



Advanced Networked Systems (SS24)

Introduction

Prof. Lin Wang, Ph.D.

Computer Networks Group

Paderborn University

<https://en.cs.uni-paderborn.de/cn>



About me

Since November 2023: Full Professor @UPB

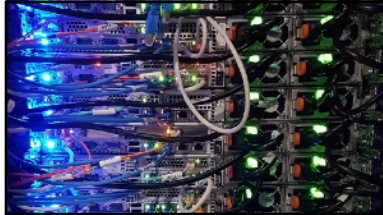
- Head of the Computer Networks group
- Office: 03.149, email: lin.wang@upb.de



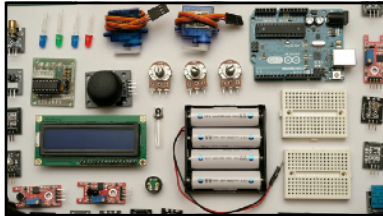
Past work experience

- 2018 - 2023: Assistant Professor, Computer Systems group, VU Amsterdam
- 2016 - 2018: Research Group Leader, TU Darmstadt
- 2015 - 2016: Research Associate, SnT Luxembourg
- 2012 - 2015: PhD in Computer Science from ICT-CAS

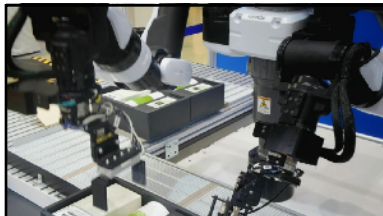
Research @ Computer Networks group



Data center in-network acceleration: leveraging programmable network devices / FPGAs to accelerate distributed systems



Sustainable Internet of Things (IoT): building sustainable and resilient IoT systems with battery-free devices



Systems for machine learning (ML): building efficient networked systems to support ML workloads

Research assistant positions or thesis projects available!

Goals of the course

To get familiar with the **state-of-the-art** of computer networking technologies

To be able to reason about the **designs/principles** in networks and networked systems

To gain **hands-on experience** with networked systems programming and outlook for research

To practice the **art of reading** research papers

It is a **big** field, but we can only focus on just **a few** topics in this course.

Course logistics

All teaching activities will be in-person

- Check PAUL for the schedule
- Lectures recorded for offline studies

Communication channels

- All announcements and all material on PANDA
- Discussions on PANDA encouraged
- For course-related questions, please write an email with subject "[ANS-SS24] XXX" where "XXX" is a keyword of your email content



Weekly schedule overview

Monday	Tuesday	Wednesday	Thursday	Friday
			Exercise A 8-11	
				Lecture 11-13
				Exercise B 14-17



For normal exercises, you can freely choose one slot to join.

When the exercise sessions are used for lab **grading**, you are expected to be available at **any of the slots** depending on the schedule.

Course grading

Study achievement

- **PASS:** if you obtain no less than 25 out of 50 lab points
- Bonuses: 40-45 (+1 step to the final grade), >45 (+2 steps to the final step)

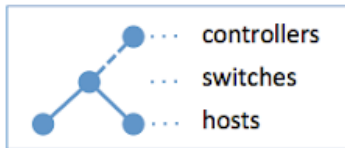


Final written exam

- **PASS:** if you obtain no less than 50 out of 100 points
- There will be non-graded exercises to help you prepare
- Two exam opportunities in total; schedule will be announced in due time



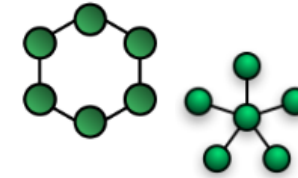
Lab assignments overview



Lab0: welcome and warm-up



Lab1: hey switches and routers



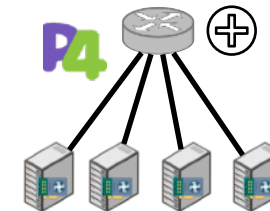
Lab2: topologies are fun



Lab3: my data center on my computer



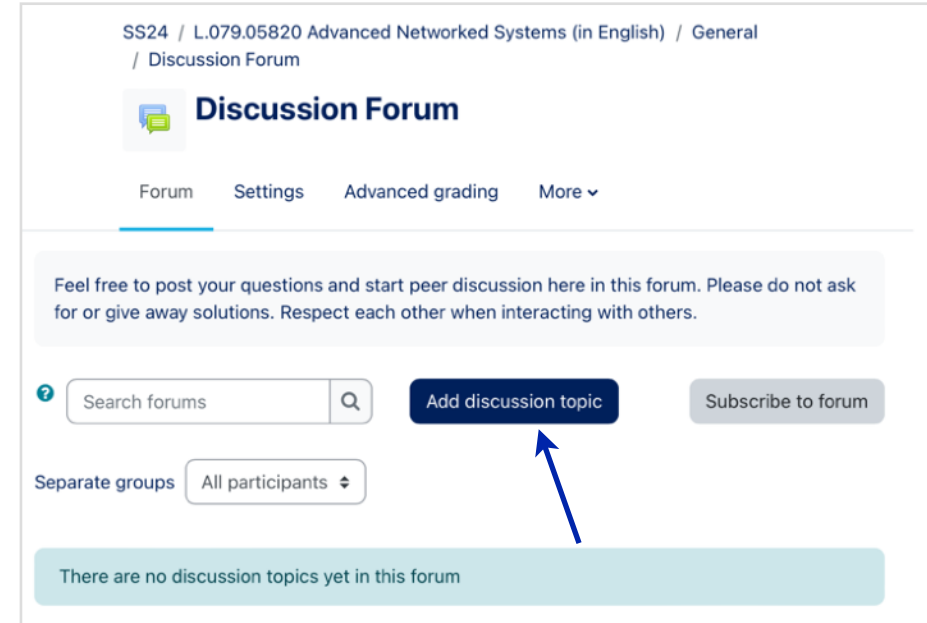
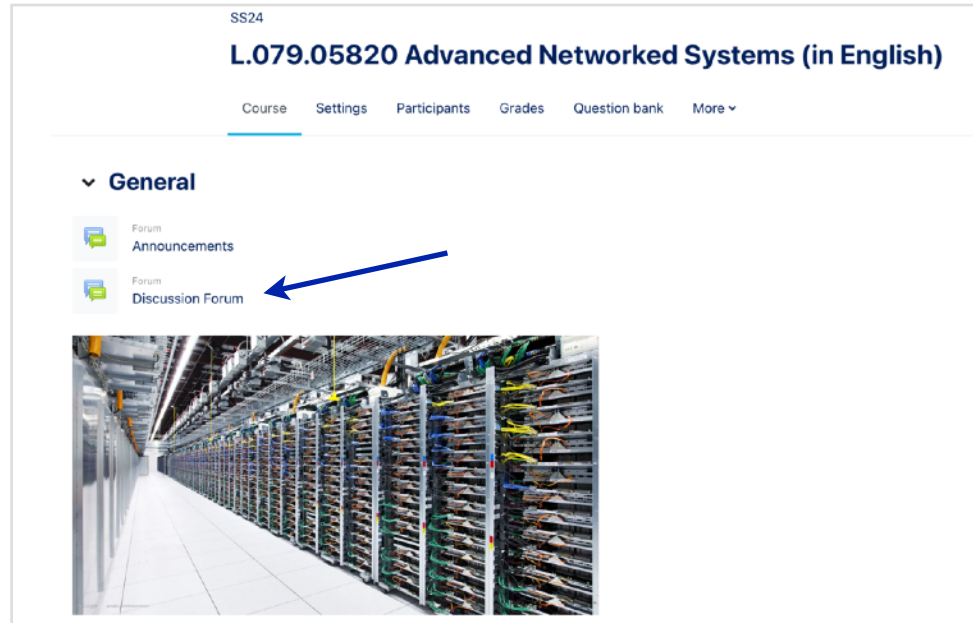
Lab4: yet another language to learn



Lab5: switches wish to do machine learning

Details and schedule will be discussed in the exercise slots next week.

Discussion on PANDA



You can post general questions/doubts in the discussion and get help from each other, but please do not post your code or spoil answers directly.

Integrity

Zero tolerance → You should **not plagiarize anything** in this course (and other courses)

The following are considered plagiarism

- Copy (part of) a solution from another team or from the Internet
- Buy a solution from any source
- Copy + make changes to any of the above

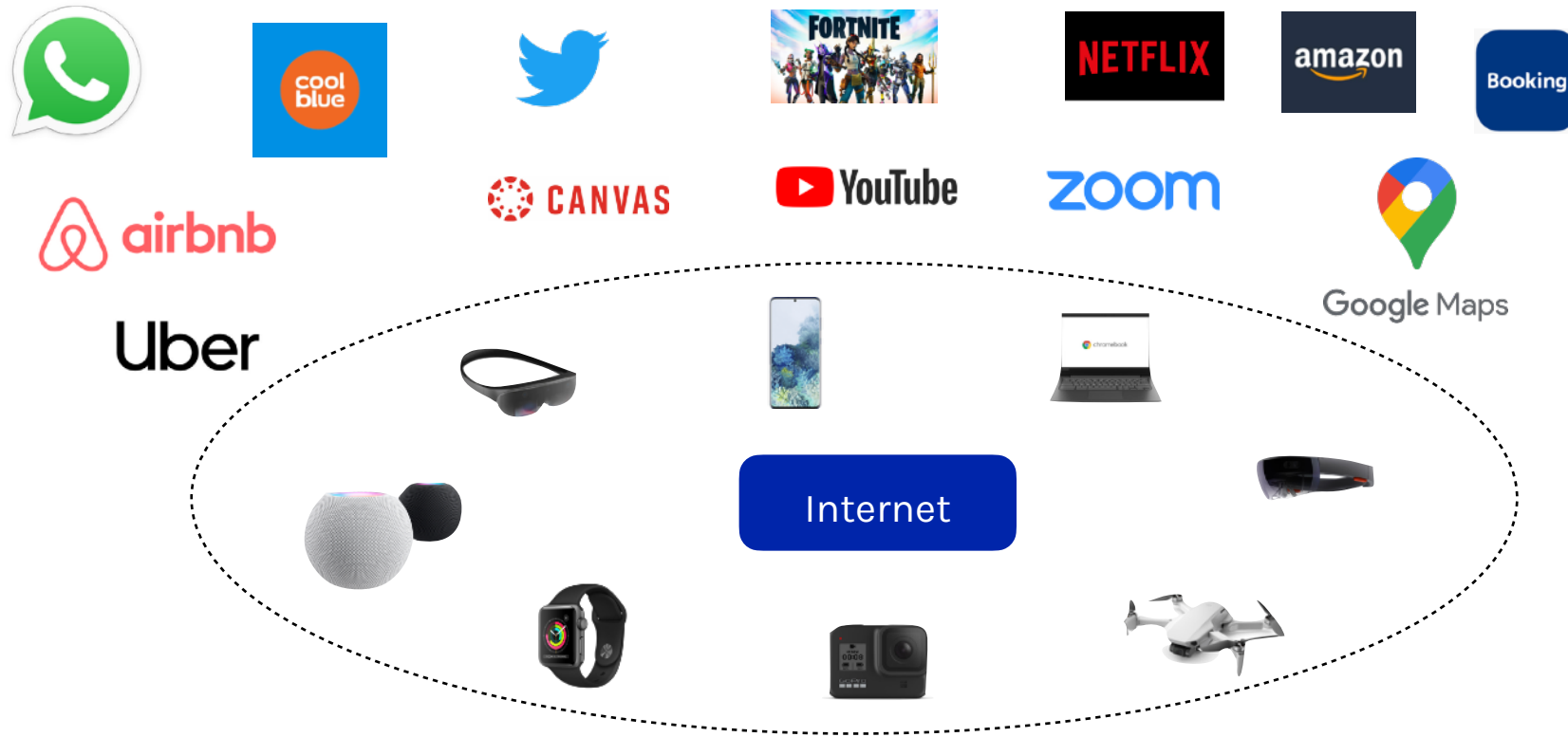
What happens if someone commits plagiarism

- The case will be reported to the exam committee

Questions?

