CSCI 1680: Computer Networks

Nick DeMarinis

https://brown-csci1680.github.io

For anonymous questions & feedback during lecture:
https://feedback.cs1680.systems/main

Based partly on lecture notes by Rodrigo Fonseca, David Mazières, Phil Levis, John Jannotti, Peterson & Davie

Explore new and existing community resources and mental health support:

BROWN

The Brown community is resilient, caring and strong.

We are ever true.

CSCI 1680: Computer Networks

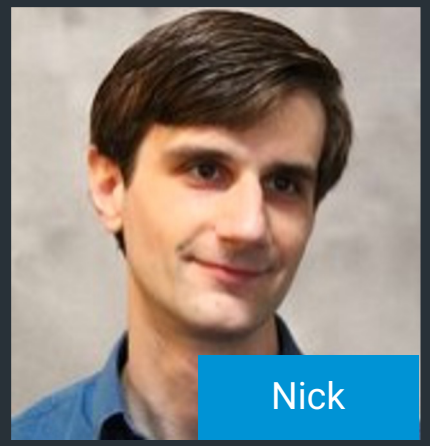
Nick DeMarinis

https://brown-csci1680.github.io

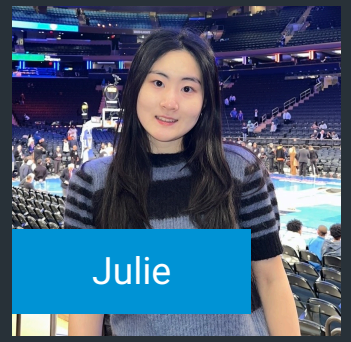
For anonymous questions & feedback during lecture:
https://feedback.cs1680.systems/main

Based partly on lecture notes by Rodrigo Fonseca, David Mazières, Phil Levis, John Jannotti, Peterson & Davie

Cast


Nick

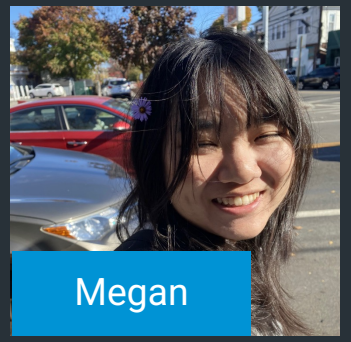

Kazuya


Julie


Alex C.

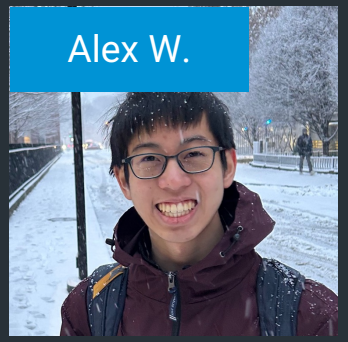


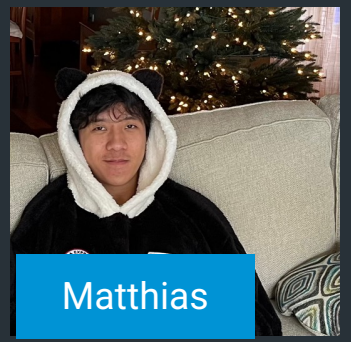
Roberto


Megan


Eddie


Praccho


Alex W.

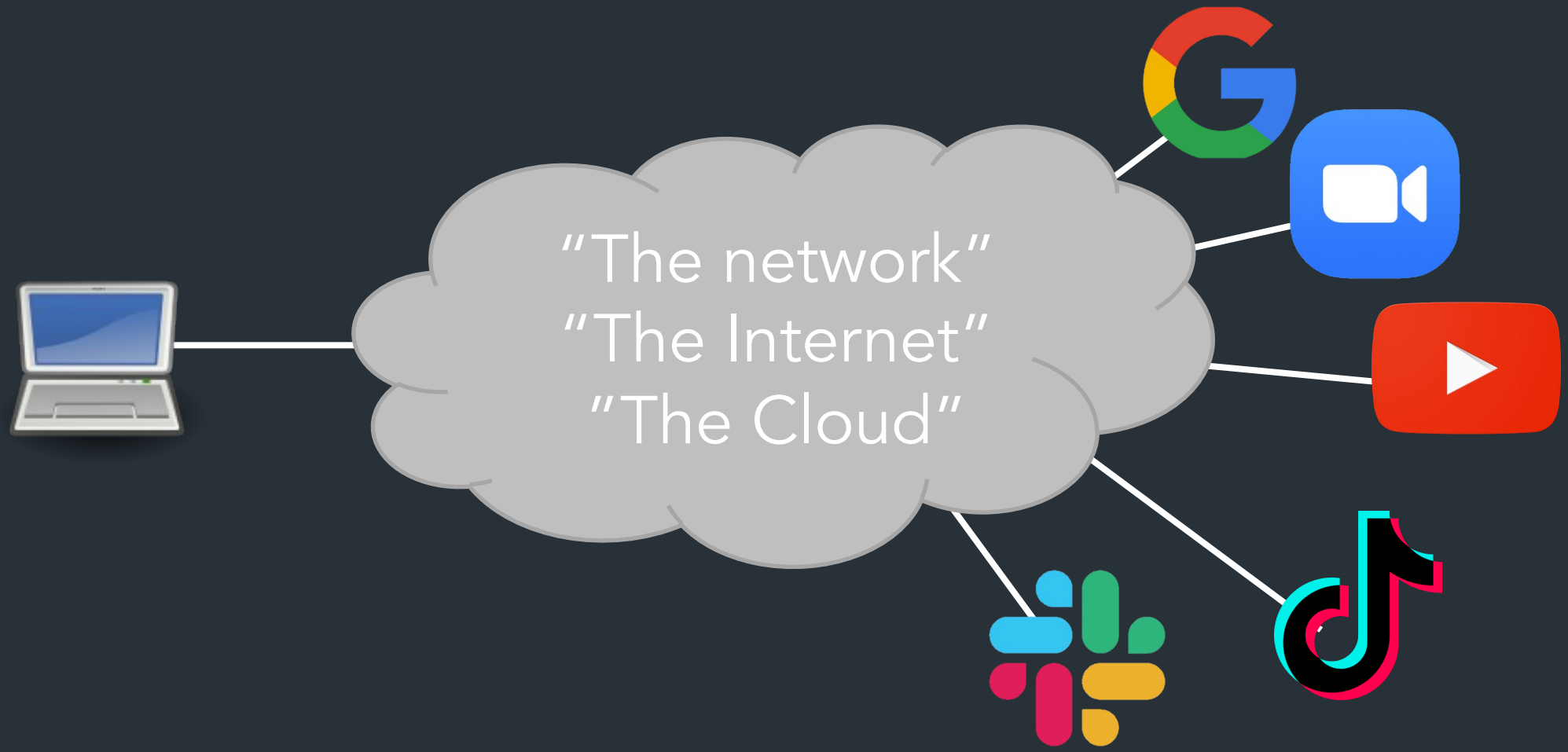

Matthias

Why are we here?

You (the user)

Applications

Why are we here?



You (the user)

Applications

Why should you care?

Networks have mostly disappeared...

…by being everywhere!

Why should you care?

Networks have mostly disappeared...

…by being everywhere!

Almost all applications are cloud-based => new challenges in system and software design

- How do we build *robust* networked applications?
- How does the “quality” of network connectivity affect users?
- What to do when networks fail?

Networks, in general

What is a network?

- System of lines/channels that interconnect
- *E.g.*, railroad, highway, plumbing, postal, telephone, social, computer

Networks, in general

What is a network?

- System of lines/channels that interconnect
- *E.g.*, railroad, highway, plumbing, postal, telephone, social, computer

A *computer* network…

- Moves *information*
- Nodes: general-purpose computers
- Links: wires, fiber optics, EM spectrum…

Why are computer networks cooler?

- Most nodes are general-purpose computers
- Very easy to innovate and develop new uses of the network: *you* can program the nodes

Why are computer networks cooler?

- Most nodes are general-purpose computers
- Very easy to innovate and develop new uses of the network: *you* can program the nodes

A landline phone

Why are computer networks cooler?

- Most nodes are general-purpose computers
- Very easy to innovate and develop new uses of the network: *you* can program the nodes

A landline phone

First "Internet" router (~1969)

The first webserver (1993)

Why are computer networks cooler?

