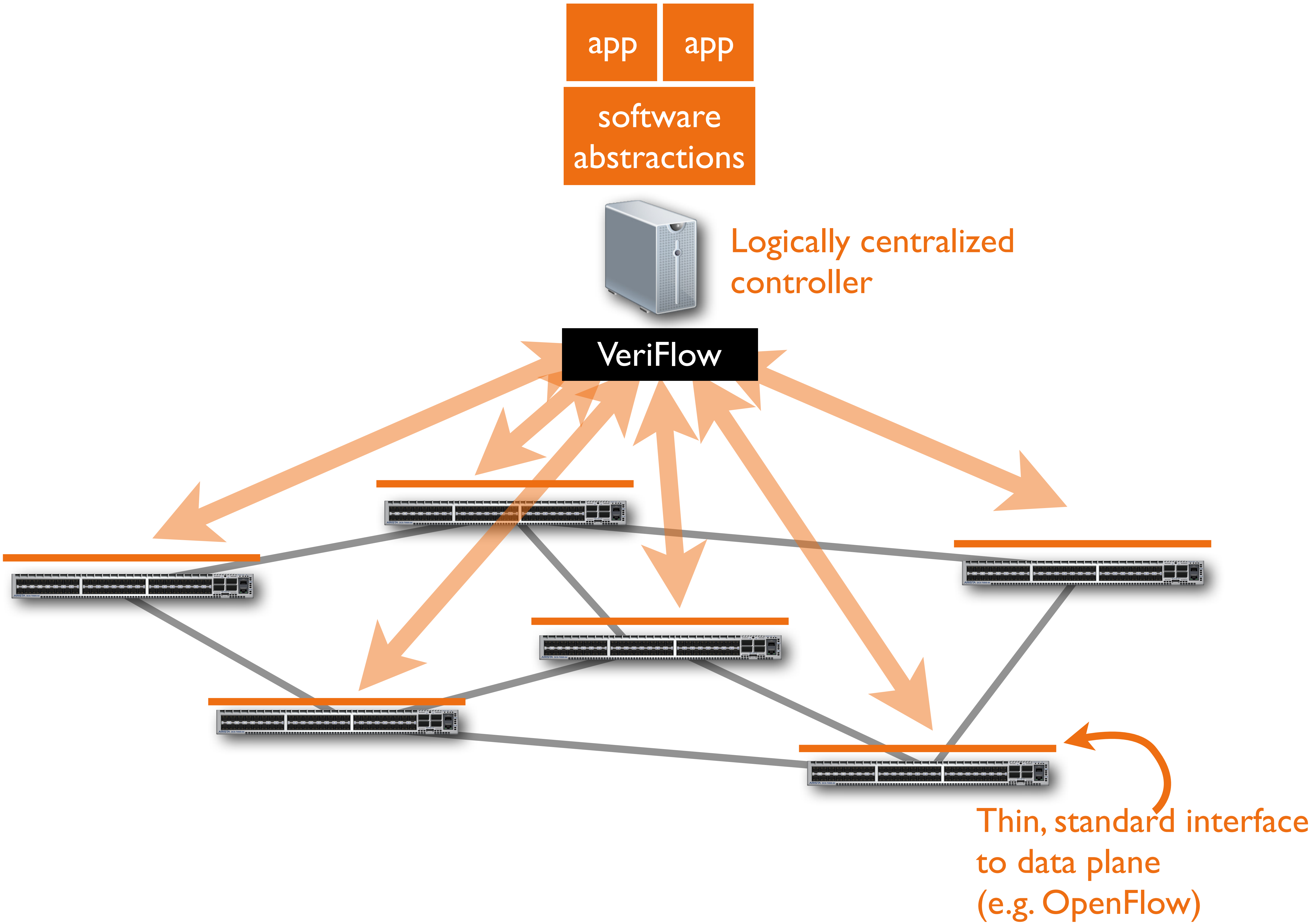
Challenge #2: Verification speed

- Solution: Algorithms :-)