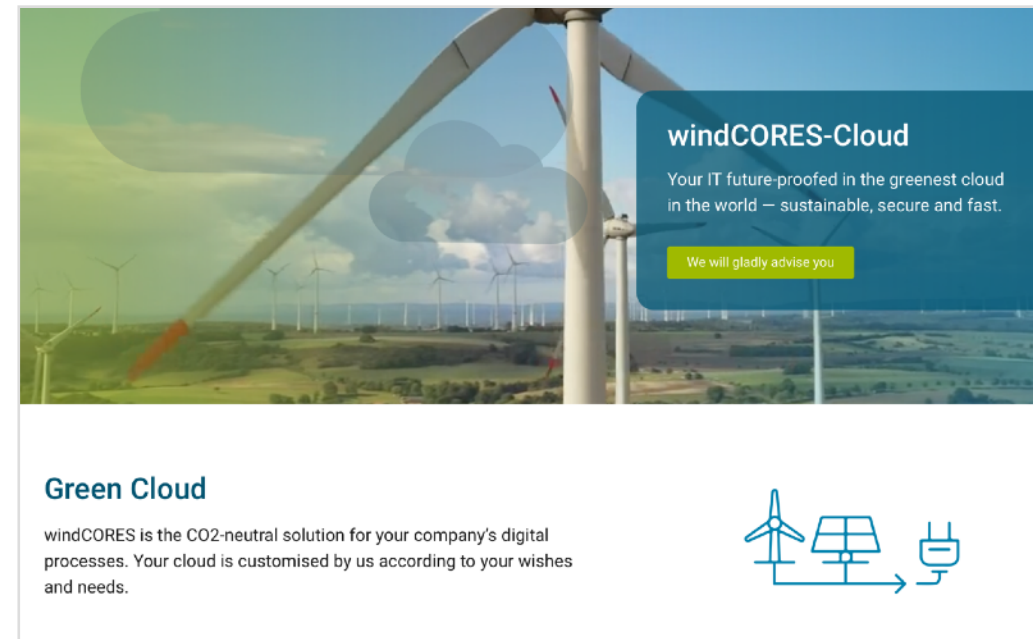
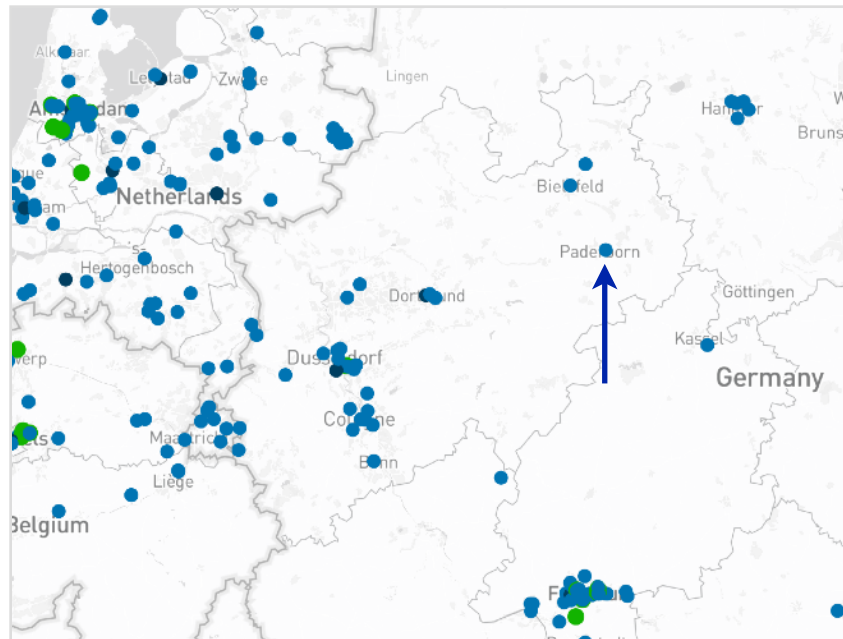
Why this course?

Because networked services are everywhere...



Why this course

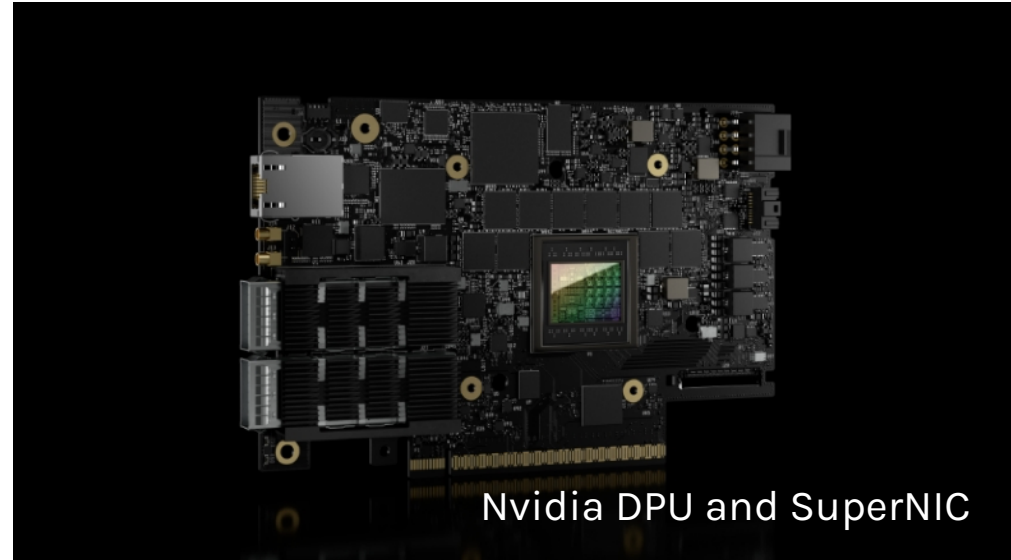
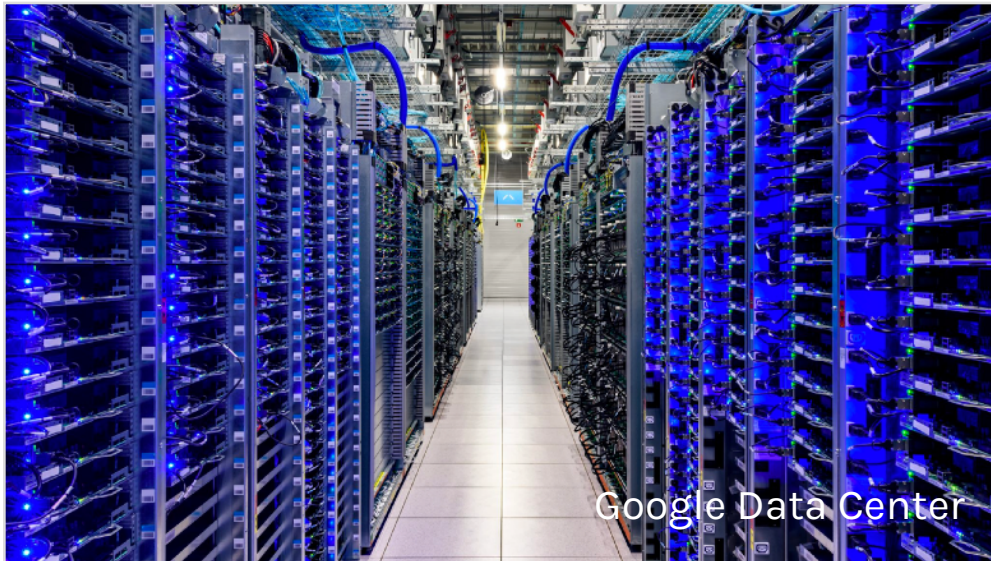
Because data centers are central to many businesses...

An advertisement for windCORES-Cloud. The top half features a background image of a wind farm with a large wind turbine in the foreground. Overlaid on this is a dark blue semi-transparent box containing the text "windCORES-Cloud" in white, followed by "Your IT future-proofed in the greenest cloud in the world — sustainable, secure and fast." and a yellow button that says "We will gladly advise you". The bottom half of the advertisement has a white background. On the left, the text "Green Cloud" is followed by a paragraph: "windCORES is the CO2-neutral solution for your company's digital processes. Your cloud is customised by us according to your wishes and needs." On the right, there is a line-art icon showing a wind turbine connected to a server rack, which is then connected to a power plug.

<https://www.datacentermap.com/>
<https://www.windcores.de/en/cloud/>

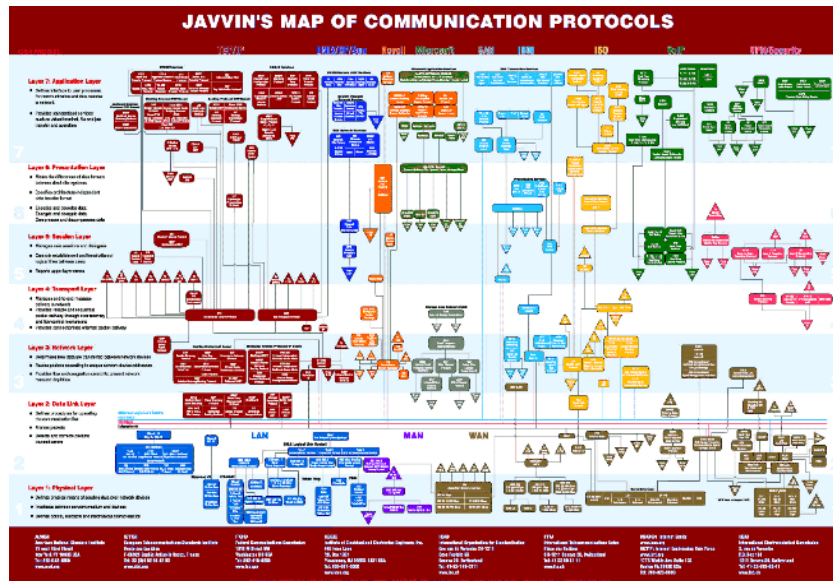
Why this course?

Because the world is now "ruled" by the big techs...