- Most nodes are general-purpose computers
- Very easy to innovate and develop new uses of the network: *you* can program the nodes

A landline phone

First "Internet" router (~1969)

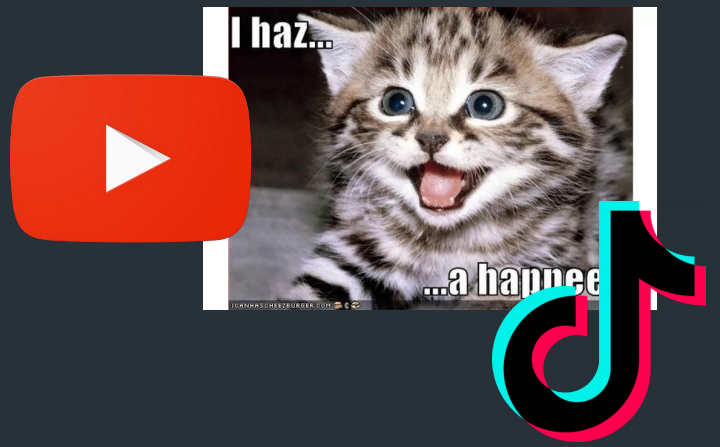
The first webserver (1993)

iPhone 1 (2007)

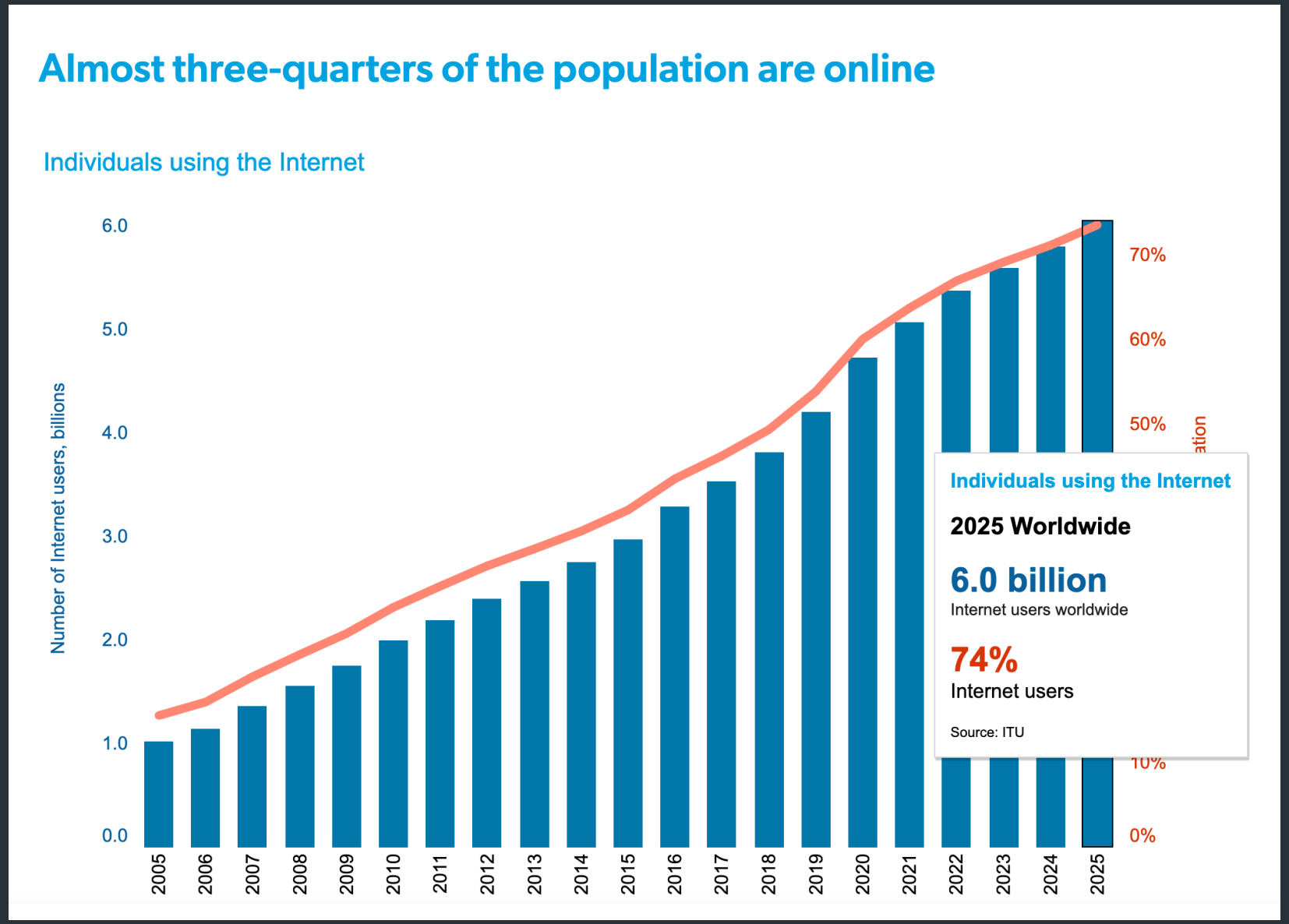
Networking => Innovation

- WhatsApp: 1B monthly active users in 7 years, now over 3B
  - 1B users => 57 engineers

- Uber, etc.: global dispatch service, enabled by smartphones
- Online communities, memes, video platforms, …