Software defined networking

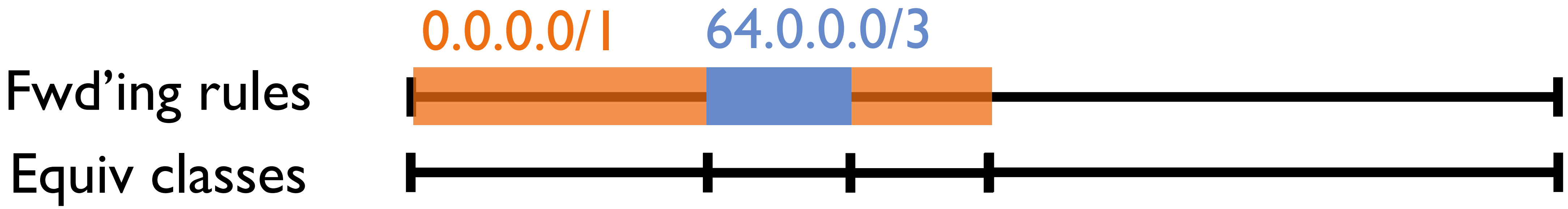


VeriFlow architecture

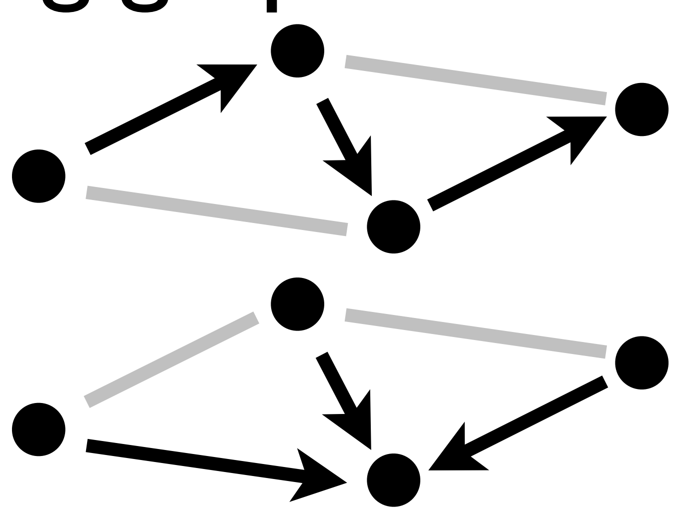


Checking in real time

