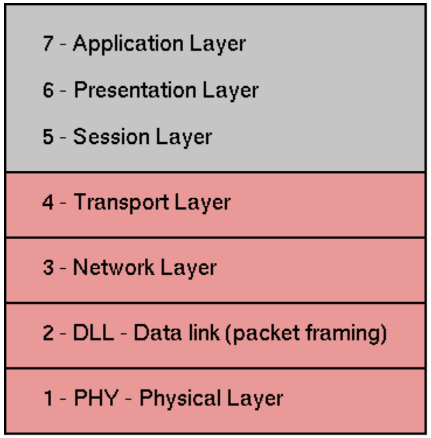
Computer Networking

Slide Set 1

D J Greaves



(thanks to Andrew W. Moore)
David.greaves@cl.cam.ac.uk

2025-26 update A2

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://github.com/SystemsApproach/book

- Computer Networking : Principles, Protocols and Practice

  Olivier Bonaventure (and friends)

  Less GitHub but more practical exercises

  https://www.computer-networking.info/

Many other textbooks are available, details on the materials tab of webpage.

Thanks

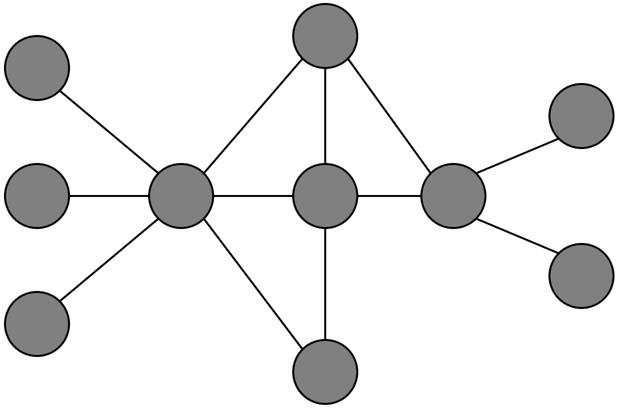
- Slides are a fusion of material from

  to Andrew Moore, Stephen Strowes, Tilman Wolf & Mike Zink, Ashish Padalkar , Evangelia Kalyvianaki, Brad Smith, Ian Leslie, Richard Black, Jim Kurose, Keith Ross, Larry Peterson, Bruce Davie, Jen Rexford, Ion Stoica, Vern Paxson, Scott Shenker, Frank Kelly, Stefan Savage, Jon Crowcroft , Mark Handley, Sylvia Ratnasamy, Adam Greenhalgh, and Anastasia Courtney.
- Supervision material is drawn from

  Stephen Kell, Andy Rice, and the TA teams of 144 and 168
- Finally thanks to the fantastic past Part Ib students and Andrew Rice for all the tremendous feedback.

What is a network?

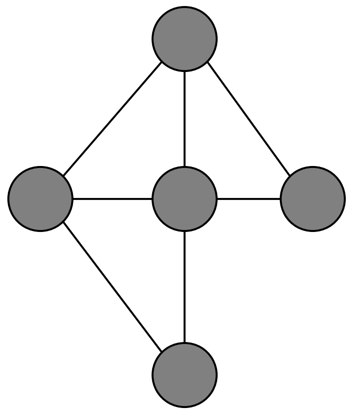
- A system of “links” that interconnect “nodes” in order to move “information” between nodes



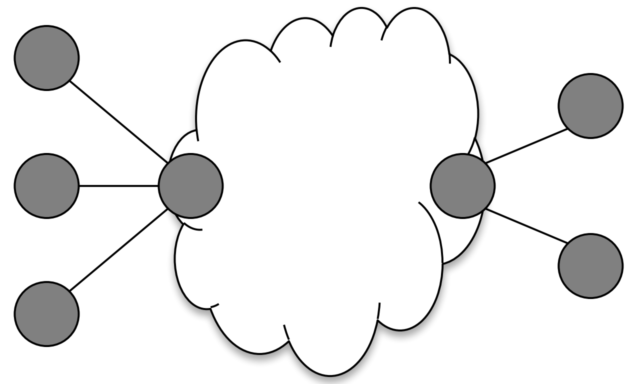
- Yes, this is all rather abstract

What is a network?

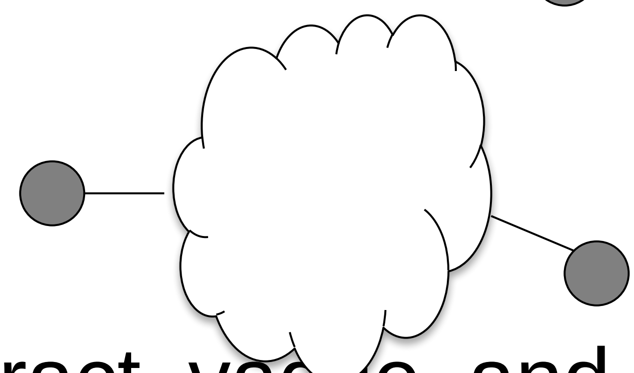
- We also talk about



or



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
- Satellite networks

We will focus almost exclusively on the Internet

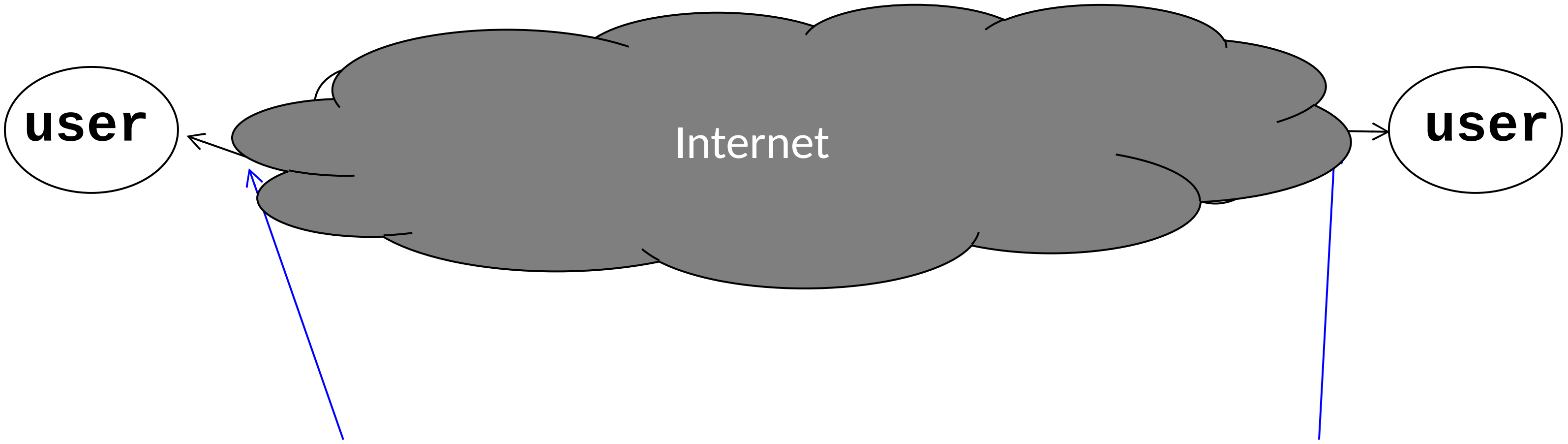
The Internet has transformed everything

- 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
  - >22,000 ISP networks (the definition is fuzzy)



Tied together by IP -- the "Internet Protocol" : a single common interface between users and the network and between (sub-)networks

A federated system

- The Internet ties together different networks
  - >22,000 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

Enormous diversity and dynamic range

- Communication latency: nanoseconds to seconds (10^9)
- Bandwidth: 100bits/second to 1.600 Terabits/second (10^{12})
- 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:

- 56kilobits/second “backbone” links
- <100 computers, a handful of sites in the US (and one UK)
- Telnet and file transfer were 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 a 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
    - Travelling at (200-)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 within 172 million instructions?

Prone to Failure

- To send a message (on a predetermined route), **all** components along a path must function correctly
  - software, wireless access point, firewall, links, network interface cards, switches,…
  - Including human operators
- Consider: 50 components in a system, each working 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 (**inter**net) hard to identify fault or assign blame

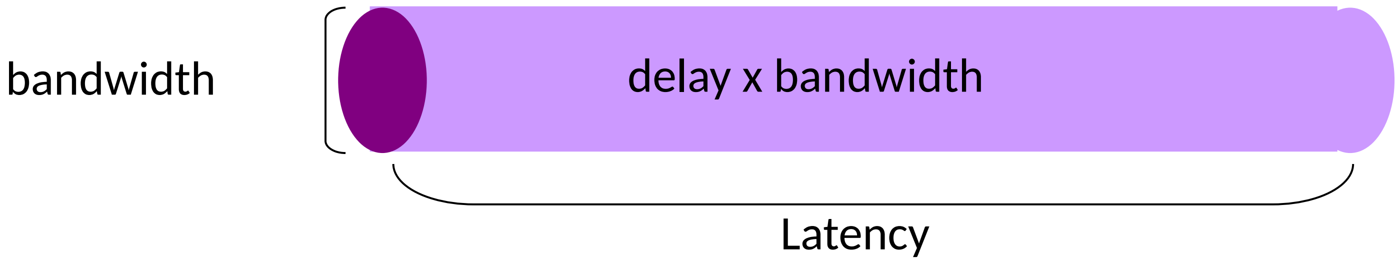
Recap: The Internet is…

- 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/any?) 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
  - …

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 ~ 10^6bits ~ 125KBytes

17km / c = 56µs << 10ms

- To California over a fast link:
  - BW~10Gbps
  - Latency~140msec
  - BDP ~ $1.4x10^9$bits ~ 175MBytes

9708km / c = 32ms << 140ms

- Intra Datacenter:
  - BW~100Gbps
  - Latency~30usec
  - BDP ~ 10^6bits ~ 375KBytes

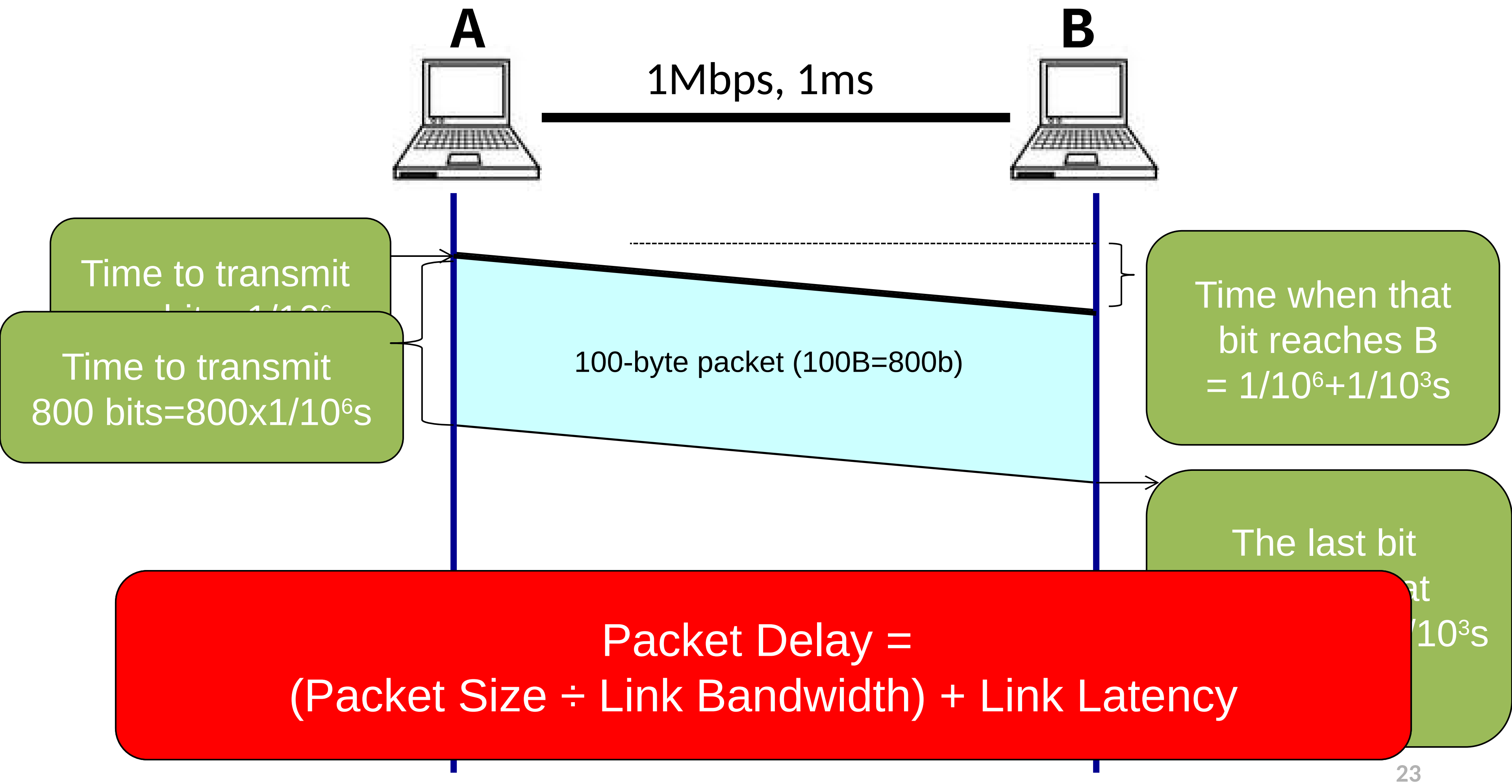
750m / c = 56µs ≅ 30µs

- Intra (inside) Host:
  - BW~800Gbps
  - Latency~16nsec
  - BDP ~ $12x10^3$bits ~ 5KBytes

25cm / c = 83ps << 16ns

Packet Delay

Sending a 100B packet from A to B?

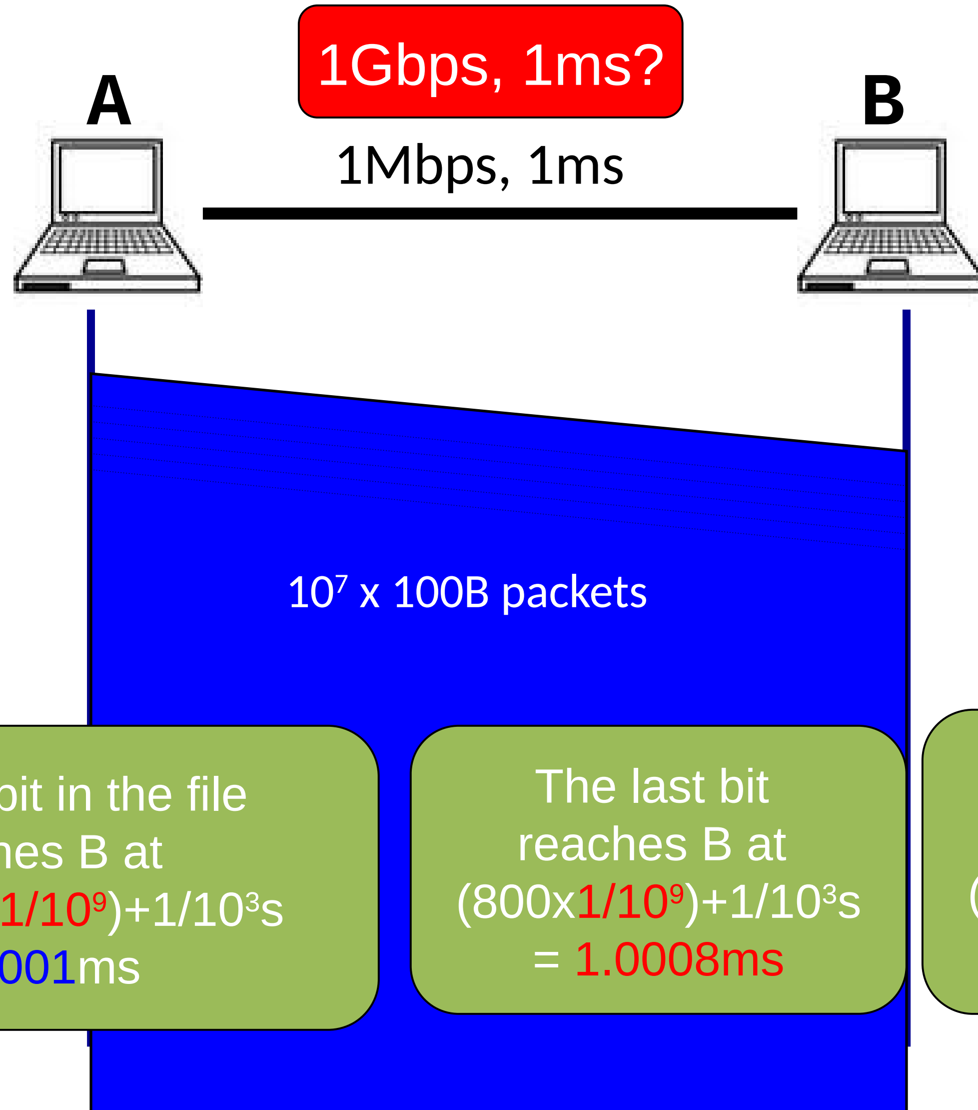


Packet Delay =
(Packet Size ÷ Link Bandwidth) + Link Latency

1GB file in 100B packets

ay

Sending a 100B packet from A to B?