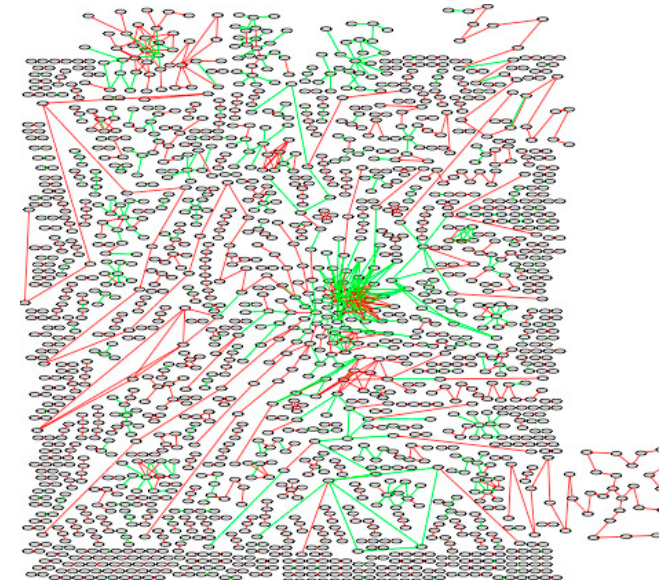


Why this course?

For the love of managing complexity and extracting simplicity...



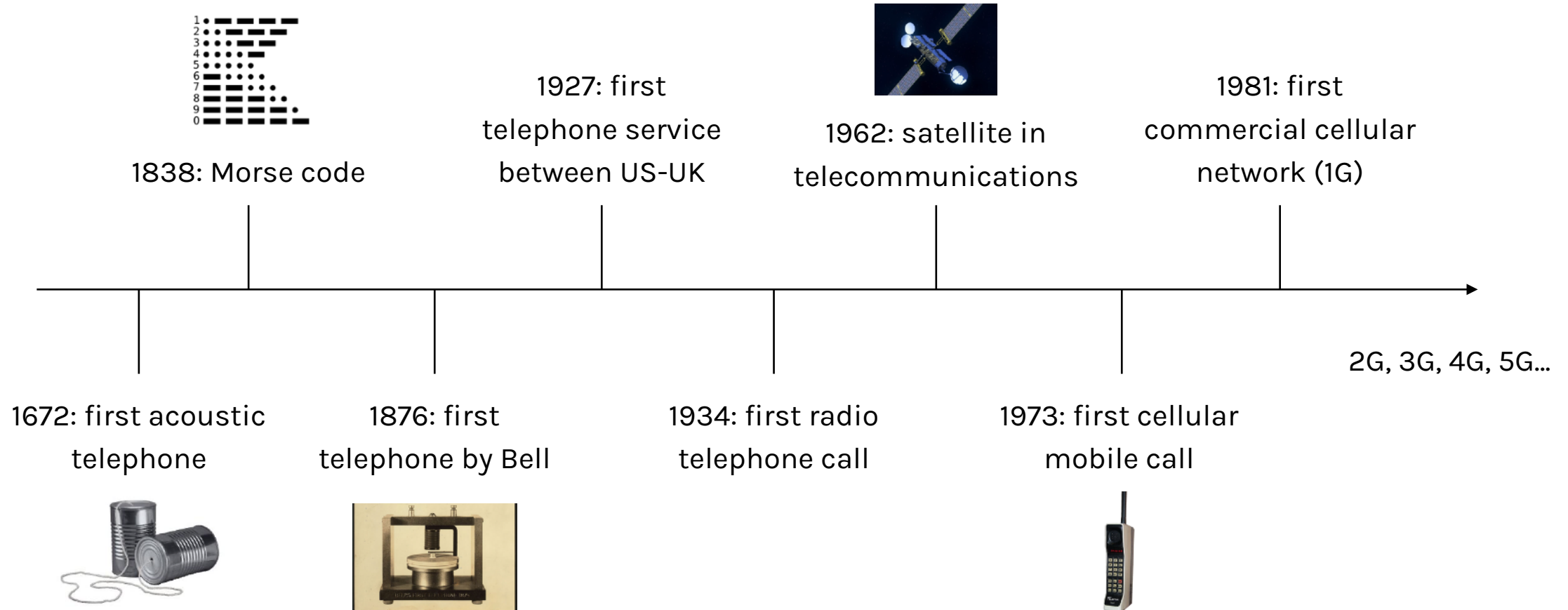
Protocol map



RFCs and their interactions

Internet: Then and Now

History of Internet: telephone network



History of Internet: visions at that time

Memex

- Vannevar Bush, “As we may think”, 1945
- A hypothetical proto-hypertext system in which individuals would compress and store all of their books, records, and communications

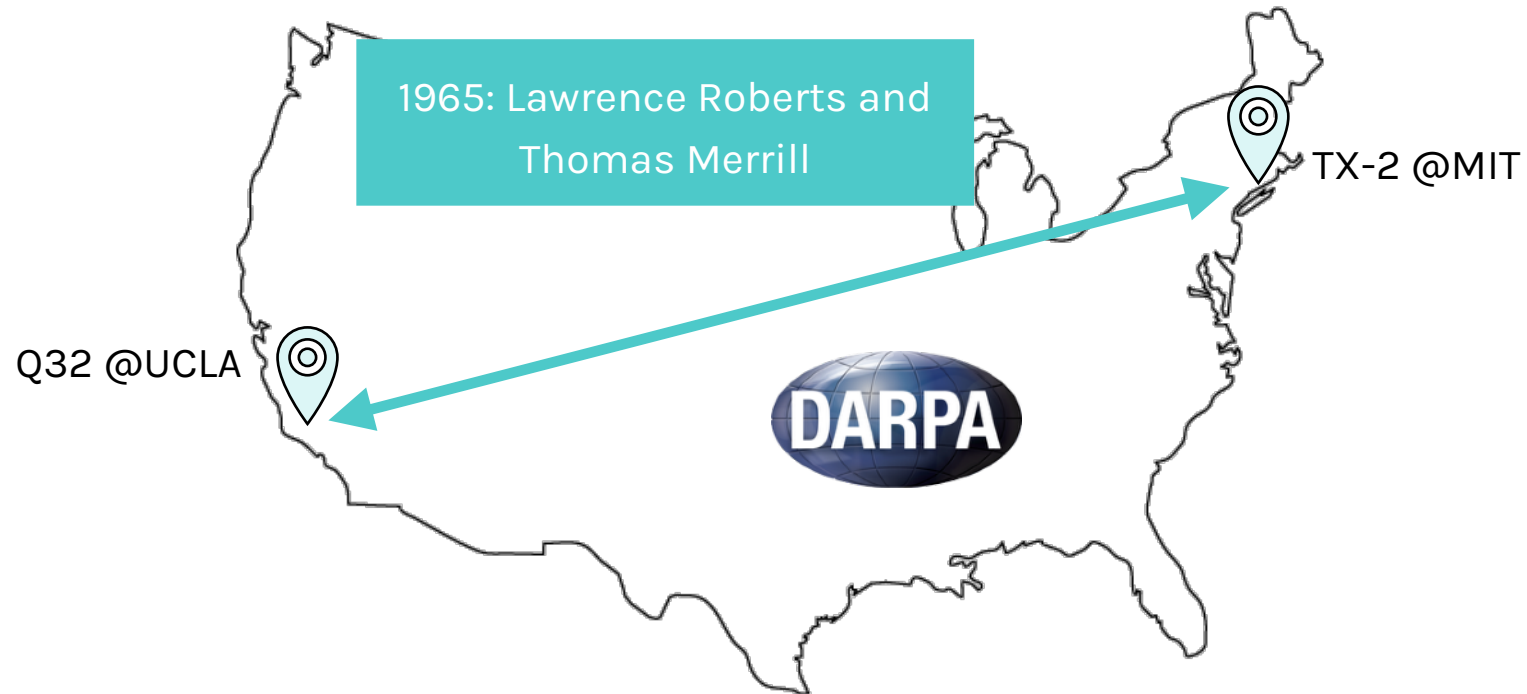


Galactic network

- J.C.R. Licklider (MIT), “Galactic network”, 1962
- Concept of a global network of computers connecting people with data and programs
- First head of DARPA (Department of Defense Advanced Research Projects Agency) computer research, October 1962

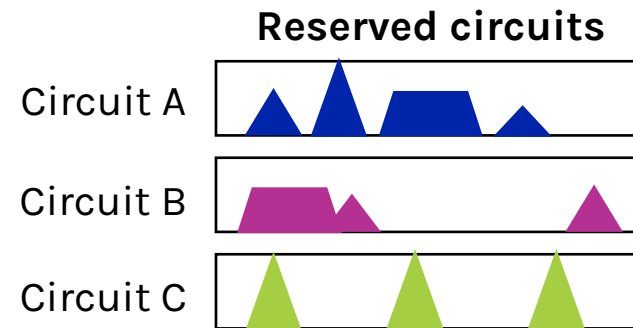


History of Internet: first wide area network



Connection is through the telephone line – it works, but it is inefficient and expensive – confirming the motivation for packet switching

Circuit switching



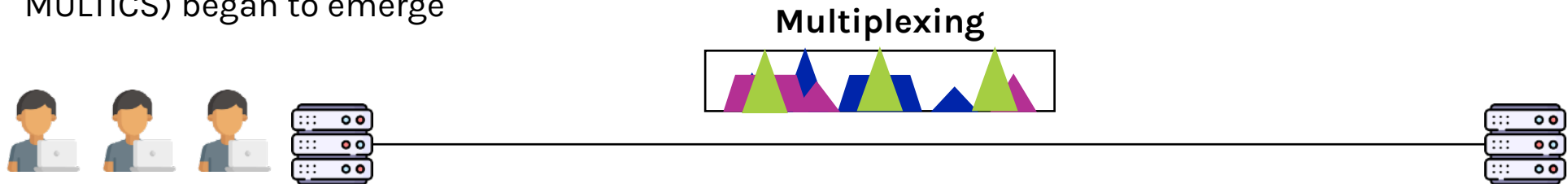
- **Physical channel** carrying stream of data from source to destination
- Three phases: setup, data transfer, tear-down
- Data transfer involves **no routing**

Packet switching

Leonard Kleinrock: queueing-theoretic analysis of packet switching in his MIT PhD thesis (1961-63) demonstrated **value of statistical multiplexing**

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

1960s: Time-sharing operating systems (e.g., MULTICS) began to emerge

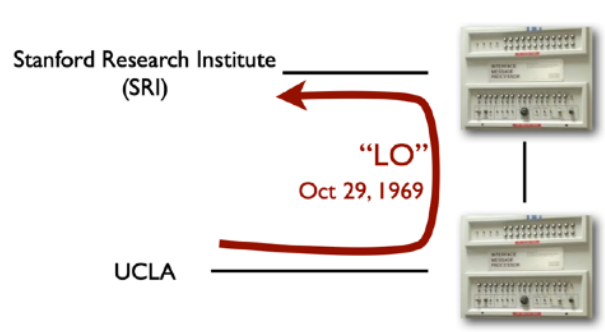


Multiplexing: multiple analog/digital signals are combined into one signal over a shared medium

- Message broken into short packets, each handled separately
- One operation: send packet
- Packets stored/queued in each router, forwarded to appropriate neighbor

History of Internet: ARPANET

- 1967 Lawrence Roberts publishes plan for the ARPANET computer network
- 1968 Bolt Beranek and Newman (BBN) wins bid to build packet switches, the Interface Message Processor (IMP)
- 1969 BBN delivers first IMP to Kleinrock's lab at UCLA



Oct 29, 1969: ARPANET went live!