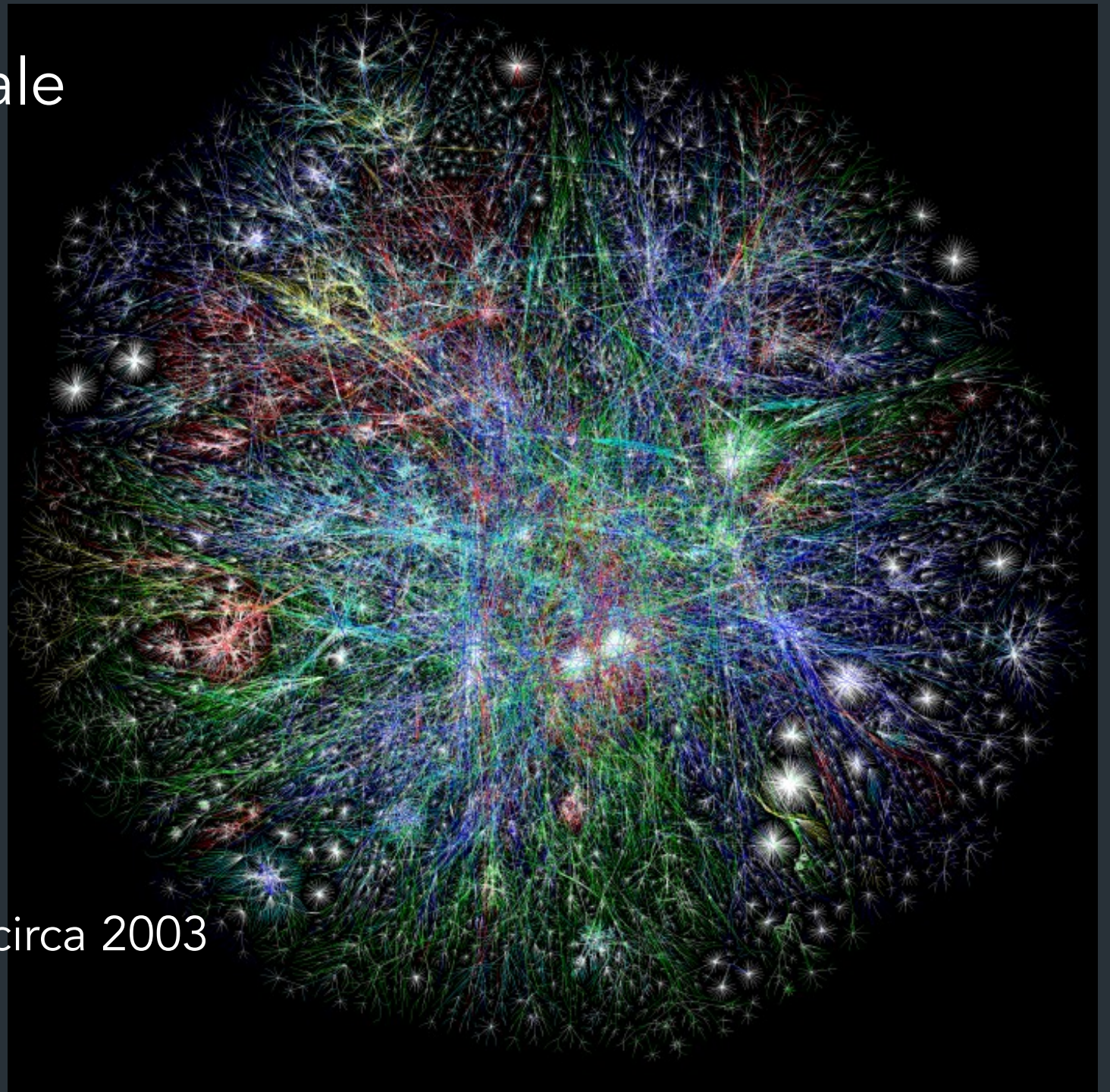


Challenge: Scale



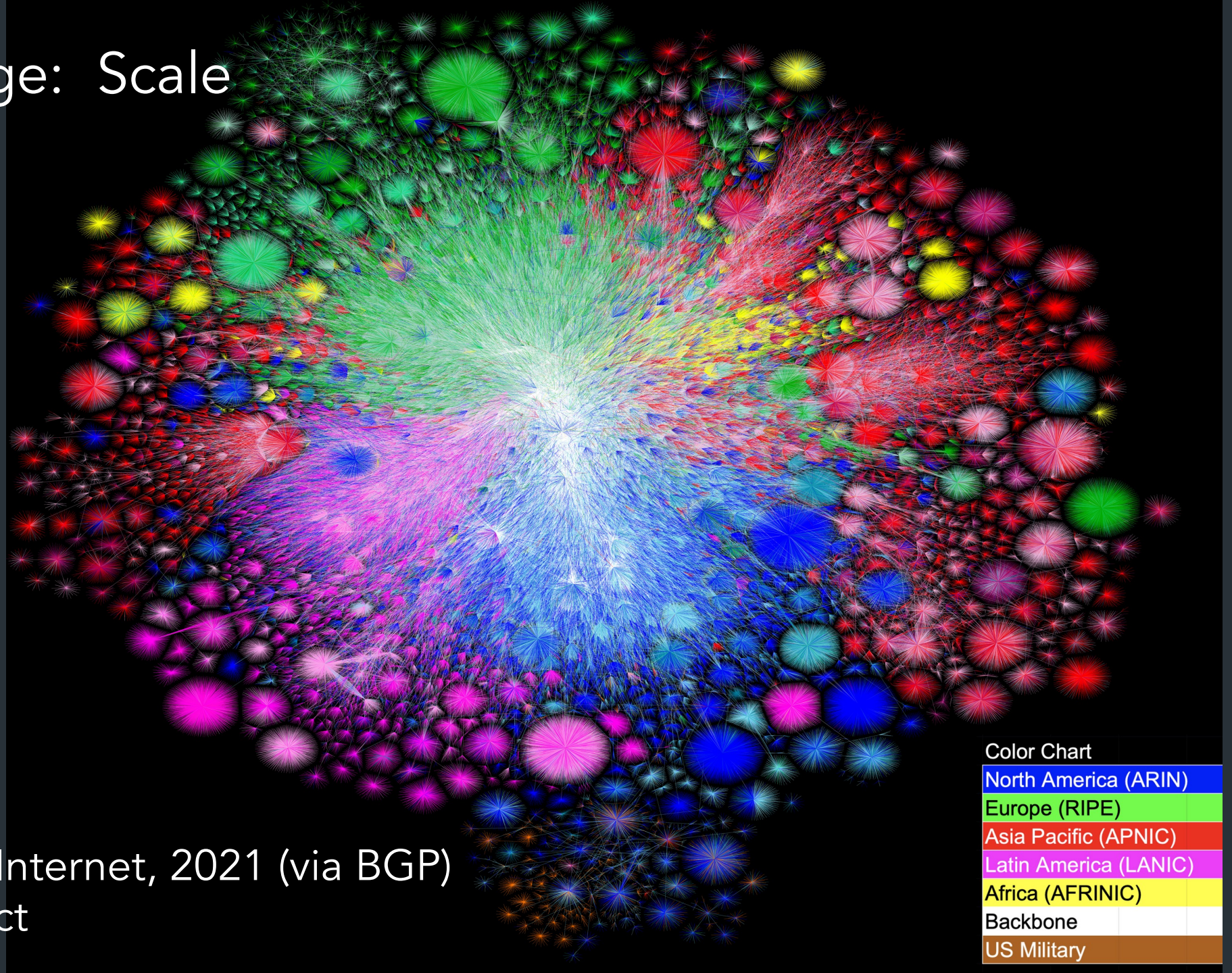
Source: [ITU Facts and Figures 2025]

Challenge: Scale



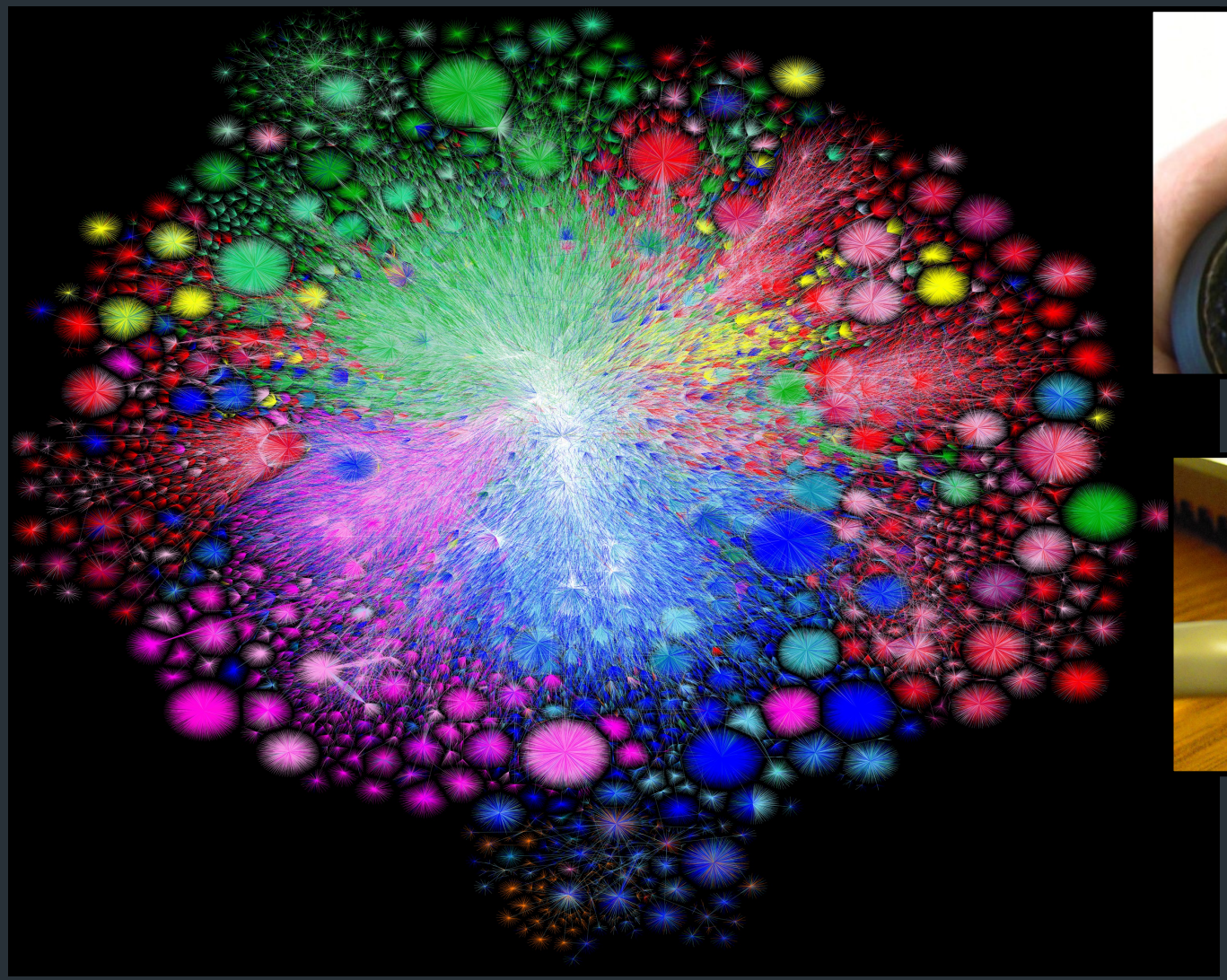
Map of the Internet, circa 2003
(via Traceroute)
OPTE project