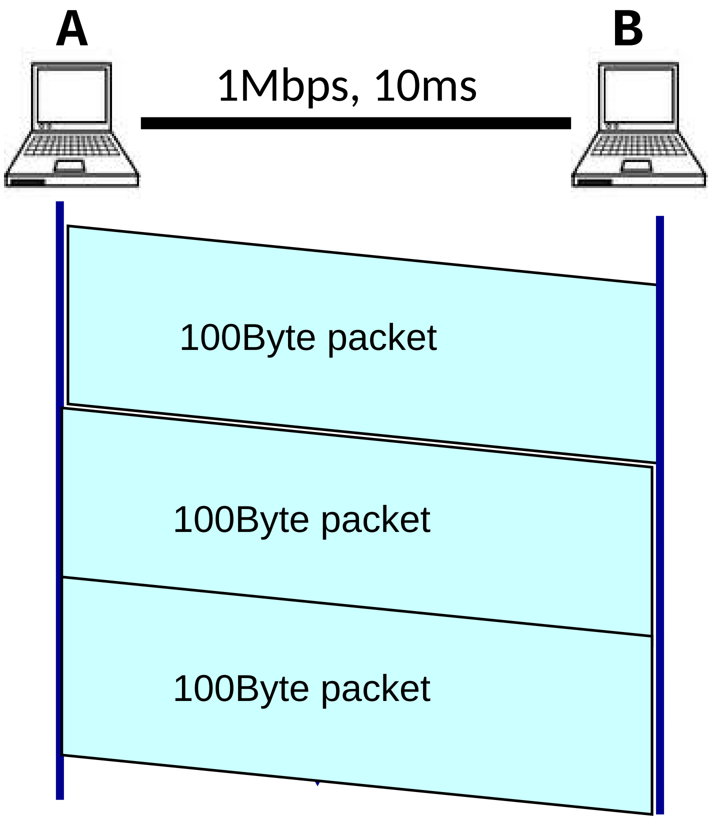
The last bit in the file reaches B at $(10^7 \times 800 \times 1/10^9) + 1/10^3$s = 8001ms

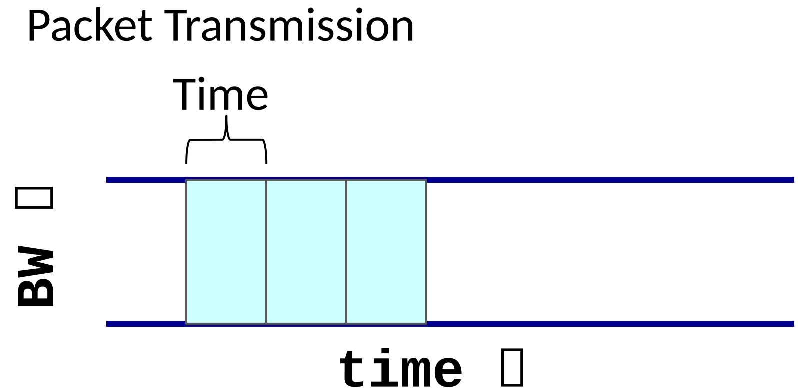
The last bit reaches B at $(800 \times 1/10^9) + 1/10^3$s = 1.0008ms

The last bit reaches B at $(800 \times 1/10^6) + 1/10^3$s = 1.8ms

Packet Delay: The "pipe" view

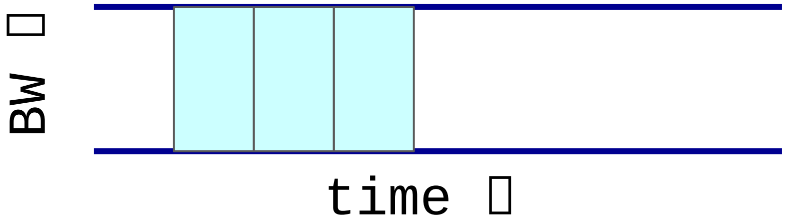
Sending 100B packets from A to B?



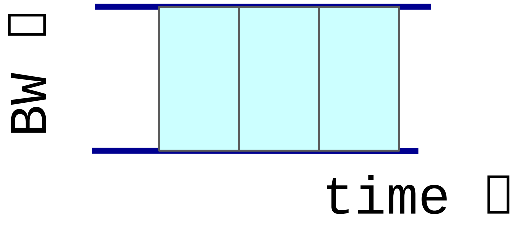


Packet Delay: The “pipe” view
Sending 100B packets from A to B?

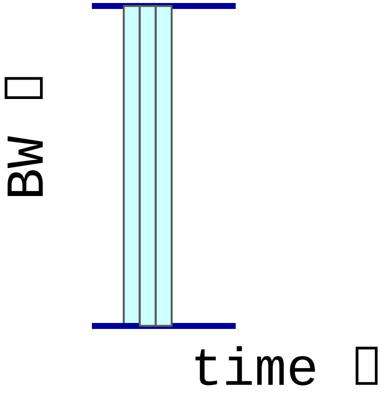
1Mbps, 10ms (BDP=10,000)



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



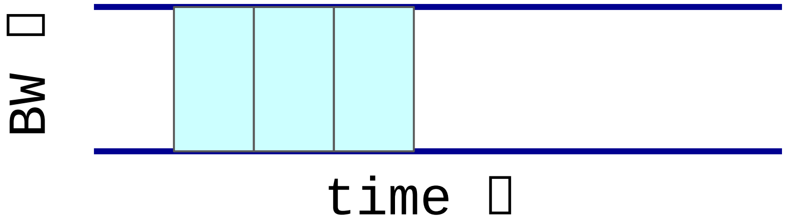
10Mbps, 1ms (BDP=10,000)



Packet Delay: The "pipe" view

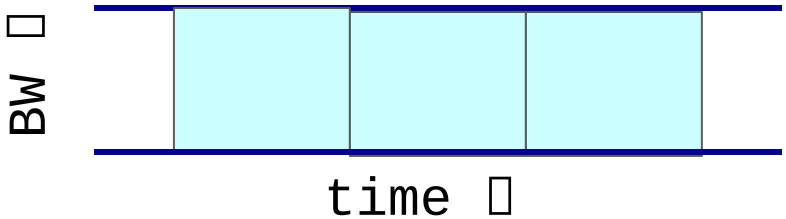
Sending 100B packets from A to B?

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

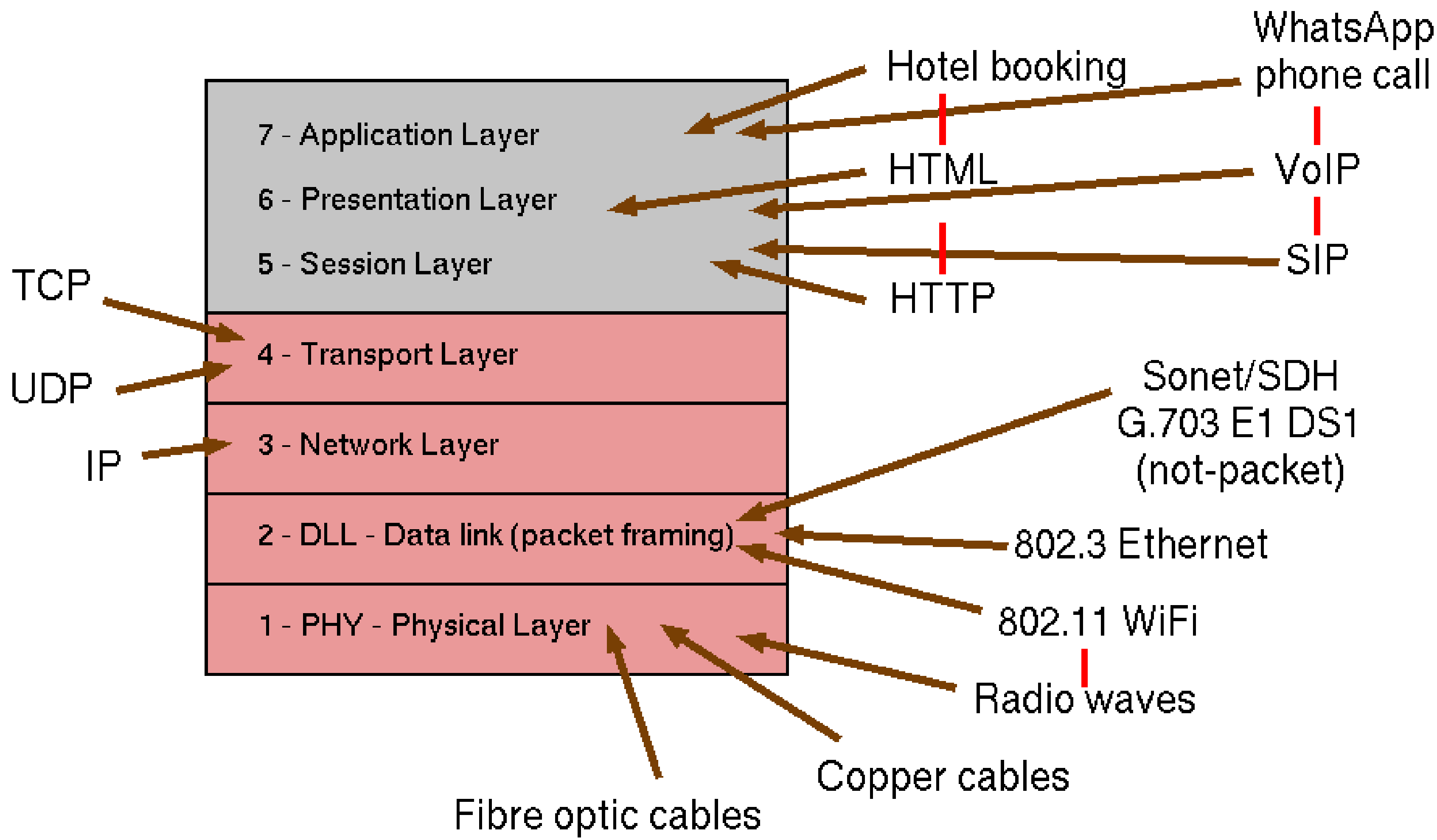


What if we used *200Byte packets??*

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

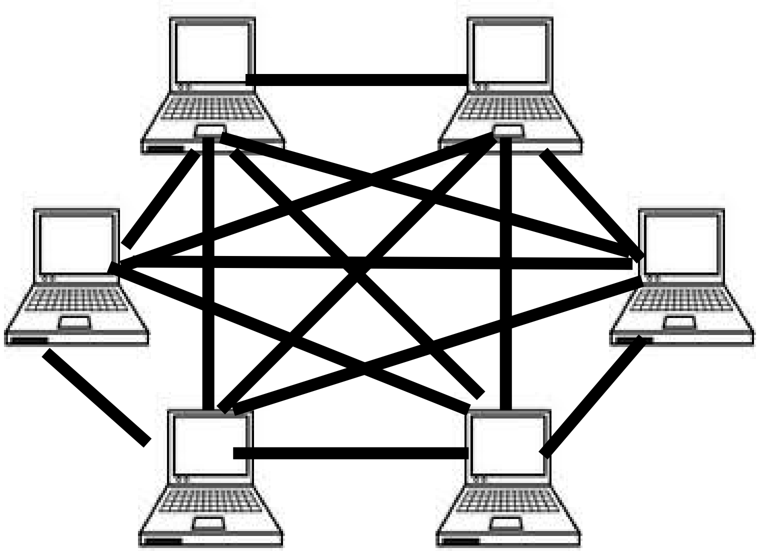


Can we ever talk about networking without talking about the OSI seven-layer reference model ?



What if we have more nodes?

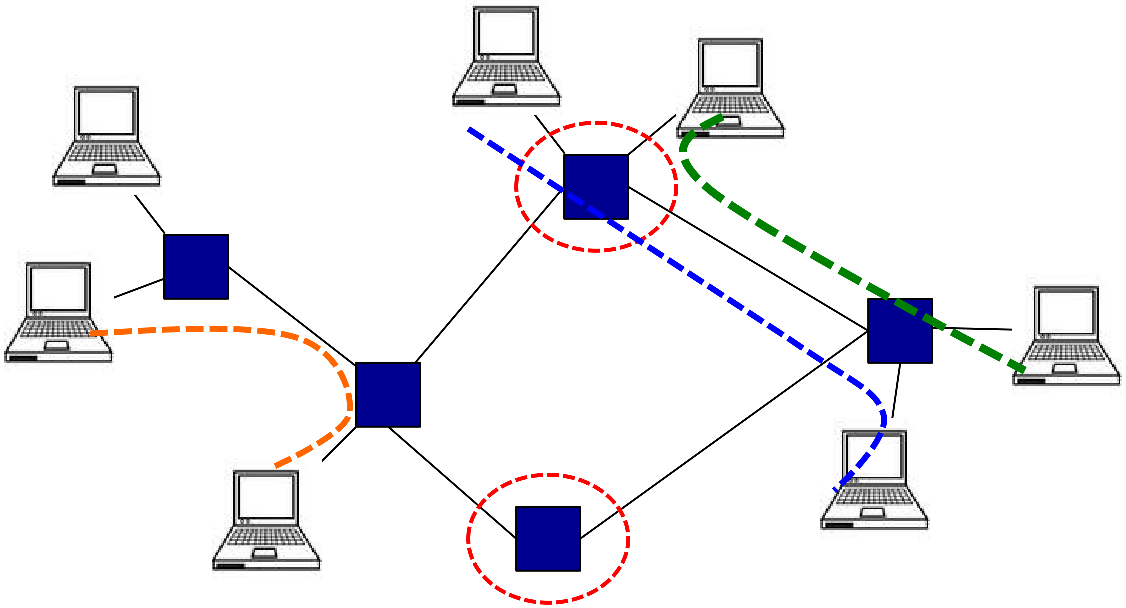
One link for every node?



Need a <u>scalable</u> way to interconnect nodes

Solution: A *switched* network

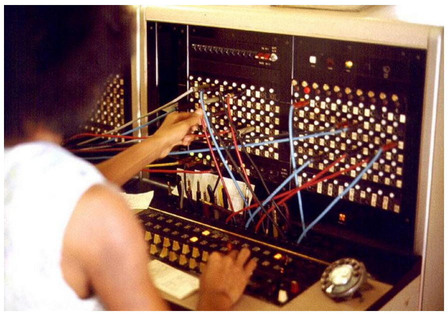
Nodes share network link resources

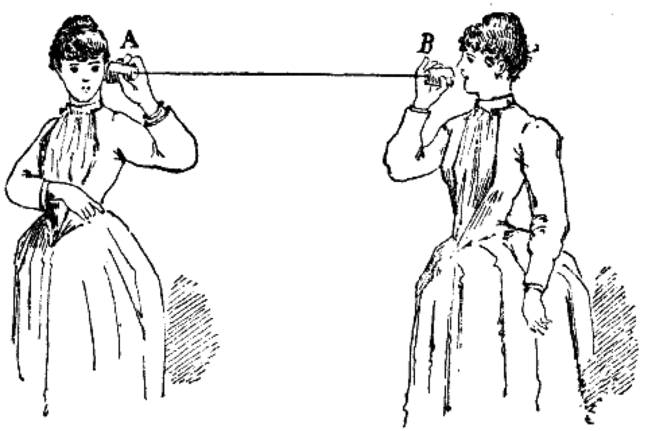


How is this sharing implemented?

Two network switching paradigms

- Circuit switching (used in the *POTS*: Plain Old Telephone Service) emphasis on old





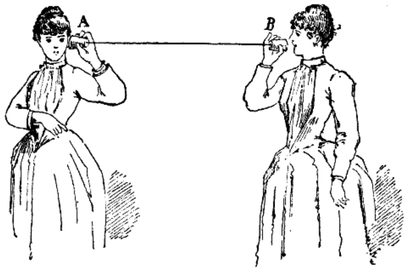
- Packet switching (used in the Internet)

Sub-paradigms

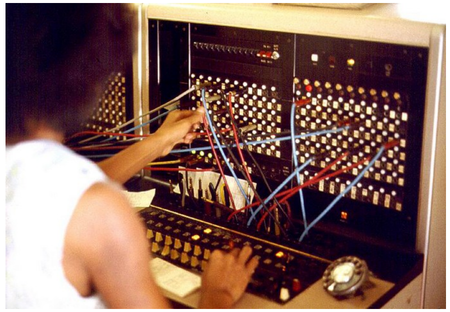
- Duplex – data flows in both directions
- Simplex – data flows in one direction only
  - Broadcast/multicast/anycast variations
- Half-duplex (often time-division duplex) – take it in turns to send in each direction
  - Also frequency-division duplexing etc..

Links considered single-cast duplex unless otherwise specified.

