Advanced Computer Networks

Introduction

Lin Wang

Period 2, Fall 2021

VU VRIJE UNIVERSITEIT AMSTERDAM

Teaching team

Lin Wang (Lecturer)
Assistant Professor

Vinod Nigade (TA)
PhD researcher

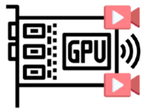
George Karlos (TA)
PhD researcher

Our research

We focus on programming models and resource management for high-performance networked systems

Programmable networks

Programmable data plane with P4, in-network computing, intent-driven networking, network configuration synthesis



Edge AI systems

Video stream analytics, video query systems, resource scaling for deep learning inference workloads

Cyber-physical systems

Battery-free intermittent computing, neuromorphic computing, embedded intelligence

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, so we can only focus on just **a few** topics.

Course logistics

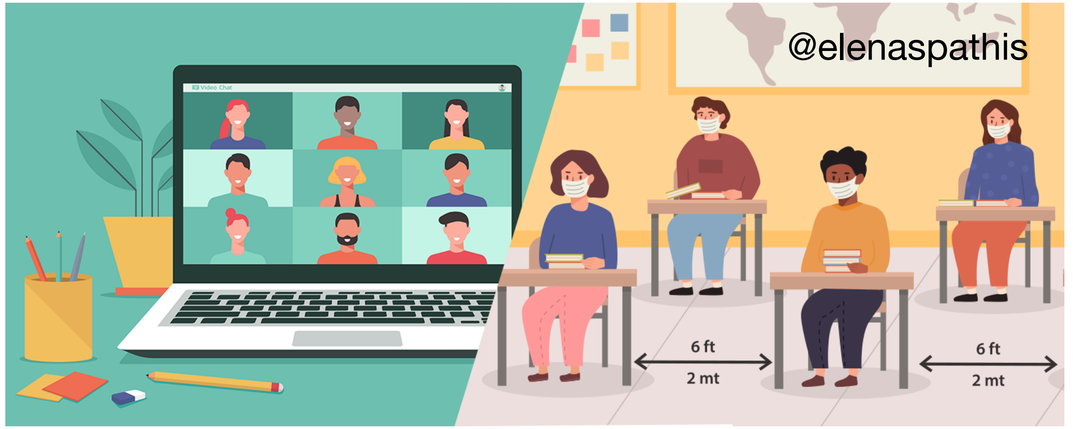
All teaching activities will be hybrid

- On-campus (check Rooster for locations)
- Live streamed via Zoom
- Recorded for offline studies
- Note: Please stay at home if you do not feel well

Exams will be on-site

- No online alternatives will be provided

If you have any difficulties, communicate to us early enough.



Course logistics

Communication channels

- All announcements and all material on Canvas
- Discussions on Canvas encouraged
- Questions: please send an email to vu.acn.ta@gmail.com
  (emails to our personal accounts may not be processed)

Policy

- Attendance is strongly encouraged

Office hours

- Every Wednesday 9:30 - 10:30
- Online via Zoom
- Not mandatory, but feel free to join

Course grading

Project: 50 points

Final exam: 50 points

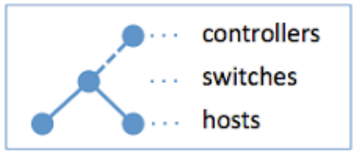
PASS condition

- If you obtain no less than 25/50 points in **both** components

Final grade scaling

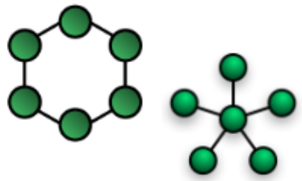
[95, 100] → 10.0	[68, 75) → 7.5
[90, 95) → 9.5	[62, 68) → 7.0
[85, 90) → 9.0	[56, 62) → 6.5
[80, 85) → 8.5	[50, 56) → 6.0
[75, 80) → 8.0	[0, 50) → FAIL

Project labs preview

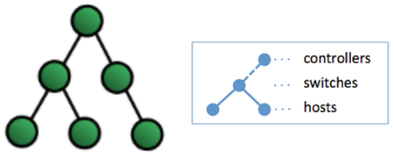


Lab0: get familiar with Mininet – a network emulator

Lab1: implement a learning switch



Lab2: generate and compare data center network topologies



Lab3: build your own data center in Mininet

Lab4: implement in-network services

Bonus (your own project ideas)

Project labs organization

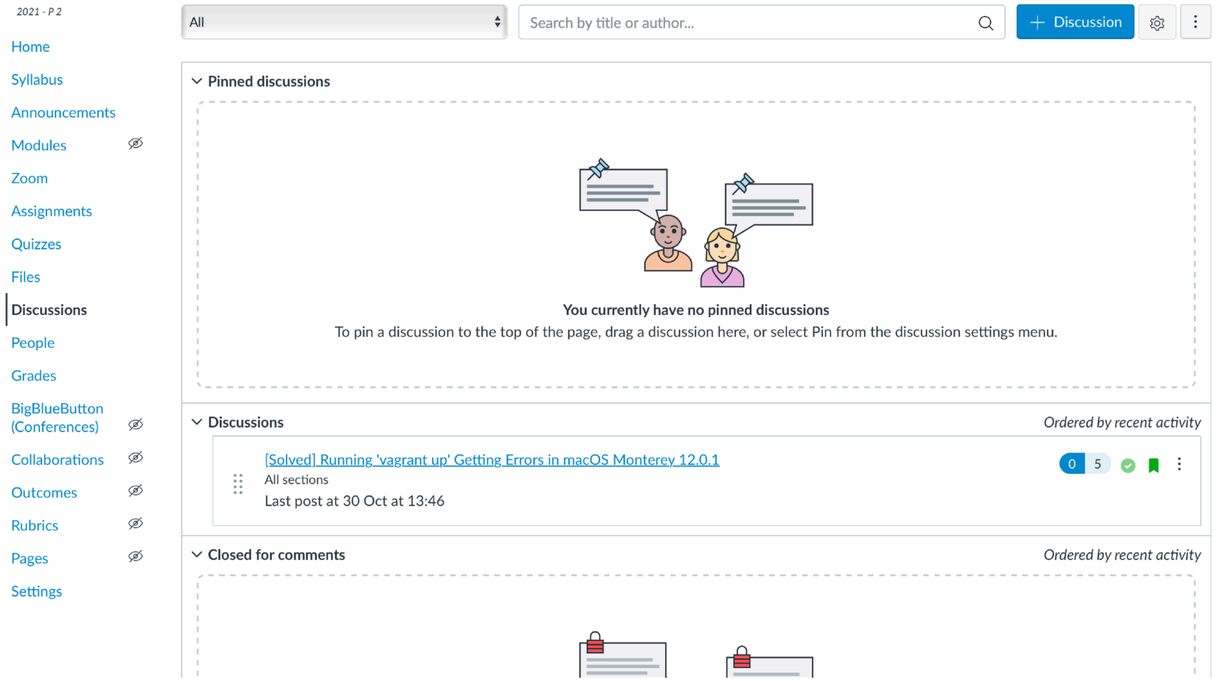
Individual assignments for lab0 and lab1

- Lab0 is just a warm up, no points and no submission needed, but do not skip it
- Lab1 will be assessed with Canvas quizzes, no code submission needed

Group assignment for lab2 through lab4

- You work in a group of max. 2
- Choose your own teammate, deadline **Friday Nov 12, 2021**
- Split the work evenly, all of you need to understand the entire code
- Submission: code + report in PDF, all in one zip file

Peer discussions are encouraged



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

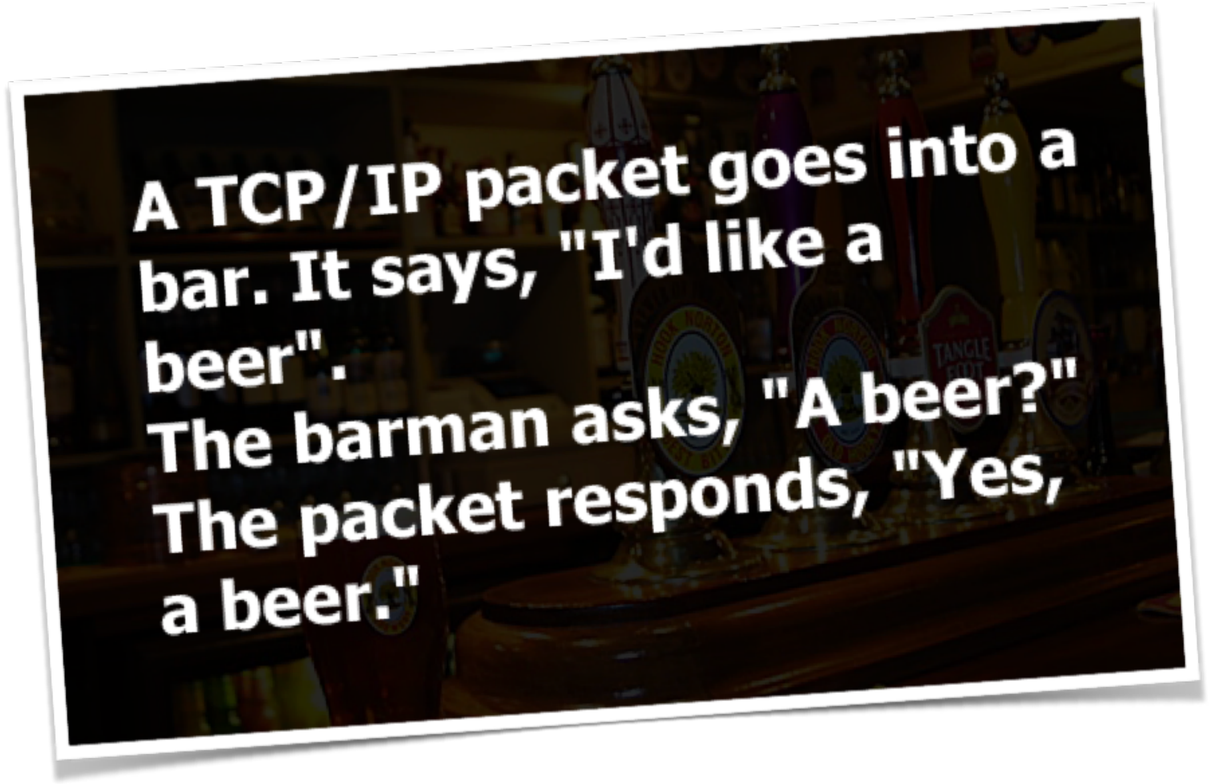
- Copy (part of) a solution from another team or from the Internet
- Buy a solution from any source
- Copy + make changes to any of the above

