Topic 1 Foundation

- Administrivia
- Networks
- Channels
- Multiplexing
- Performance: loss, delay, throughput
Course Administration

Commonly Available Texts

- **Computer Networks: A Systems Approach**
  Peterson and Davie
  [https://book.systemsapproach.org](https://book.systemsapproach.org)

- **Computer Networking : Principles, Protocols and Practice**
  Olivier Bonaventure (and friends)
  Less GitHub but more practical exercises
  [https://www.computer-networking.info/](https://www.computer-networking.info/)
  [Version 3 draft (UCAM access only)](https://www.computer-networking.info/)

Other textbooks are available.
Thanks

• Slides are a fusion of material from

• Supervision material is drawn from
  Stephen Kell, Andy Rice, and the TA teams of 144 and 168

• Finally thanks to the fantastic past Part 1b students and Andrew Rice for all the tremendous feedback.
What is a network?

• A system of “links” that interconnect “nodes” in order to move “information” between nodes

• Yes, this is all rather abstract
What is a network?

- We also talk about

or even

- Yes, abstract, vague, and under-defined....
There are many different types of networks

- Internet
- Telephone network
- Transportation networks
- Cellular networks
- Supervisory control and data acquisition networks
- Optical networks
- Sensor networks

We will focus almost exclusively on the Internet
The Internet has transformed everything

• The way we do business
  – E-commerce, advertising, cloud-computing
• The way we have relationships
  – Facebook friends, E-mail, IM, virtual worlds
• The way we learn
  – Wikipedia, search engines
• The way we govern and view law
  – E-voting, censorship, copyright, cyber-attacks
A few defining characteristics of the Internet
A federated system

- The Internet ties together different networks
  - >20,000 ISP networks (the definition is fuzzy)

Tied together by IP -- the “Internet Protocol” : a single common interface between users and the network and between networks
A federated system

- The Internet ties together different networks
  - >20,000 ISP networks

- A single, common interface is great for interoperability…
- …but tricky for business

- Why does this matter?
  - ease of interoperability is the Internet’s most important goal
  - practical realities of incentives, economics and real-world trust, drive topology, route selection and service evolution
Tremendous scale
(2020 numbers – so some ‘weird’)

- **4.57 Billion** users (58% of world population)
- **1.8 Billion** web sites
  - 34.5% of which are powered by the WordPress!
- **4.88 Billion** smartphones (45.4% of population)
- **500 Million** Tweets a day
- **100 Billion** WhatsApp messages per day
- **1 Billion** hours of YouTube video watched per day
- **500 hours** of Youtube video added per minute
- **2+ billion** TikTok installs
- **60% video streaming**
  - 12.5% of the Internet traffic is native Netflix
Tremendous scale
(2020 numbers – so some ‘weird’)

• 4.57 Billion users (58% of world population)
• 4.88 Billion smartphones (45.4% of population)
• 1.8 Billion web sites
  – 34.5% of which are powered by WordPress!
• 500 Million Tweets a day
• 1 Billion WhatsApp messages per day
• 1 Billion hours of YouTube video watched per day
• 500 hours of Youtube video added per minute
• 2+ billion TikTok installs
• 60% video streaming
  – 12.5% of the Internet traffic is native Netflix

“Internet Scale” refers to such systems
Enormous diversity and dynamic range

- Communication latency: nanoseconds to seconds ($10^9$)
- Bandwidth: 100 bits/second to 400 Gigabits/second ($10^9$)
- Packet loss: 0 – 90%

- Technology: optical, wireless, satellite, copper

- Endpoint devices: from sensors and cell phones to datacenters and supercomputers
- Applications: social networking, file transfer, skype, live TV, gaming, remote medicine, backup, IM
- Users: the governing, governed, operators, malicious, naïve, savvy, embarrassed, paranoid, addicted, cheap …
Constant Evolution

1970s:
• 56 kilobits/second “backbone” links
• <100 computers, a handful of sites in the US (and one UK)
• Telnet and file transfer are the “killer” applications

Today
• 400+ Gigabits/second backbone links
• 40B+ devices, all over the globe
  – 27B+ IoT devices alone
Asynchronous Operation

- Fundamental constraint: **speed of light**

- Consider:
  - How many cycles does your 3GHz CPU in Cambridge execute before it can possibly get a response from a message it sends to a server in Palo Alto?
    - Cambridge to Palo Alto: 8,609 km
    - Traveling at 300,000 km/s: 28.70 milliseconds
    - Then back to Cambridge: 2 x 28.70 = 57.39 milliseconds
    - $3,000,000,000$ cycles/sec * 0.05739 = 172,179,999 cycles!