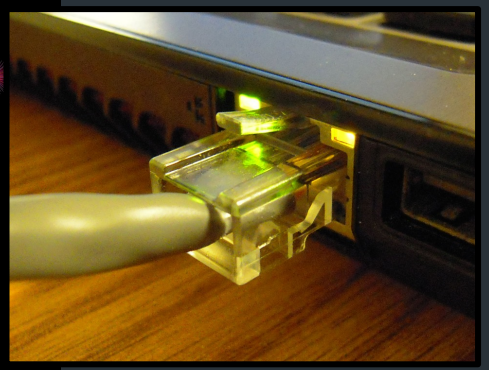
Challenge: Scale

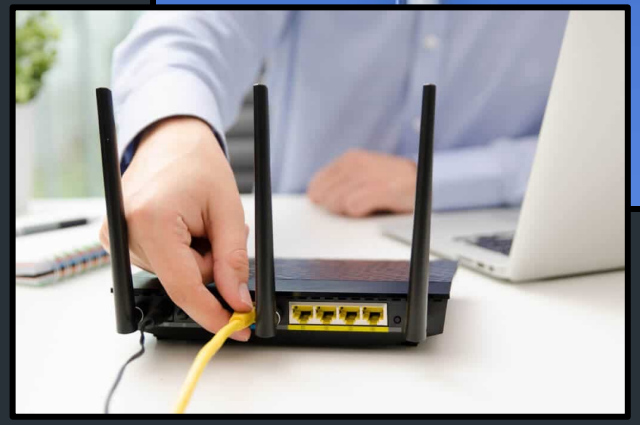


Map of the Internet, 2021 (via BGP)
OPTE project

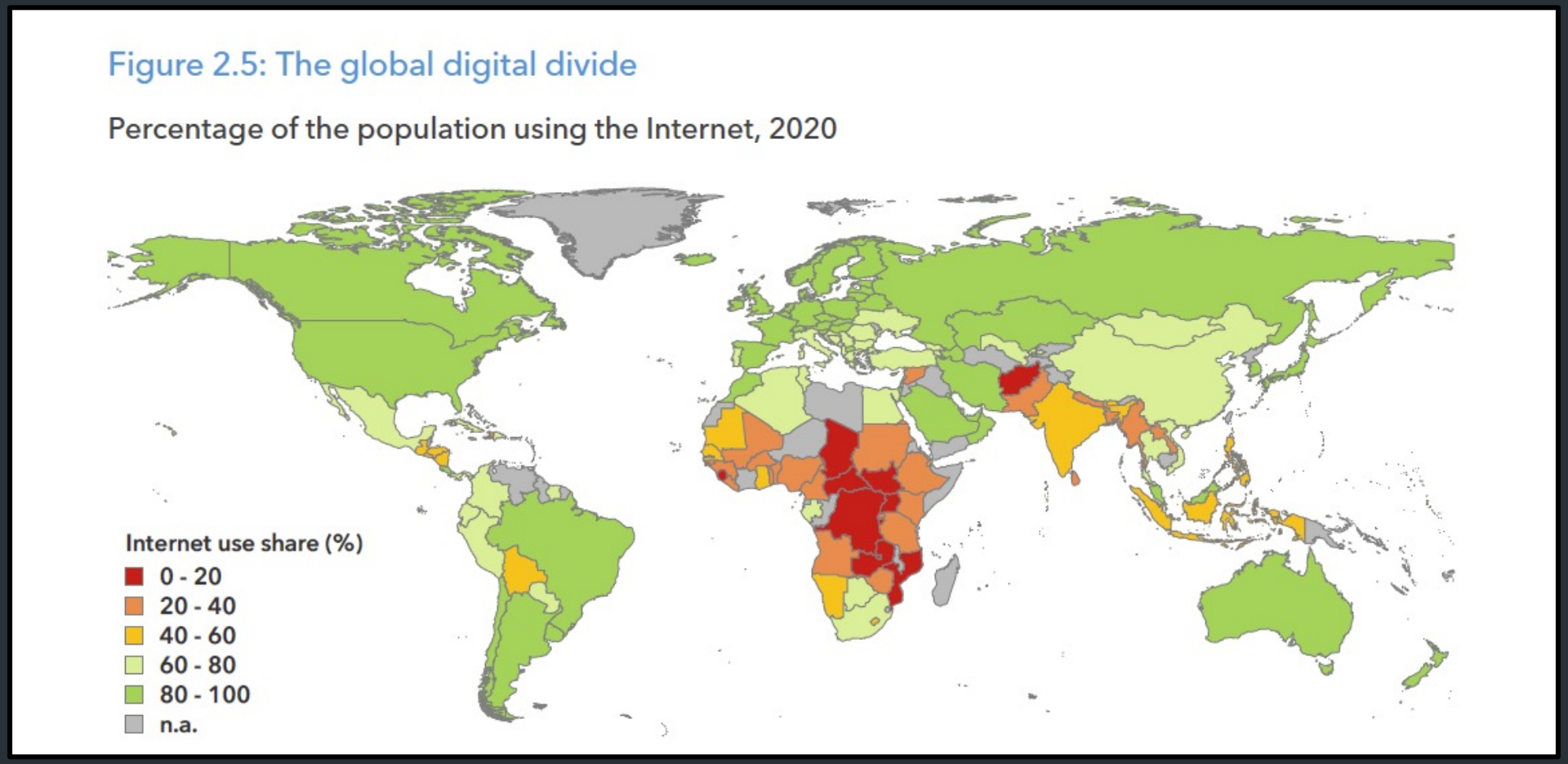
Challenge: Heterogeneity







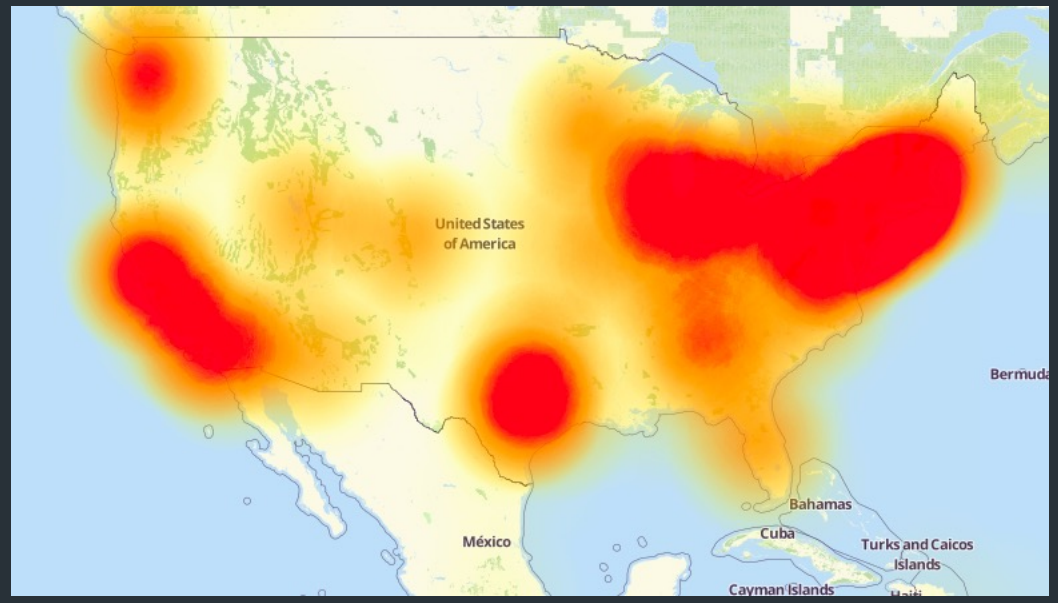
Challenges: Access



Challenge: Security

Mirai Botnet (2016)

- Vulnerable DVRs, Home Routers, Cameras disrupted Dyn
  - DNS provider for Twitter, Netflix, Reddit, others
- Largest denial-of-service (DDoS) attack at the time, over 1TBit/s



Challenges: Politics and Oversight

DDoS attack that disrupted internet was largest of its kind in history, experts say

Explainer

Facebook outage: what went wrong and why did it take so long to fix after social platform went down?

RYAN SINGEL SECURITY FEB 25, 2008 10:37 AM

Pakistan's Accidental YouTube Re-Routing Exposes Trust Flaw in Net

INNOVATION > CYBERSECURITY

Feds Warn iPhone And Android Users—Stop Using Your VPN

By Zak Doffman, Contributor. ⓘ Zak Doffman writes about security, surveillanc... Follow Author

TECHNOLOGY

How Was Egypt's Internet Access Shut Off?

DYLAN TWENEY BUSINESS FEB 3, 2011 9:58 AM

No Easy Fixes as Internet Runs Out of Addresses

MARVIN AMMORI OPINION JAN 18, 2014 7:00 AM

Internet Freedom Day: This Year We Go to War for Net Neutrality

How Russia Took Over Ukraine's Internet in Occupied Territories

By Adam Satariano and
Graphics by Scott Reinhard
Aug. 9, 2022

Why should you take this course?

- Networks have a huge impact on computing on society

- Continuously changing and evolving
  - Incredible complexity
  - *Facts* you learn will be out of date quickly
  - Instead: learn how to navigate the underlying *principles and abstractions*

Why should you take this course?

- Networks have a huge impact on computing on society

- Continuously changing and evolving
  - Incredible complexity
  - *Facts* you learn will be out of date quickly
  - Instead: learn how to navigate the underlying *principles and abstractions*

Networks are cool…
and you will learn to program them!