What happens if someone commits plagiarism

- The case will be reported to the exam committee
- It is up to them to decide on disciplinary actions

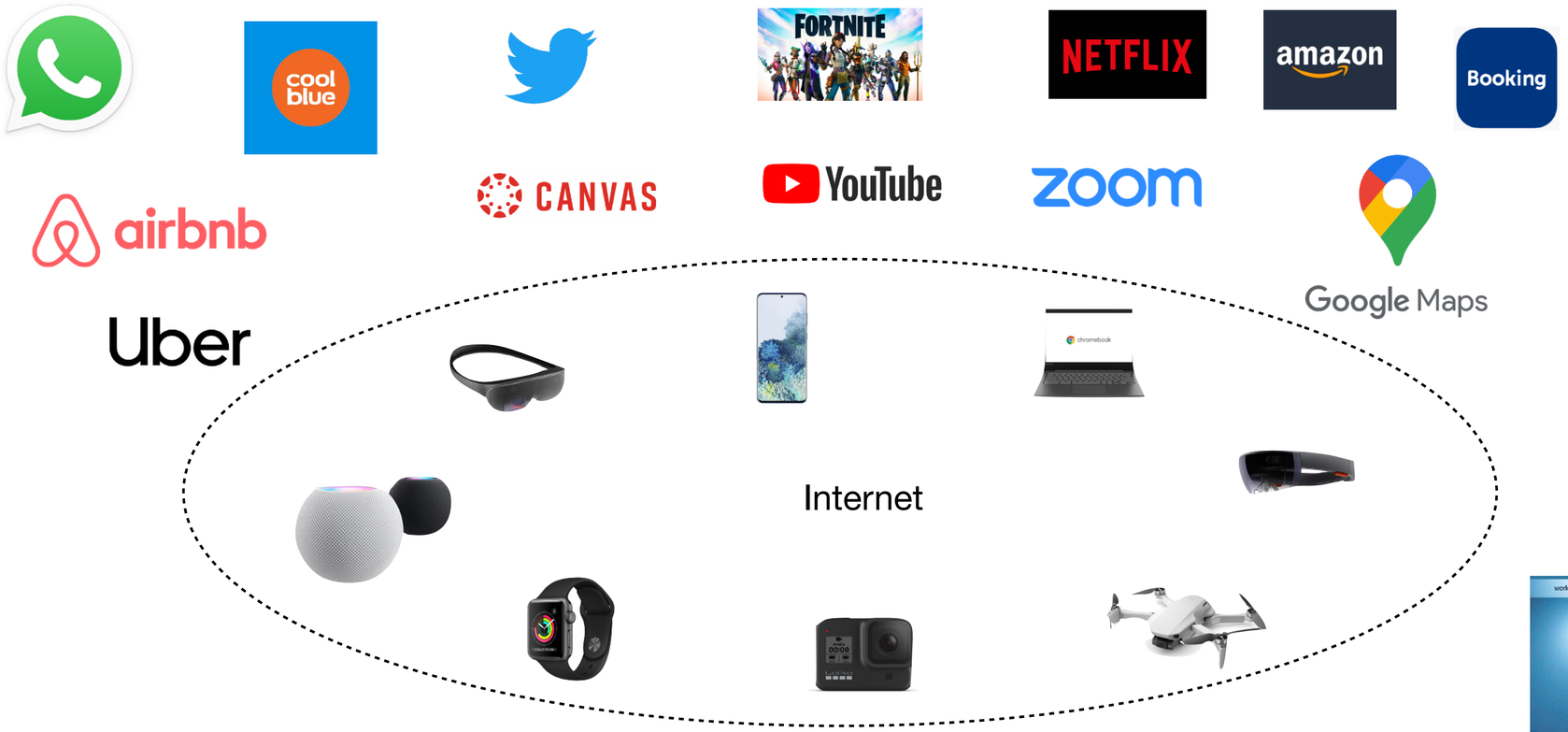
Questions?

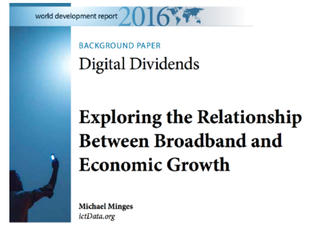
Why this course?



For the love of terrible jobs…

Internet: a fundamental societal infrastructure





The Internet is behind most of our daily activities nowadays,
and it has a huge impact on our society!

History of Internet: visions at that time

Memex

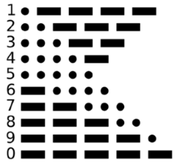
- Vannevar Bush, “As we may think”, 1945
- A hypothetical proto-hypertext system in which individuals would compress and store all of their books, records, and communications

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

https://www.internetsociety.org/internet/history-internet/brief-history-internet/

History of Internet: telephone network



1838: Morse code

1927: first telephone service (US-UK)

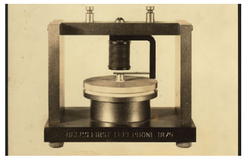
1962: satellite in telecommunications

1981: first commercial cellular network (1G)

2G, 3G, 4G, 5G…

1672: first acoustic telephone

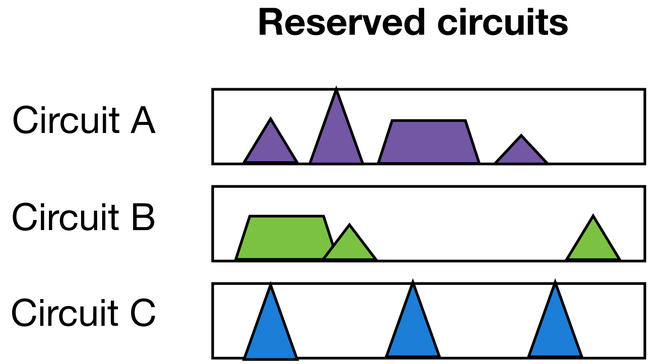
1876: first telephone by Bell



1934: first radio telephone call

1973: first cellular mobile call

Circuit switching



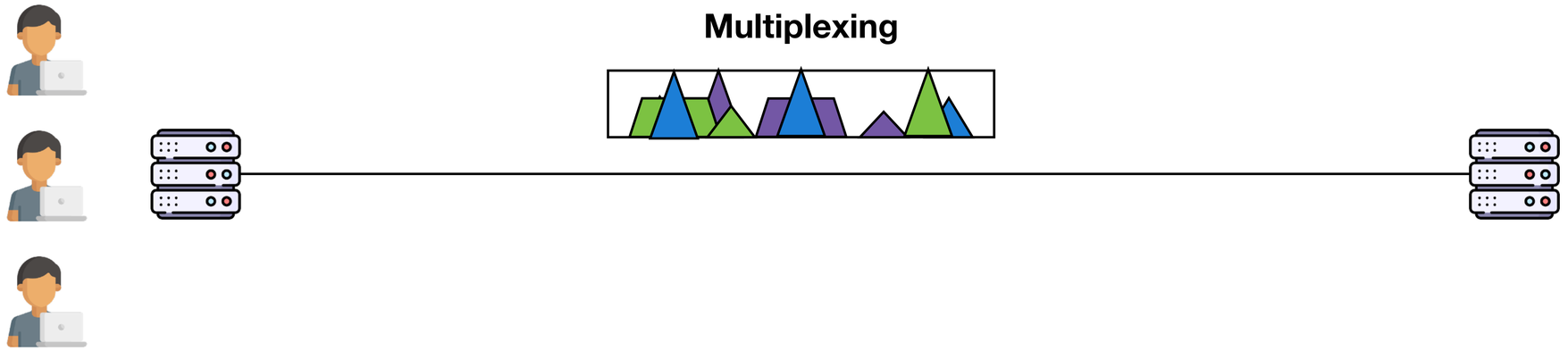
- Physical channel carrying stream of data from source to destination
- Three phases: setup, data transfer, tear-down
- Data transfer involves no routing