The intended message was "login", but the system crashed after "o"

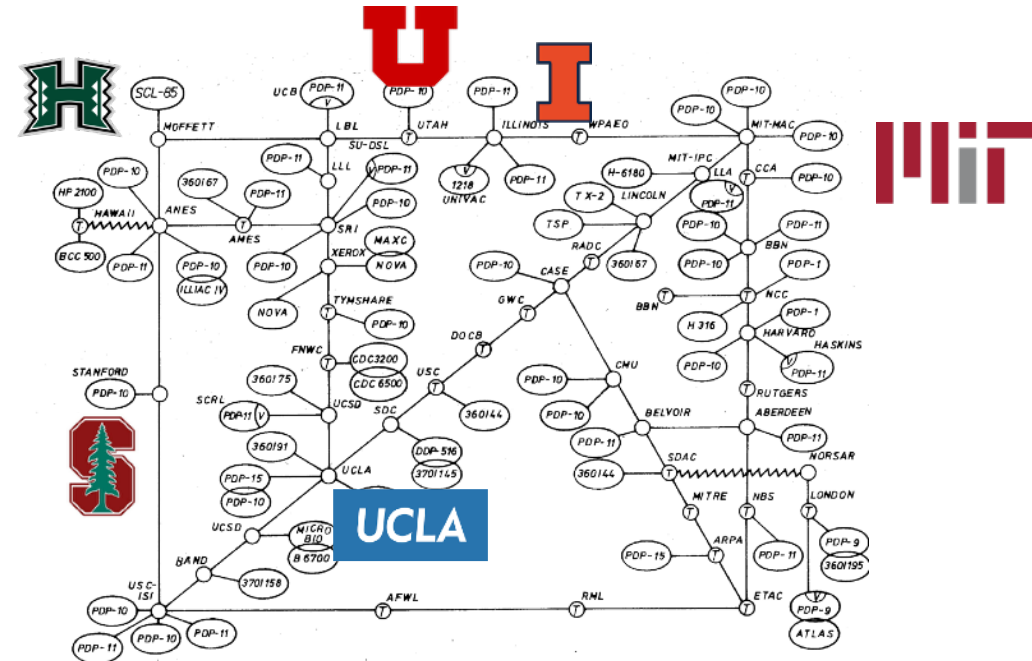
History of Internet: ARPANET grows

Originally for military computer networking, later expanded for universities

- | | |
|------|---------------------------------------|
| 1971 | Telnet, FTP |
| 1972 | Email (Ray Tomlinson, BBN) |
| 1973 | USENET (precursor of Internet forums) |

GOOD TO KNOW

ALOHAnet@Hawaii: first public wireless data network (inspiration for Ethernet and WiFi networks)



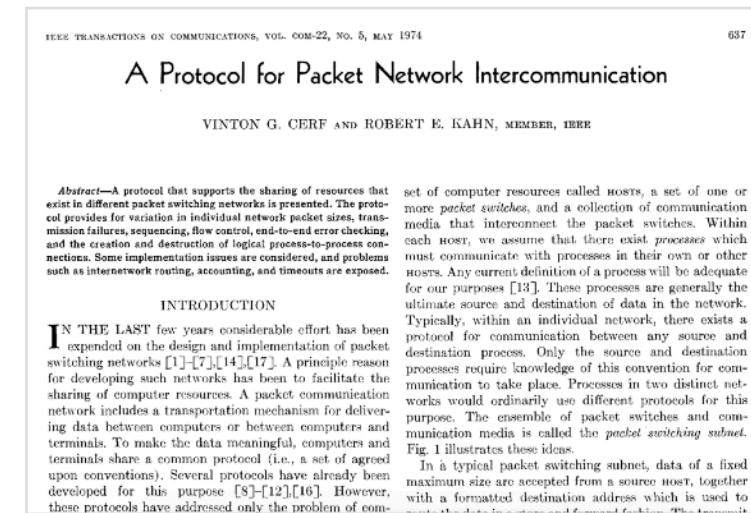
History of Internet: network of networks

In the meantime of ARPANET

- Other networks, such as PRnet, SATNET developed
- May 1973: Vinton G. Cerf and Robert E. Kahn present first paper on interconnecting networks



Concept of connecting diverse networks, unreliable datagrams, global addressing, etc. → became TCP/IP



IEEE TCOM 1974



History of Internet: standards

TCP/IP: interconnecting networks

- TCP/IP implemented on mainframes by groups at Stanford, BBN, and UCL
- David Clark implements it on Xerox Alto and IBM PC
- 1982: International Organization for Standards (ISO) releases Open Systems Interconnection (OSI) reference model
- Jan 1, 1983: “flag day” NCP (Network Control Protocol) to TCP/IP transition on ARPANET

Ethernet: local area networking

- 1976: R. Metcalfe and D. Boggs
- 1985: Radia Perlman, Spanning Tree Protocol (STP)

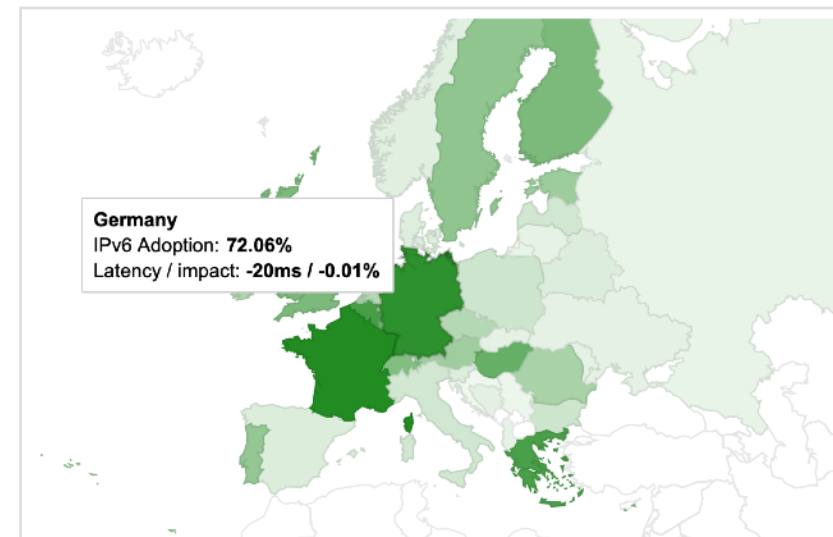
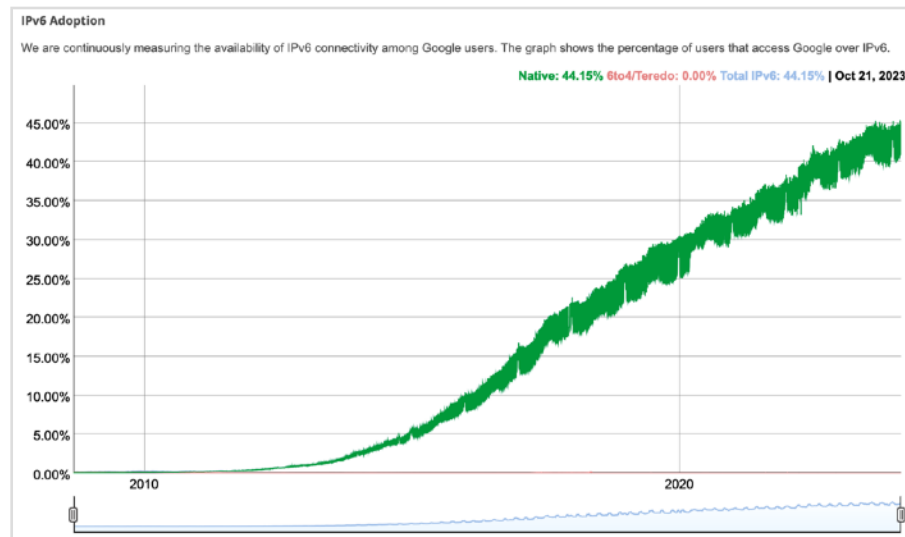
ACM A.M. Turing Award Honors Bob Metcalfe for Invention, Standardization, and Commercialization of Ethernet

Metcalfe is recognized for creating the foundational technology of the Internet which supports more than 5 billion users and enables much of modern life.



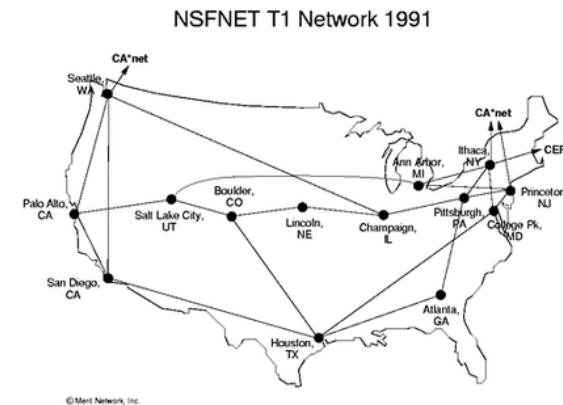
Another flag day is almost impossible nowadays

The global IPv4 → IPv6 transition is extremely low...



History of Internet: fast development

- 1983 DNS developed by Jon Postel, Paul Mockapetris (USC/ISI), Craig Partridge (BBN)
- 1984 Hierarchical routing: EGP, IGP (later to become eBGP and iBGP)
 NSFNET for US higher education
 Served many users, not just one field
 Encouraged development of private infrastructure (e.g., backbone required to be used for research and education)
 Stimulated investment in commercial long-haul networks
- 1988 Morris worm – first computer worm
- 1990 ARPANET ends
- 1995 NSFNET decommissioned



History of Internet in Germany

"Welcome to CSNET!"

Short history of the first German Internet E-Mail

On August 3, 1984 Professor Werner Zorn and his staff Michael Rotert received **the first Internet Email in Germany**. In it, Laura Breeden from the Massachusetts Institute of Technology (MIT) in Boston welcomed the German scientists to CSNET, one of the forerunners of today's Internet. Germany was the second country outside the USA to be connected to the Internet.

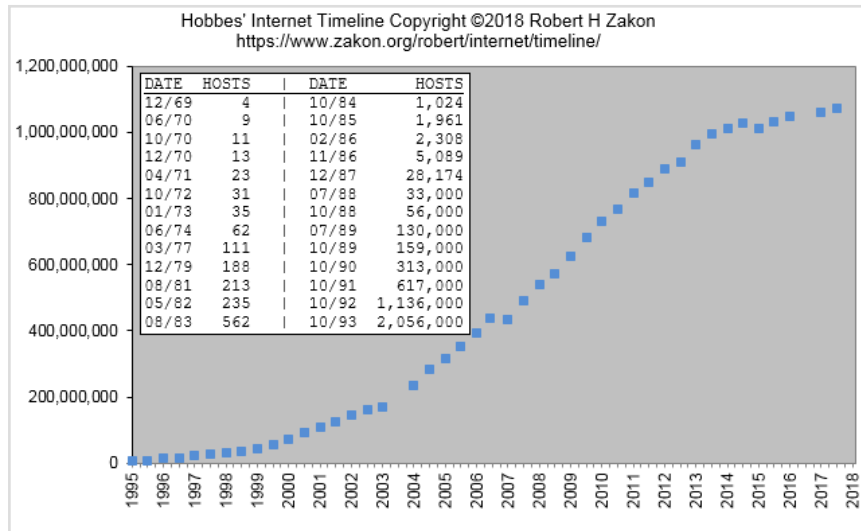
The CSNET (Computer Science Network) founded by Prof. Larry Landweber was put into operation in the USA in 1981. It was intended to give colleges and universities free access to a communication network with which the institutions could exchange information. CSNET provided a solution to the problems of scientific institutions with the ARPANET (Advanced Research Projects Agency Network), which they could not dial into due to a lack of authentication and the actual benefit for military research.