Circuit switching



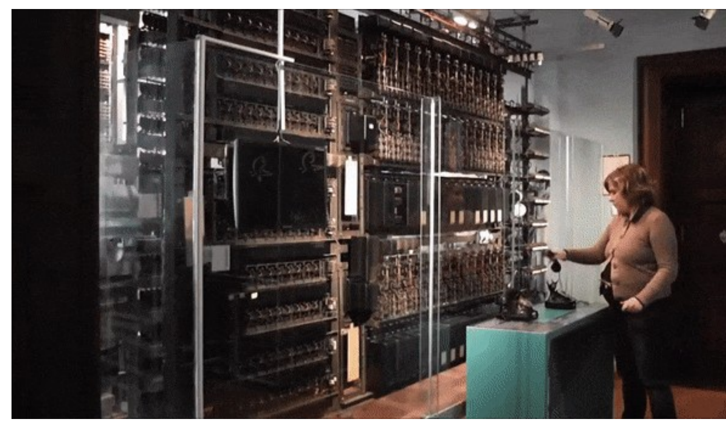
Telephone



Exchange

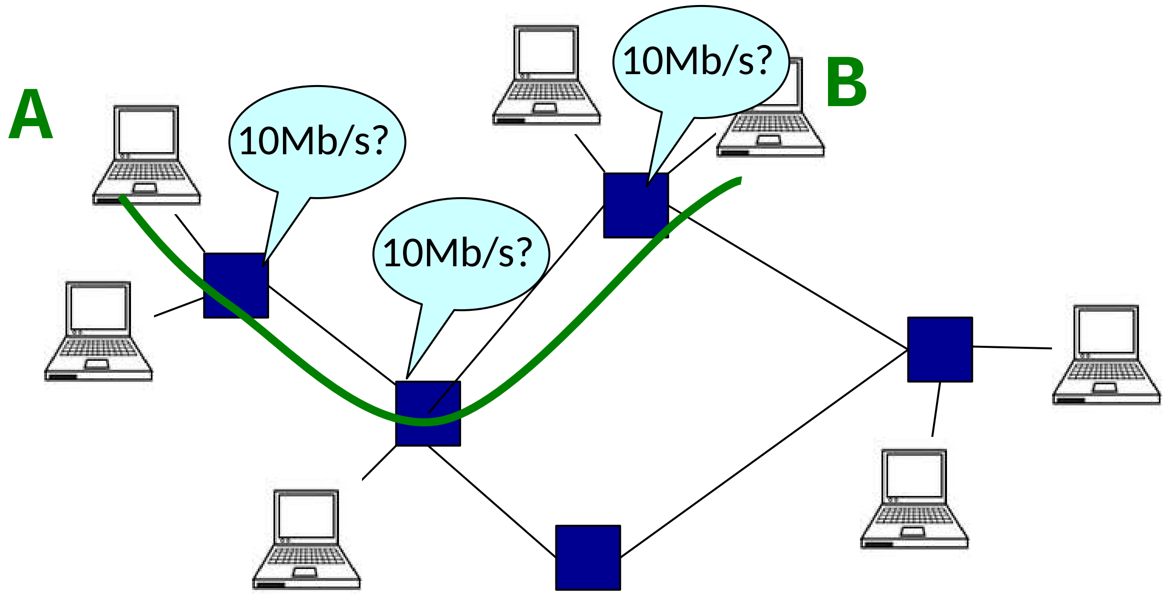
Exchange

Telephone



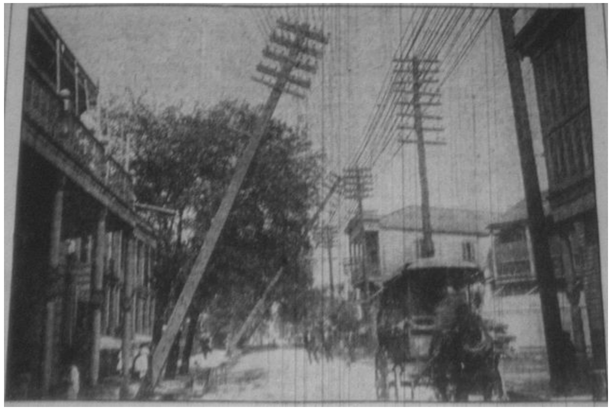
Circuit switching

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

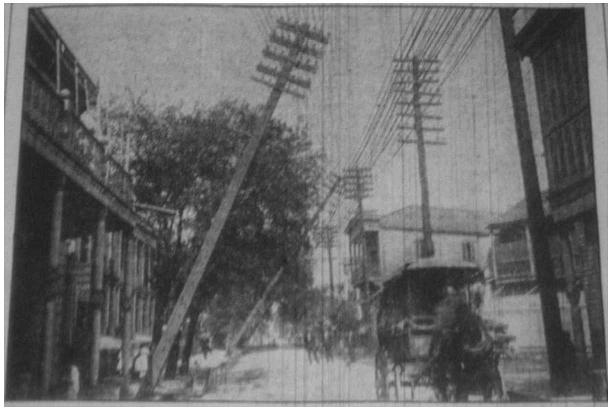
Multiplexing



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

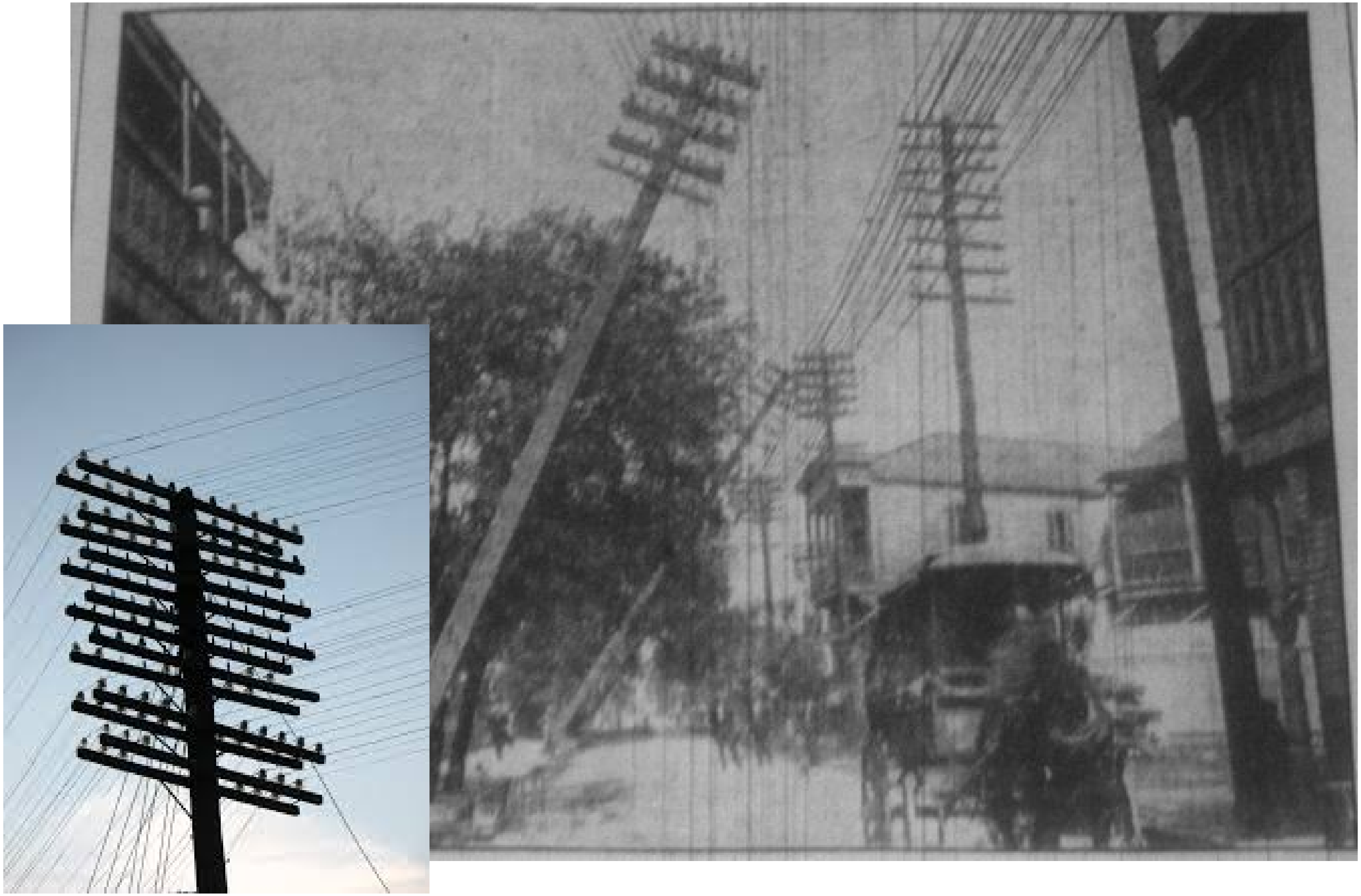
Multiplexing



Sharing makes things efficient (cost less)

- One airplane/train for 100’s of people
- One telephone for many calls
- One ~~lecture theatre~~ Lecturer? for many classes
- One computer for many tasks
- One network for many computers
- One datacenter many applications

Old Time (spatial) Multiplexing

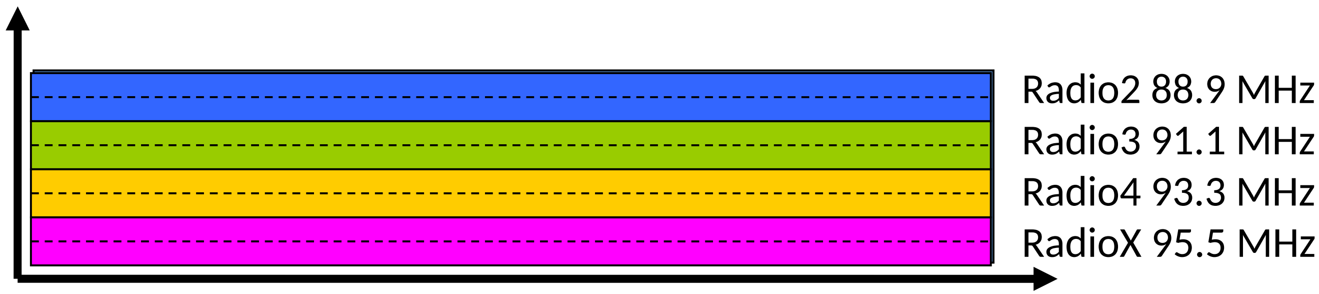


Sharing Circuit Switching: FDM and TDM

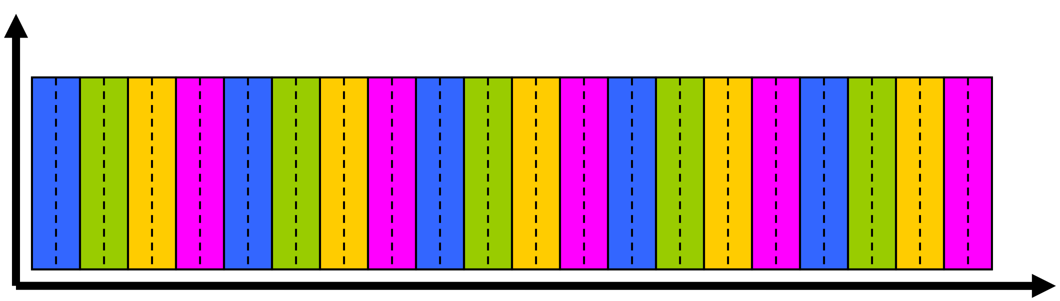
Example:

4 users

Frequency Division Multiplexing

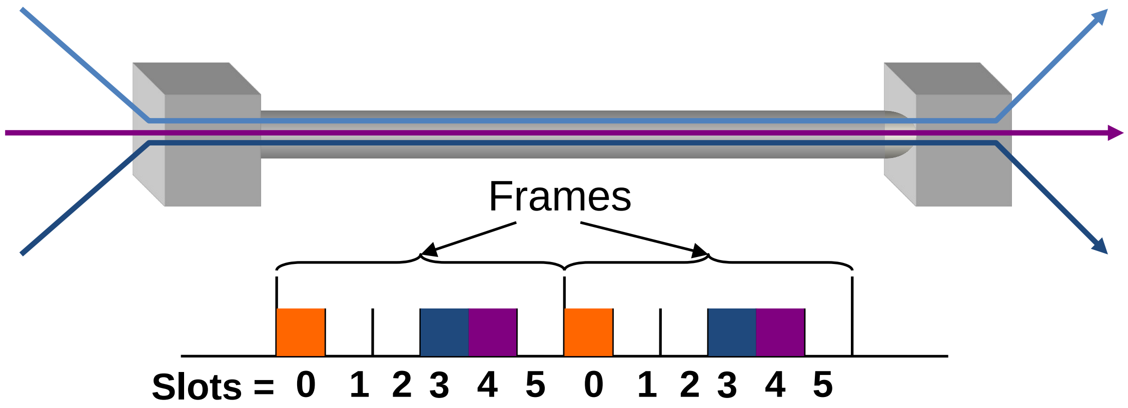


Time Division Multiplexing



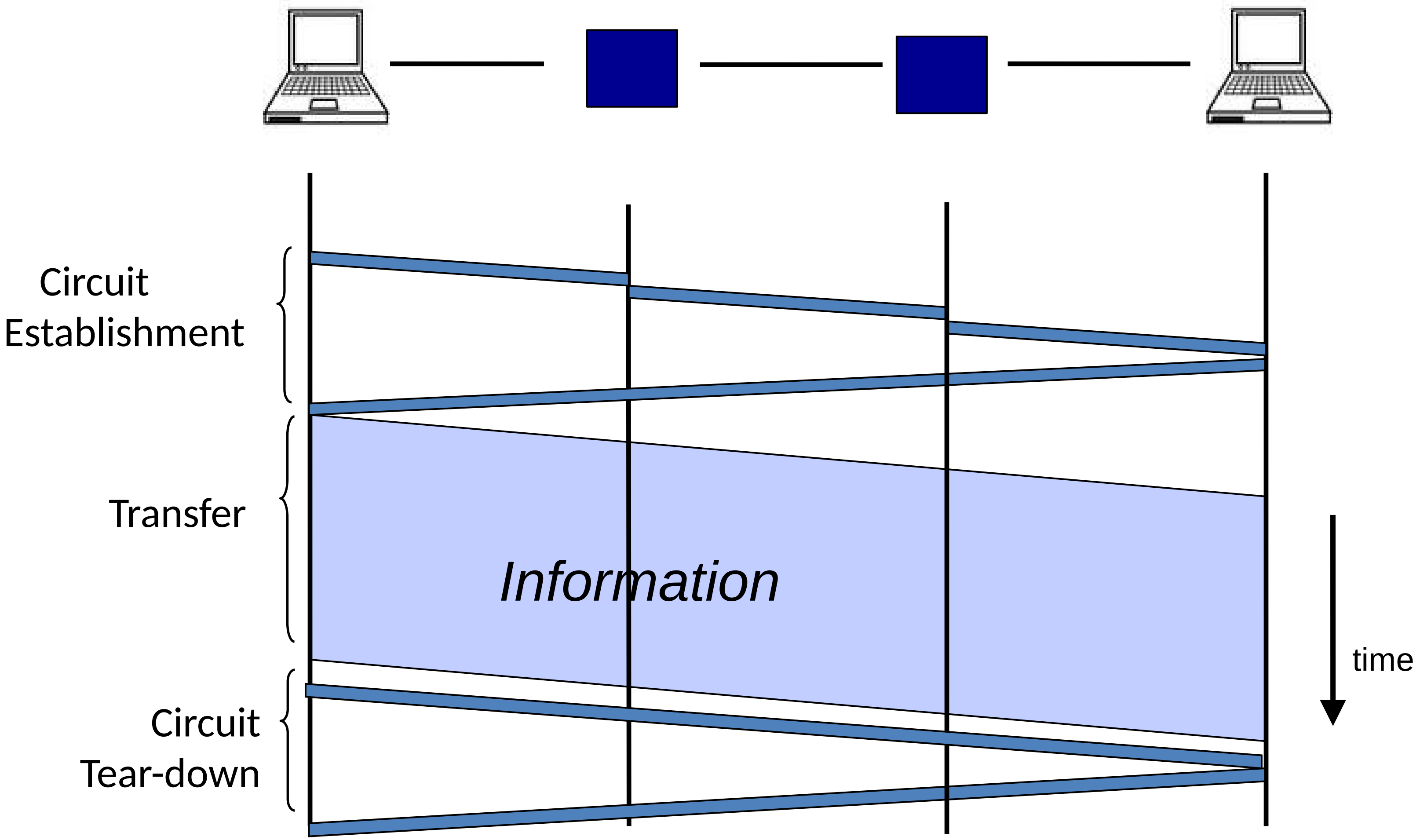
Radio schedule is very coarse TDM:
…,News, Sports, Weather, Local, News, Sports,…