How will we do this?

Build networks from the ground up

- How to link 2 things, then ~10 things
- Then, how to link all the things (the Internet)
- Transport: how to send *messages* between *programs*
- Applications: how key protocols/apps make our modern Internet

How will we do this?

Build networks from the ground up

- How to link 2 things, then ~10 things
- Then, how to link all the things (the Internet)
- Transport: how to send *messages* between *programs*
- Applications: how key protocols/apps make our modern Internet

A few cross-cutting issues:

=> Security, access, social impact, key abstractions…

Logistics

How will we do this?

- 4 Programming Projects (65%)
- ~5 Written homeworks (35%)
- No exams!

Mechanics: Resources

Lecture slides/notes: authoritative content

Tools

- Course website (notes, handouts, guides): https://brown-csci1680.github.io
- Discussions: **EdStem**
- You are responsible for checking Ed for announcements/updates

Mechanics: Resources

Lecture slides/notes: authoritative content

Tools

- Course website (notes, handouts, guides): https://brown-csci1680.github.io
- Discussions: **EdStem**
- You are responsible for checking Ed for announcements/updates

Complete HW0 (on website) ASAP so we can add you to resources!

Prerequisites

Prereqs: CS330/CS1330, CS300/CS1310 (or equivalent)

- You should have worked with systems programming and basic OS concepts before
  - Threads, processes, Kernel vs. Userspace
  - Bits and bytes, memory management, synchronization, …

Prerequisites

Prereqs: CS330/CS1330, CS300/CS1310 (or equivalent)

- You should have worked with systems programming and basic OS concepts before
  - Threads, processes, Kernel vs. Userspace
  - Bits and bytes, memory management, synchronization, …

- Graduate students: our systems courses are intensive, make sure your background is equivalent

If you aren't sure if the course is right for you, please talk to us!

Lectures

- T/Th 9-10:20am, Salomon DECI
- Recorded and streamed live on zoom
- Lots of live demos/coding, time for discussion, etc.

Lectures

- T/Th 9-10:20am, Salomon DECI
- Recorded and streamed live on zoom
- Lots of live demos/coding, time for discussion, etc.

I will do my best to make 9am worth it!

Mechanics: Homeworks

- Short problems based on lecture material
  - Some "on paper", some code, etc.

- Experiment with some fundamental networking tools, tutorial-style

Mechanics: Homeworks

- Short problems based on lecture material
  - Some "on paper", some code, etc.
- Experiment with some fundamental networking tools, tutorial-style

<u>How these work</u>

- Shouldn’t take a huge amount of time
- 4 or 5 homeworks total (TBA based on enrollment)
- Hopefully fun!

Mechanics: Projects

Build fundamental Internet protocols and client/server apps from the ground up

- 4 Programming projects
  - Snowcast: streaming music server
  - IP: build your own networking library
  - TCP: extending your IP
  - Final (shorter, choose-your-own)
- First project is individual, others in groups of 2

Mechanics: Projects

Build fundamental Internet protocols and client/server apps from the ground up

- 4 Programming projects
  - Snowcast: streaming music server
  - IP: build your own networking library
  - TCP: extending your IP
  - Final (shorter, choose-your-own)
- First project is individual, others in groups of 2

You get: lots of freedom to design your own system

Languages

You can use any systems language

- Go (recommended)
- C
- C++
- Rust

- You'll be making a lot of threads, manipulating bits/bytes, …

Languages

You can use any systems language

- Go (recommended)
- C
- C++
- Rust

- You'll be making a lot of threads, manipulating bits/bytes, …

We recommend Go, even if it's new to you.
We'll do a bunch of live coding in Go in lectures 2-3

How projects work

- This is where you will spend most of your time.

Learn how to design big systems… that happen to use the network

  - No stencil code! You get to design everything
  - We will provide lots of examples and meetings to help

What this means

nick: no stencil code

everyone:

The projects are a lot of work.

If you meet the prereqs, you have the skills.

But you need to be prepared for the time commitment.

The projects are a lot of work.

If you meet the prereqs, you have the skills.

But you need to be prepared for the time commitment.

How to succeed? **Start early**, take advantage of the tools and resources we provide (there's a lot!)

How to compare this class to others

CS1680 + GenAI

Goal: you will write code to learn *concepts*, both in networking and system design

While we want you to learn how to use AI, you must use it in a way that supports your learning process, rather than replacing it

CS1680 + GenAI

Goal: you will write code to learn *concepts*, both in networking and system design

While we want you to learn how to use AI, you must use it in a way that supports your learning process, rather than replacing it

Each project will have its own AI policy. Some ground rules:

- No autonomous agents
- Don't use AI on written work (not worth it, pls don't waste our time)
- You will be asked to tell us how you used AI
- Ultimately, you are responsible for the work you submit

CS1680 + GenAI

Overall, you are responsible for everything you submit:

- Larger projects are graded by code review
  => You'll need to explain and demonstrate your work for credit

- You will need to maintain and debug your code, over 8+ weeks

=> It's in your interest to make sure you understand your work!

CS1680 + GenAI

Overall, you are responsible for everything you submit:

- Larger projects are graded by code review
  => You'll need to explain and demonstrate your work for credit

- You will need to maintain and debug your code, over 8+ weeks

How we support you on projects

- Lots of posted code examples
- Live demos during class
- “Warmup” tutorials to get started with mechanics

Most of our time is spent here too

How we support you on projects

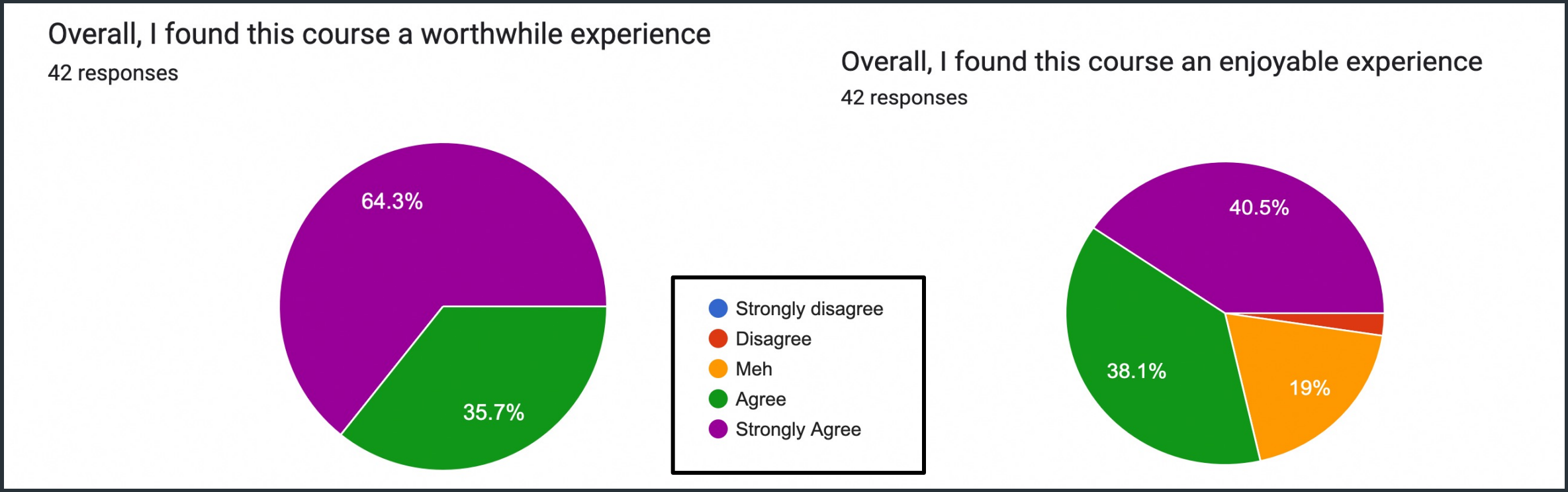
- Lots of posted code examples
- Live demos during class
- "Warmup" tutorials to get started with mechanics
- Gearups at least once per project
  - When scheduled: Thursdays 5-7pm in CIT165 (+ recorded)

Most of our time is spent here too

How we support you on projects

- Lots of posted code examples
- Live demos during class
- "Warmup" tutorials to get started with mechanics
- Gearups at least once per project
  - When scheduled: Thursdays 5-7pm in CIT165 (+recorded)
- Milestone meetings with TA to check in about design
- Grading by code review => discussion, partial credit

Most of our time is spent here too



- *"If you spend an hour or two hours on IP and TCP every day it is *very* manageable"*
- *"I am really glad that I took this course"*
- *"it's an amazing course and we had a wonderful staff to help us out"*
- *"Just take it, worth it"*
- *"Pace yourself, and don't wait until the last minute"*

Why should you listen to me?