The preparatory work for connecting Germany to CSNET was carried out by Professor Werner Zorn from the then University of Karlsruhe (TH), who proposed to the Federal Ministry of Education and Research as early as 1982 that the German Research Network (DFN) be connected to the American CSNET. Professor Zorn's idea and the subsequent implementation using Dave Farber's CSNET software from the University of Delaware resulted in the e-mail from Boston received in 1984. One of Professor Zorn's further successes is the connection of China to CSNET. Through the work of Professor Wang Yunfeng from Beijing, China was able to connect to the Internet via Germany for the first time in 1987. Until a functioning infrastructure for the People's Republic of China was established, all domains were administered on the computers of Professor Zorn.



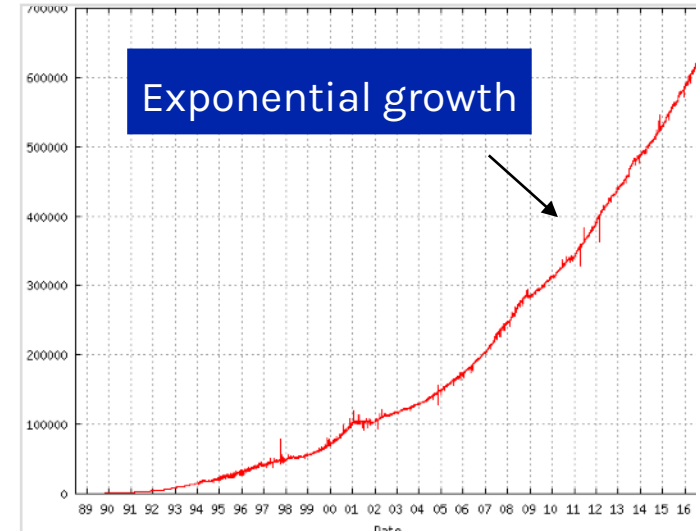
Internet growth

Number of connected hosts



Year

Internet forwarding table size



Year

<https://www.zakon.org/robert/internet/timeline/>

<https://www.redpill-linpro.com/sysadvent/2016/12/09/slimming-routing-table.html>

Questions?

What do we want from the network?

Performance: latency?
bandwidth?

Reliability, availability,
security?

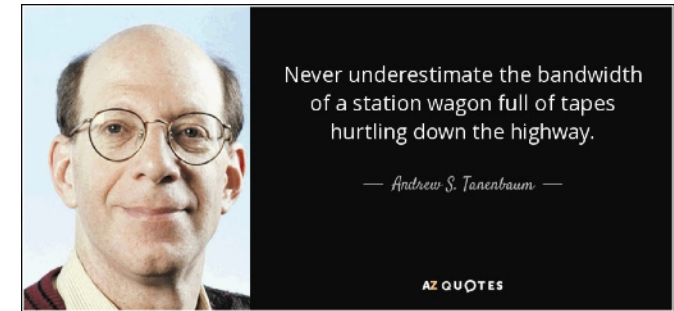
Flexibility, manageability?

Others?

Network performance

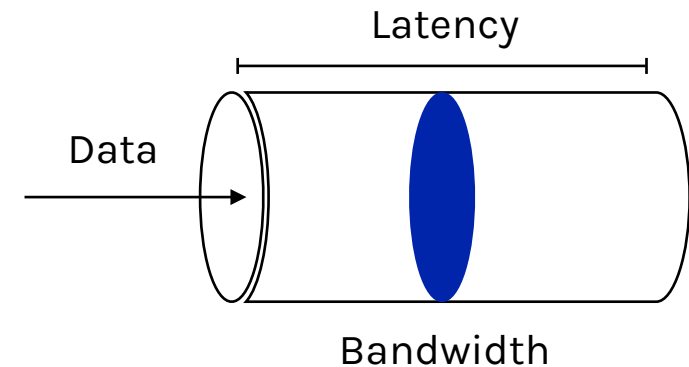
Bandwidth

- Maximum amount of data transfer across a given network path in a given amount of time
- *"Never underestimate the bandwidth of a station wagon full of tapes hurtling down the highway."* [1]



Latency

- Amount of time it takes to deliver some data from the source to the destination across the network
- "Latency lags bandwidth" — David A. Patterson



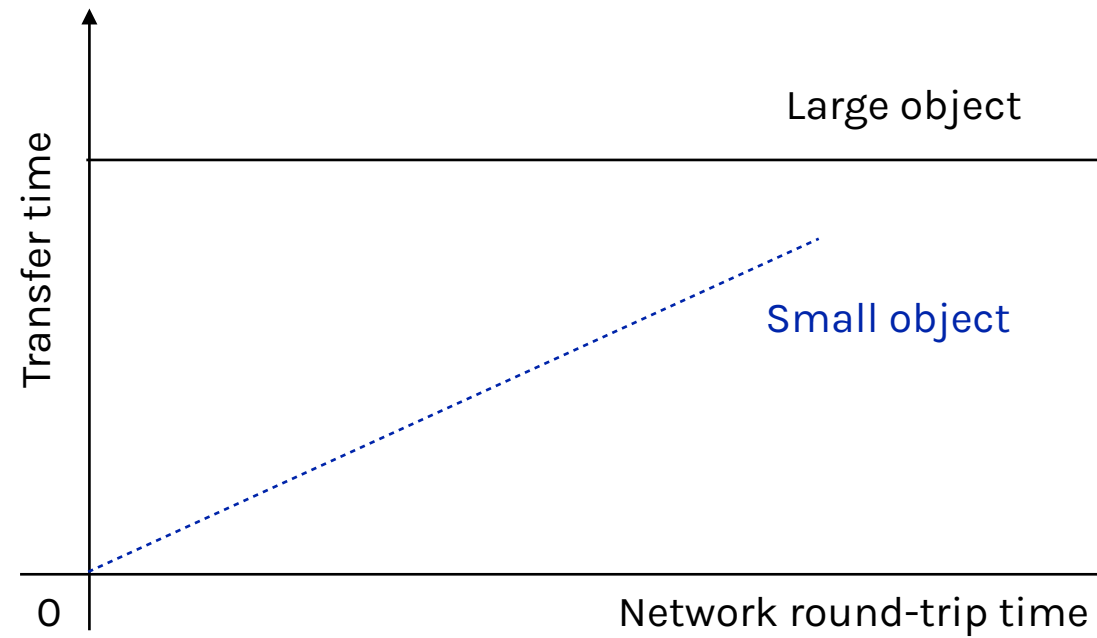
[1] Andrew S. Tanenbaum paraphrasing Dr. Warren Jackson, Director, University of Toronto Computing Services (UTCS) circa 1985

Network latency

		Barcelona ❌	Paris ❌	Tokyo ❌	Toronto ❌	Washington ❌				
Amsterdam ❌	🟢	34.708ms	🟢	10.532ms	🟡	92.104ms	🟢	80.861ms		
Auckland ❌	🔴	307.64ms	🔴	283.922ms	🟠	164.018ms	🟠	190.802ms	🟠	207.26ms
Copenhagen ❌	🟢	47.48ms	🟢	22.418ms	🔴	280.054ms	🟡	101.744ms	🟡	90.201ms
Dallas ❌	🟡	128.866ms	🟡	117.32ms	🟡	138.833ms	🟢	35.853ms	🟢	32.903ms
Frankfurt ❌	🟢	23.923ms	🟢	10.483ms	🔴	255.117ms	🟡	99.297ms	🟡	153.283ms
London ❌	🟢	27.671ms	🟢	8.079ms	🔴	275.808ms	🟢	85.952ms	🟢	79.133ms
Los Angeles ❌	🟠	168.615ms	🟡	150.986ms	🟡	107.913ms	🟢	70.805ms	🟢	76.082ms
Moscow ❌	🟢	71.174ms	🟢	47.362ms	🔴	273.371ms	🟡	134.005ms	🟠	197.867ms
New York ❌	🟡	101.057ms	🟢	73.186ms	🟠	176.173ms	🟢	17.528ms	🟢	8.143ms
Paris ❌	🟢	22.572ms	—	🔴	258.173ms	🟡	89.393ms	🟢	77.849ms	
Stockholm ❌	🟢	59.122ms	🟢	30.716ms	🔴	281.31ms	🟡	97.392ms	🟡	97.165ms
Tokyo ❌	🟠	213.031ms	🔴	258.553ms	—	🟡	159.63ms	🟠	161.622ms	

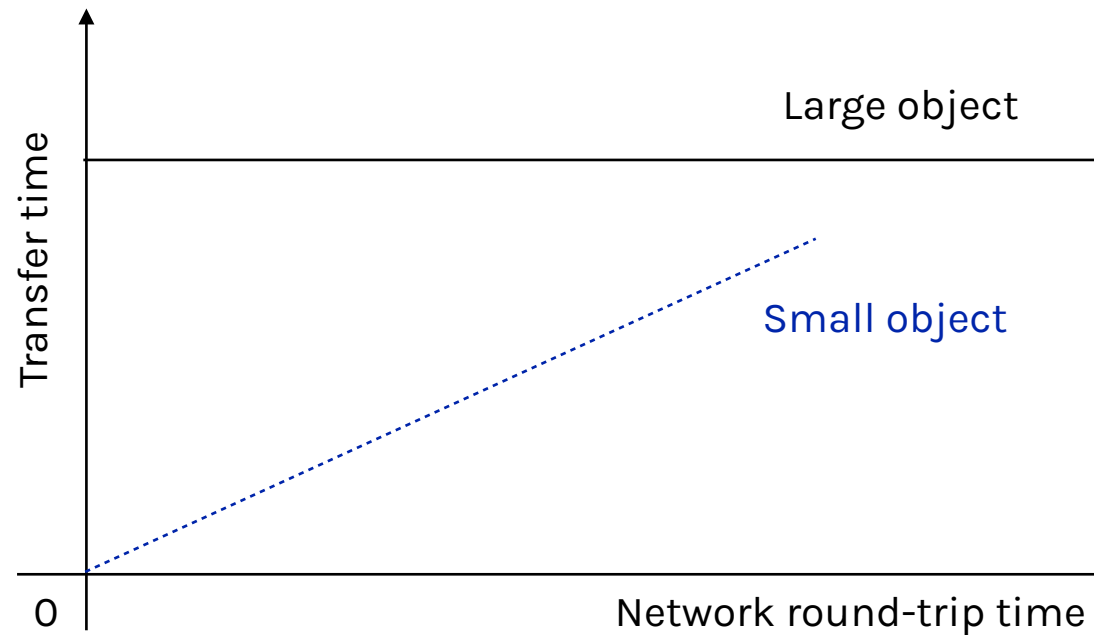
What factors are involved in these latency numbers?