- Thus, communication feedback is always *dated*

How much can change with 172 Million instructions
Prone to Failure

• To send a message, all components along a path must function correctly
  – software, wireless access point, firewall, links, network interface cards, switches,…
  – Including human operators

• Consider: 50 components, that work correctly 99% of time → 39.5% chance communication will fail

• Plus, recall
  – scale → lots of components
  – asynchrony → takes a long time to hear (bad) news
  – federation (internet) → hard to identify fault or assign blame
A detour
8 fallacies of Distributed Systems
Recap: The Internet is...

• A complex federation
• Of enormous scale
• Dynamic range
• Diversity
• Constantly evolving
• Asynchronous in operation
• Failure prone
• Constrained by what’s practical to engineer
• Too complex for (simple) theoretical models
• “Working code” doesn’t mean much
• Performance benchmarks are too narrow
An Engineered System

- Constrained by what technology is practical
  - Link bandwidths
  - Switch port counts
  - Bit error rates
  - Cost
  - …
Nodes and Links

Channels = Links
Peer entities = Nodes
Properties of Links (Channels)

- Bandwidth (capacity): “width” of the links
  - number of bits sent (or received) per unit time (bits/sec or bps)
- Latency (delay): “length” of the link
  - propagation time for data to travel along the link (seconds)
- Bandwidth-Delay Product (BDP): “volume” of the link
  - amount of data that can be “in flight” at any time
  - propagation delay × bits/time = total bits in link
Examples of Bandwidth-Delay

• Same city over a slow link:
  – BW~100Mbps
  – Latency~10msec
  – BDP ~ $10^6$bits ~ 125KBytes
    \[17\text{km} \times \text{c} = 56\mu\text{s} \ll 10\text{ms}\]

• To California over a fast link:
  – BW~10Gbps
  – Latency~140msec
  – BDP ~ $1.4 \times 10^9$bits ~ 175MBytes
    \[9708\text{km} \times \text{c} = 32\text{ms} \ll 140\text{ms}\]

• Intra Datacenter:
  – BW~100Gbps
  – Latency~30usec
  – BDP ~ $10^6$bits ~ 375KBytes
    \[750\text{m} \times \text{c} = 56\mu\text{s} \approx 30\mu\text{s}\]

• Intra Host:
  – BW~100Gbps
  – Latency~16nsec
  – BDP ~ 1600bits ~ 200Bytes
    \[25\text{cm} \times \text{c} = 83\text{ps} \ll 16\text{ns}\]
Packet Delay

Sending a 100B packet from A to B?

Time to transmit one bit = \( \frac{1}{10} \) s

Time to transmit 800 bits = \( 800 \times \frac{1}{10} \times 10^6 \) s

The last bit reaches B at \( \left( \frac{800}{10} \times 10^6 \right) + \frac{1}{10} \times 10^3 \) s

\[ \text{Packet Delay} = \left( \frac{\text{Packet Size}}{\text{Link Bandwidth}} \right) + \text{Link Latency} \]
Packet Delay

Sending a 100B packet from A to B?

A

1Mbps, 1ms

1Gbps, 1ms?

B

10^7 x 100B packets

The last bit in the file reaches B at
(10^7 \times 800 \times \frac{1}{10^9}) + \frac{1}{10^3}s
= 8001ms

The last bit reaches B at
(800 \times \frac{1}{10^9}) + \frac{1}{10^3}s
= 1.0008ms

The last bit reaches B at
(800 \times \frac{1}{10^6}) + \frac{1}{10^3}s
= 1.8ms
Packet Delay: The “pipe” view

Sending 100B packets from A to B?

A

B

1Mbps, 10ms

Packet Transmission

Time

BW

100Byte packet

100Byte packet

100Byte packet

41
Packet Delay: The “pipe” view

Sending 100B packets from A to B?

