552: Computer Networks

Srinivas Narayana
Fall 2018 (MW 8.40—10 AM in SEC 218)
The Internet: it’s exciting!
The Internet: it’s exciting!

- A research experiment that escaped the lab
  - ... to become a global communication infrastructure

- Ever-expanding reach
  - Now: 3B people online
  - Future: Large potential for user and machine growth

- Consistent innovation with low barriers
  - Apps: streaming video, smart grids, telesurgery, …
  - Technologies: Ethernet, cellular, optics, human-body (!), …
The Internet has transformed everything

• How we communicate with other humans
• How we learn and acquire knowledge
• How we transact and do business
• How we entertain ourselves
• How we govern ourselves
• How warfare is conducted (!)
Why should we study networking?

- Utility: tangible real-world impact
  - Easy to measure and build things!
  - Artifacts can go a long way
  - *You* can build something that *you* want to use
    - … that other people then build on

- Intellectual rewards: interdisciplinary & evolving problems
  - So much to learn and apply from other fields
  - Several principles to contribute to other fields
Why should we study networking?

• Academic impact: a young and relatively immature field
  • *You* get to decide what the field looks like in a few years!
  • Several highly cited CS papers are networking papers
  • At least 2 Turing awards (Cerf&Kahn’04, Berners-Lee’16)

• Significant opportunities lie ahead!
  • *Define* the most interesting problems
  • *Build* your ideas using freely and openly available software
  • *Deploy* your ideas and technology over cheap platforms!
So, how *should* we study networking?
So, how *should* we study networking?

• A soup of acronyms? IP, TCP, HTTP, RSVP, …
So, how *should* we study networking?

• The soup of acronyms? IP, TCP, HTTP, RSVP, …
• The stack of layers and headers? port, option, flags, …
So, how *should* we study networking?

- The soup of acronyms? IP, TCP, HTTP, RSVP, ...
- The stack of layers and headers? port, option, flags, ...
- The suite of tools? ping, traceroute, wireshark, ...
- The wire protocols? SCTP, DNS, SSH, ...
- The middle boxes? CDN, firewall, NAT, proxy, ...
These categorizations turn out not to be very helpful.
So, how do we do networking research?
What we will study in this course

• Principles used to build computer networks
  • How to design and manage communication infrastructure
  • … to **effectively** meet application and network designer goals
  • … in ways that can be composed constructively

• Ways to partition functionality
  • Among entities (ISPs, users, …)
  • Among machines (end points, switches, middleboxes, …)
  • Among modules inside a machine (OS, apps, hardware, …)

Mix of classic and recent research papers
Course syllabus

1. Foundational principles
2. Inter-domain networking (how is the Internet glued together?)
3. Intra-domain networking (how do ISPs run their networks?)
4. Software-defined networking (how to make networks fungible?)
5. Data center networking (how do your fav Internet services run?)
6. Applications (how do your apps use networks effectively?)
7. Looking forward
Quick class introductions
Go around and say a few things…
Course logistics: Structure

• Reading papers
  • Summarize and critique the papers you read
  • Find ways to improve and generalize the ideas and principles

• Classroom time
  • Some background material
  • Discussion of papers

• Paper reviews

• Final project
Course logistics: Assessments

- Class participation (20%)
  - Ask a lot of questions
  - Speak up, debate, and challenge ideas with your own proposals
  - But keep the conversations technical: no \textit{ad hominem}
  - Read “how to disagree” in the optional readings

- Project (40%)

- Paper reviews (40%)
Course logistics: Class project

• Could be …
  • A solution to an open research question
  • A reproduction of results from one of the papers you’ve read
  • A reimplementation of an existing system for a different platform
  • A new tool that makes further research possible or easy
  • … or come talk to me about your ideas

• *Must* involve a significant programming component
Course logistics: Class project

• Teams of 2—3 (preferred) or work alone

• Submit initial proposal (1 page) by Friday 11/02

• 6-page project write-up due end of semester (date TBA)

• More info and ideas over the next few lectures
Course logistics: Paper reviewing

• Reviews due on Fridays 09/21, 10/12, 10/19, 11/09

• Paper assignments TBA
  • You’ll review 1 of the 4 from the week of the due date

• You are welcome to discuss and collaborate. However:
  • All written work must be your own (no exceptions)
  • You must name all your collaborators in your review
  • Do not consult public reviews (where available)
    • You can read them after you’ve submitted your own
How do you read and critique a paper?
You’ll spend a lot of time reading

- Graduate courses
- Reviewing conference papers
- Research literature
- Understanding a closely related paper deeply
- Staying broadly educated
- Branching out into closely related areas
- Learning how to write better papers!