Packet switching

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

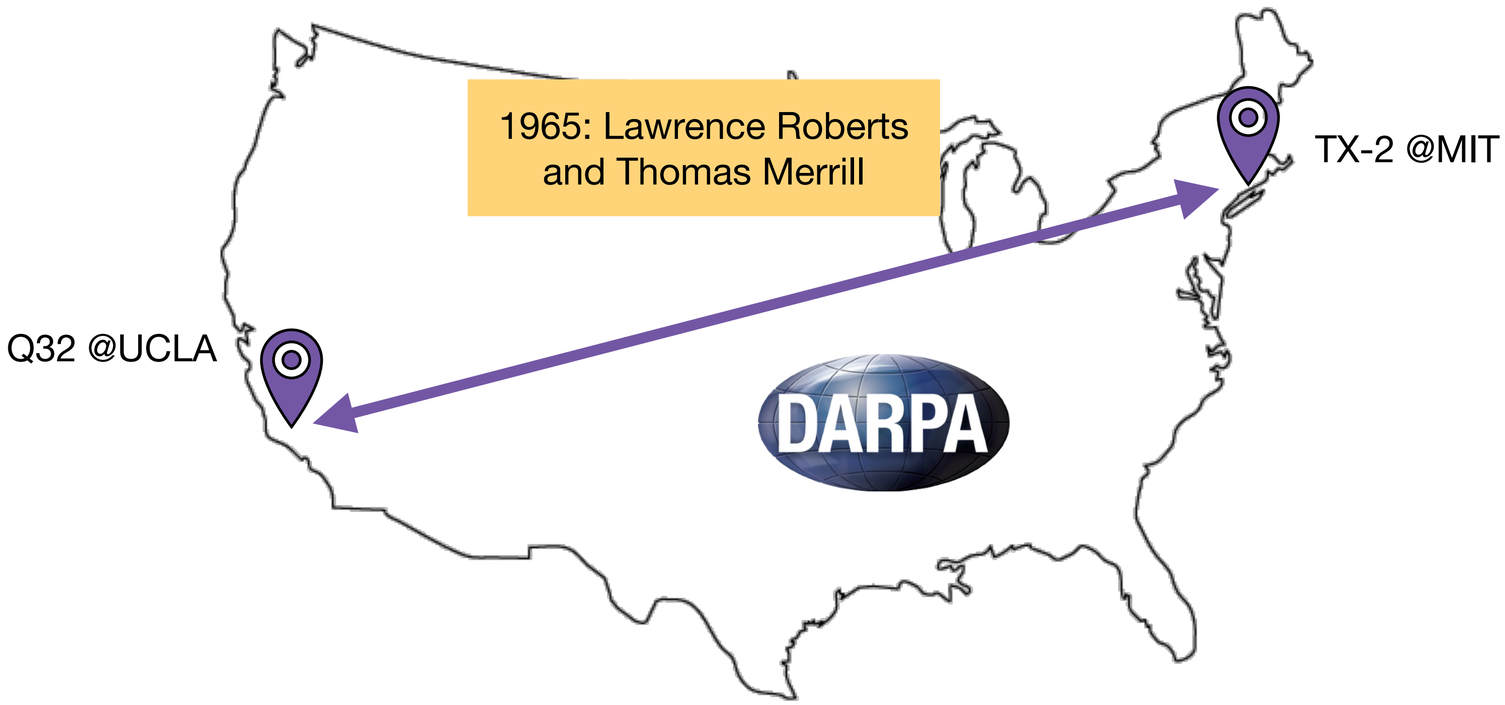
Concurrent work from Paul Baran (RAND), Donald Davies (National Physical Labratories, UK)

1960s: Time-sharing operating systems began to emerge



- 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: first wide area network



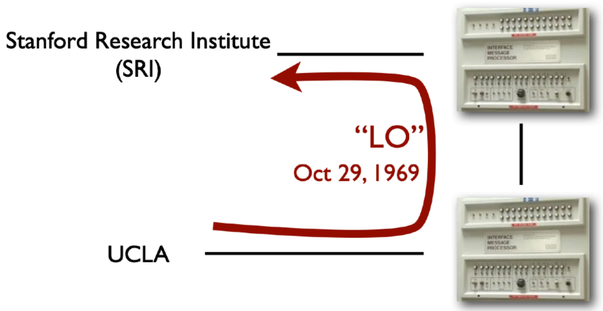
Connection is through the telephone line – it works, but it is inefficient and expensive – confirming the motivation for packet switching

History of Internet: ARPANET

1967 Larry 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

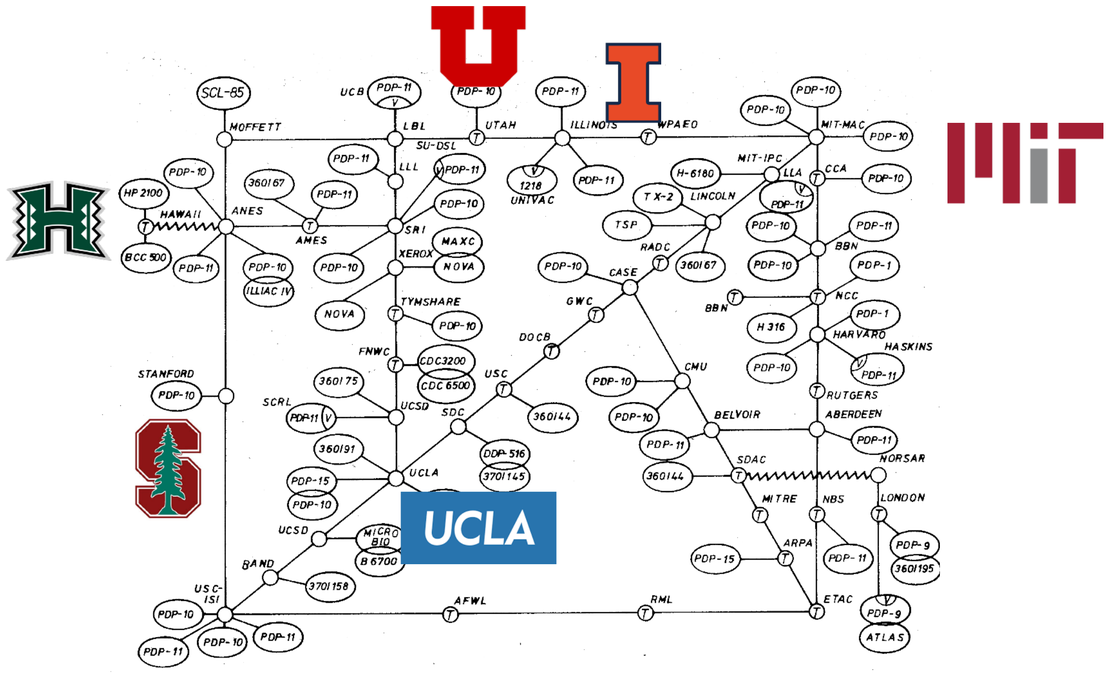
1971 Telnet, FTP

1972 Email (Ray Tomlinson, BBN)

1973 USENET

GOOD TO KNOW

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



History of Internet: network of networks

In the mean time 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 TRANSACTIONS ON COMMUNICATIONS, VOL. COM-22, NO. 5, MAY 1974

A Protocol for Packet Network Intercommunication

VINTON G. CERF AND ROBERT E. KAHN, MEMBER, IEEE

***Abstract*—A protocol that supports the sharing of resources that exist in different packet switching networks is presented. The protocol provides for variation in individual network packet sizes, transmission failures, sequencing, flow control, end-to-end error checking, and the creation and destruction of logical process-to-process connections. Some implementation issues are considered, and problems such as internetwork routing, accounting, and timeouts are exposed.**

INTRODUCTION

IN THE LAST few years considerable effort has been expended on the design and implementation of packet switching networks [1]–[7],[14],[17]. A principle reason for developing such networks has been to facilitate the sharing of computer resources. A packet communication network includes a transportation mechanism for delivering data between computers or between computers and terminals. To make the data meaningful, computers and terminals share a common protocol (i.e., a set of agreed upon conventions). Several protocols have already been developed for this purpose [8]–[12],[16]. However, these protocols have addressed only the problem of computer resources called HOSTS, a set of one or more *packet switches*, and a collection of communication media that interconnect the packet switches. Within each HOST, we assume that there exist *processes* which must communicate with processes in their own or other HOSTS. Any current definition of a process will be adequate for our purposes [13]. These processes are generally the ultimate source and destination of data in the network. Typically, within an individual network, there exists a protocol for communication between any source and destination process. Only the source and destination processes require knowledge of this convention for communication to take place. Processes in two distinct networks would ordinarily use different protocols for this purpose. The ensemble of packet switches and communication media is called the *packet switching subnet*. Fig. 1 illustrates these ideas.