Split possible packet headers into equivalence classes



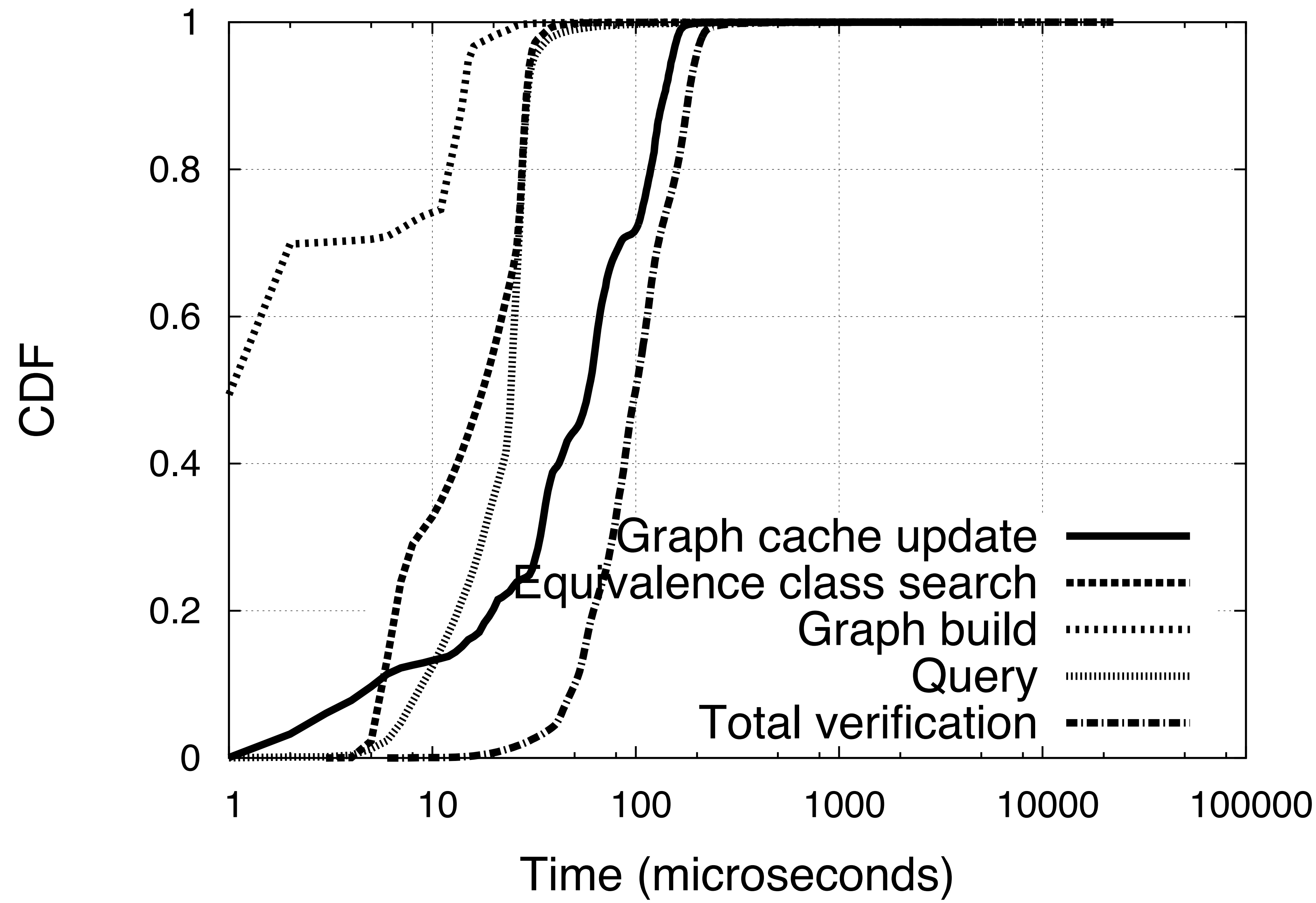
Construct forwarding graph for each class



On rule insert/delete:

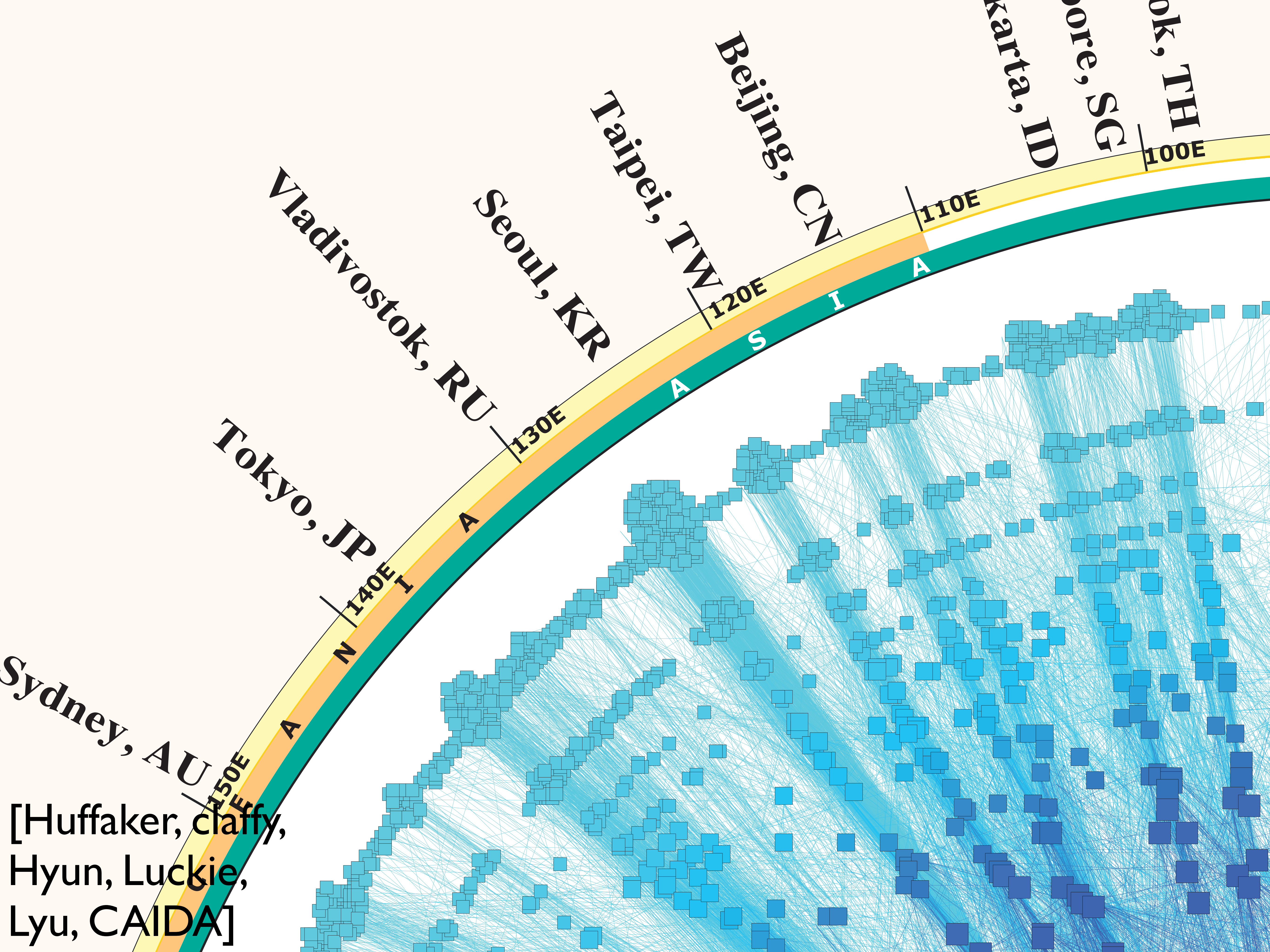
- Update equivalence classes
- For modified classes, update graphs & verify invariants

Results: Verification time



Simulated network: BGP DFZ RIBs and update trace from RouteViews injected into 172-router AS 1755 topology, checking reachability invariant