- My background
  - Received my PhD from Brown in 2021
  - My areas: software security, networking, network security
  - My third year as Lecturer, was a long-time TA before that

- No one knows everything about networks, and I am no exception!

Brief history of this class

- Fall 2019: ~35 students

Brief history of this class

Fall 2019: ~35 students

Spring 2022

Fall 2022

Nick joins as instructor (capped at 40)
(+ demos, examples, tutorials, dev environment)

Brief history of this class

Fall 2019: ~35 students

Spring 2022

Fall 2022

Nick joins as instructor (capped at 40)
(+ demos, examples, tutorials, dev environment)

Fall 2023

Scaling up (127 students!)
(+ Gearups)

Fall 2024

Stability fixes, refinements
(+ New lecture content, HW tutorials)

Brief history of this class

Fall 2019: ~35 students

Spring 2022

Fall 2022

Nick joins as instructor (capped at 40)
(+ demos, examples, tutorials, dev environment)

Fall 2023

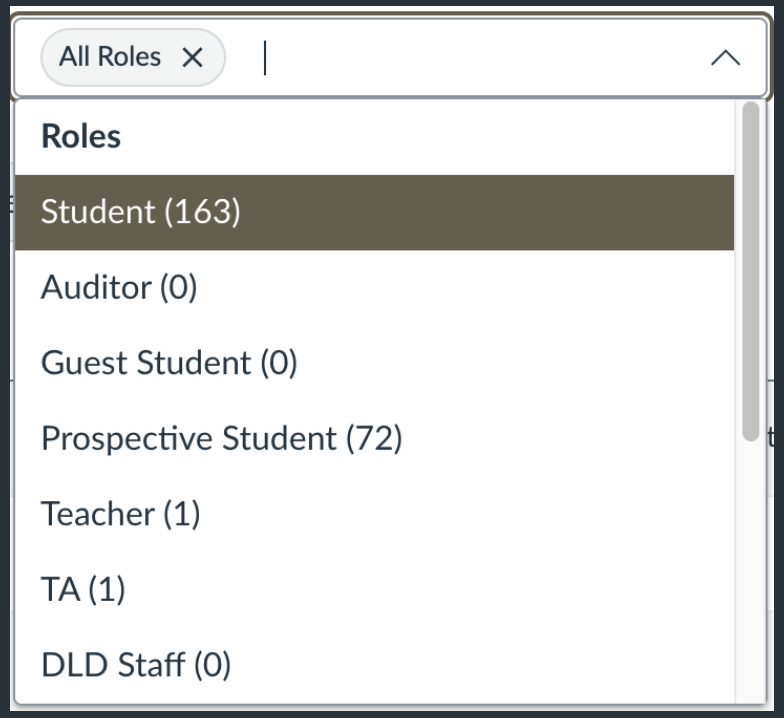
Scaling up (127 students!)
(+ Gearups)

Fall 2024

Stability fixes, refinements
(+ New lecture content, HW tutorials)

Spring 2026

You are here
(+ More lab-style HW, SRC, demos, and back to spring!)



Our course is now more stable, but we're still evolving and learning how to scale.

We may adjust course content/policies over time, paying equal attention to:

- Making sure we provide support for everyone
- Managing TA workloads

We value your feedback! (Ed, email, anonymous form, …)

Asking for help

- Online help: EdStem
- Office hours: collaborative (group, in-person)

Can help with…

- Debugging
- Project planning/design
- Concepts
- We're here for you—don't be afraid to ask!

I don't bite—Instructor's office hours are just like regular hours!

Asking for help

Collaboration: work with your peers!

- Collaboration policy on course website
- I encourage you to collaborate, *so long as the code you write down is your own*

- Your health is most important
  - If you have concerns, feel free to talk to us

We're listening!

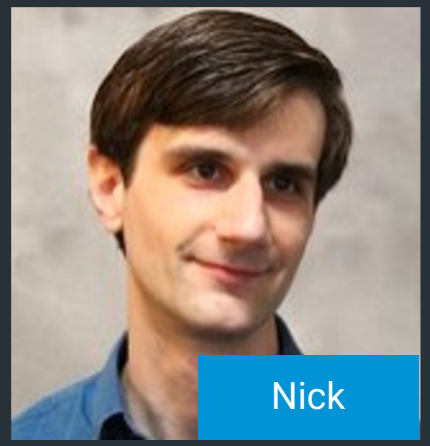
Our course is now more stable, but we're still evolving and learning how to scale.

We may adjust course content/policies/the schedule over time, paying equal attention to:

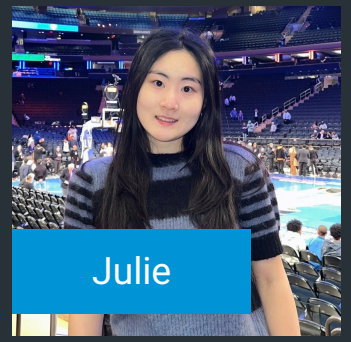
- Making sure we provide support for everyone
- Managing TA workloads

We value your feedback! (Ed, email, anonymous form, …)

Cast


Nick

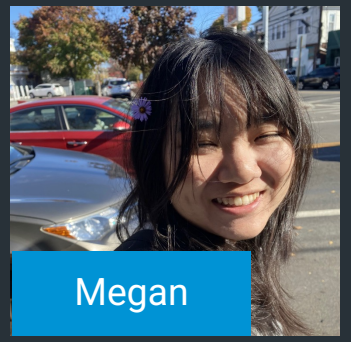

Kazuya


Julie


Alex C.

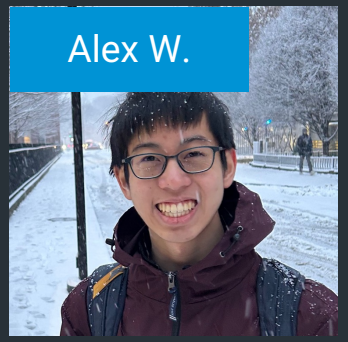


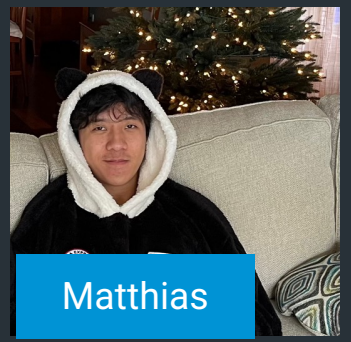
Roberto


Megan


Eddie


Praccho


Alex W.

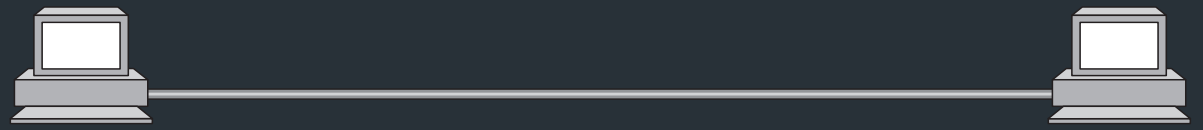

Matthias

Stretch

(and I won’t look if you are shopping and want to flee)

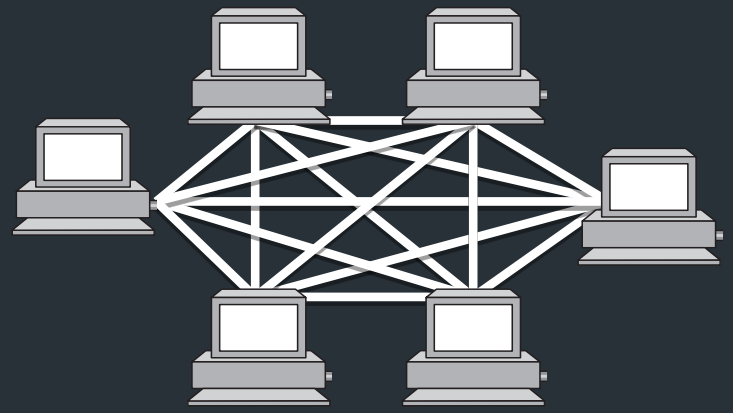
Building Blocks

- Nodes: Computers (hosts), dedicated routers, …
- Links: Coax, twisted pair, fiber, radio, …