In a typical packet switching subnet, data of a fixed maximum size are accepted from a source HOST, together with a formatted destination address which is used to

http://pbg.cs.illinois.edu/courses/cs598fa09/readings/ck74.pdf


I didn't invent
the Internet
I did not
invent the Web

History of Internet: standards

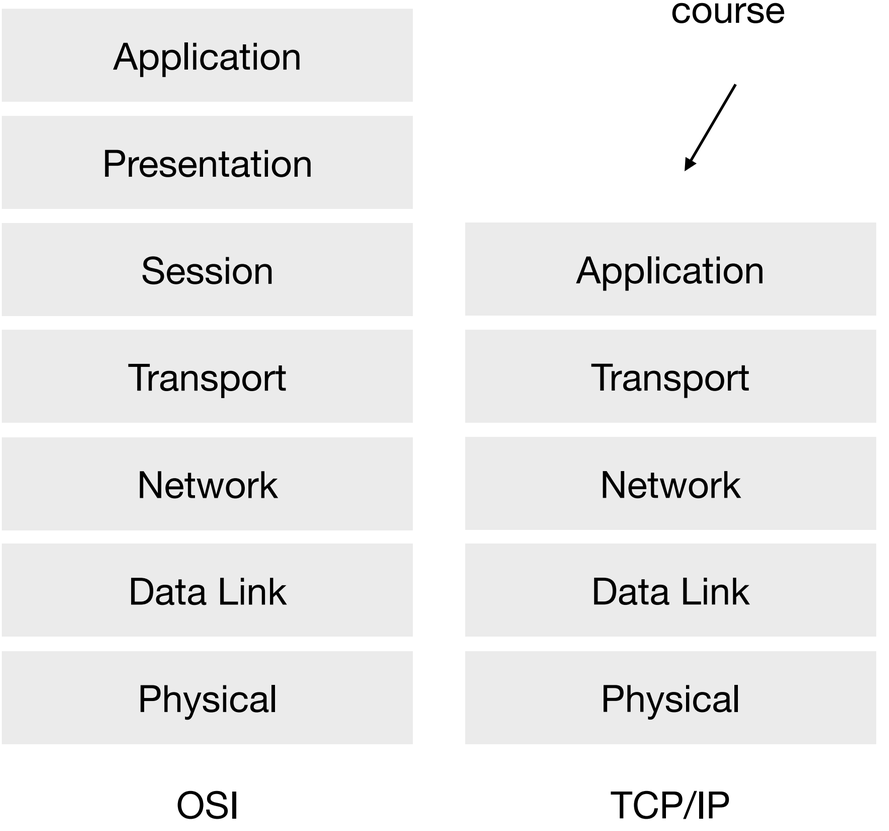
TCP/IP: interconnecting networks

- 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 to TCP/IP transition on ARPANET

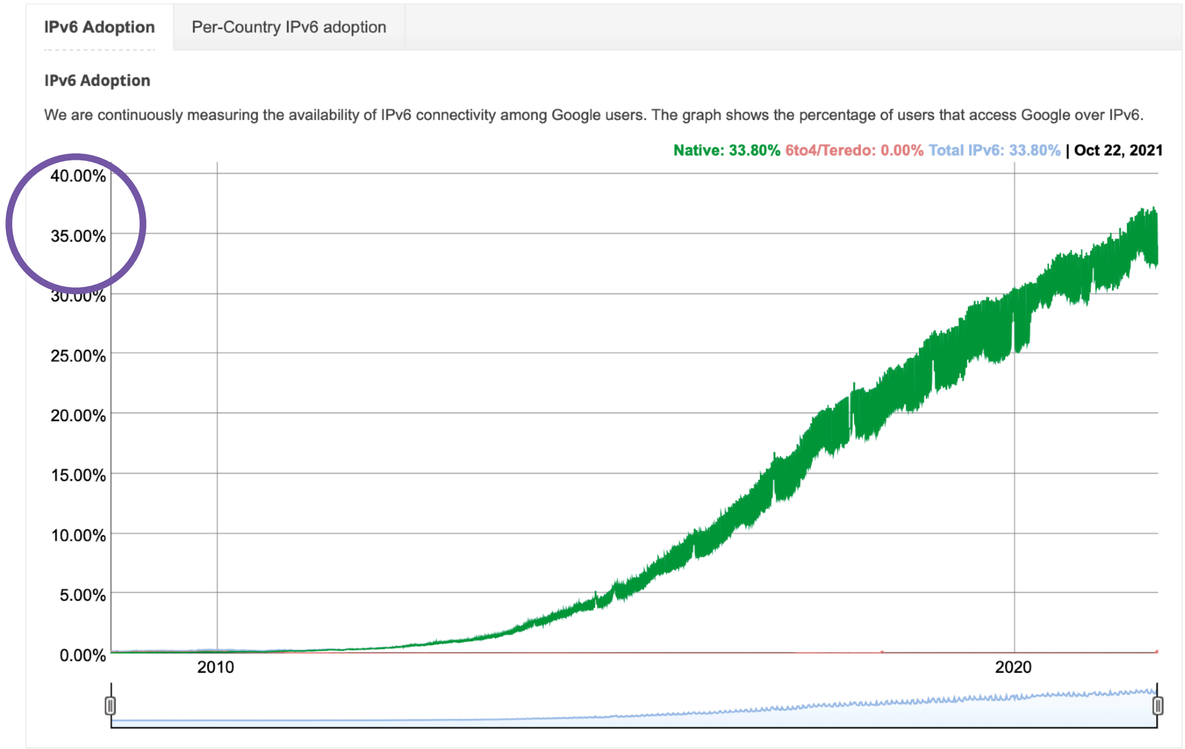
Ethernet: local area networking

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

Likely what you have learned in the “Computer Networks” course



Another flag day is almost impossible nowadays



The global IPv4 → IPv6 transition is extremely low…

https://www.google.com/intl/en/ipv6/statistics.html

History of Internet: fast development

1983 DNS developed by Jon Postel, Paul, Mockapetris (USC/ISI), Craig Partridge (BBN)

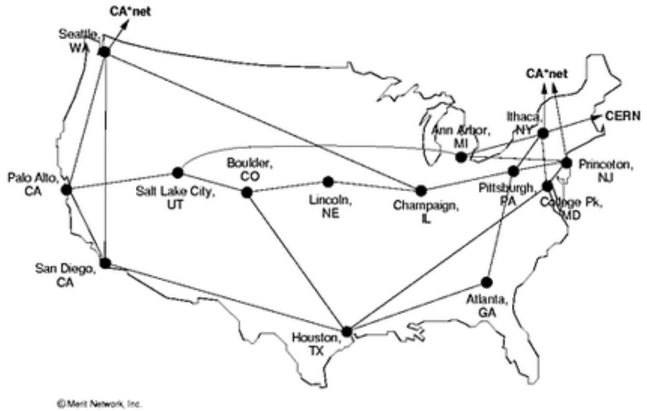
1984 Hierarchical routing: EGP, IGP (later to become eBGP and iBGP)
NSFNET for US higher education
- Served many users, not just one field
- Encouraged development of private infrastructure (e.g., backbone required to be used for research and education)
- Stimulated investment in commercial long-haul networks

1988 Morris worm – first computer worm

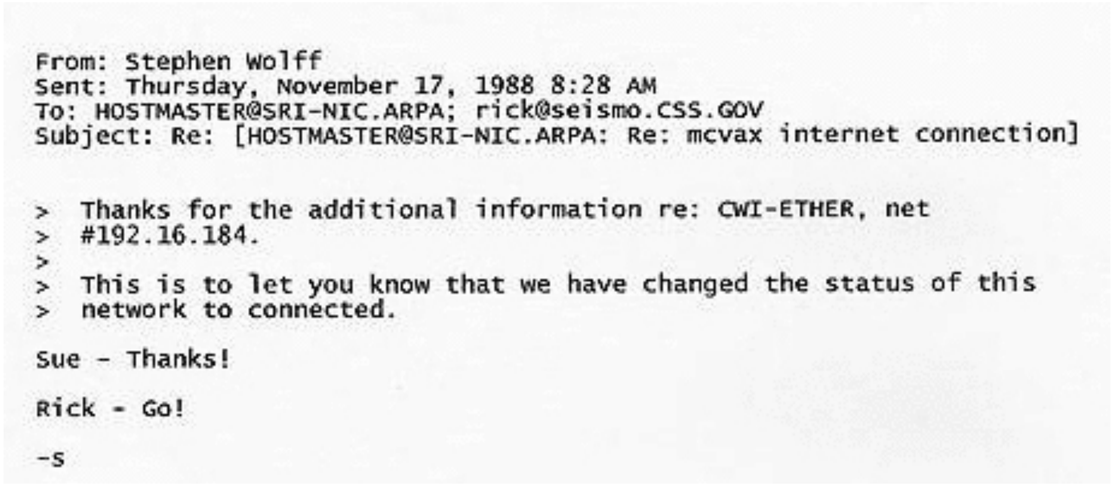
1990 ARPANET ends

1995 NSFNET decommissioned

NSFNET T1 Network 1991



Internet in The Netherlands

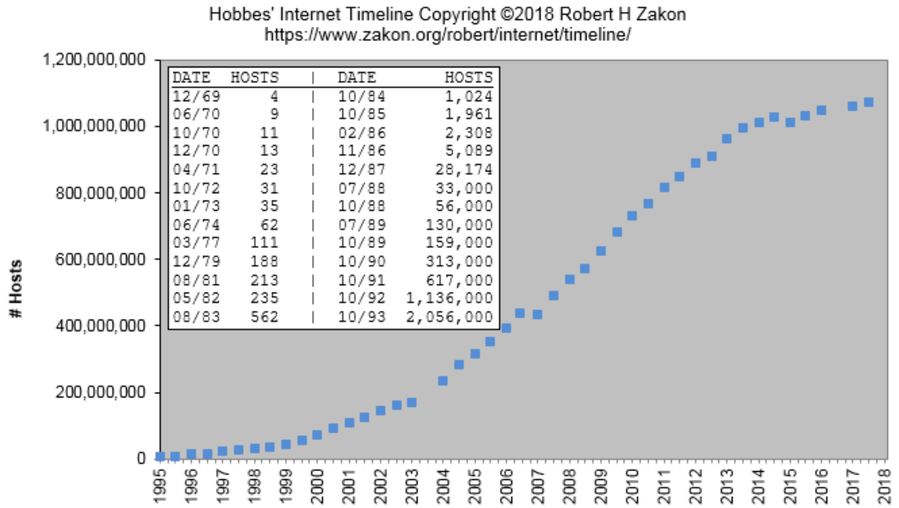


```
From: Stephen Wolff
Sent: Thursday, November 17, 1988 8:28 AM
To: HOSTMASTER@SRI-NIC.ARPA; rick@seismo.CSS.GOV
Subject: Re: [HOSTMASTER@SRI-NIC.ARPA: Re: mcvax internet connection]


>  Thanks for the additional information re: CWI-ETHER, net
>  #192.16.184.
>
>  This is to let you know that we have changed the status of this
>  network to connected.

Sue - Thanks!

Rick - Go!

-S
```