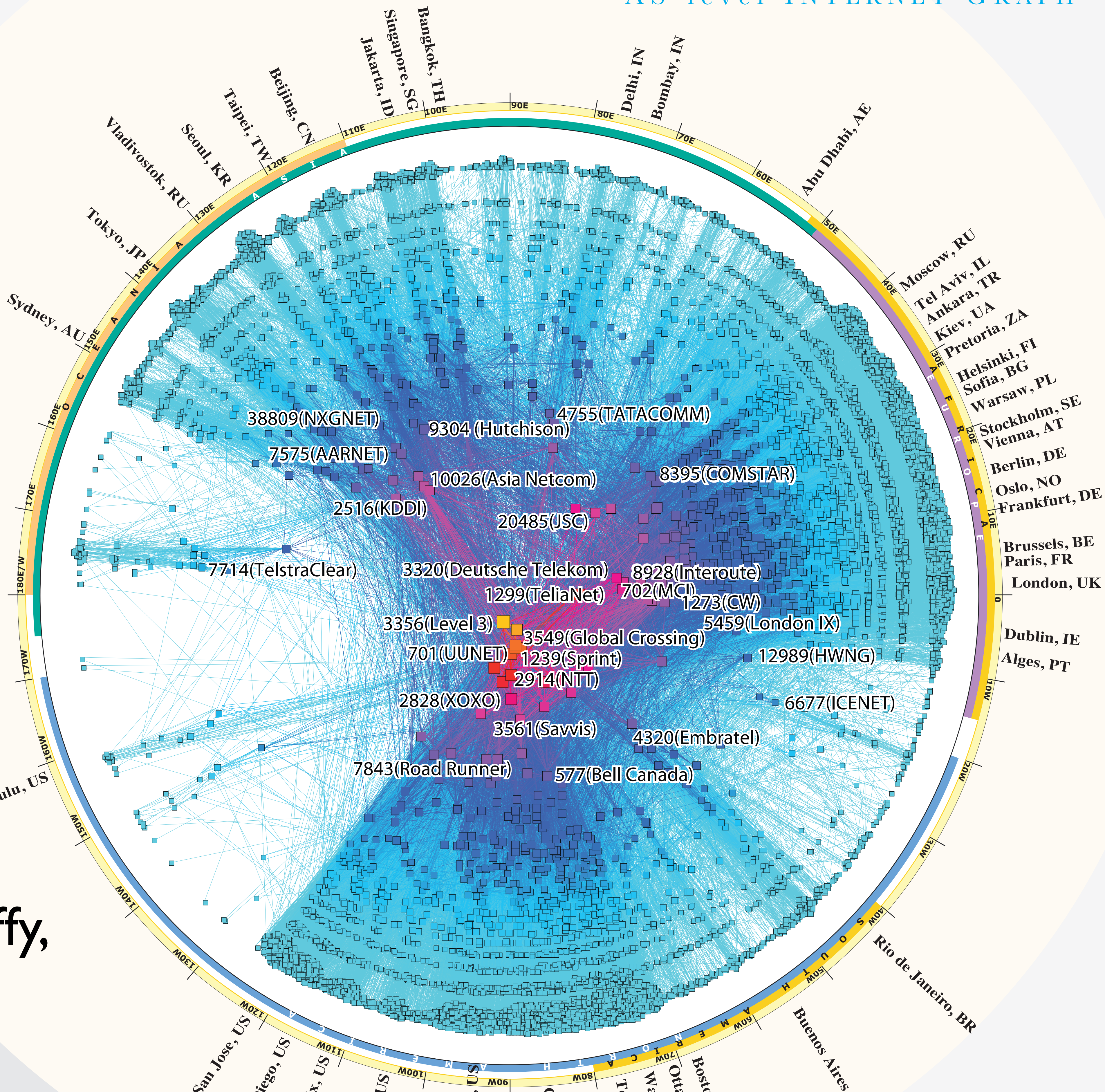
Closing Thoughts:
Why Networking Research?



[Huffaker, claffy, Hyun, Luckie, Lyu, CAIDA]