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 wasted!**

Timing in Circuit Switching

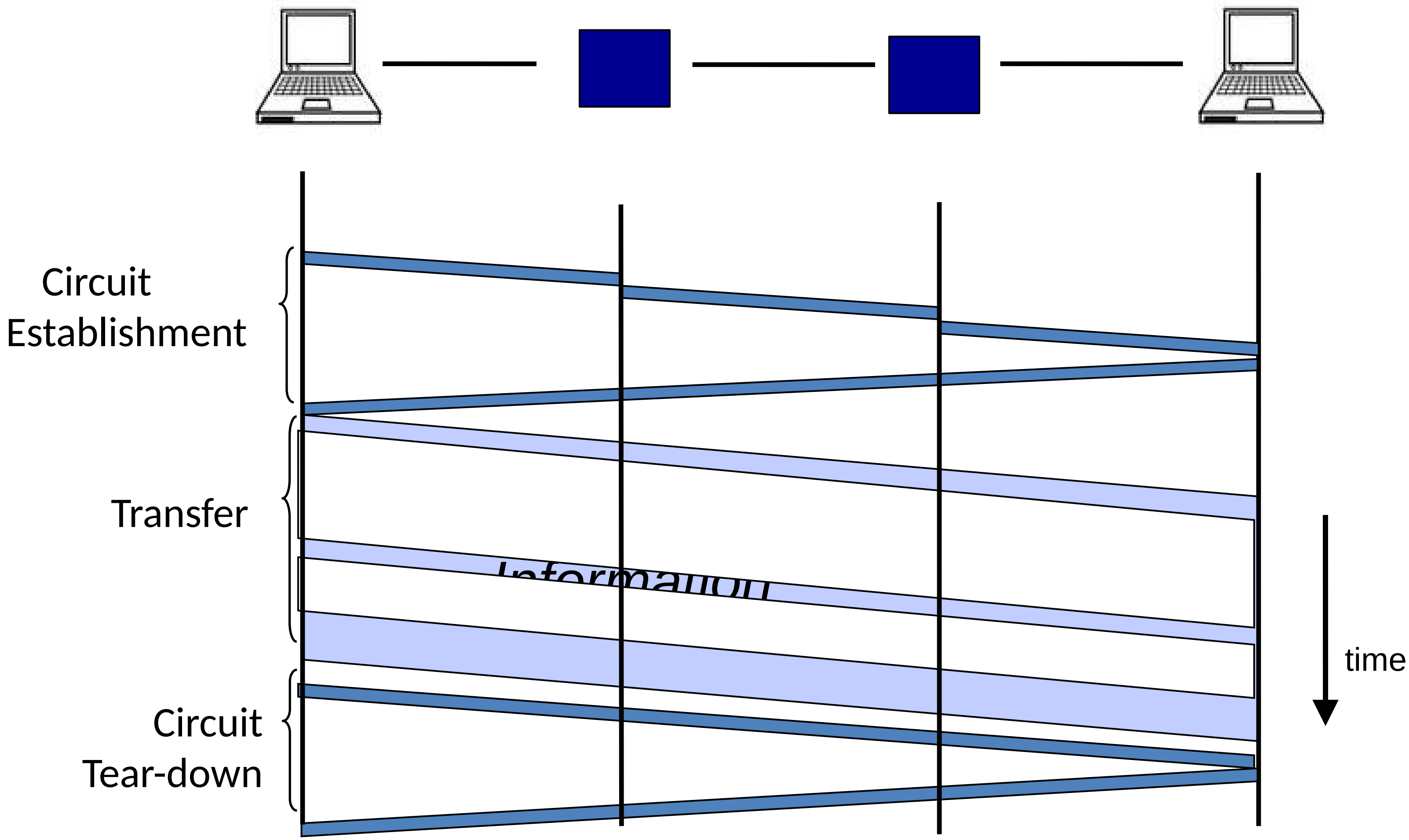


Circuit switching: pros and cons

- Pros
  - guaranteed performance
  - fast transfer (once circuit is established)

- Cons

Timing in Circuit Switching



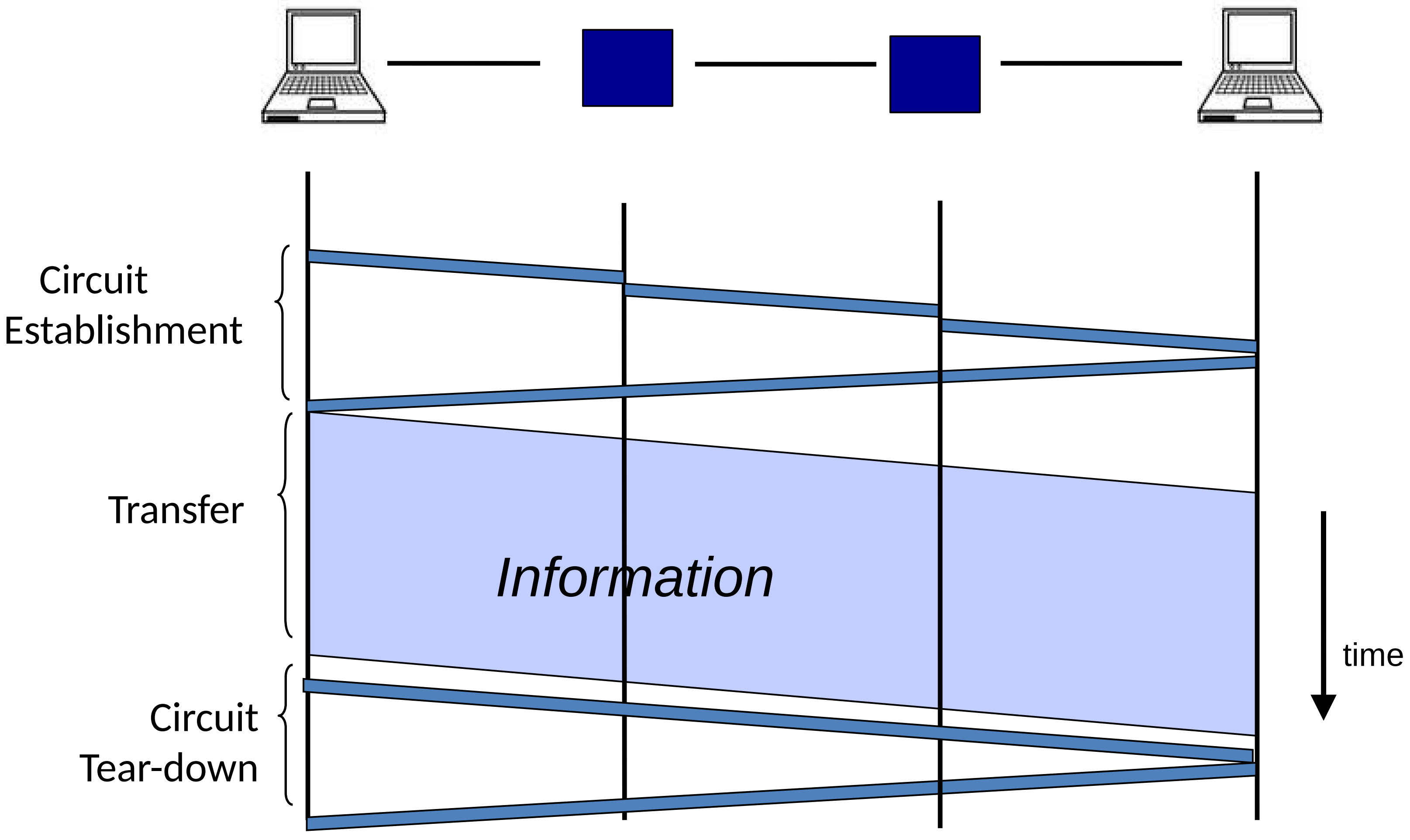
Circuit switching: pros and cons

- Pros
  - guaranteed performance
  - fast transfer (once circuit is established)

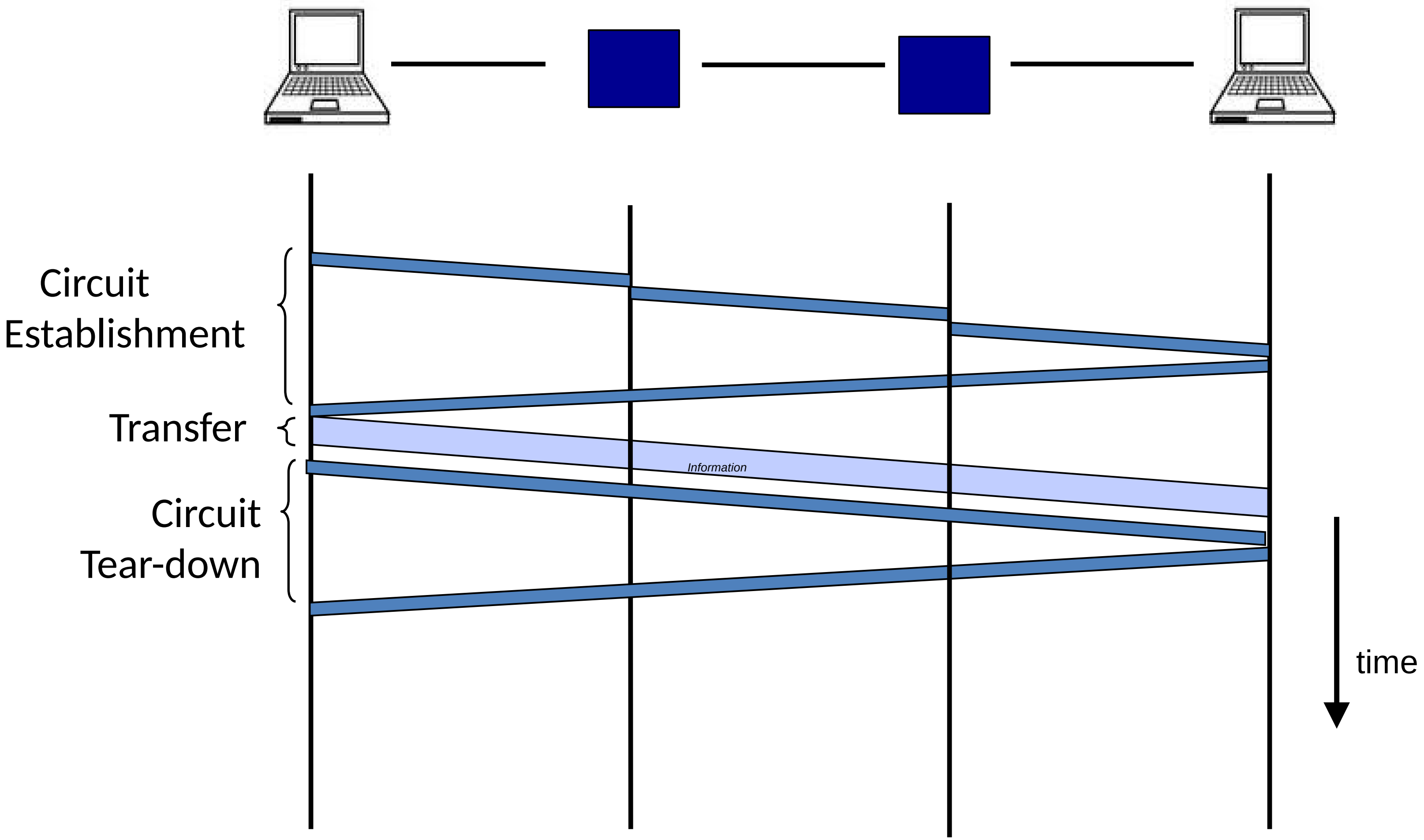
- Cons
  - **wastes bandwidth if traffic is “bursty”**

Traditional computer data is very, very bursty (peak/average >> 1000)
Multimedia (voice/video/music) is far less bursty.

Timing in Circuit Switching



Timing in Circuit Switching

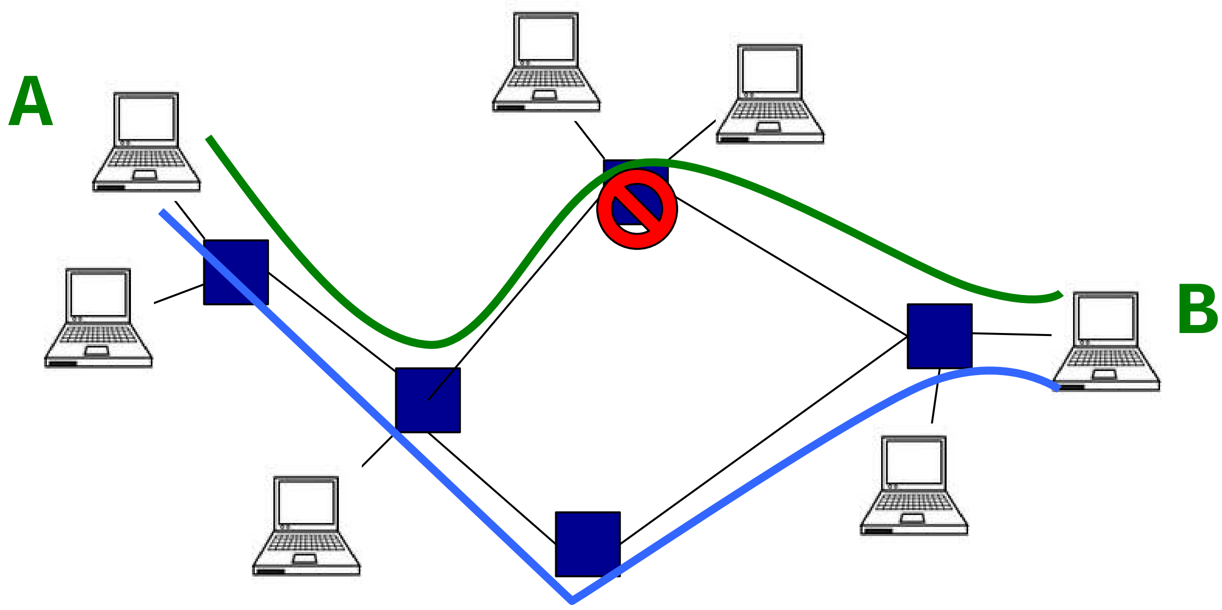


Circuit switching: pros and cons

- Pros
  - guaranteed performance
  - fast transfers (once circuit is established)

- Cons
  - wastes bandwidth if traffic is “bursty”
  - **connection setup time is overhead**

Circuit switching



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 slot frame (DS1)
  - 500 msec to establish end-to-end circuit

Let's work it out!

Two examples of switched networks

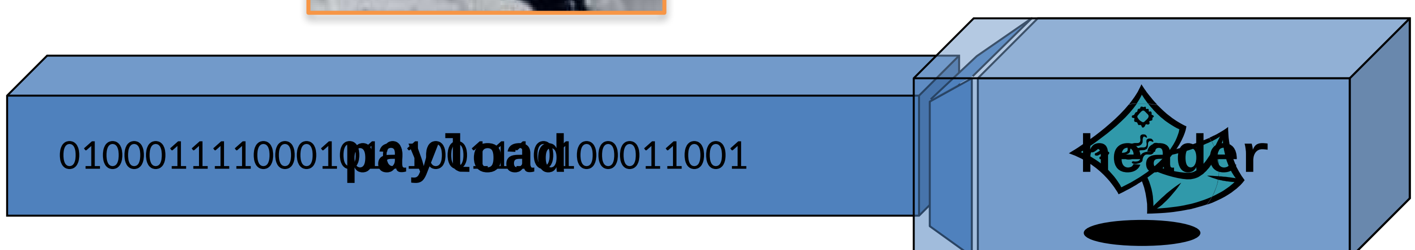
- Circuit switching (used in the *POTS*: Plain Old Telephone system)

- Packet switching (used in the Internet)


Packet Switching

- Data is sent as chunks of formatted bits (Packets)
- Packets consist of a "header" and "payload"*

Destination Address



Connectionless 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
  - 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)
    - Do I acknowledge this? Who signed for it?
    - Check digits to determine contents intact?