First email via the first transatlantic connection (Nov 17, 1988)

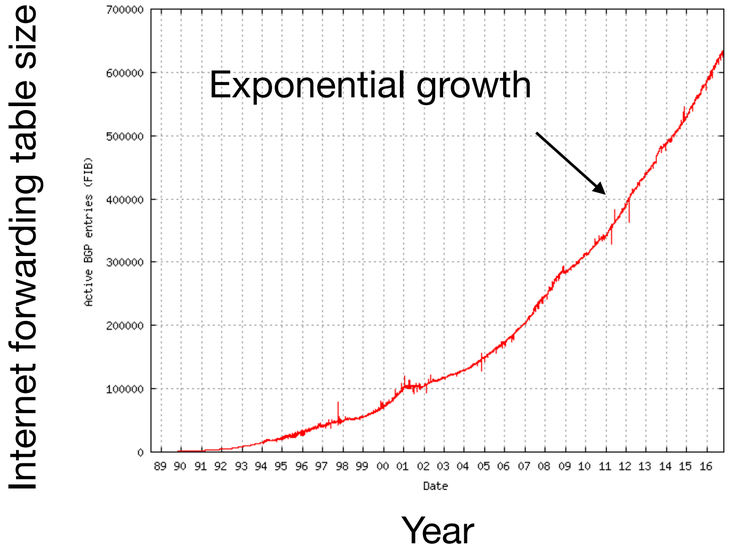
Piet Beertema (CWI): https://godfatherof.nl

Internet growth: network size



DATE	HOSTS	DATE	HOSTS
12/69	4	10/84	1,024
06/70	9	10/85	1,961
10/70	11	02/86	2,308
12/70	13	11/86	5,089
04/71	23	12/87	28,174
10/72	31	07/88	33,000
01/73	35	10/88	56,000
06/74	62	07/89	130,000
03/77	111	10/89	159,000
12/79	188	10/90	313,000
08/81	213	10/91	617,000
05/82	235	10/92	1,136,000
08/83	562	10/93	2,056,000

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?

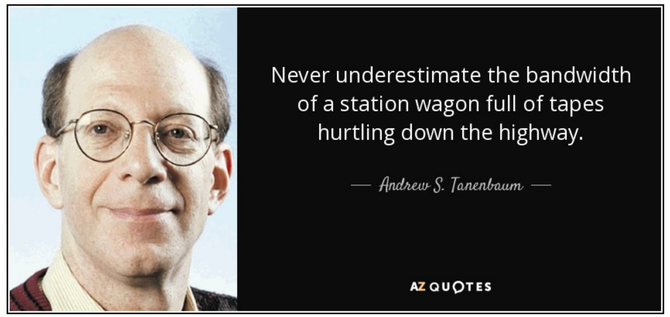
Flexibility, manageability?

Others?

Network performance

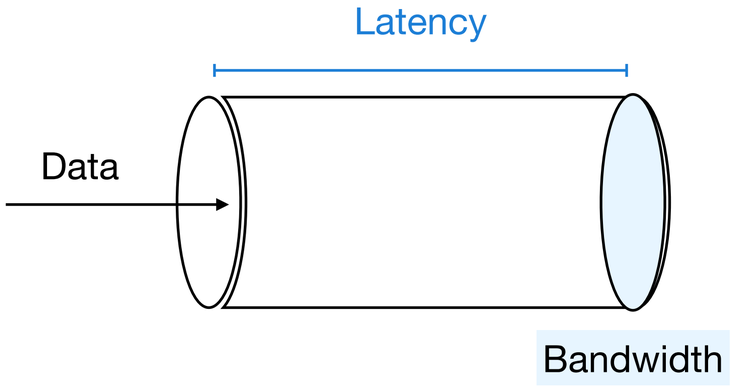
Bandwidth

- Definition: maximum rate of data transfer across a given network path
- "*Never underestimate the bandwidth of a station wagon full of tapes hurtling down the highway.*" [1]



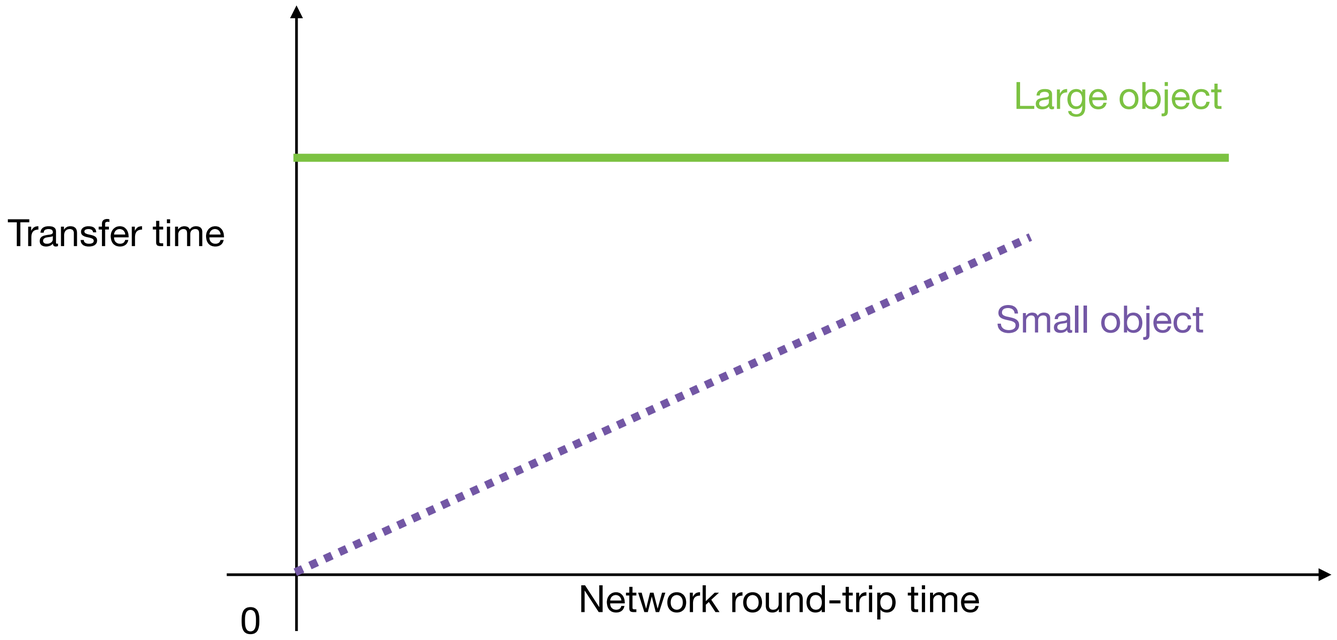
Latency

- Definition: the 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

Bandwidth vs. latency



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

With **TCP**: 100KB = 1460B * (0 + 2 + 4 + 8 + 16 + 32 + 6) → 7ms in the best case, 0.0143Gbps

Do you know how we come to this?

Performance metric

Flow completion time

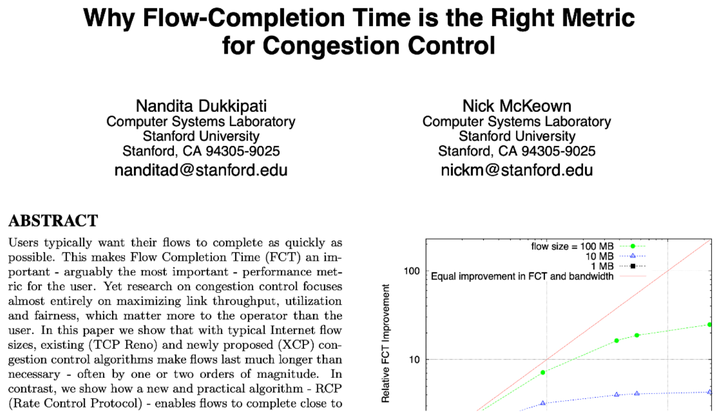
- How long does it take to complete a traffic flow?
- How long does it take to complete a set of correlated flows (co-flows)?

What else?

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

Not just about performance

- But also **consistent**, **predictable** performance



Why Flow-Completion Time is the Right Metric for Congestion Control

Nandita Dukkipati
Computer Systems Laboratory
Stanford University
Stanford, CA 94305-9025
nanditad@stanford.edu

Nick McKeown
Computer Systems Laboratory
Stanford University
Stanford, CA 94305-9025
nickm@stanford.edu

ABSTRACT

Users typically want their flows to complete as quickly as possible. This makes Flow Completion Time (FCT) an important - arguably the most important - performance metric for the user. Yet research on congestion control focuses almost entirely on maximizing link throughput, utilization and fairness, which matter more to the operator than the user. In this paper we show that with typical Internet flow sizes, existing (TCP Reno) and newly proposed (XCP) congestion control algorithms make flows last much longer than necessary - often by one or two orders of magnitude. In contrast, we show how a new and practical algorithm - RCP (Rate Control Protocol) - enables flows to complete close to

ACM SIGCOMM CCR 2006

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

ACM SIGCOMM CCR 2007

“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 that needs reliability has to engineer it from scratch: programming burden, bugs…

What if the network layer tried to provide reliable delivery?