IPv4 & IPv6 INTERNET TOPOLOGY MAP JANUARY 2009

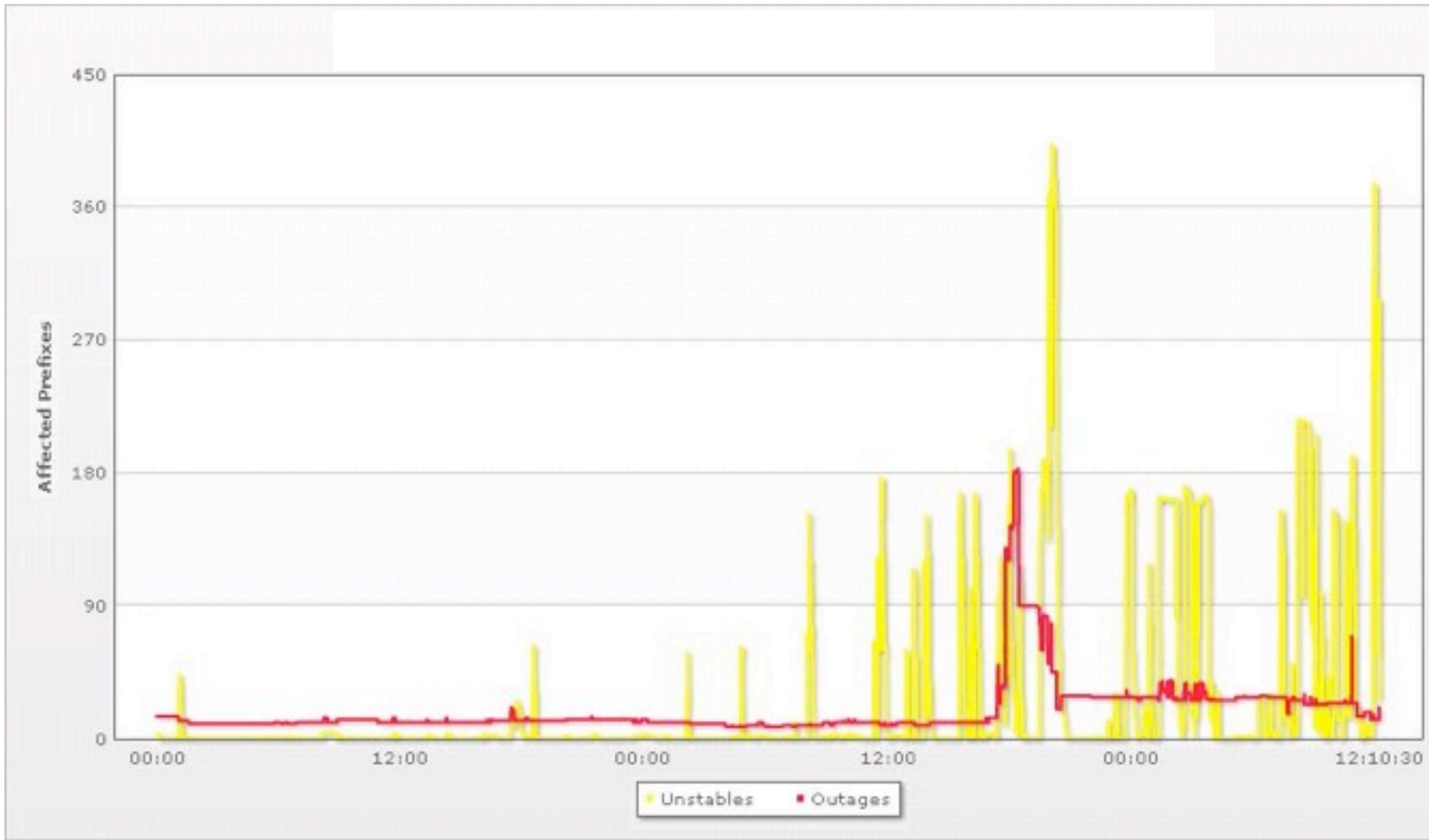
AS-level INTERNET GRAPH



[Huffaker, claffy, Hyun, Luckie, Lyu, CAIDA]

Routing instabilities and outages in Iranian prefixes following 2009 presidential election