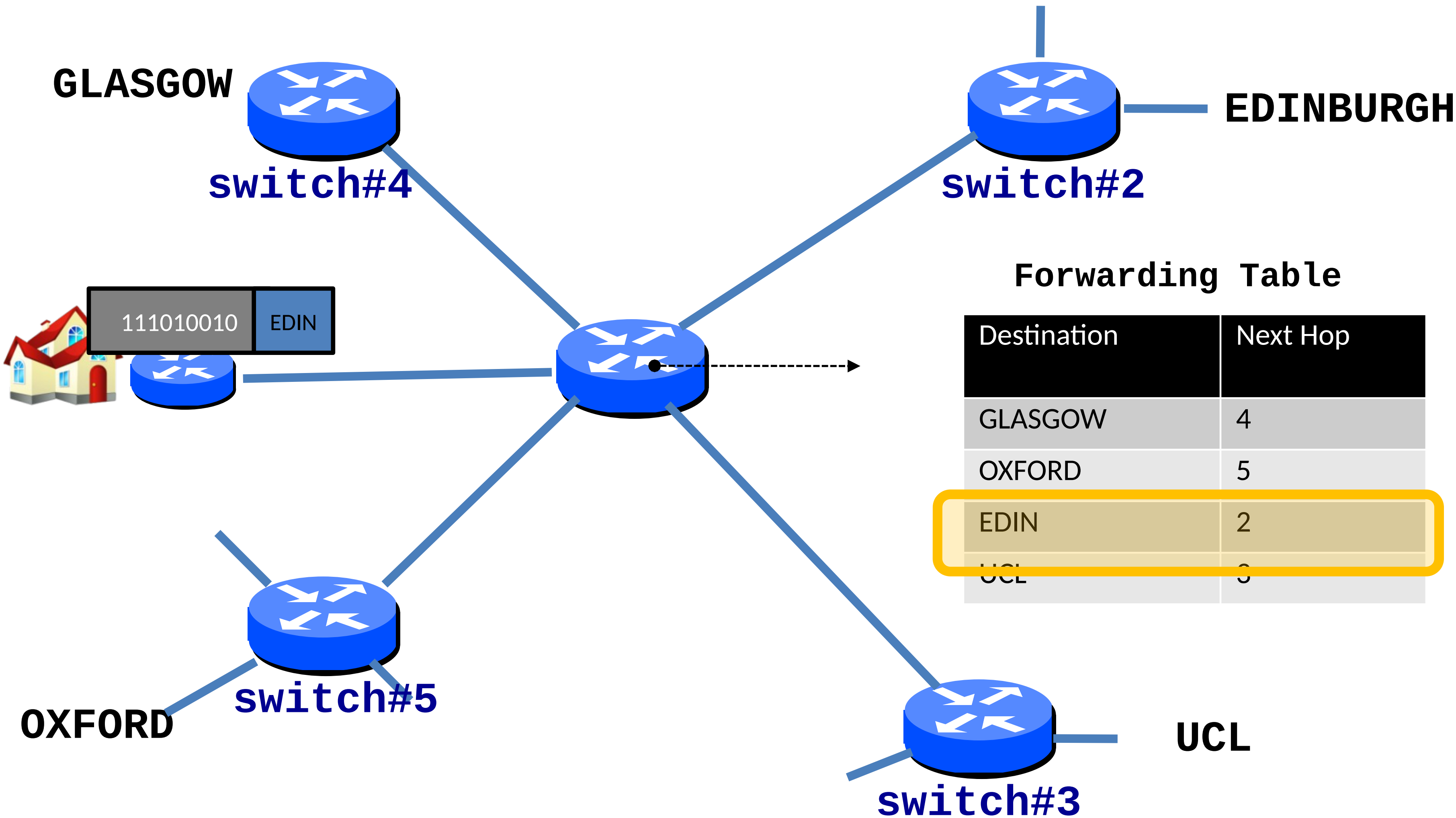
Packet Switching

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

A switch looks at the header and immediately decides which physical port to send it out on.

→ Address somehow implies output port no.

Switches forward packets

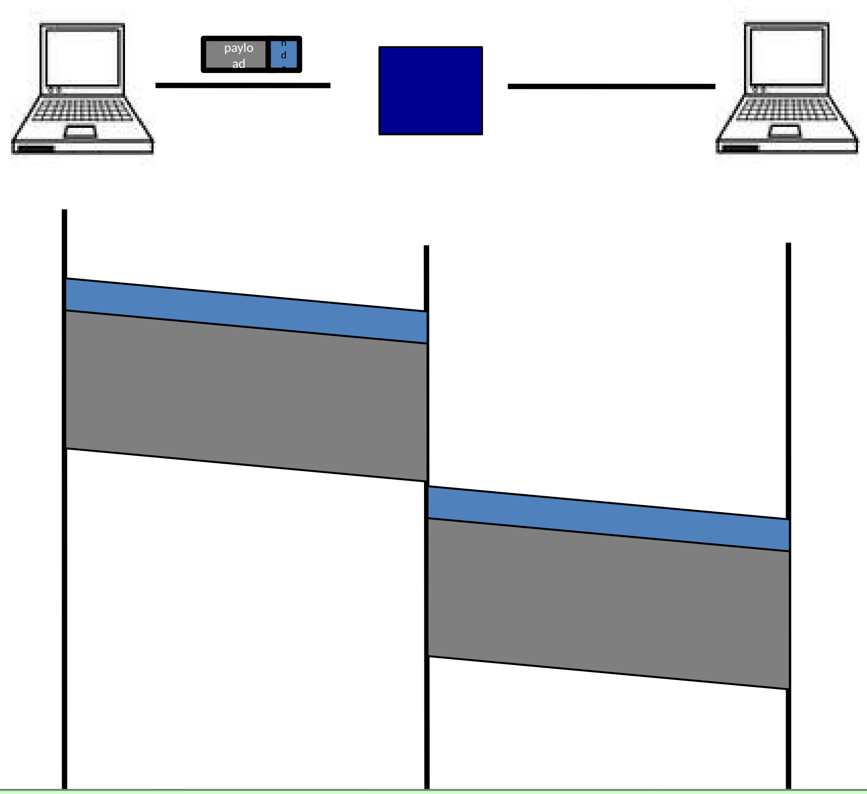


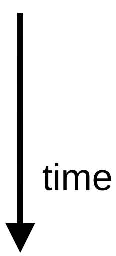
Forwarding Table

Destination	Next Hop
GLASGOW	4
OXFORD	5
EDIN	2
UCL	3

Also need to know port_no to switch_no mapping.

Timing in Packet Switching

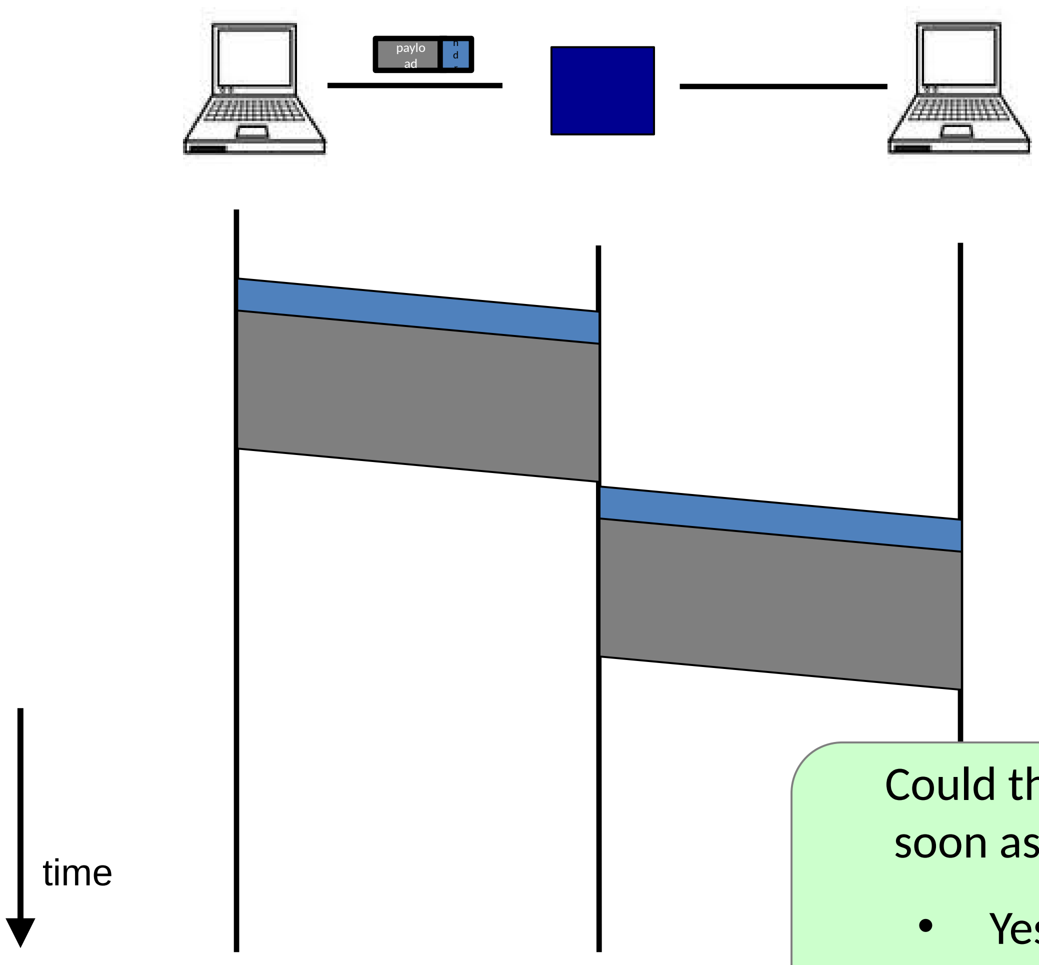




What about the time to process the packet at the switch?

- We’ll assume it’s relatively negligible (mostly true)

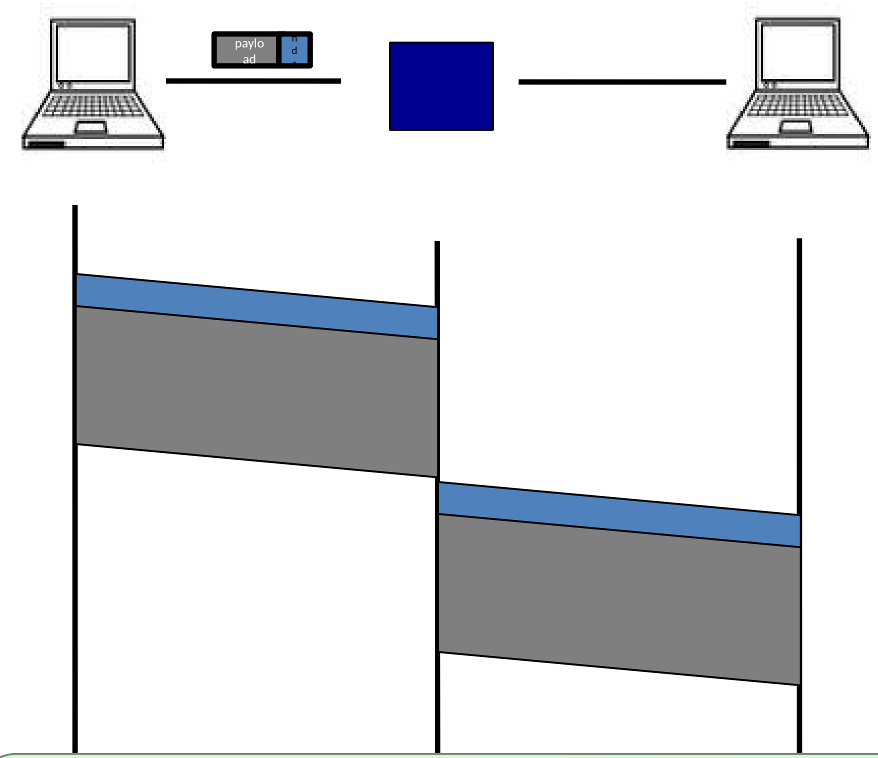
Timing in Packet Switching



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

- Yes! This is called “cut-through” switching (SoC/supercomputers).

Timing in Packet Switching



time

We’ll normally assume a switch processes/forwards
a packet after it has received it entirely.
This is called “store-and-forward” switching

Connectionless 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

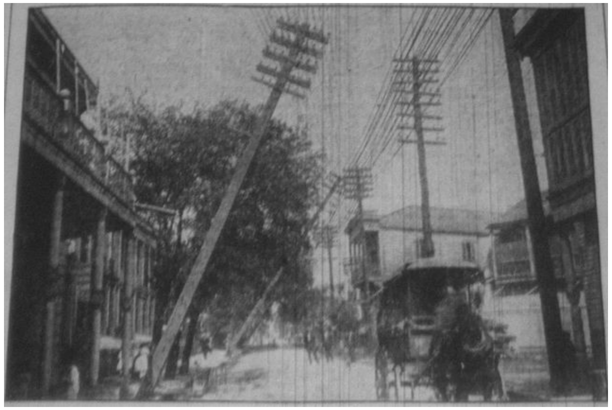
Connectionless Packet Switching

- 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”

Connectionless 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 need be reserved in advance. Instead packet switching leverages statistical multiplexing (stat muxing)

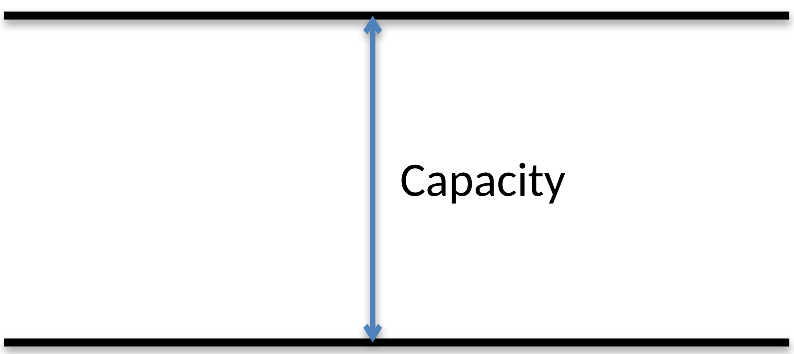
Multiplexing



Sharing makes things efficient (cost less)

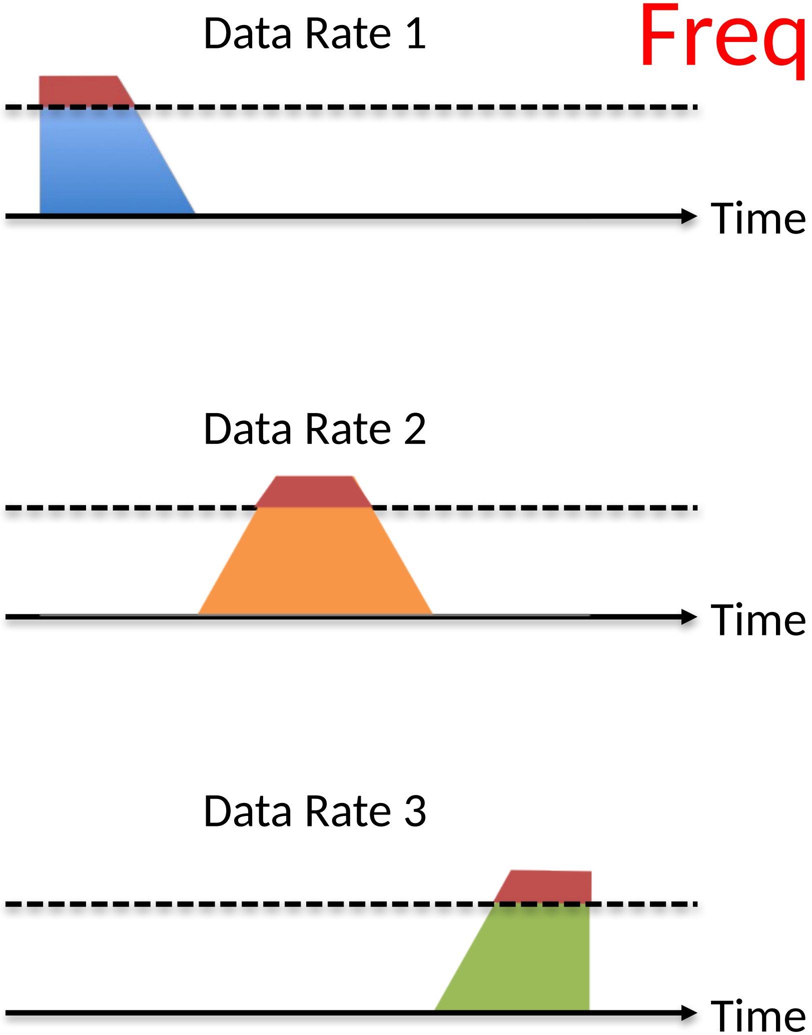
- One airplane/train for 100's of people
- One telephone for many calls
- One lecture theatre for many classes
- One computer for many tasks
- One network for many computers
- One datacenter many applications

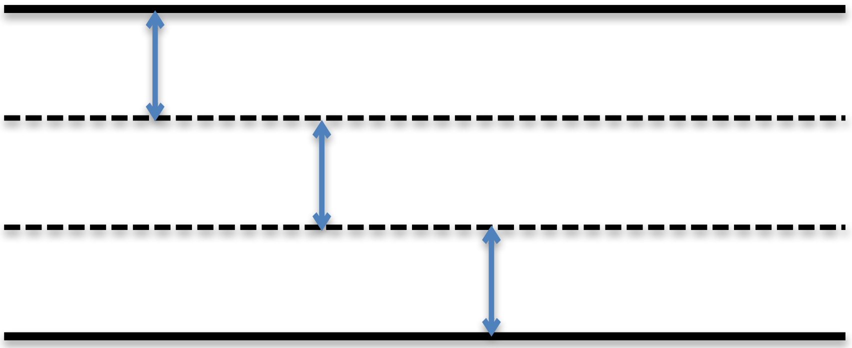
Three Flows with Bursty Traffic



Static allocation: give each flow 1/3rd of channel capacity

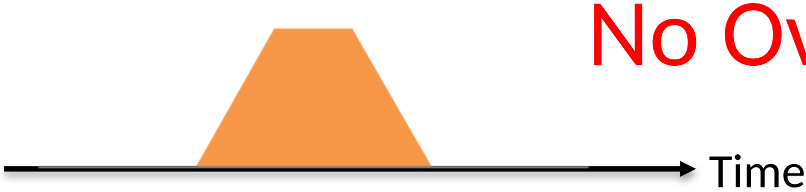
Frequent Overloading





When Flows Share Total Capacity

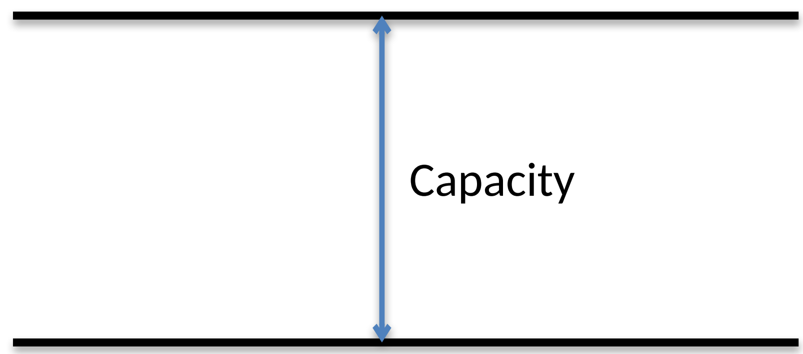
No Overloading



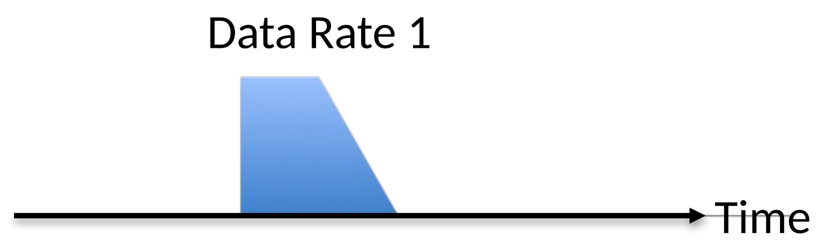
Statistical multiplexing relies on the probability of not all flows bursting at the same time.