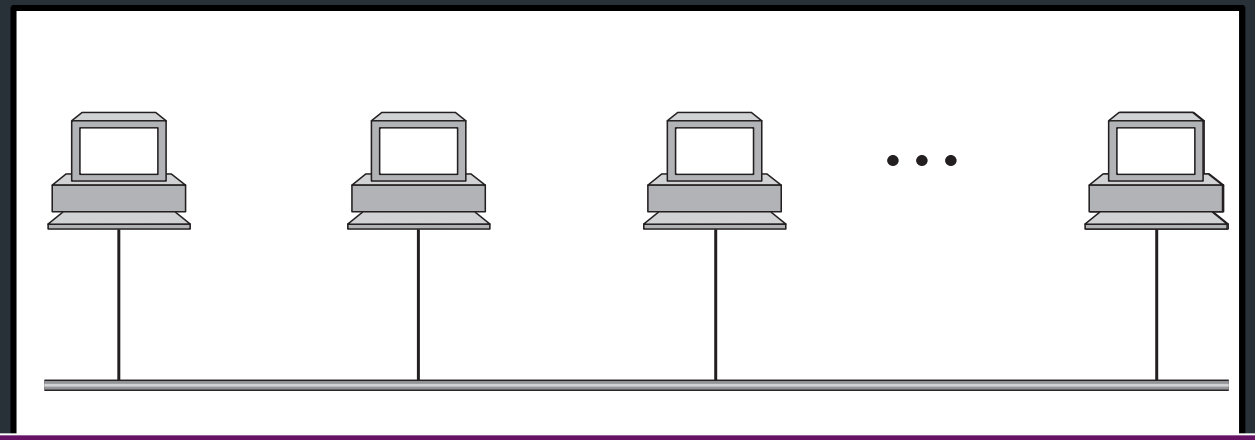
How to connect more nodes?

How to connect more nodes?



Multiple wires

Shared medium



Need a mechanism to share network resources

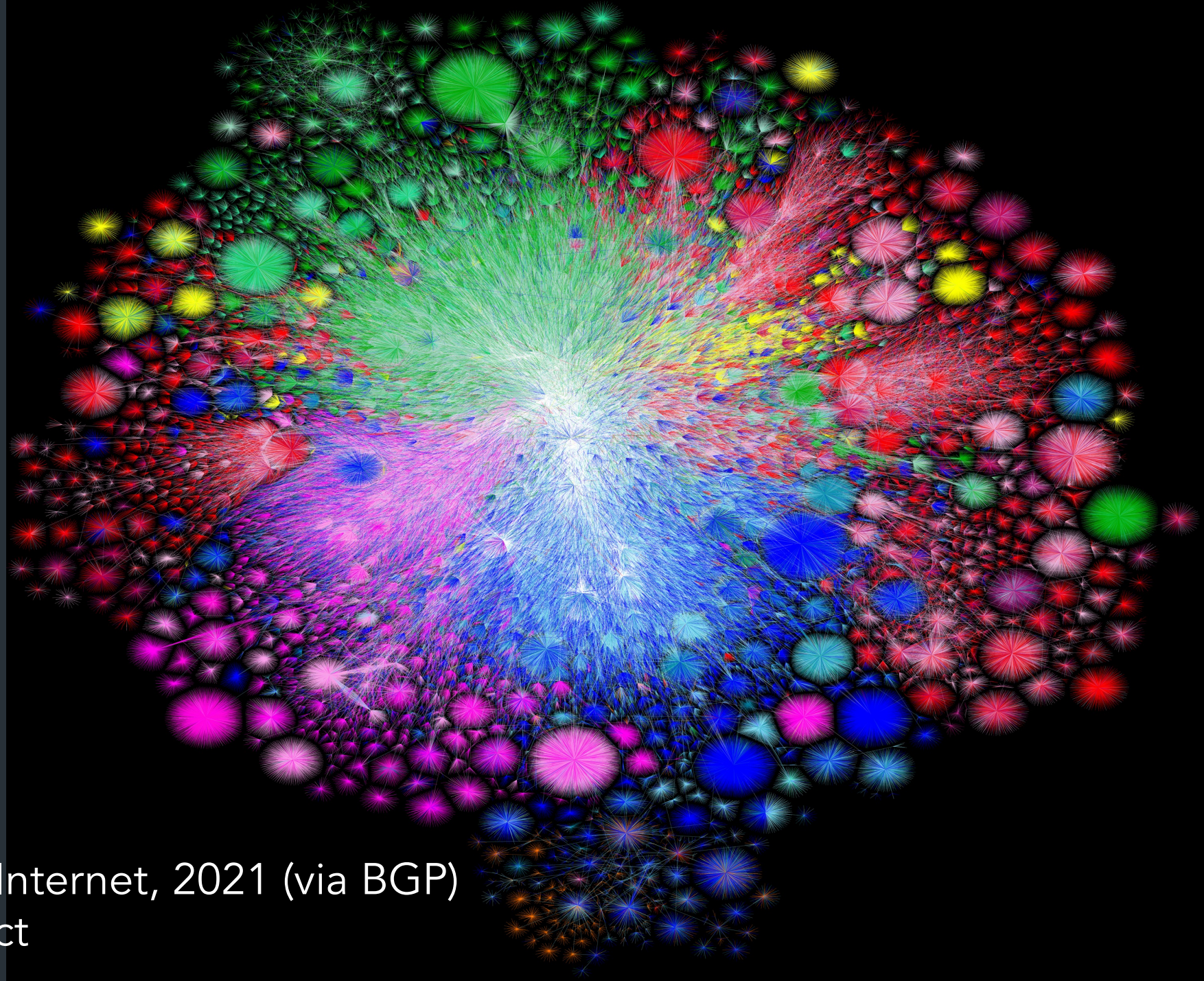
Multiplexing

- Mechanisms for simultaneous communication on the same channel
  - (or at least *nearly* simultaneous)

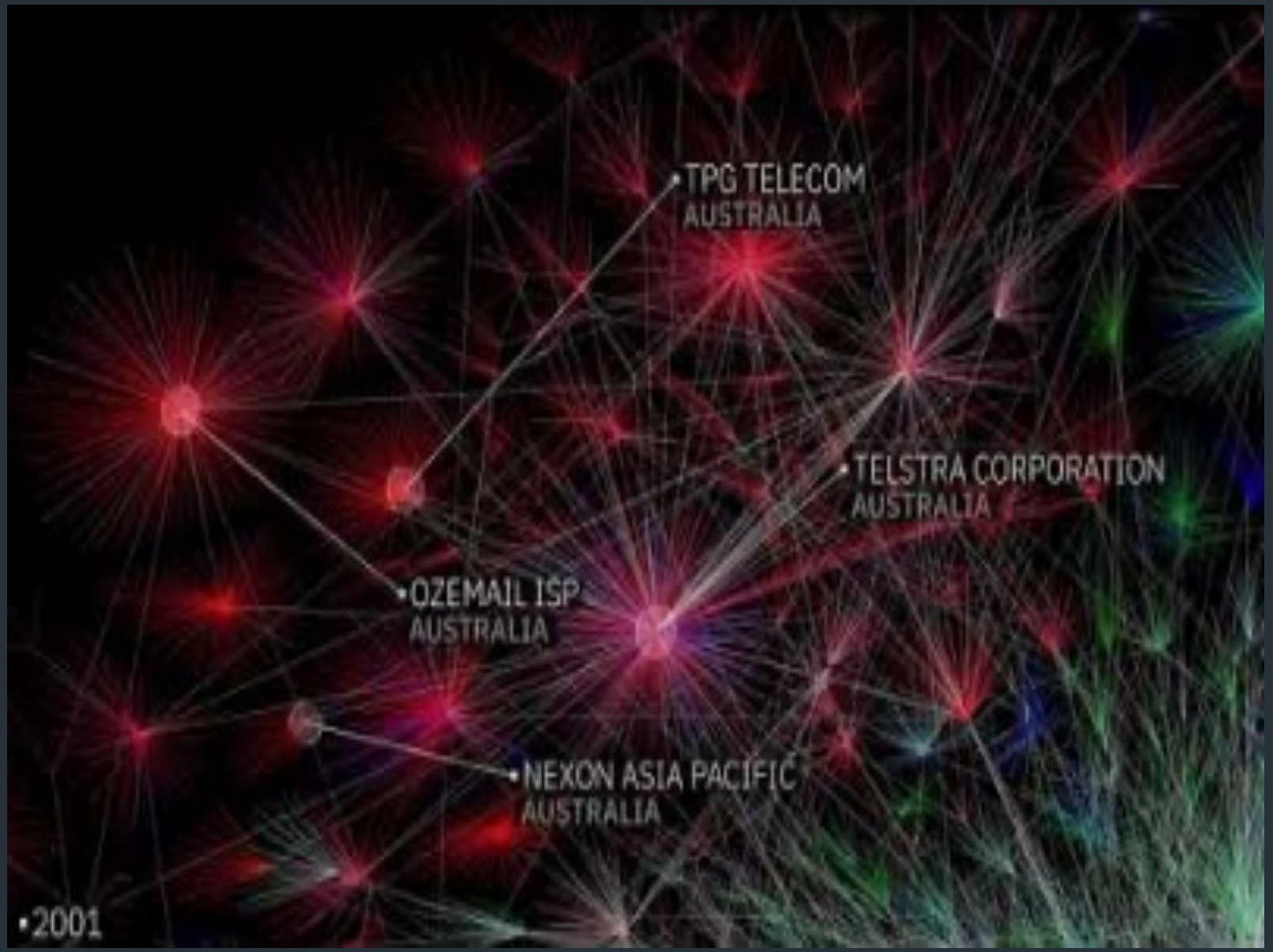
- Lots of different methods, depending on the medium and abstraction

How do you connect *apps*?