Affected prefixes



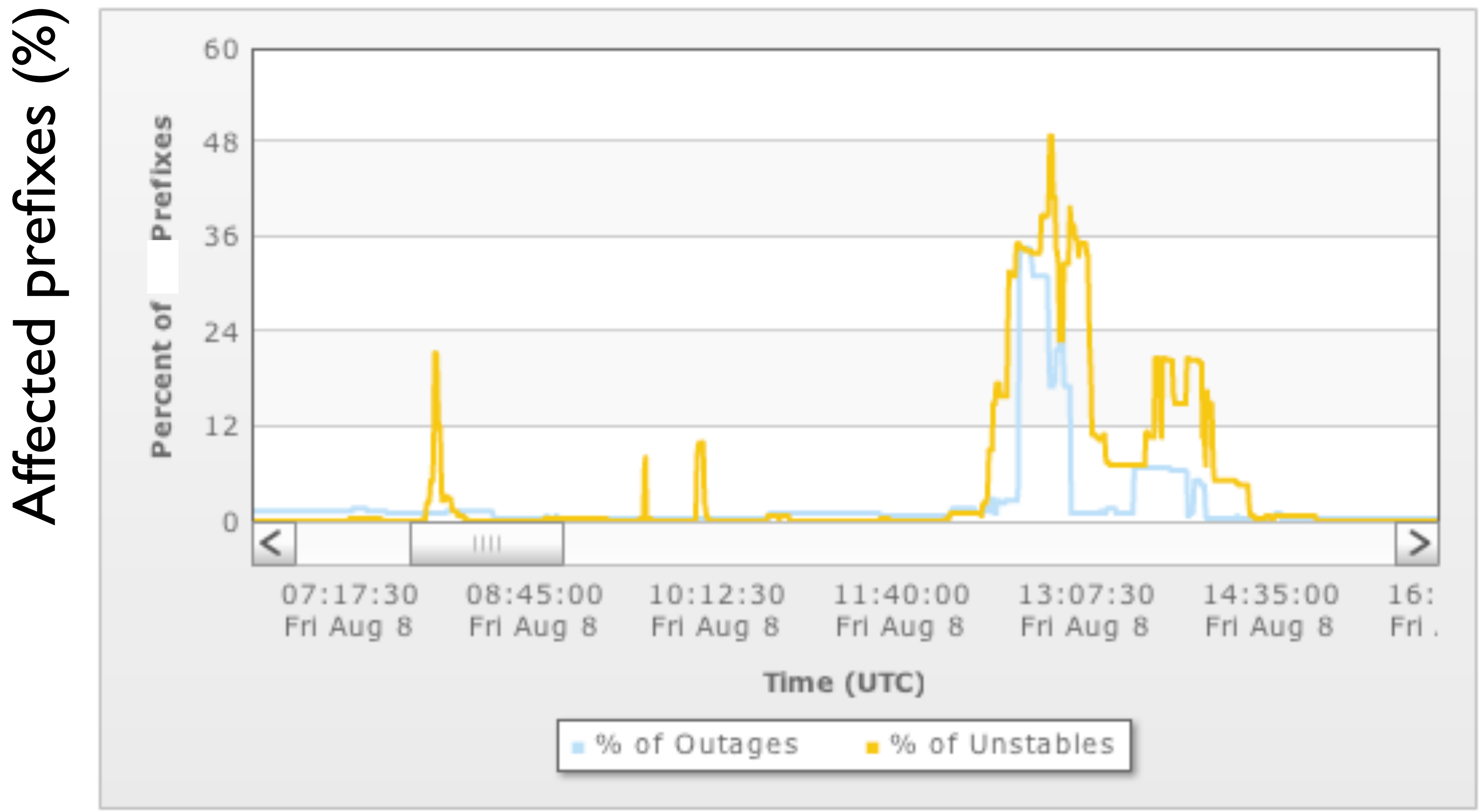
Friday June 12

Saturday June 13

Sunday June 14

[James Cowie, Renesys Corporation]

Routing instabilities and outages in Georgian prefixes following 2008 South Ossetia War