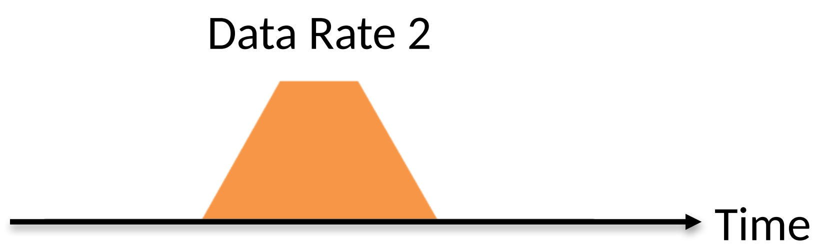
Very similar to insurance, and has same failure case.

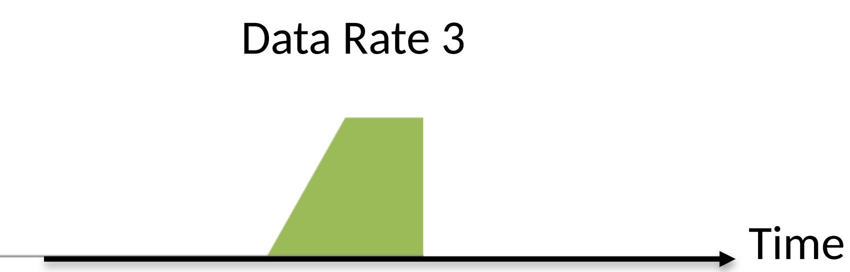
Three Flows with Bursty Traffic

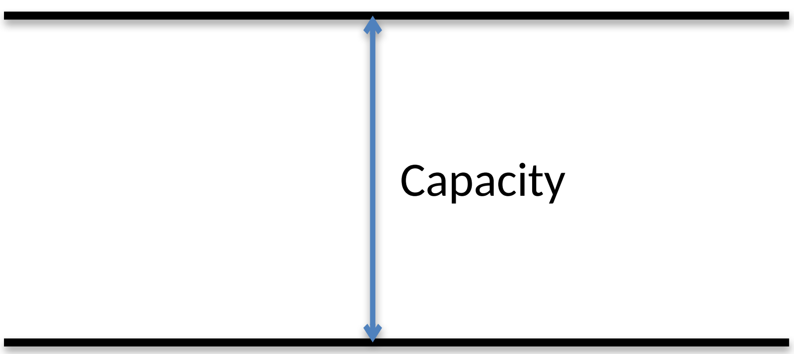


Three Flows with Bursty Traffic

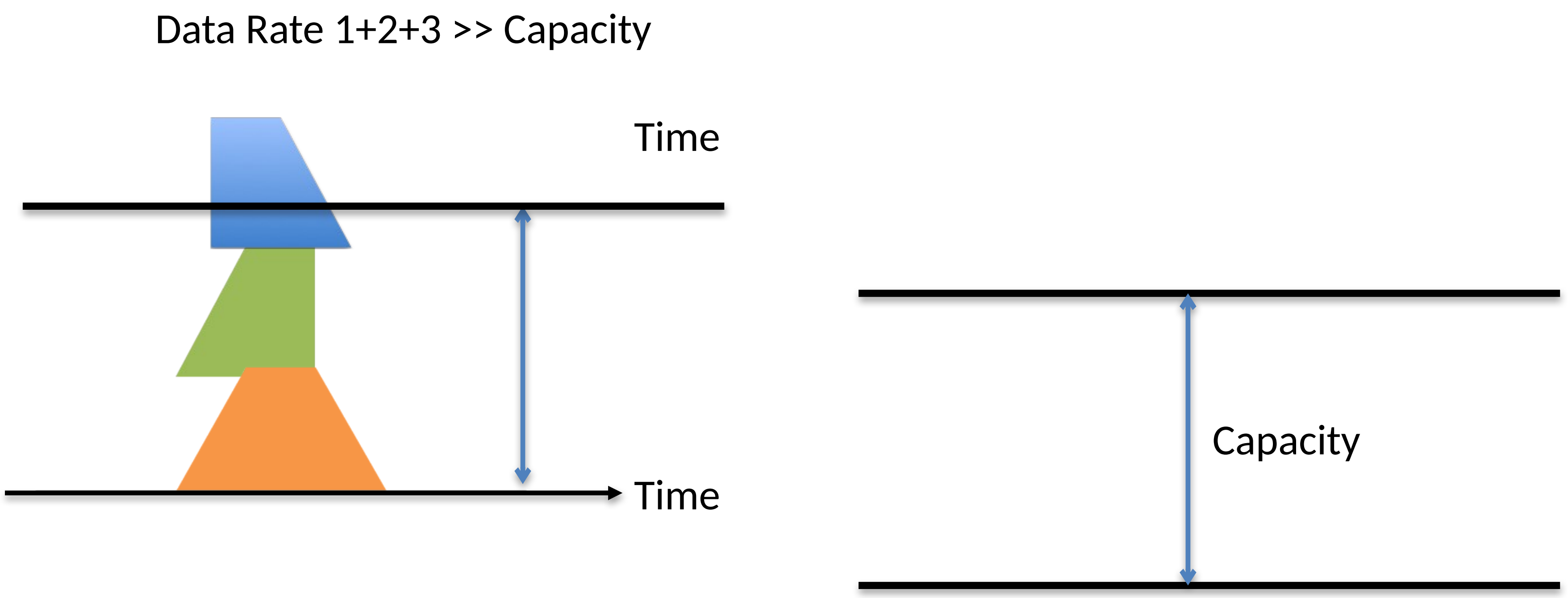






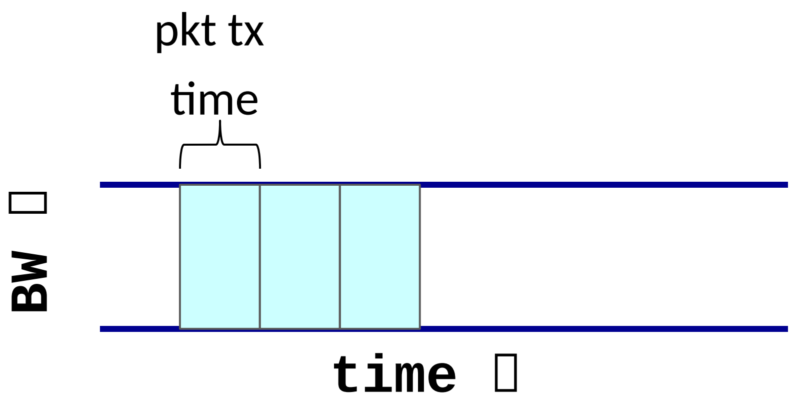


Three Flows with Bursty Traffic

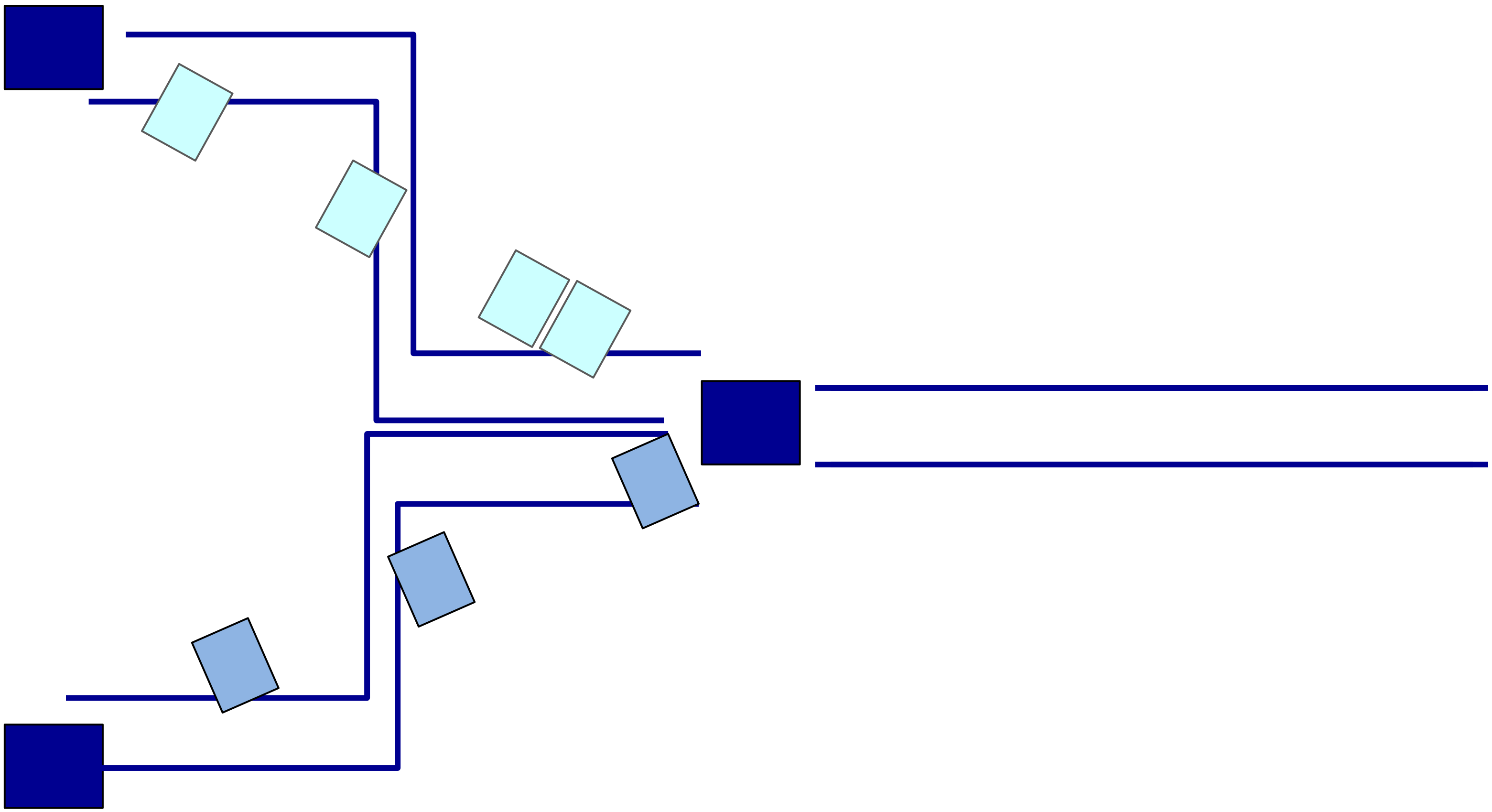


How to handle temporary overload?