- Your computer runs multiple applications
- How to make them share the same resources?



Map of the Internet, 2021 (via BGP)
OPTE project

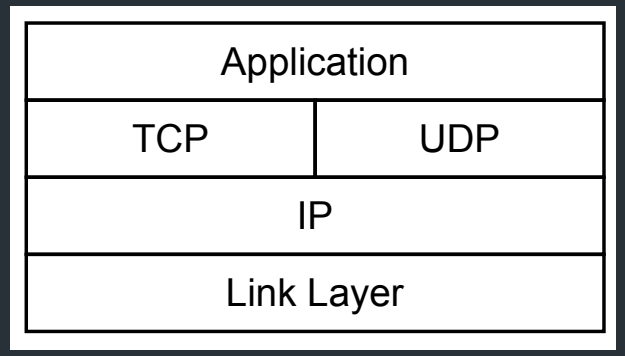


OPTE Internet map, 1997-2021: https://youtu.be/DdaElt6oP6w

How do we make sense of all this?

- *Very* large number of computers
- Incredible variety of technologies
  - Each with very different constraints
- Lots of *multiplexing*
- No single administrative entity
- Evolving demands, protocols, applications
  - Each with very different requirements!

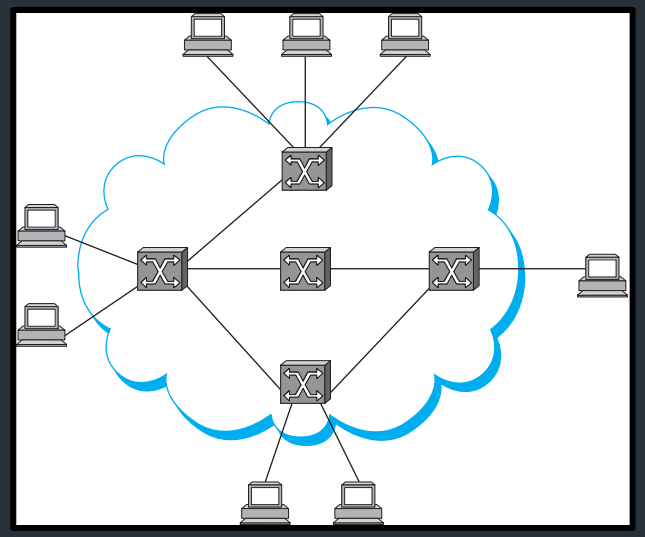
Layering

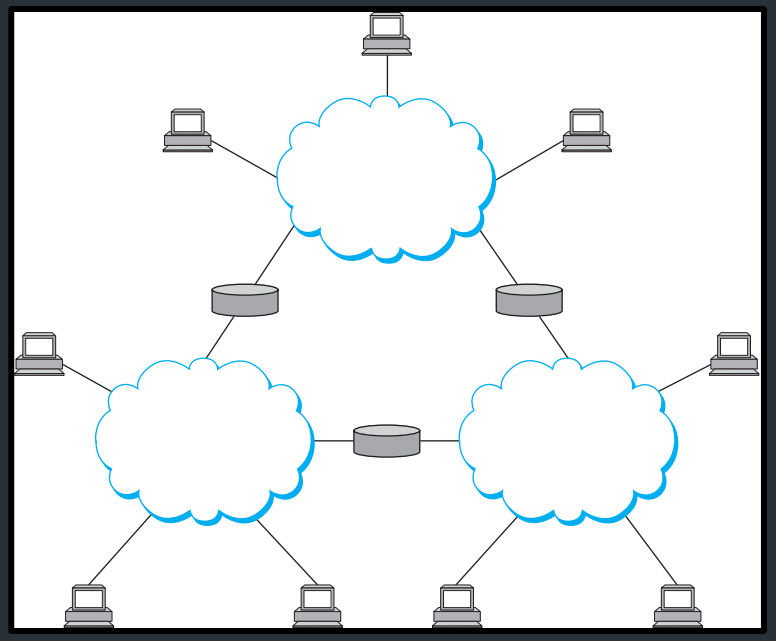


Separation of concerns

- Break problem into separate parts
- Solve each one independently
- Abstract data from the layer above inside data from the layer below
  - “Encapsulation”
- Different implementations at one “layer” use same interface
- Allows independent evolution

From Links to Networks

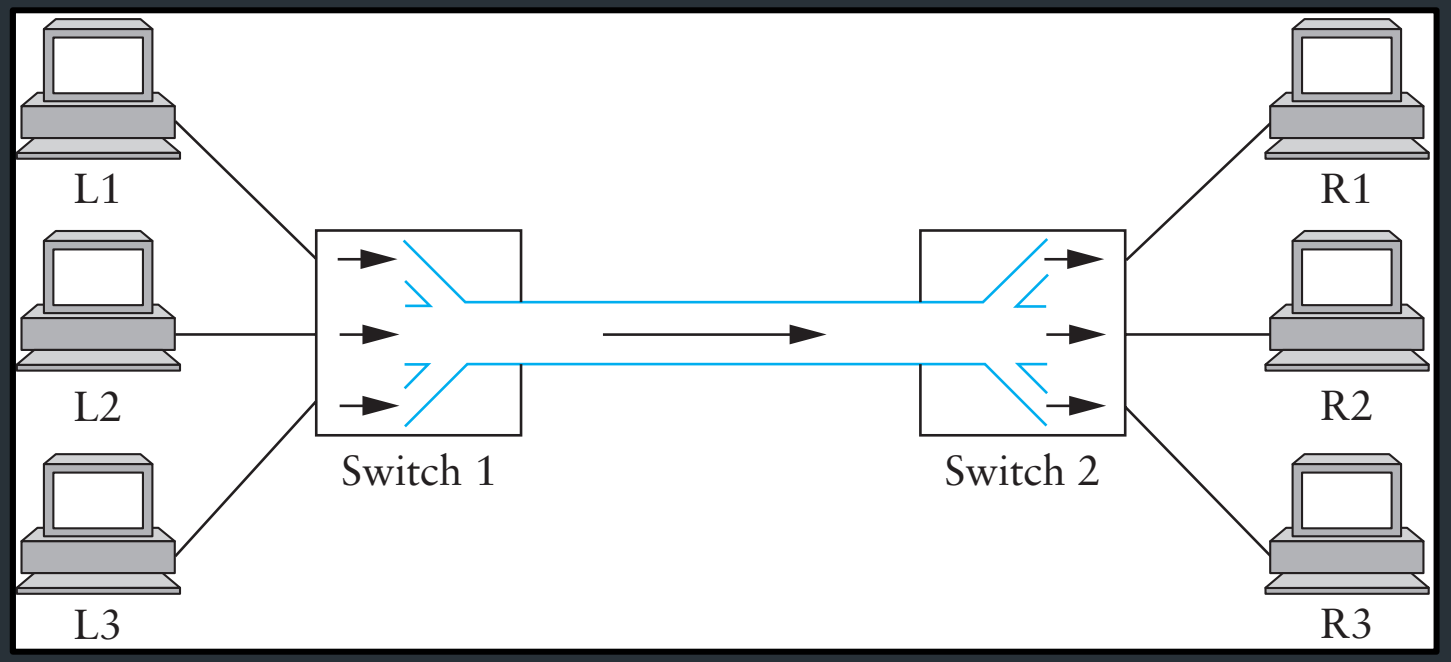




To scale to more nodes, use *switching*

- Nodes can connect to multiple other nodes
- Recursively, one node can connect to multiple networks

Multiplexing



- What to do when multiple flows must share a link?

Multiplexing strategies

- **Physical/Link layer** (Copper wires, wireless): Signalling strategies
- **Network layer**: IP addresses to identify one host
- **Transport layer**: One host has many ports, used for different applications
- Application layer: One program can have multiple "sockets", which are an interface to one network "connection"

For next class

- HW0: Survey (please fill out ASAP)
- Project 0: dev environment setup (due next Thursday)
- Project 1: out on Tuesday
  - Gearup: Thursday, Sept 14, 5-7pm in CIT368 (+ Zoom, recorded)