Fri, Aug 8, 2008

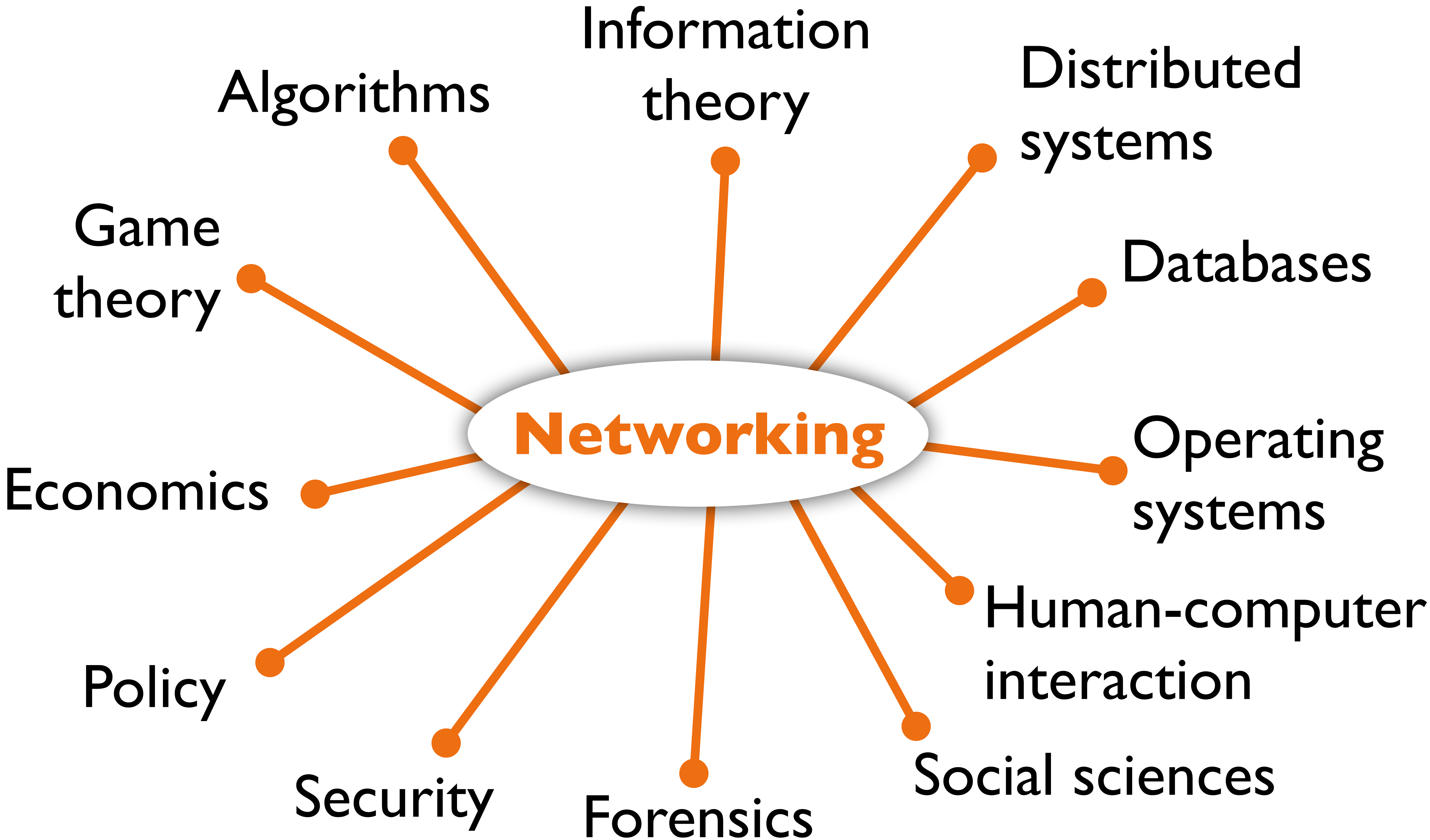
[Earl Zmijewski, Renesys Corporation]

1. It's relevant

Majority of new developments in computer systems are dependent on networking

Far-reaching impacts beyond systems & networking

1. It's relevant



2. It's new

~35 years since the birth of the field

But only ~15 years since networks in widespread use

- tussles between businesses, peer-to-peer systems, malware, denial of service attacks, content distribution networks, all fundamental but relatively new!

Operating systems: ~30 years in widespread use

Physics: ~13.75 billion years in widespread use

3. It's changing

Network new people, new technologies, connect disciplines, "make order out of chaos" (– Jen Rexford)

Start a new subfield!

- In the last decade: Peer-to-peer, sensor networks, data centers, cloud, energy, Internet architecture, cell, ...
- A new subfield every ~2 years – rapid change!

You can change not just the technology, but the field!

3. It's changing

About 2/3 of the world not yet online!

> *It is anticipated that the whole of the populous parts of the United States will, within two or three years, be covered with net-work like a spider's web.*

— *The London Anecdotes*, 1848

What it all adds up to...

You have the opportunity for big impact!