Impact of bandwidth vs. latency



Assume a network link with 10 Gbps bandwidth and 1 ms latency?
How long does it take to transfer 1 B, 100 KB, or 10 GB of data?

Impact of bandwidth vs. latency



Assume a network link with 10 Gbps bandwidth and 1 ms latency?

How long does it take to transfer 1 B, 100 KB, or 10 GB of data?

With **TCP**: $100 \text{ KB} = 1460 \text{ B} * (0 + 2 + 4 + 8 + 16 + 32 + 6) \rightarrow 7 \text{ ms}$ in the best case, effective bandwidth of 0.0143 Gbps only!

Performance metric

Flow completion time

- How long does it take to complete a traffic flow?
- How long does it take to complete a set of correlated flows (co-flows)?

What else?

- How long does <https://www.google.com/?q=cool+network> take?
- What is the best video quality I can watch with, without the annoying “buffering”?
- How to guarantee the per-frame latency (e.g., 20 ms) in AR?

Not just about performance

- But also **consistent, predictable** performance

What about fairness?

Suppose a network is flow-fair. How useful is that?

Flow Rate Fairness: Dismantling a Religion

Bob Briscoe
BT Research & UCL
bob.briscoe@bt.com

“Both the thing being allocated (rate) and what it is allocated among (flows) are **completely daft** – both unrealistic and impractical.”

Food for thought:

- How to translate microbenchmarks to app-level metrics?
- How to make service providers accountable?
- How to improve the performance and approach fairness?

Network reliability

Three important considerations in network reliability

**The end-to-end
argument**

**The fate-sharing
principle**

**Packet vs. circuit
switching**

The end-to-end argument

TCP provides reliable transport

What if no reliable transport is provided?

- Every application engineers it from scratch: programming burden, bugs...

What if the network layer tried to provide reliable delivery?

Reliable (or ~~unreliable~~) transport

built on...

Best-effort global packet delivery

Reliable (~~or unreliable~~) transport

built on...

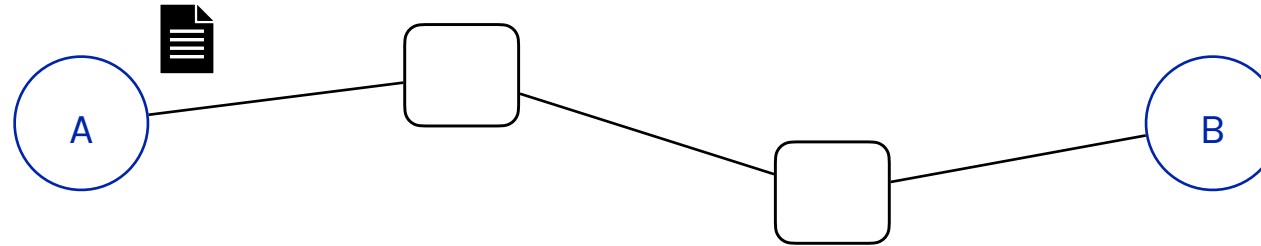
Reliable global packet delivery

What are the problems?

The end-to-end argument

Problem #1: there are applications that require speedy delivery, even if it is lossy.

Problem #2: can the network even achieve reliable global packet delivery?



Check reliability at every step (on the network layer)

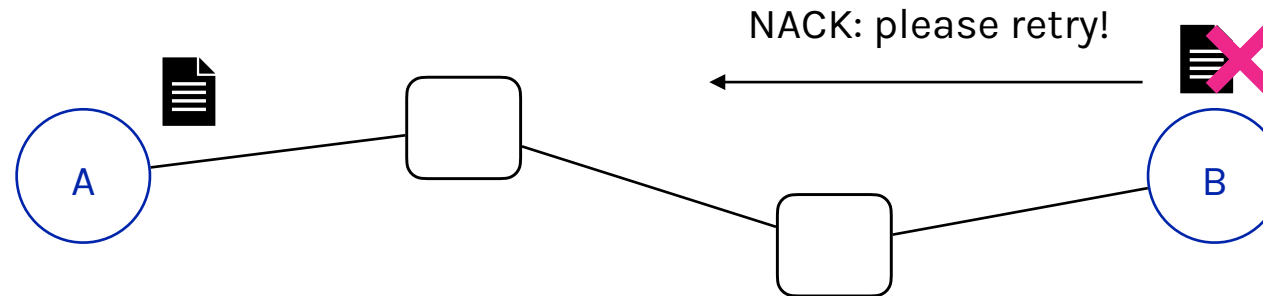
Problems: bugs, failures
are a truth of life



The end-to-end argument

“If a function can only be correctly implemented end-to-end, it must be implemented in the end systems. Implementing it in the network can, at best, only be an optimization.”

Any examples?

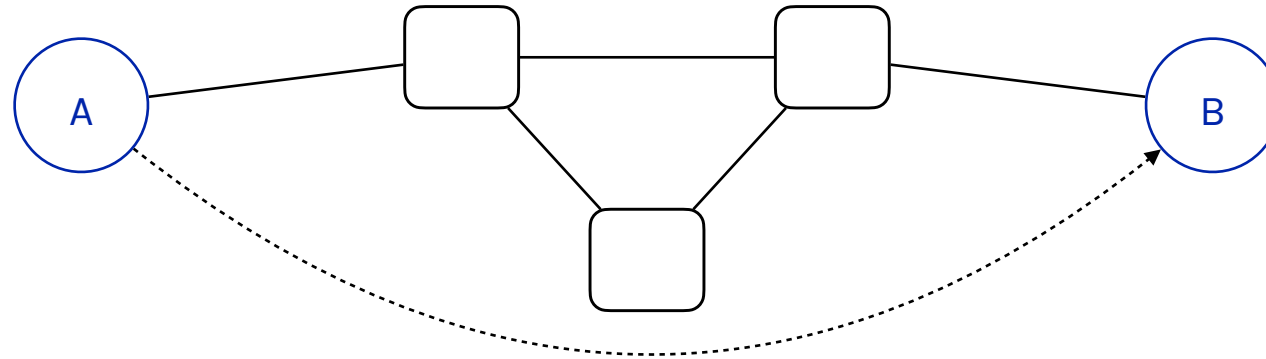


Allow unreliable steps (network layer is best-effort).

B checks correctness. On failure, B tells A to retry.

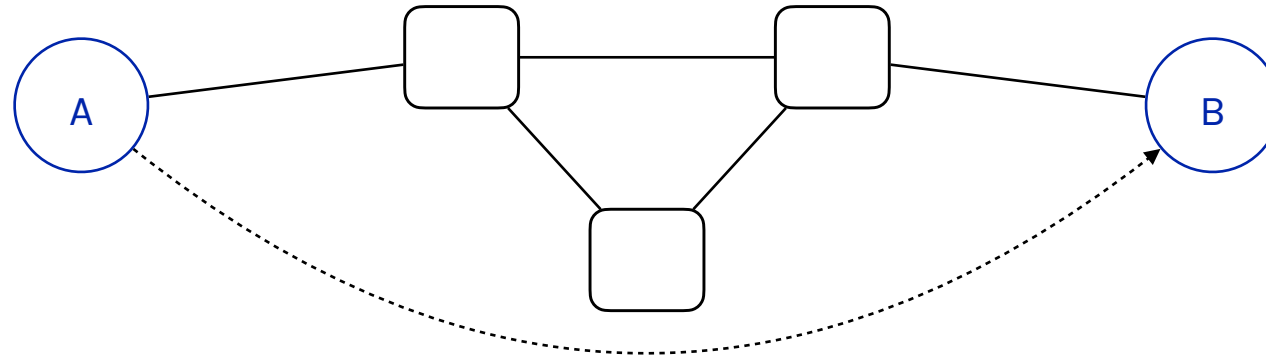
Can still fail, but only if A/B themselves fail → depends on what end-points themselves control

The fate-sharing principle



Where to maintain $A \rightarrow B$ connection state?

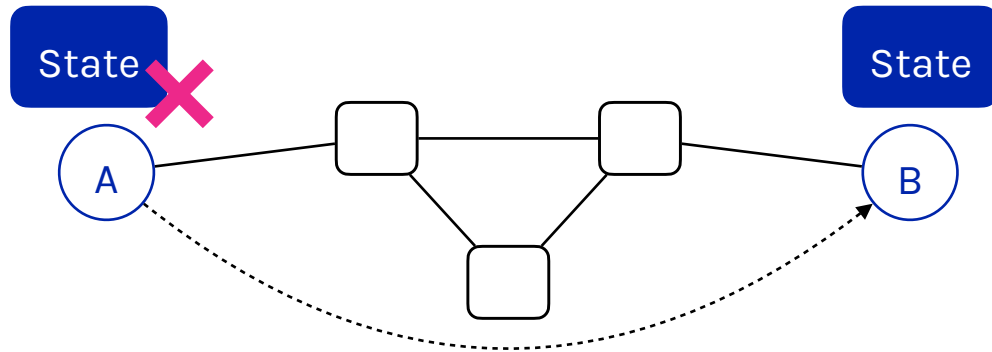
The fate-sharing principle



To deal with potential failures, store critical system state **at the nodes which rely on that state**.
Only way to lose that state is if the node that relies on it fails, in which case it does not matter.

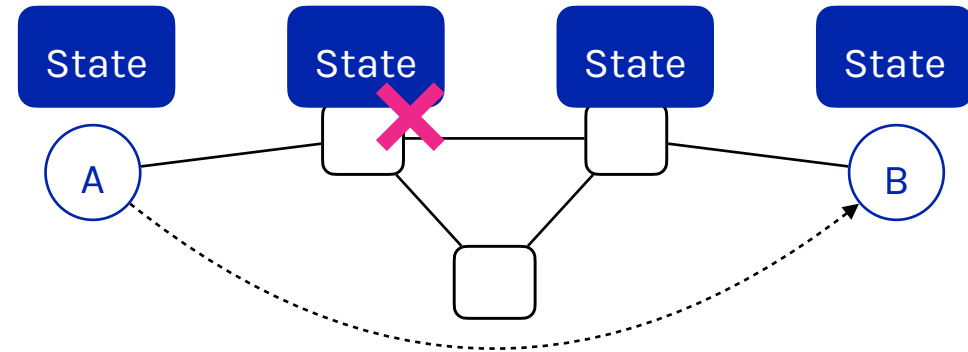
The fate-sharing principle

Keep state on end hosts



In the end-host fails, the connection state is irrelevant anyway

Keep state on network devices



Failures of network devices require the state to be cleaned up or recreated → complex consistency issue!