1Mbps, 10ms (BDP=10,000)

1Mbps, 5ms (BDP=5,000)

10Mbps, 1ms (BDP=10,000)
Packet Delay: The “pipe” view

Sending 100B packets from A to B?

What if we used 200Byte packets??
Recall Nodes and Links
What if we have more nodes?

One link for every node?

Need a scalable way to interconnect nodes
Solution: A switched network

Nodes share network link resources

How is this sharing implemented?
Two examples of switched networks

- Circuit switching (used in the *POTS*: Plain Old Telephone system)

- Packet switching (used in the Internet)
Circuit switching
Circuit switching

Idea: source reserves network capacity along a path

(1) Node A sends a reservation request
(2) Interior switches establish a connection -- i.e., “circuit”
(3) A starts sending data
(4) A sends a “teardown circuit” message
Multiplexing

Sharing makes things efficient (cost less)

• One airplane/train for 100’s of people
• One telephone for many calls
• One lecture theatre for many classes
• One computer for many tasks
• One network for many computers
• One datacenter many applications
Multiplexing

Sharing makes things efficient (cost less)
• One airplane/train for 100’s of people
• One telephone for many calls
• One lecture theatre for many classes
• One computer for many tasks
• One network for many computers
• One datacenter many applications
Old Time Multiplexing
Circuit Switching: FDM and TDM

Frequency Division Multiplexing

Example:
4 users

Radio Schedule
..., News, Sports, Weather, Local, News, Sports, ...

Frequency

Time Division Multiplexing

Radio Schedule
..., News, Sports, Weather, Local, News, Sports, ...

Frequency
Time-Division Multiplexing/Demultiplexing

- Time divided into frames; frames into slots
- Relative slot position inside a frame determines to which conversation data belongs
  - e.g., slot 0 belongs to orange conversation
- Slots are reserved (released) during circuit setup (teardown)
- If a conversation does not use its circuit capacity is lost!
Timing in Circuit Switching

- Circuit Establishment
- Transfer
- Circuit Tear-down

Information
Circuit switching: pros and cons

• Pros
  – guaranteed performance
  – fast transfer (once circuit is established)

• Cons
Timing in Circuit Switching

Circuit Establishment
Transfer
Circuit Tear-down
Circuit switching: pros and cons

• Pros
  – guaranteed performance
  – fast transfer (once circuit is established)

• Cons
  – wastes bandwidth if traffic is “bursty”
Timing in Circuit Switching

- Circuit Establishment
- Transfer
- Circuit Tear-down

Information
Timing in Circuit Switching

Circuit Establishment
Transfer
Circuit Tear-down
Circuit switching: pros and cons

• Pros
  – guaranteed performance
  – fast transfers (once circuit is established)

• Cons
  – wastes bandwidth if traffic is “bursty”
  – connection setup time is overhead
Circuit switching doesn’t “route around failure”
Circuit switching: pros and cons

• Pros
  – guaranteed performance
  – fast transfers (once circuit is established)

• Cons
  – wastes bandwidth if traffic is “bursty”
  – connection setup time is overhead
  – recovery from failure is slow
Numerical example

• How long does it take to send a file of 640,000 bits from host A to host B over a circuit-switched network?
  – All links are 1.536 Mbps
  – Each link uses TDM with 24 slots/sec
  – 500 msec to establish end-to-end circuit

Let’s work it out!

\[
\frac{1}{24} \times 1.536 \text{ Mbps} = 64 \text{ kb/s}
\]

\[
\frac{640,000}{64 \text{ kb/s}} = 10 \text{ s}
\]

\[
10 \text{ s} + 500 \text{ ms} = 10.5 \text{ s}
\]
Two examples of switched networks

- Circuit switching (used in the POTS: Plain Old Telephone system)

- Packet switching (used in the Internet)
Packet Switching

• Data is sent as chunks of formatted bits (Packets)
• Packets consist of a “header” and “payload”*
Packet Switching

• Data is sent as chunks of formatted bits (Packets)
• Packets consist of a “header” and “payload”*
  – payload is the data being carried
  – header holds instructions to the network for how to handle packet (think of the header as an API)
  – In this example, the header has a destination address
  – More complex headers may include
    • How this traffic should be handled? (first class, second class, etc)
    • Who signed for it?
    • Were the contents ok?
Packet Switching