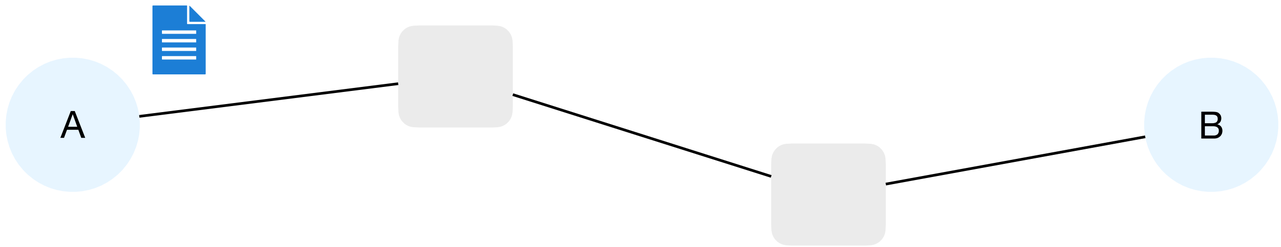
Reliable (or **unreliable**) transport	**Reliable** ~~(or unreliable)~~ transport
built on…	built on…
Best-effort global packet delivery	**Reliable** global packet delivery

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?



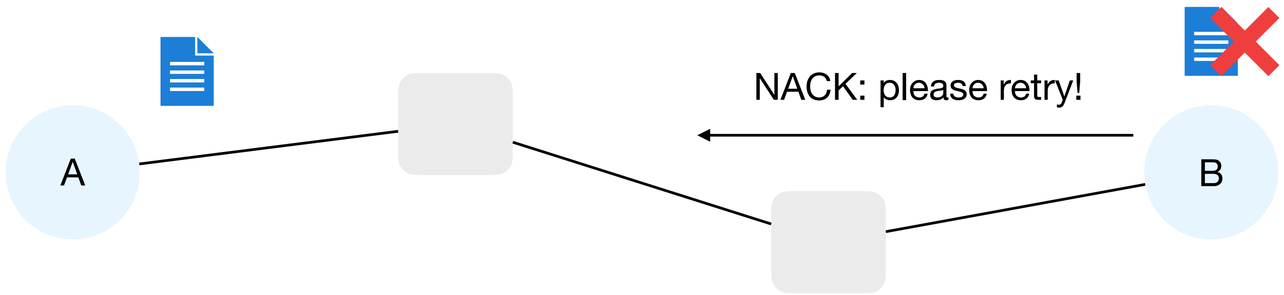
Check reliability at every step (on the network layer).

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."

Examples?

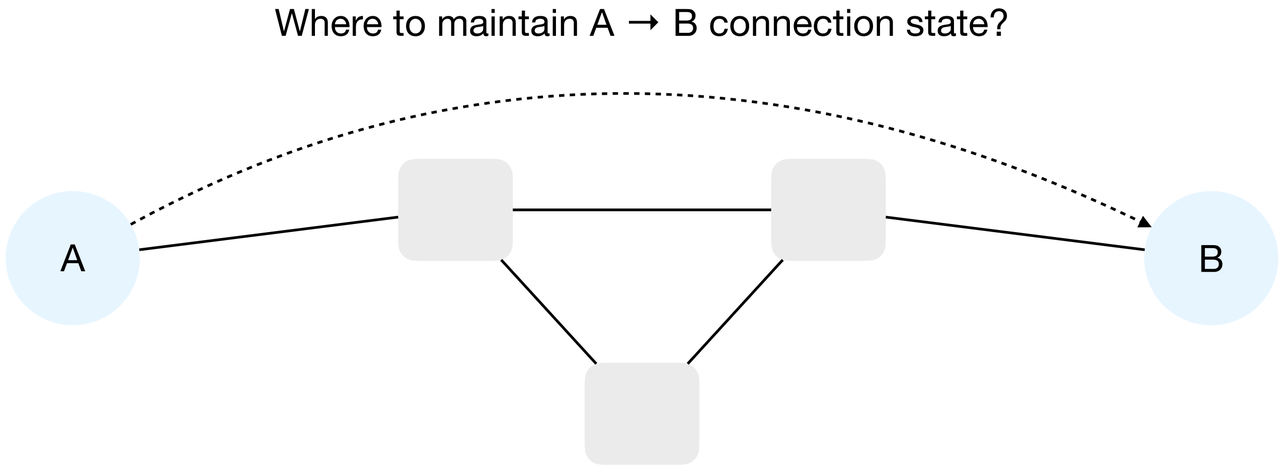


Allow unreliable steps (network layer is best-effort).
B checks correctness. On failure, B tells A to retry.

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

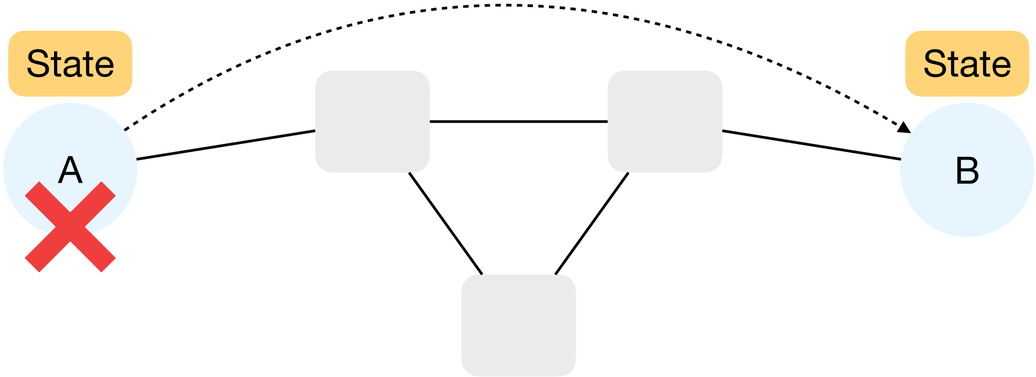
The fate-sharing principle

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.



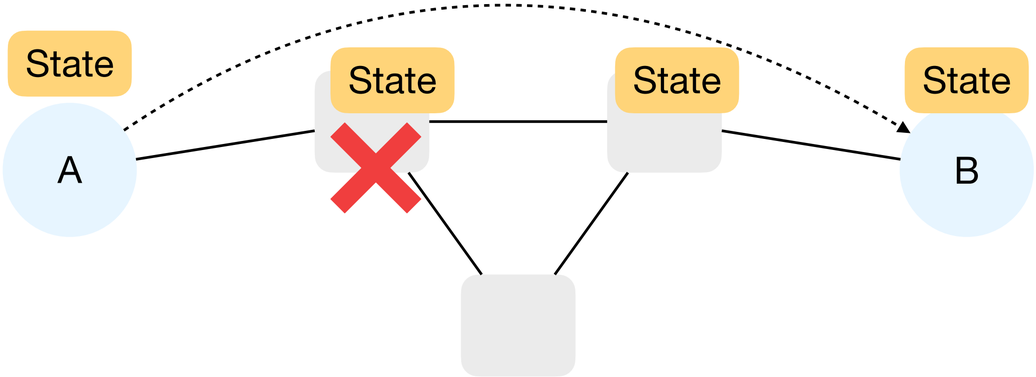
The fate-sharing principle

Keep state on end hosts



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

Keep state on network devices



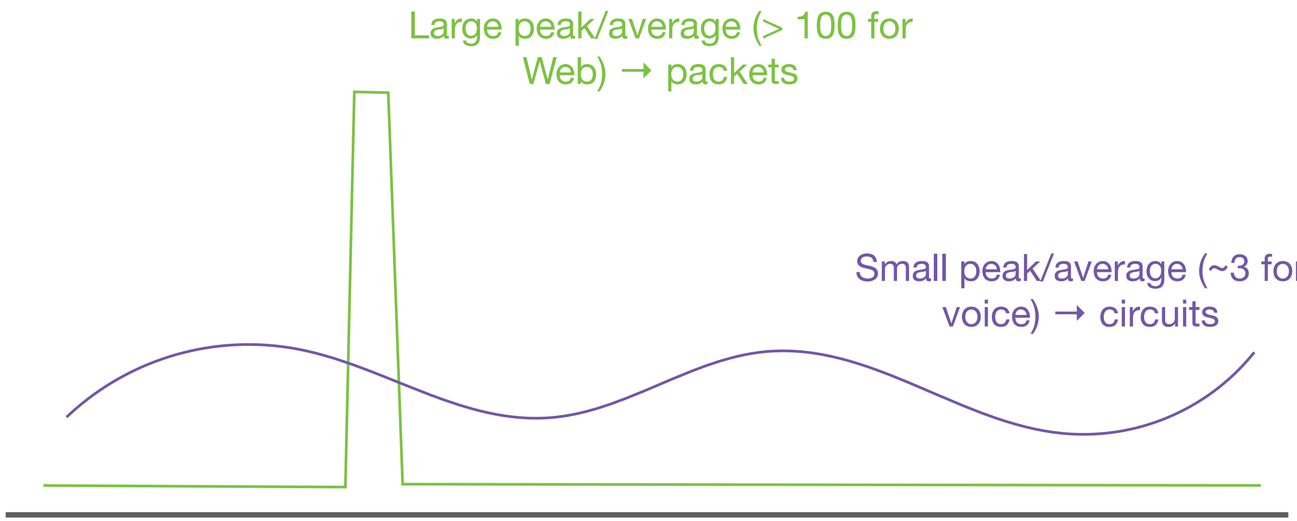
Failures of network devices require the state to be cleaned up or recreated → complex consistency issue!