Packet vs. circuit switching

Circuit switching

- Predictable performance

- Wasteful, if packet is bursty and short
- Large latency for small messages, as they wait for circuits creation
- Require new circuit setup upon failure

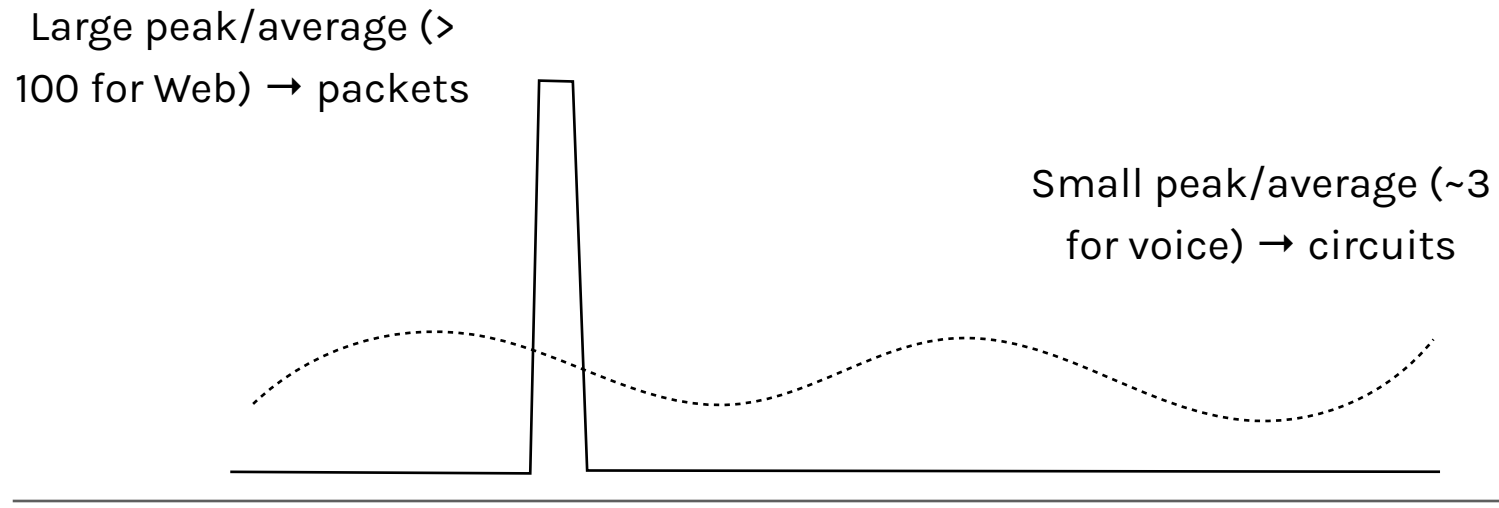
Packet switching

- Efficient use of resources
- Automatic, in-network rerouting on failures

- Unpredictable performance
- Requires buffer management and congestion control

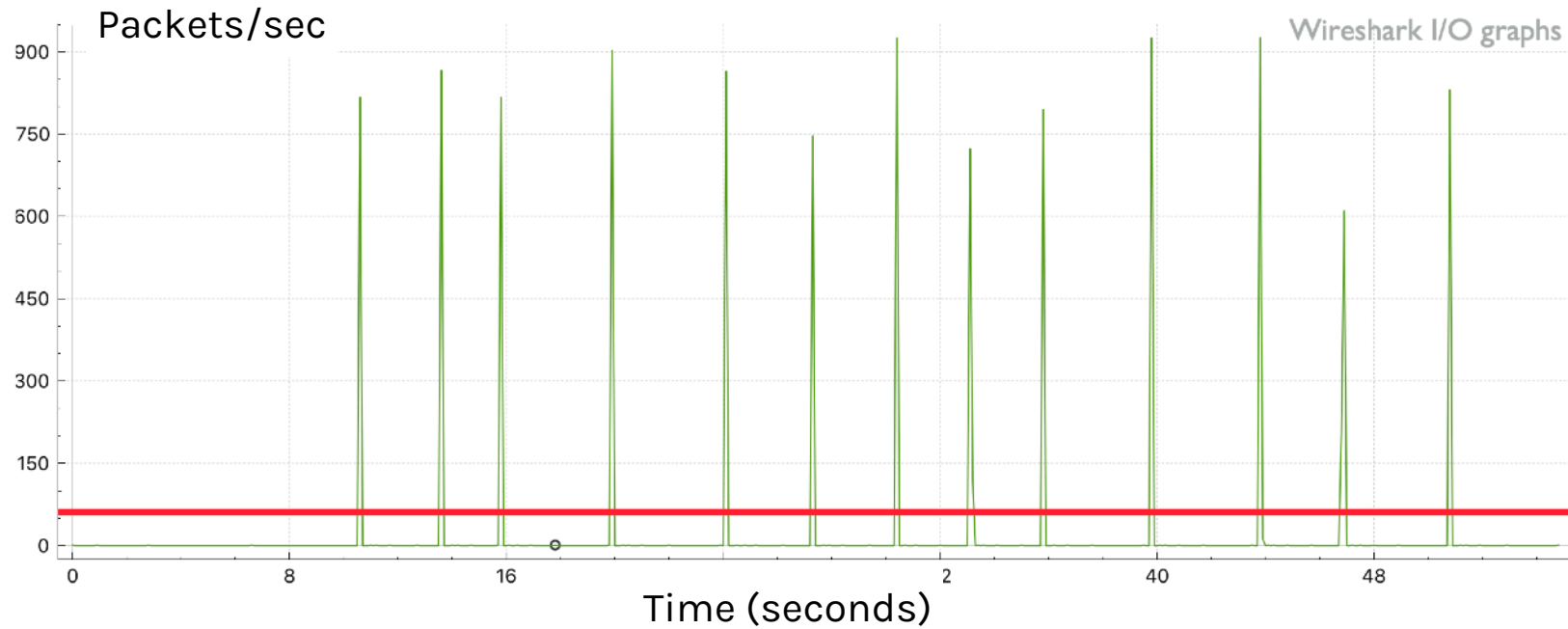
Packet switching beats circuit switching with respect to **resilience** and **efficiency**

Packet vs. circuit switching examples



Which pattern do you think modern video streaming (e.g., YouTube) will follow?

Video streaming today



Videos are split into small chunks and transmitted chunk by chunk

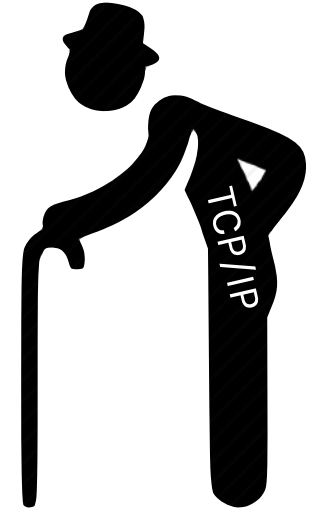
Internet: current status

Internet is there for more than 50 years

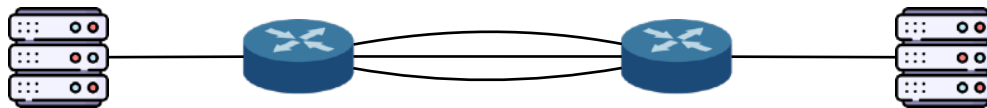
- Networks keep growing and more applications are developed
- TCP/IP is the norm: to program a network application, you simply use the socket APIs

So, the network will keep growing with the same set of technologies?

- Partially, yes, we are still using these old technologies (TCP/IP)
- But, there are also new developments
- This course is to reveal the **state-of-the-art** of computer networking



Still TCP? Yes, but there are more!



Multipath TCP, QUIC

facebook Engineering

[Open Source](#) [Platforms](#) [Infrastructure Systems](#) [Physical Infrastructure](#) [Video Engineering & AR/VR](#)

POSTED ON OCT 21, 2020 TO ANDROID, DATA INFRASTRUCTURE, IOS, NETWORKING & TRAFFIC, WEB

How Facebook is bringing QUIC to billions



We Need a Replacement for TCP in the Datacenter

John Ousterhout

Stanford University

(paper currently under submission)

Abstract

In spite of its long and successful history, TCP is a poor transport protocol for modern datacenters. Every significant element of TCP, from its stream orientation to its requirement of in-order packet delivery, is wrong for the datacenter. It is time to recognize that TCP's problems are too fundamental and interrelated to be fixed; the only way to harness the full performance potential of modern networks is to introduce a new transport protocol into the datacenter. Homa demonstrates that it is possible to create a transport protocol that avoids all of TCP's problems. Although Homa is not API-compatible with TCP, it should be possible to bring it into widespread usage by integrating it with RPC frameworks.

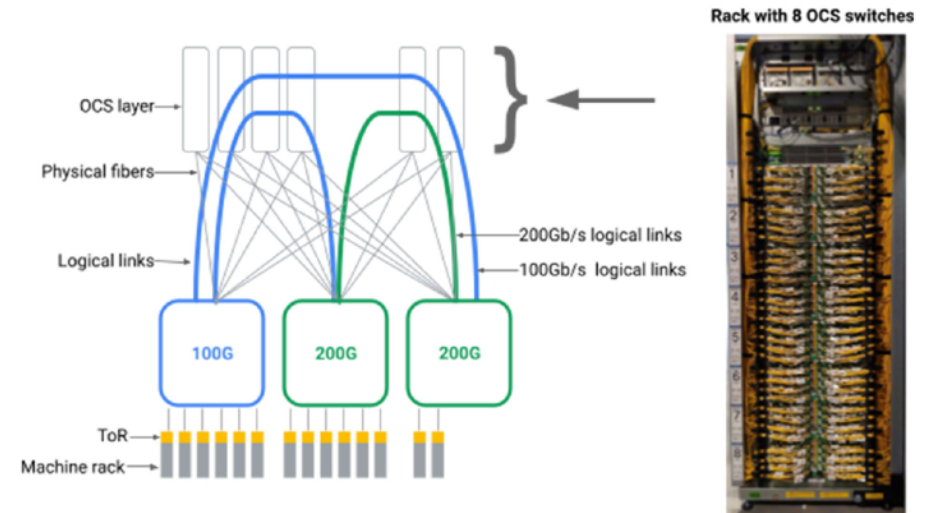
1 Introduction

The TCP transport protocol [6] has proven to be phenomenally successful and adaptable. At the time of TCP's design in the late 1970's, there were only about 100 hosts attached

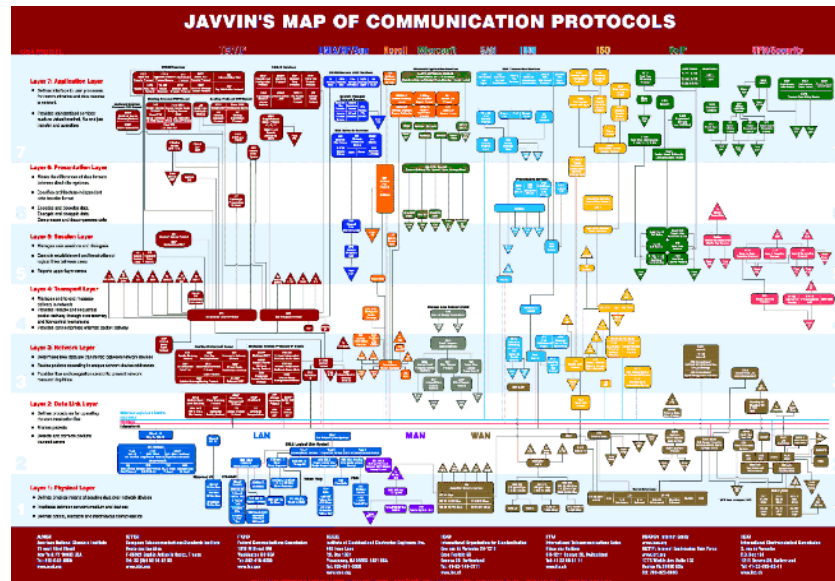
One example is load balancing, which is essential in datacenters in order to process high loads currently. Load balancing did not exist at the time TCP was designed, and TCP interferes with load balancing both in the network and in software.