- Data is sent as chunks of formatted bits (Packets)
- Packets consist of a “header” and “payload”
- Switches “forward” packets based on their headers
Switches forward packets

Forwarding Table

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLASGOW</td>
<td>4</td>
</tr>
<tr>
<td>OXFORD</td>
<td>5</td>
</tr>
<tr>
<td>EDIN</td>
<td>2</td>
</tr>
<tr>
<td>UCL</td>
<td>3</td>
</tr>
</tbody>
</table>

111010010  EDIN
What about the time to process the packet at the switch?

- We’ll assume it’s relatively negligible (mostly true)
Timing in Packet Switching

Could the switch start transmitting as soon as it has processed the header?

• Yes! This would be called a “cut through” switch
We will always assume a switch processes/forwards a packet after it has received it entirely. This is called “store and forward” switching.
Packet Switching

• Data is sent as chunks of formatted bits (Packets)
• Packets consist of a “header” and “payload”
• Switches “forward” packets based on their headers
Packet Switching

- Data is sent as chunks of formatted bits (Packets)
- Packets consist of a “header” and “payload”
- Switches “forward” packets based on their headers
- Each packet travels independently
  – no notion of packets belonging to a “circuit”
Packet Switching

- Data is sent as chunks of formatted bits (Packets)
- Packets consist of a “header” and “payload”
- Switches “forward” packets based on their headers
- Each packet travels independently
- No link resources are reserved in advance. Instead packet switching leverages statistical multiplexing (stat muxing)
Multiplexing

Sharing makes things efficient (cost less)
• One airplane/train for 100’s of people
• One telephone for many calls
• One lecture theatre for many classes
• One computer for many tasks
• One network for many computers
• One datacenter many applications
Three Flows with Bursty Traffic

Data Rate 1

Data Rate 2

Data Rate 3

Time

Capacity
When Each Flow Gets $\frac{1}{3}$rd of Capacity

**Frequent Overloading**
When Flows Share Total Capacity

Statistical multiplexing relies on the assumption that not all flows burst at the same time.

Very similar to insurance, and has same failure case

No Overloading
Three Flows with Bursty Traffic

Data Rate 1

Data Rate 2

Data Rate 3

Time

Capacity
Three Flows with Bursty Traffic

Data Rate 1

Data Rate 2

Data Rate 3

Capacity
Three Flows with Bursty Traffic

Data Rate 1+2+3 >> Capacity

What do we do under overload?
Statistical multiplexing: pipe view
Statistical multiplexing: pipe view
Statistical multiplexing: pipe view

No Overload
Statistical multiplexing: pipe view

Queue overload into Buffer

Transient Overload
Not such a rare event
Statistical multiplexing: pipe view

Queue overload into Buffer

Transient Overload
Not such a rare event
Statistical multiplexing: pipe view

Queue overload into Buffer

Transient Overload
Not such a rare event
Statistical multiplexing: pipe view

Transient Overload
Not such a rare event
Statistical multiplexing: pipe view

Queue overload into Buffer

Transient Overload
Not such a rare event
Statistical multiplexing: pipe view

Buffer absorbs transient bursts
But NOT additional capacity
What about persistent overload?
Will eventually drop packets
Queues introduce queuing delays

- Recall,
  
  \[ \text{packet delay} = \text{transmission delay} + \text{propagation delay} \] (*)

- With queues (statistical multiplexing)
  
  \[ \text{packet delay} = \text{transmission delay} + \text{propagation delay} + \text{queuing delay} \] (*)

- Queuing delay caused by “packet interference”

- Made worse at high load
  - less “idle time” to absorb bursts
  - think about traffic jams at rush hour or rail network failure

(* plus per-hop *processing* delay that we define as negligible)
Queuing delay extremes

- \( R = \) link bandwidth (bps)
- \( L = \) packet length (bits)
- \( a = \) average packet arrival rate

Traffic intensity = \( \frac{La}{R} \)

- \( \frac{La}{R} \approx 0 \): average queuing delay small
- \( \frac{La}{R} \rightarrow 1 \): delays become large
- \( \frac{La}{R} > 1 \): more “work” arriving than can be serviced, average delay infinite – or data is lost (dropped).
Recall the Internet *federation*