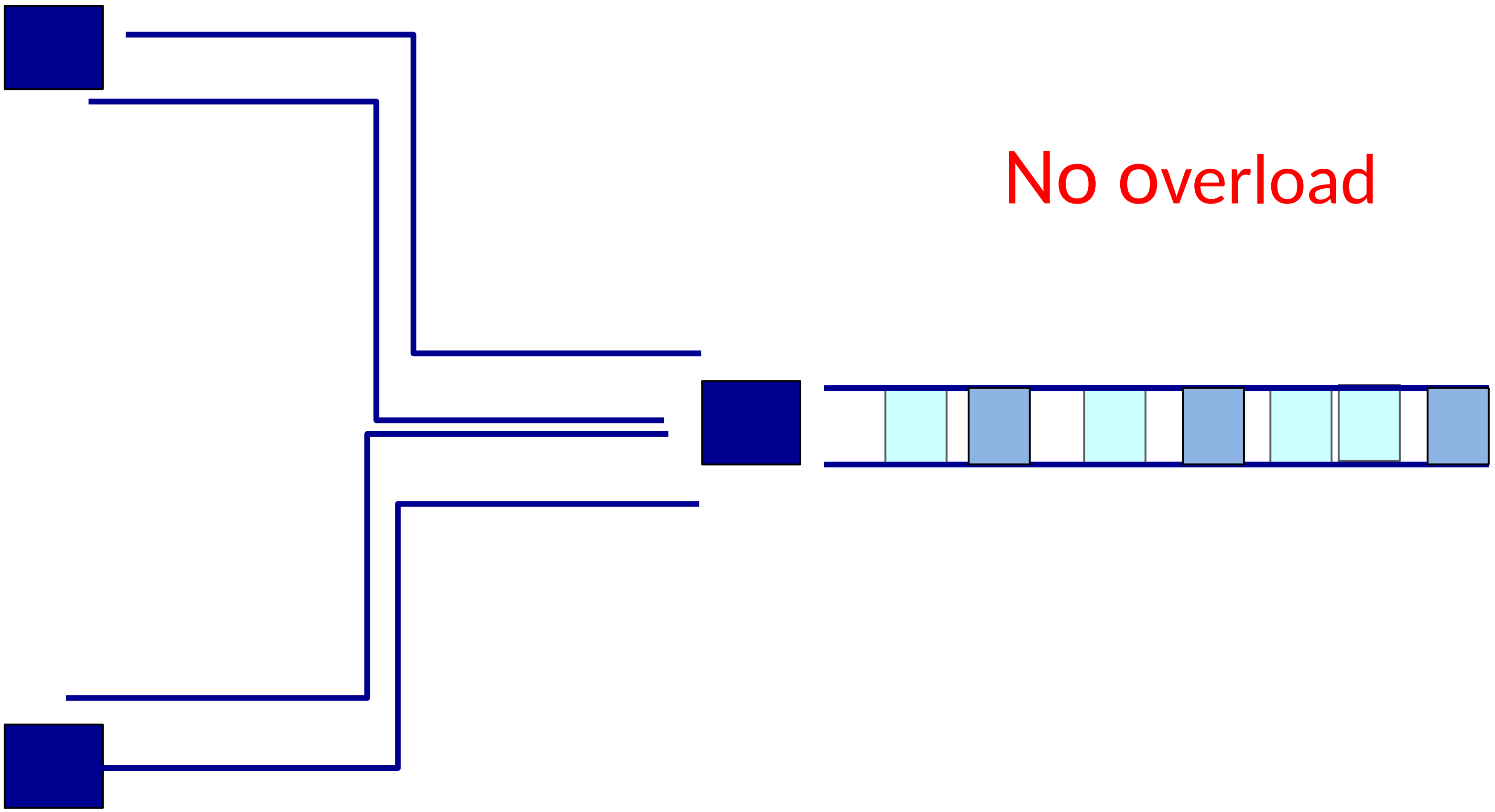
Statistical multiplexing: pipe view



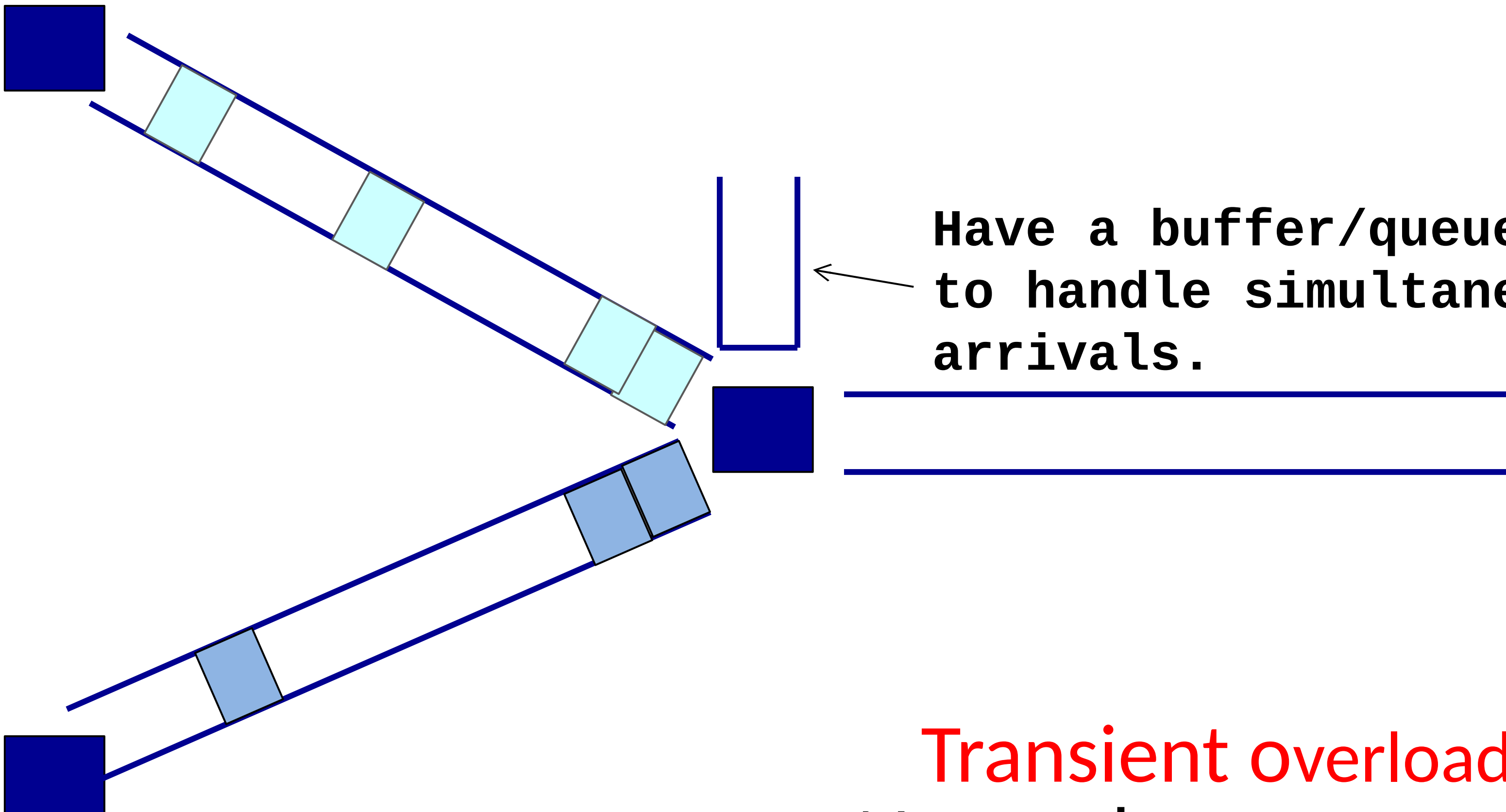
Statistical multiplexing: pipe view



Statistical multiplexing: pipe view



Statistical multiplexing: pipe view

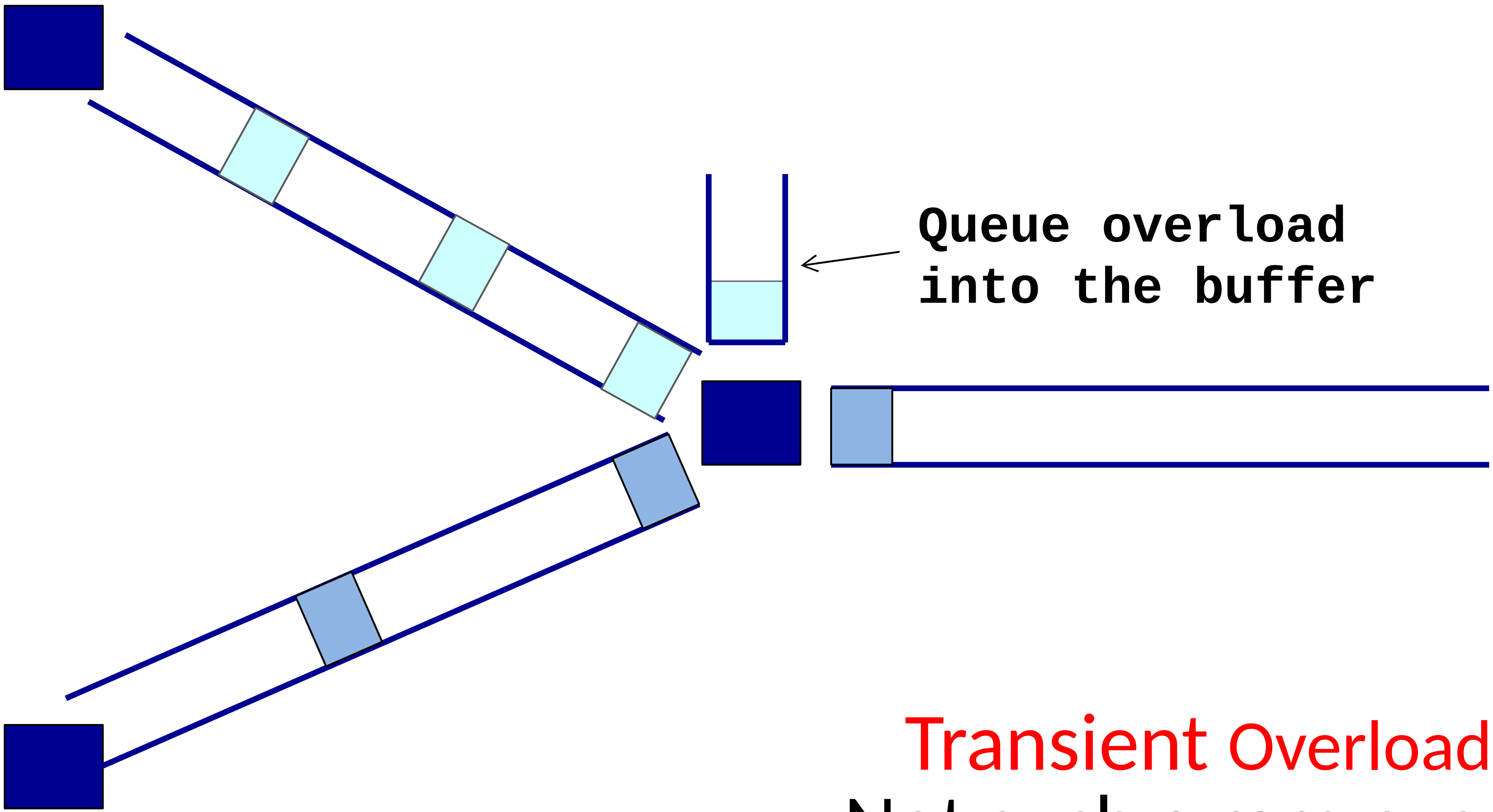


Have a buffer/queue to handle simultaneous arrivals.

Transient overload

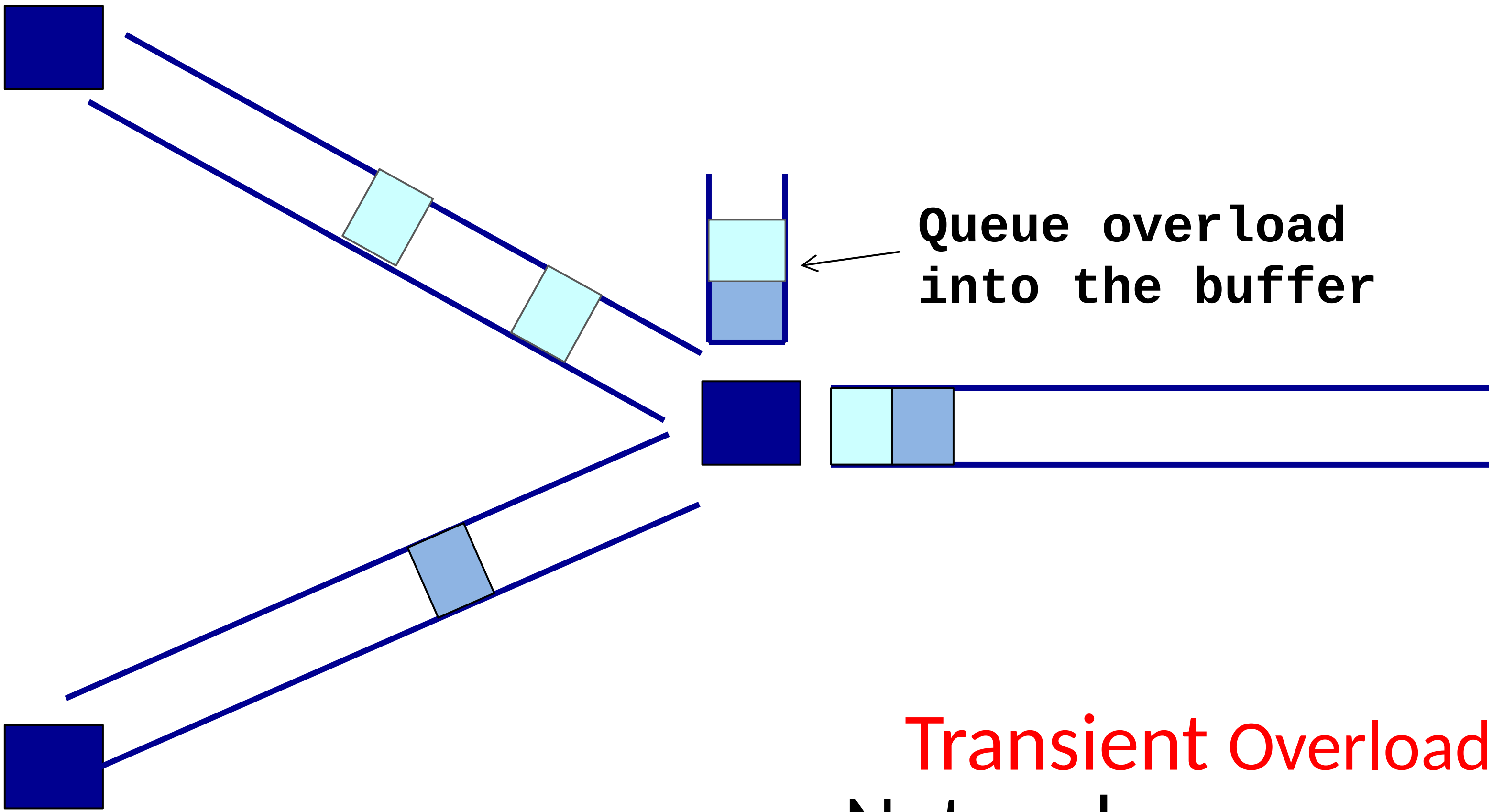
Not such a rare event

Statistical multiplexing: pipe view



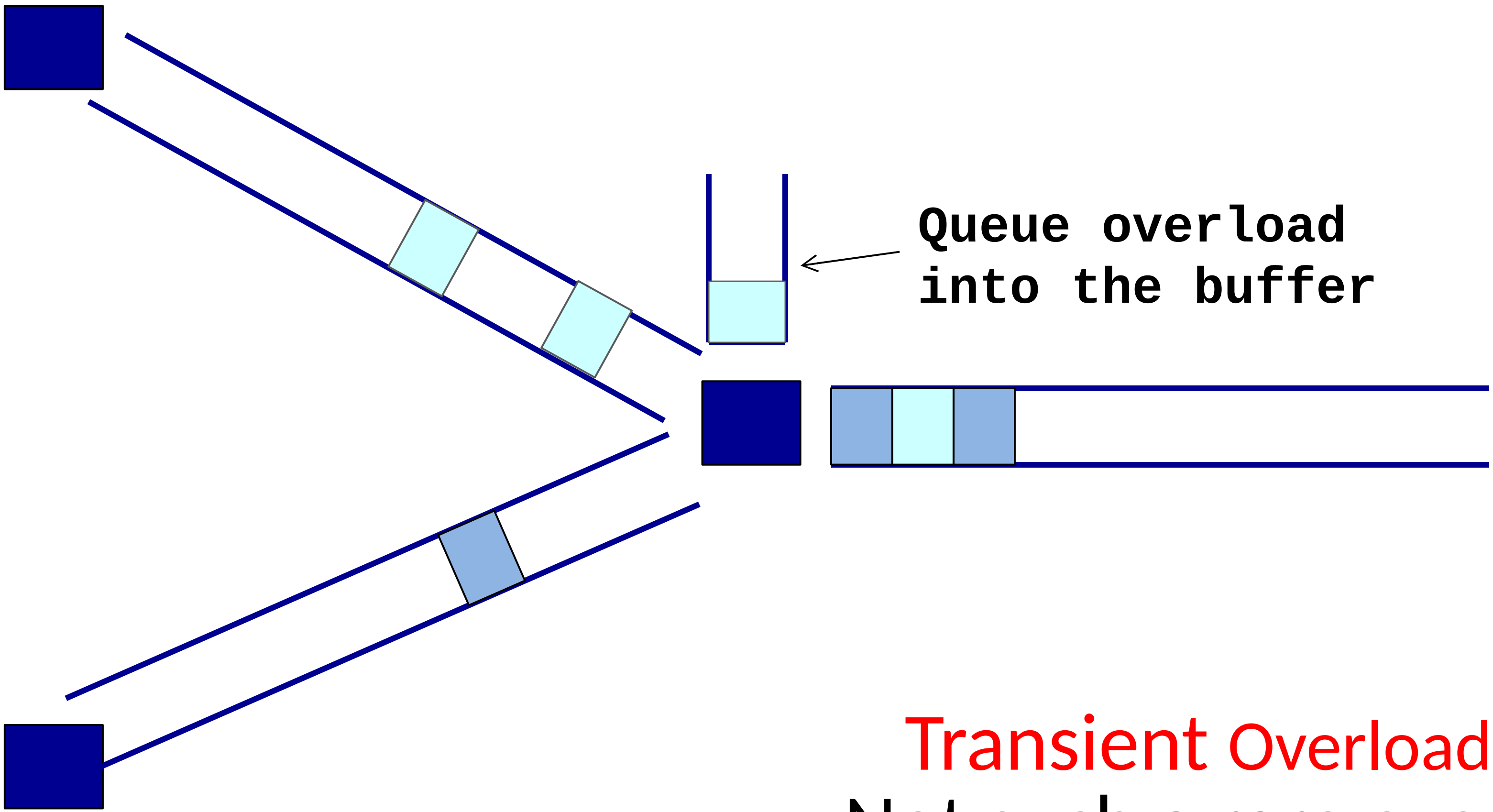
Transient Overload
Not such a rare event

Statistical multiplexing: pipe view



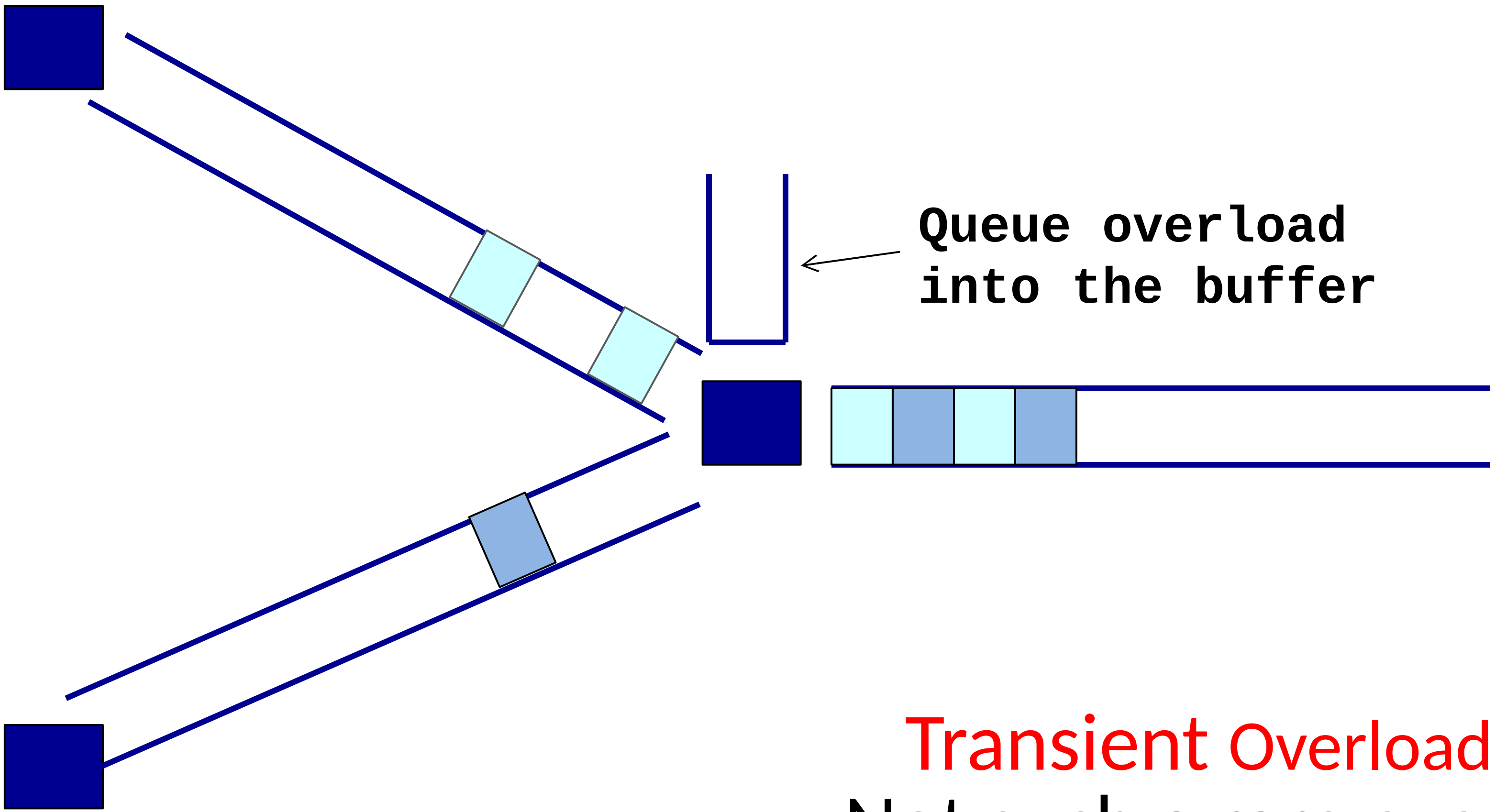
Transient Overload
Not such a rare event