Packet vs. circuit switching

Circuit switching	Packet switching
Predictable performance	Efficient use of resources
Wasteful, if packet is bursty and short	Automatic, in-network rerouting on failures
Large latency for small messages, as they wait for circuits creation	Unpredictable performance
Require new circuit setup upon failure	Requires buffer management and congestion control

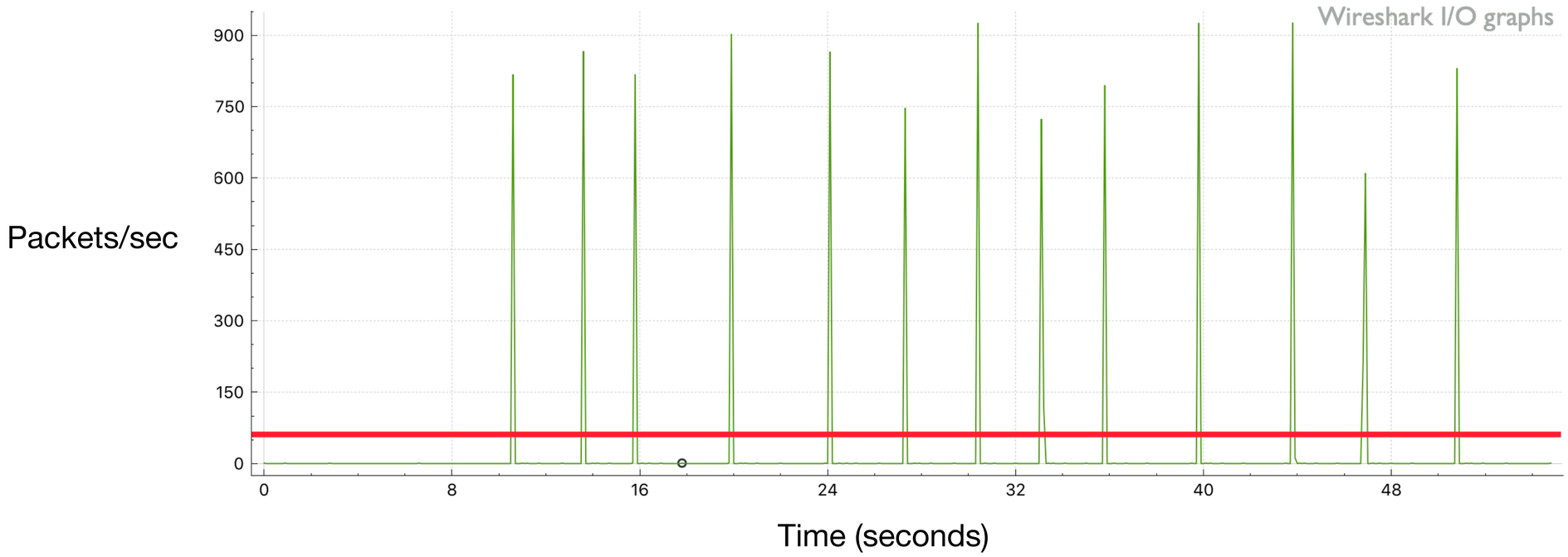
Packet switching beats circuit switching with respect to resilience and efficiency

Packet vs. circuit switching examples



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

Video streaming today



Do you know why?

Questions?

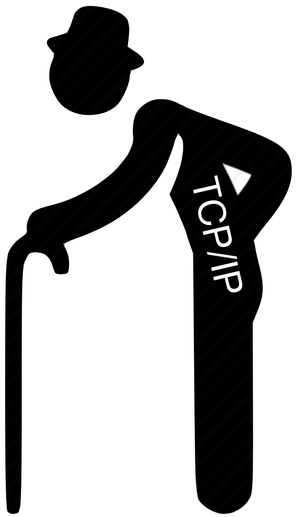
Internet: current status

Internet is there for more than 50 years

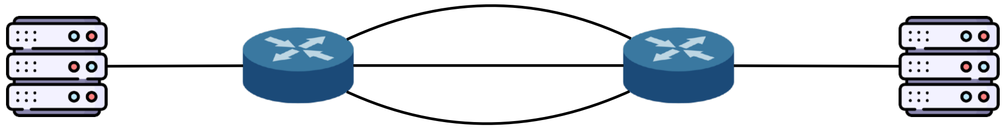
- Networks keep growing and more applications are developed
- TCP/IP is the norm: to program a network application, you simply use the socket APIs

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



Innovation in network protocols



Multipath TCP, QUIC

facebook Engineering

Open Source Platforms Infrastructure Systems Physical Infrastructure Video Engineering & AR/VR

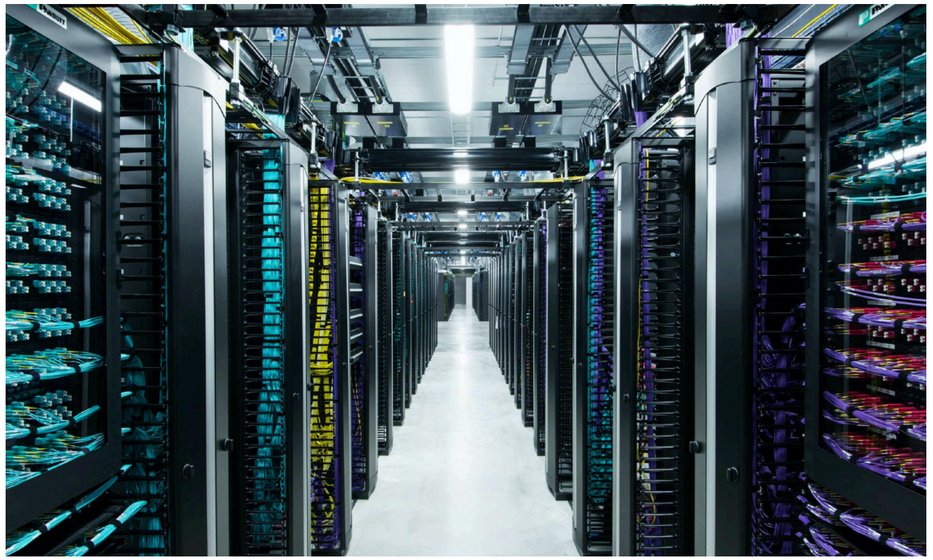
POSTED ON OCT 21, 2020 TO ANDROID, DATA INFRASTRUCTURE, IOS, NETWORKING & TRAFFIC, WEB

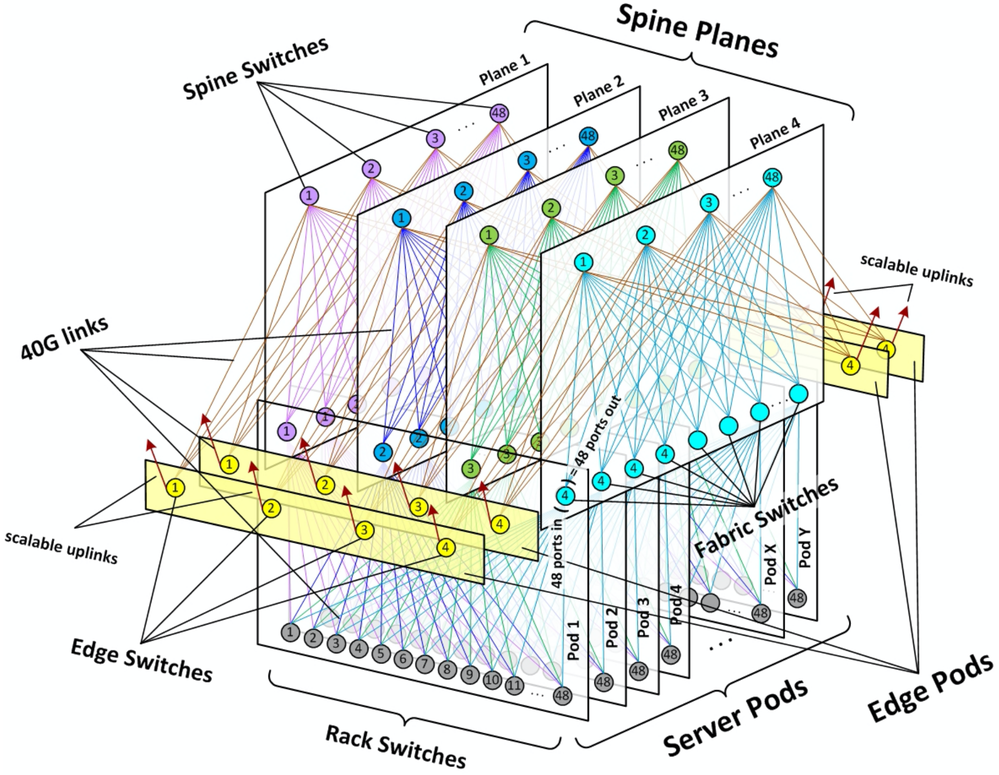
How Facebook is bringing QUIC to billions