- The Internet ties together different networks
  - >20,000 ISP networks

We can see (hints) of the nodes and links using traceroute...
“Real” Internet delays and routes

traceroute: rio.cl.cam.ac.uk to people.eng.unimelb.edu.au
(tracepath on winows is similar)

awm22@rio:~$ traceroute people.eng.unimelb.edu.au
traceroute to people.eng.unimelb.edu.au (128.250.59.37), 30 hops max, 60 byte packets
1 vlan101.gatwick.net.cl.cam.ac.uk (128.232.32.2) 1.520 ms 1.822 ms 0.709 ms
2 cl-wgb.d-mw.net.cam.ac.uk (193.60.89.5) 0.259 ms 0.256 ms 0.227 ms
3 d-mw.c-ce.net.cam.ac.uk (131.111.6.53) 0.231 ms 0.381 ms 0.357 ms
4 c-ce.b-ec.net.cam.ac.uk (131.111.6.82) 0.317 ms 0.481 ms 0.476 ms
5 ae0.lowdss-ban1.ja.net (146.97.41.37) 2.842 ms 2.846 ms 2.821 ms
6 ae26.lowdss-sbr1.ja.net (146.97.35.245) 2.877 ms 2.805 ms 2.795 ms
7 ae28.londhx-sbr1.ja.net (146.97.33.17) 6.191 ms 6.109 ms 6.325 ms
8 janet.mx1.lon.uk.geant.net (62.40.124.197) 6.319 ms 6.245 ms 6.258 ms
9 138.44.226.6 (138.44.226.6) 169.704 ms 169.722 ms 169.682 ms
10 et-7-3-0.pe1.wmlb.vic.aarnet.net.au (113.197.15.28) 250.954 ms 251.163 ms 251.116 ms
11 ***
12 4000v-eng-web-people-l.eng.unimelb.edu.au (128.250.59.37) 251.943 ms 251.952 ms 251.962 ms
13 4000v-eng-web-people-l.eng.unimelb.edu.au (128.250.59.37) 252.053 ms 252.018 ms 251.966 ms
14 ***
15 4000v-eng-web-people-l.eng.unimelb.edu.au (128.250.59.37) 252.215 ms 252.088 ms 252.118 ms
16 4000v-eng-web-people-l.eng.unimelb.edu.au (128.250.59.37) 253.361 ms 253.109 ms 253.461 ms
17 4000v-eng-web-people-l.eng.unimelb.edu.au (128.250.59.37) 253.077 ms 253.832 ms 253.298 ms
18 ***
....
29 ***
30 ***

* means no response (probe or reply lost, router not replying)
traceroute: rio.cl.cam.ac.uk to www.caida.org

rio:~$ traceroute --resolve-hostnames www.caida.org
traceroute to www.caida.org (192.172.226.122), 64 hops max
  1 128.232.64.2 (vlan398.gatwick.net.cl.cam.ac.uk) 3.760ms 2.060ms 1.226ms
  2 193.60.89.5 (cl-wgb.d-mw.net.cam.ac.uk) 53.777ms 67.458ms 0.556ms
  3 131.111.7.53 (d-mw.c-hi.net.cam.ac.uk) 0.638ms 0.621ms 0.658ms
  4 131.111.7.82 (ips-out.b-jc.net.cam.ac.uk) 0.353ms 0.346ms 0.338ms
  5 146.97.41.37 (ae0.lowdss-ban1.ja.net) 2.754ms 2.648ms 2.701ms
  6 146.97.35.245 (ae26.lowdss-sbr1.ja.net) 5.412ms 5.177ms 4.474ms
  7 146.97.33.21 (ae31.londpg-sbr2.ja.net) 0.582ms 0.441ms 0.397ms
  8 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
  9 146.97.35.24 (ae26.erdiss-sbr1.ja.net) 0.582ms 0.441ms 0.397ms
 10 146.97.33.21 (ae31.londpg-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
 11 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 0.582ms 0.441ms 0.397ms
 12 146.97.35.24 (ae26.erdiss-sbr1.ja.net) 2.728ms 2.648ms 2.701ms
 13 146.97.33.21 (ae31.londpg-sbr2.ja.net) 0.582ms 0.441ms 0.397ms
 14 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
 15 146.97.33.21 (ae31.londpg-sbr2.ja.net) 0.582ms 0.441ms 0.397ms
 16 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
 17 146.97.33.21 (ae31.londpg-sbr2.ja.net) 0.582ms 0.441ms 0.397ms
 18 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
 19 146.97.33.21 (ae31.londpg-sbr2.ja.net) 0.582ms 0.441ms 0.397ms
 20 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
 21 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
 22 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
 23 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
 24 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
 25 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms
 26 146.97.33.25 (ae30.erdiss-sbr2.ja.net) 2.728ms 2.648ms 2.701ms