- It’s worth knowing how to read *effectively*
Keshav’s top-down, 3-pass approach (1/3)

• Title, abstract, introduction, section titles

• Category of paper: analysis? position? new capability? measurement?

• Context: what broader body of work is this related to?

• Correctness: are conclusions plausible? are assumptions valid?

• Contributions: reusable principles? previously unknown insights?

• Clarity: Can you understand it?
Keshav’s top-down, 3-pass approach (2/3)

• Read in more detail to understand main technical ideas
  • Understand graphs and illustrations

• You must be able to summarize the paper’s main technical contribution and supporting evidence to others!

• Ignore highly detailed aspects
  • Proofs
  • Appendices
  • Mark relevant but unknown references for later reading
Keshav’s top-down, 3-pass approach (3/3)

- Re-implement the paper’s solution from scratch
- … starting from its assumptions
- Identify its innovations
- Identify implicit or flawed assumptions
- Identify hidden flaws in the solution
Tim Roscoe: How to structure a review?

• What do you think are the technical contributions of the paper?
  • What’s new about the work?
  • What are the reusable principles and insights?

• What did you find to be cool or interesting about it?
  • Specific design techniques?
  • A method of measurement, evaluation, or something else?
  • A direction that may lead to interesting follow-up work?
Tim Roscoe: How to structure a review?

• Are there technical flaws in the solution or its evaluation?
  • All research is flawed or incomplete in some way!
  • Do the flaws fundamentally invalidate the claimed contributions?

• How can you improve the clarity of the paper?
  • Restructure sections?
  • Rephrase specific sentences?
  • Typos?
Course logistics summary

• Class web page: http://web.mit.edu/~alephtwo/www/552-F18/

• Instructor: Srinivas Narayana (alephtwo@csail.mit.edu)
  • Office hours (core 312) on Mondays after class (10am—noon)

• No official class textbook. Follow the assigned readings
  • Use supplementary texts (listed on web page) as needed

• Prerequisite: Undergrad computer networks (352) or by permission
  • Speak to me if you’re in doubt
Computer Networks
Edge and core: a useful distinction

- Edge: data origins or sinks ("endpoints")
  - Your laptop, mobile phone, Google’s servers
- Core: machines processing & transmitting data
  - Your WiFi router, Rutgers’s firewall, Verizon’s routers
- Varies by context: one person’s core is another’s edge
Best-effort packet delivery in the core

- Packet delivery: divide messages into sequence of packets
  - Core transmits each packet independently
  - Think “postal letters” instead of “phone calls”

- Best-effort delivery:
  - A packet may be lost
  - A packet may be corrupted
  - Packets may arrive out of order
What are the benefits of packet switching?
Best-effort packet delivery in the core

• The core becomes very simple to build
  • Don’t have to be reliable
  • Don’t need to make any performance guarantees
  • Don’t need to maintain packet ordering
  • Almost any medium can deliver individual packets
    • RFC 1149: IP Datagrams over Avian Carriers

• The early Internet thrived since (transient) disruptions are okay
But apps may need good performance!

• *Partition of labor*
  • Let the endpoints handle the guarantees for applications
  • “End to end argument”

• Endpoints run a *transport mechanism* for
  • Reliable, error-corrected, ordered delivery
  • Optimizing app metrics
  • Whatever your requirement might be!
For next week

• Read the papers & prepare for discussion

• Send your full name and netid in an email to alephtwo@csail.mit.edu with the subject “552”
  • Class announcements over email (for the time-being)

• http://web.mit.edu/~alephtwo/www/552-F18/