QUIC

mvfst

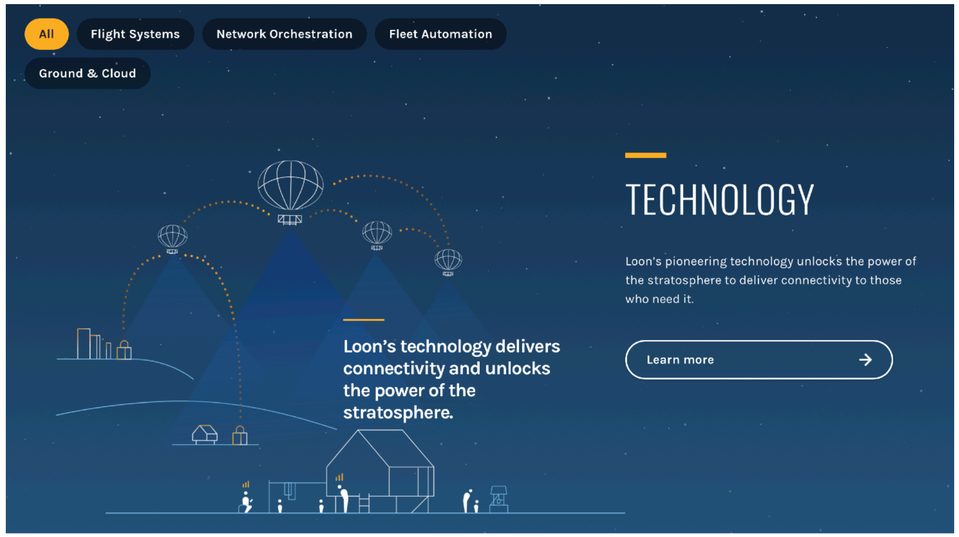
Innovations in network design



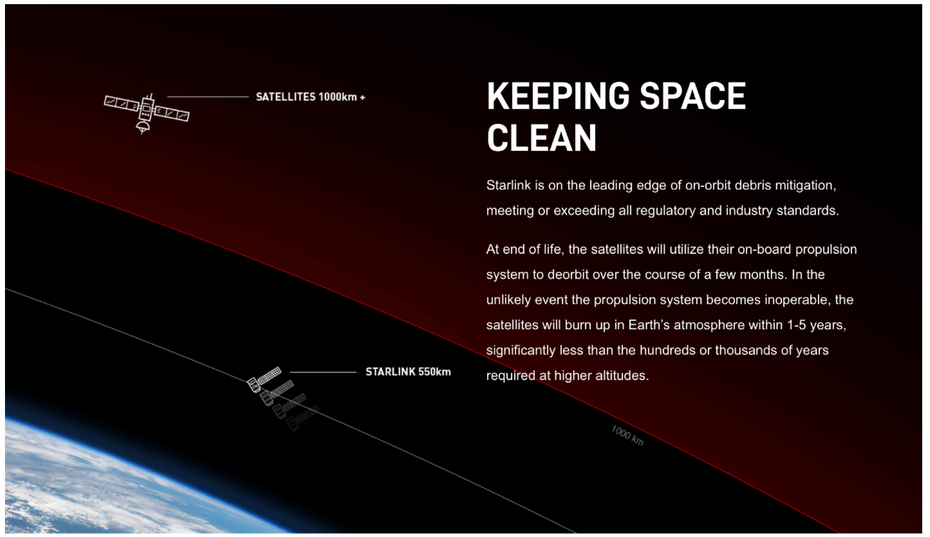


https://engineering.fb.com/production-engineering/introducing-data-center-fabric-the-next-generation-facebook-data-center-network/

Innovations in network design

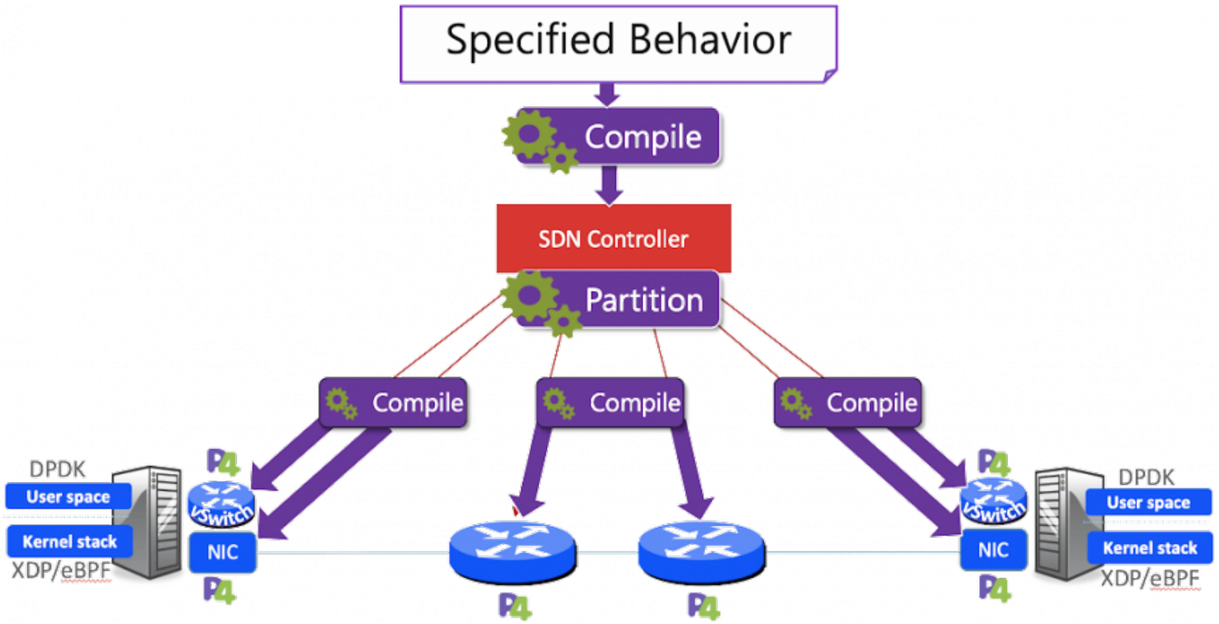


https://loon.com (discontinued)



https://www.starlink.com

Innovations in network management



https://prontoproject.org

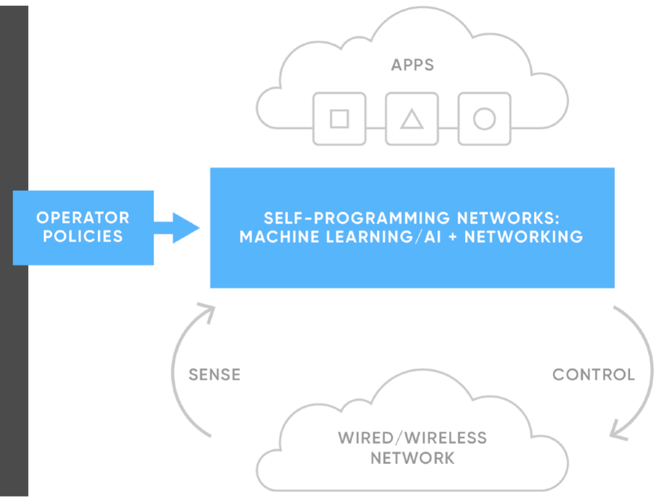
When ML meets networking

Imagine a network that can program itself
based on high level policies of its operator

A Self-Programming Network Will:

- do its own configuration, maintenance, and verification
- maximize its resource utilization
- provide differentiated and deterministic SLAs to applications and users
- support varying workloads
- provide an interactive dashboard for users and operators

all by itself with high level policies by the operator



https://platformlab.stanford.edu/platform-self-programming-networks.php

Course structure

Introduction (course logistics, Internet history)

Networking basics (main concepts and terminologies)

Data structures and algorithms (hash, Bloom filter, sketch)

Network transport (congestion control, multi-path, QUIC)

Data center networking (topologies and addressing)

Data center transport (congestion control, flow scheduling)

Software defined networking (network control/management, SDN architecture, applications)

Network automation (network configuration synthesis and verification)

Course structure

Network function virtualization (Click, ClickOS)

Programmable data plane (P4/PISA, hardware architecture)

Network monitoring (INT, sketch-based solutions)

In-network computing (NetCache, NetChain)

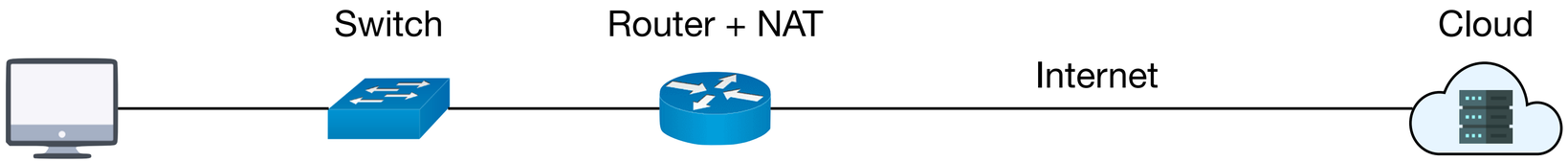
Machine learning for networking (Pensieve, NeuroCut)

Reserved (guest lecture or course summary)

Exam

Next lecture: networking basics

What happens when you visit Google in your browser?



https://www.google.com

DNS, TCP, NAT, IP routing, ARP, Ethernet