A little more interesting because each hop resolves to a name (caida is in San Diego)
Internet structure: network of networks

- a packet passes through many networks!
Internet structure: network of networks

- “Tier-3” ISPs and local ISPs
  - last hop (“access”) network (closest to end systems)

Local and tier-3 ISPs are *customers* of higher tier ISPs connecting them to rest of Internet.
Internet structure: network of networks

- "Tier-2" ISPs: smaller (often regional) ISPs
  - Connect to one or more tier-1 ISPs, possibly other tier-2 ISPs

Tier-2 ISP pays tier-1 ISP for connectivity to rest of Internet
- tier-2 ISP is customer of tier-1 provider

Tier-2 ISPs also peer privately with each other.
Internet structure: network of networks

- roughly hierarchical
- at center: “tier-1” ISPs (e.g., Verizon, Sprint, AT&T, Cable and Wireless), national/international coverage
  - treat each other as equals
Tier-1 ISP: e.g., Sprint

POP: point-of-presence

to/from backbone

peering

to/from customers
Packet Switching

- Data is sent as chunks of formatted bits (Packets)
- Packets consist of a “header” and “payload”
- Switches “forward” packets based on their headers
- Each packet travels independently
- No link resources are reserved in advance. Instead packet switching uses statistical multiplexing
  - allows efficient use of resources
  - but introduces queues and queuing delays
Packet switching versus circuit switching

Packet switching may (does!) allow more users to use network

- 1 Mb/s link
- each user:
  - 100 kb/s when “active”
  - active 10% of time
- circuit-switching:
  - 10 users
- packet switching:
  - with 35 users, probability > 10 active at same time is less than .0004

Q: how did we get value 0.0004?
Packet switching versus circuit switching

Q: how did we get value 0.0004?

- 1 Mb/s link
- each user:
  - 100 kb/s when “active”
  - active 10% of time

- circuit-switching:
  - 10 users

- packet switching:
  - with 35 users, probability
    > 10 active at same time is less than .0004

Let $U$ be number of users active
$N$ the total users
$P$ is 0.1 in our example to get 0.0004
\[ P(U = k) = \binom{n}{k} p^k (1-p)^{n-k} \]

\[
\begin{align*}
\therefore P(U \leq K) &= \sum_{k=0}^{K} \binom{n}{k} p^k (1-p)^{n-k} \\
&\geq P(U > K) = 1 - \sum_{k=0}^{K} \binom{n}{k} p^k (1-p)^{n-k} 
\end{align*}
\]

for \( n = 35, \ K = 10 \)

\[ P(U \leq 10) = \sum_{k=0}^{10} \binom{35}{k} p^k (1-p)^{35-k} \]

where \( p = 0.1 \):

\[ P(U \leq 10) = 0.99958 \]

\[ P(U > 10) = 0.00042 \]
Circuit switching: pros and cons

• Pros
  – guaranteed performance
  – fast transfers (once circuit is established)

• Cons
  – wastes bandwidth if traffic is “bursty”
  – connection setup adds delay
  – recovery from failure is slow
Packet switching: pros and cons

• Pros
  – efficient use of bandwidth (stat. muxing)
  – no overhead due to connection setup
  – resilient -- can `route around trouble’

• Cons
  – no guaranteed performance
  – header overhead per packet
  – queues and queuing delays
Summary

• A sense of how the basic `plumbing’ works
  – links and switches
  – packet delays = transmission + propagation + queuing + (negligible) per-switch processing
  – statistical multiplexing and queues
  – circuit vs. packet switching