Section 4 argues that TCP cannot be fixed in an evolutionary fashion; there are too many problems and too many interlocking design decisions. Instead, we must find a way to introduce a radically different transport protocol into the datacenter. Section 5 discusses what a good transport protocol for datacenters should look like, using Homa [16, 18] as an example. Homa was designed in a clean-slate fashion to meet the needs of datacenter computing, and virtually every one of its major design decisions was made differently than for TCP. As a result, some problems, such as core congestion, are eliminated entirely. Other problems, such as congestion control and load balancing, become much easier to address. Homa demonstrates that it is possible to solve all of TCP's problems.

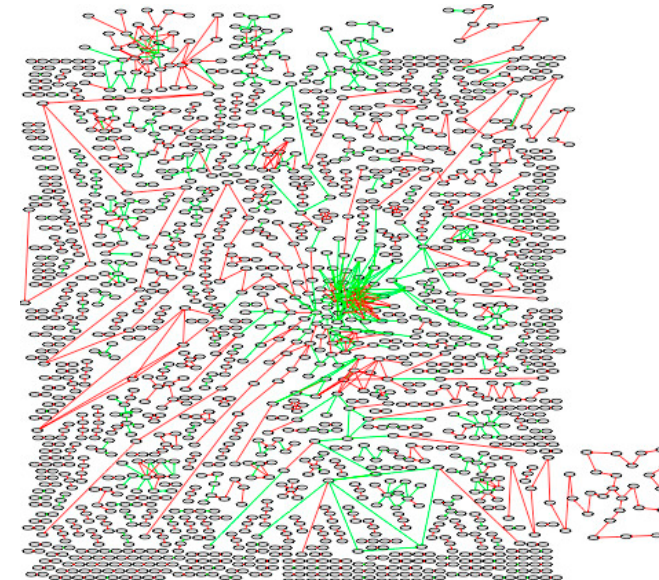
Google data centers: an inside perspective



How to extract simplicity?

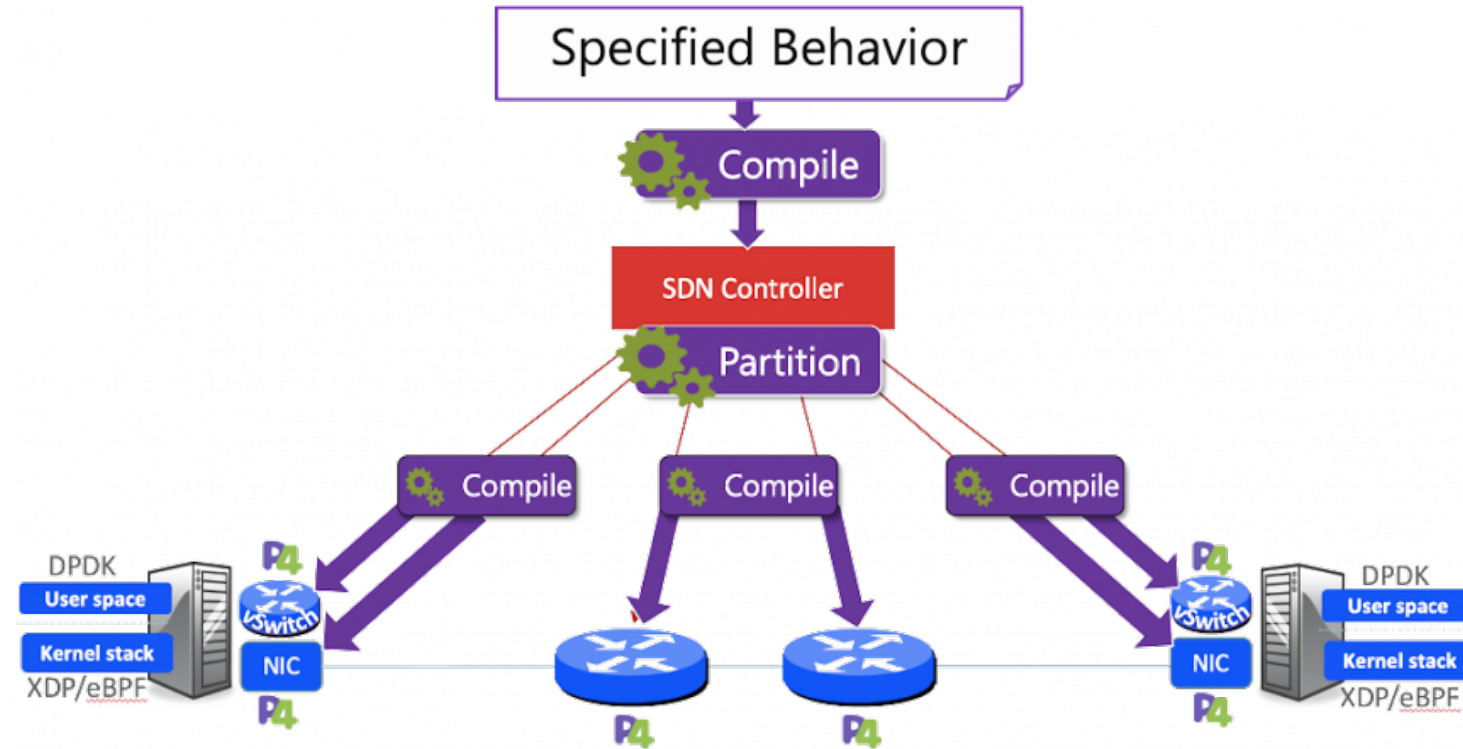


Protocol map

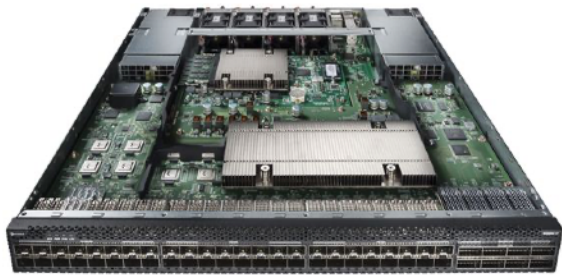


RFCs and their interactions

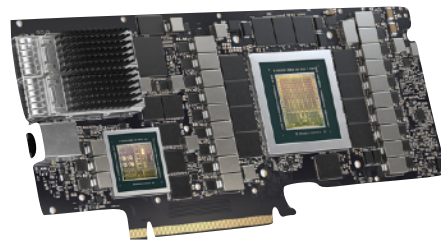
Programmable networks



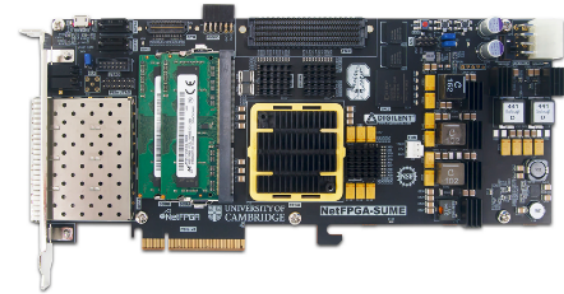
New network hardware



Intel Tofino switching ASIC

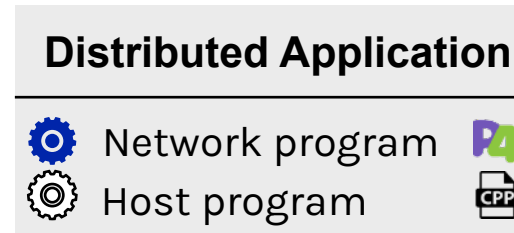
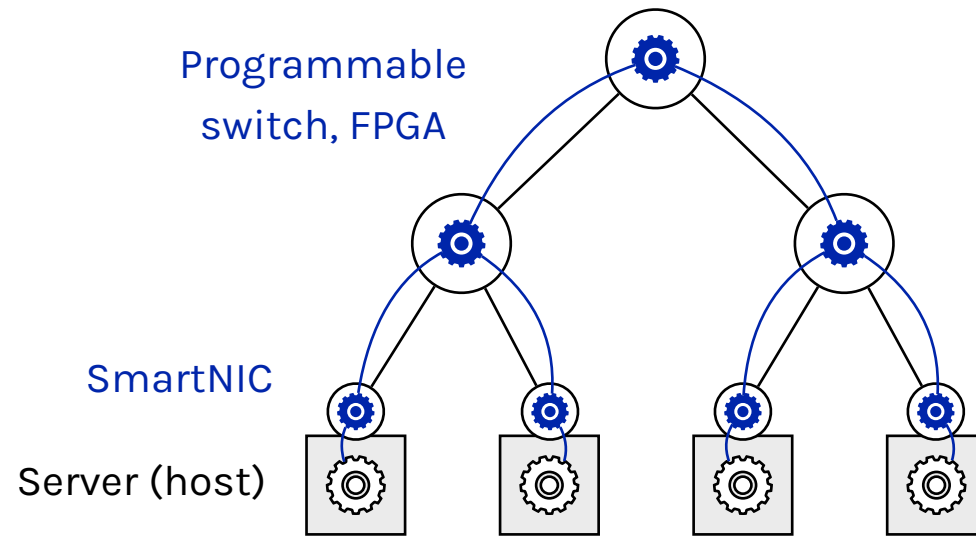


Nvidia SmartNICs and DPUs



NetFPGA

Network innovations: in-network computing

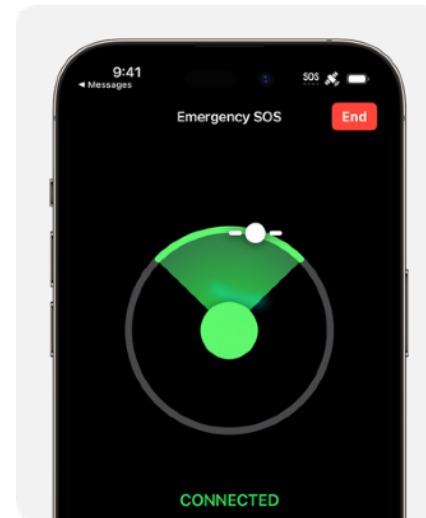


Developer

Example services: aggregation, caching, agreement, DB query acceleration, machine learning...

New forms of networks

Not covered in this course, but you have a better idea where to start after this course...



Course structure

Basics (recap)

Network forwarding
and routing

Congestion control

Core topics

Data center networking
(architecture, transport)

Programmable networks
(SDN, NFV, P4, hardware)

End-host networking
(eBPF/XDP, DPDK)

Advanced

In-network computing

Network monitoring

Machine learning for
networking

Next lecture: networking basics

What happens when you visit Google in your browser?