Statistical multiplexing: pipe view



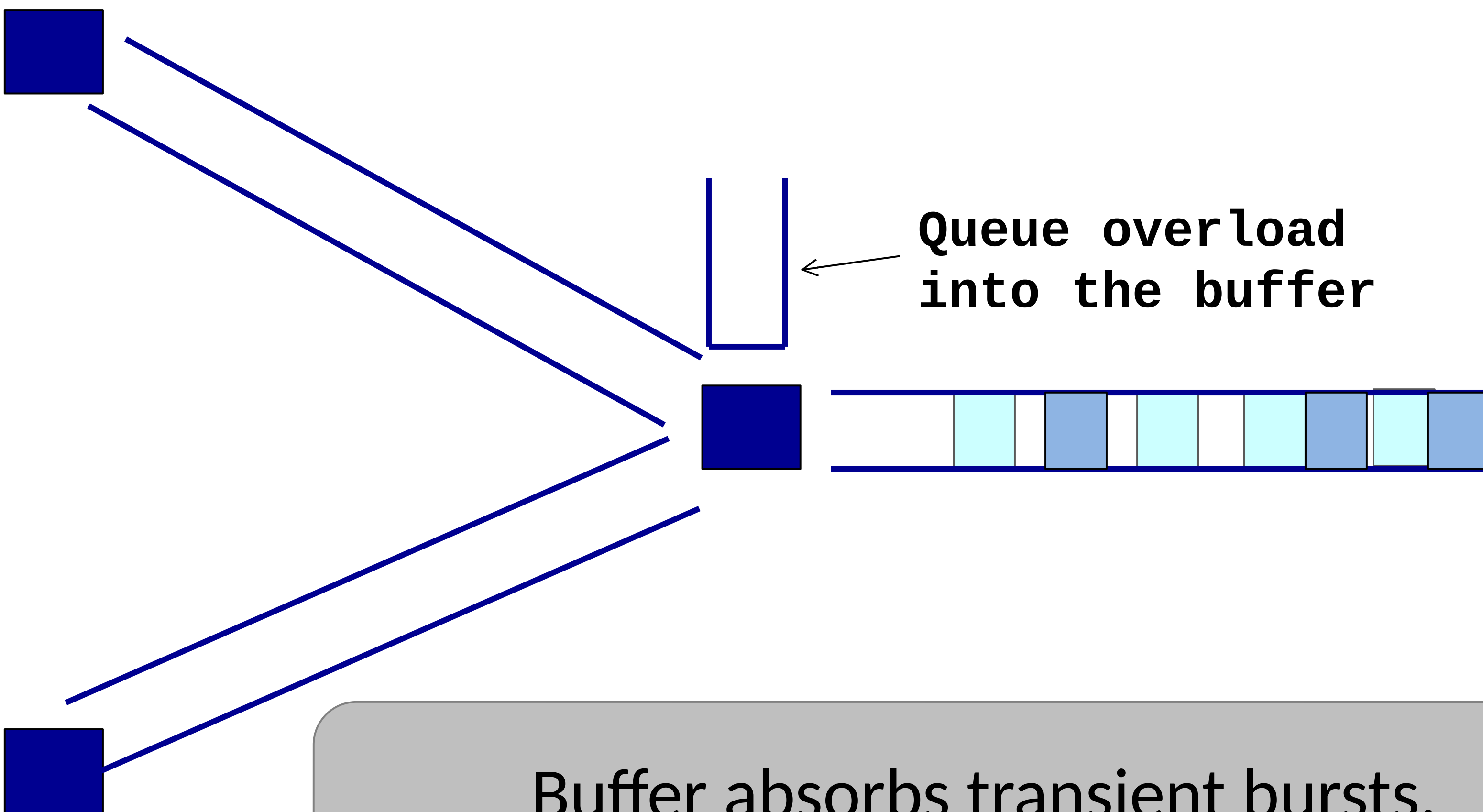
Transient Overload
Not such a rare event

Statistical multiplexing: pipe view



Transient Overload
Not such a rare event

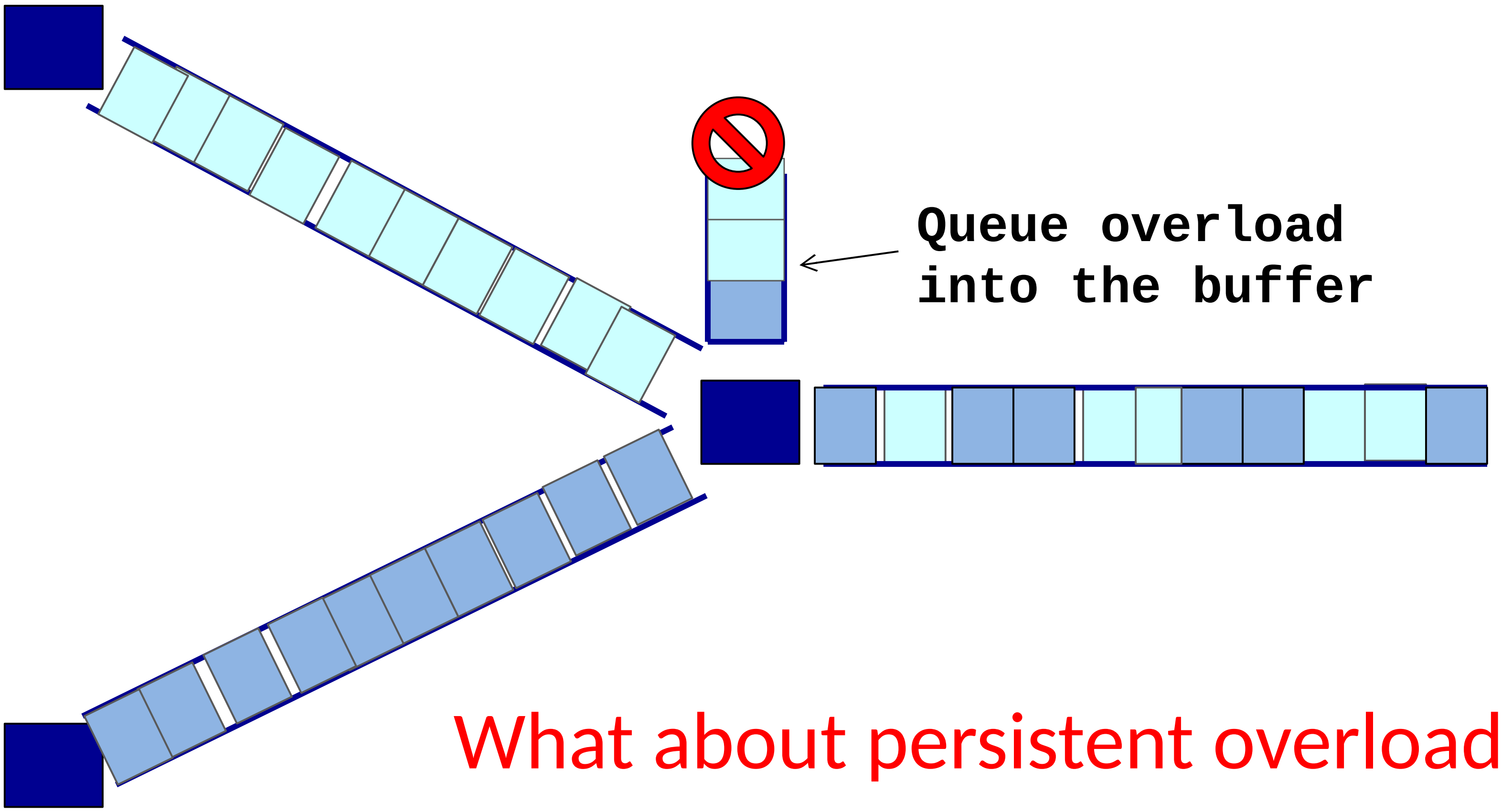
Statistical multiplexing: pipe view



Buffer absorbs transient bursts.

But it has no effect on average flow rates.

Statistical multiplexing: pipe view



What about persistent overload?

Eventually must drop packets.

Queues introduce queuing delays

- Recall,

  packet delay = transmission delay + propagation delay (*)

- With queues (statistical multiplexing)

  packet delay = transmission delay + propagation delay + queuing delay (*)

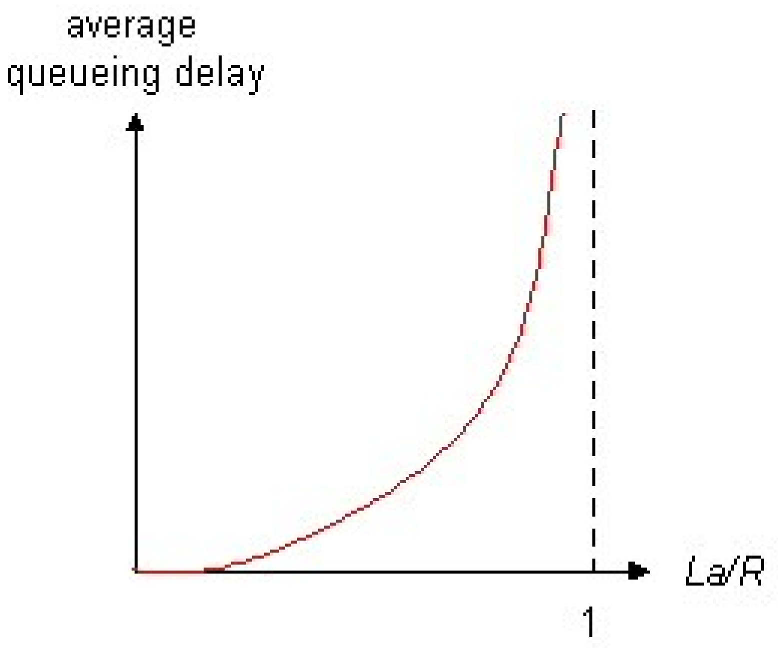
- Queuing delay caused by “packet interference”

- 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



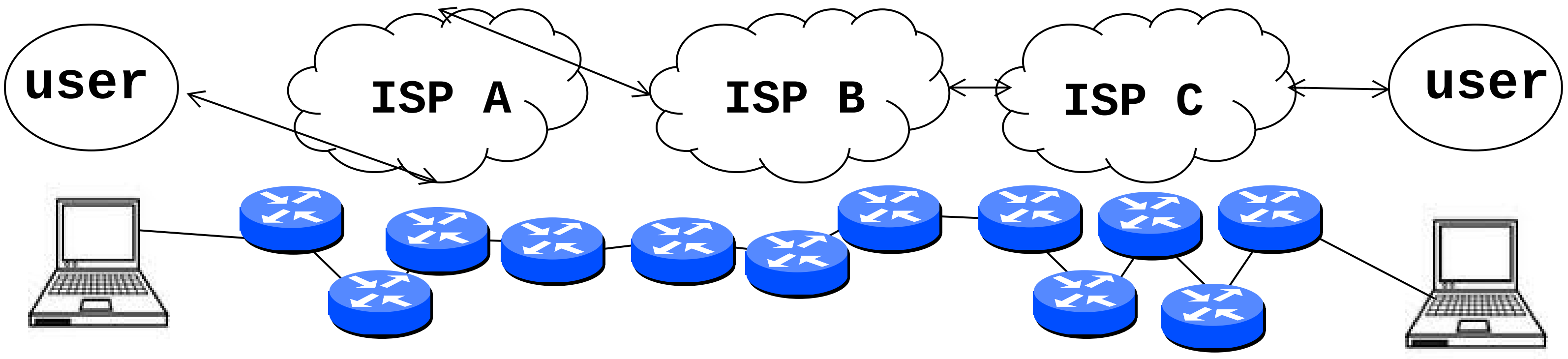
Relative traffic intensity = La/R

Also known as the utilization (rho)

- La/R ~ 0: average queuing delay small
- La/R -> 1: delays become large
- 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: department ssh server to melbourneisp.com (Melbourne

(tracepath on winows is similar)

```
awm22@svr-ssh-0:~$ traceroute melbourneisp.com
traceroute to melbourneisp.com (116.206.130.24), 30 hops max, 60 byte packets
 1  vlan398.gatwick.net.cl.cam.ac.uk (128.232.64.2)  10.299 ms  10.593 ms  11.064 ms
 2  cl-wgb.d-mw.net.cam.ac.uk (193.60.89.5)  0.743 ms  0.709 ms  0.506 ms
 3  d-mw.c-ce.net.cam.ac.uk (131.111.6.53)  0.917 ms  1.123 ms  0.906 ms
 4  c-ce.b-jc.net.cam.ac.uk (131.111.6.82)  0.795 ms  0.771 ms  0.751 ms
 5  ips-out.b-jc.net.cam.ac.uk (131.111.7.217)  1.096 ms  0.906 ms  1.032 ms
 6  ae0.lowdss-ban1.ja.net (146.97.41.37)  3.404 ms  3.019 ms  3.076 ms
 7  ae26.lowdss-sbr1.ja.net (146.97.35.245)  3.740 ms  3.431 ms  3.374 ms
 8  ae31.londtw-sbr2.ja.net (146.97.33.30)  7.034 ms  6.831 ms  6.962 ms
 9  ae28.londtt-sbr1.ja.net (146.97.33.61)  8.829 ms  16.976 ms  16.954 ms
10  ae0.londtt-ban2.ja.net (146.97.35.194)  7.320 ms  6.467 ms  6.387 ms
11  ldn-b11-link.ip.twelve99.net (62.115.175.106)  6.476 ms  6.234 ms  6.585 ms
12  ldn-bb1-link.ip.twelve99.net (62.115.138.168)  7.473 ms *  7.710 ms
13  nyk-bb2-link.ip.twelve99.net (62.115.139.246)  76.167 ms nyk-bb5-link.ip.twelve99.net (62.115.139.244)  75.315 ms *
14  * chi-bb2-link.ip.twelve99.net (62.115.132.135)  140.582 ms  140.036 ms
15  * den-bb1-link.ip.twelve99.net (62.115.115.76)  115.630 ms  114.872 ms
16  den-bb2-link.ip.twelve99.net (62.115.137.114)  113.977 ms *  113.656 ms
17  palo-bb2-link.ip.twelve99.net (62.115.139.112)  143.238 ms  144.527 ms *
18  * tpg-ic-387776.ip.twelve99-cust.net (62.115.188.5)  295.626 ms  296.153 ms
19  tpg-ic-387776.ip.twelve99-cust.net (62.115.188.5)  295.143 ms syd-apt-ros-crt3-be-100.tpgi.com.au (203.29.134.43)  295
20  syd-apt-ros-crt3-be-100.tpgi.com.au (203.29.134.43)  295.545 ms syd-sot-ken-crs2-Te-0-6-0-9.tpgi.com.au (203.26.22.122
21  syd-sot-ken-crs2-Te-0-6-0-9.tpgi.com.au (203.26.22.122)  300.416 ms AU-VI-1015-IPG-221-Bundle-Ether1.tpgi.com.au (27.
22  AU-VI-4901-IPE-01-Eth-Trunk21.tpgi.com.au (203.220.216.30)  301.396 ms AU-VI-1015-IPG-221-Bundle-Ether2.tpgi.com.au (2
23  AU-VI-4901-IPE-01-Eth-Trunk21.tpgi.com.au (203.220.216.30)  300.724 ms 14-202-130-170.static.tpgi.com.au (14.202.130.1
24  14-202-130-170.static.tpgi.com.au (14.202.130.170)  303.742 ms *  303.963 ms
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
awm22@svr-ssh-0:~$
```

Three delay measurements from rio.cl.cam.ac.uk to London JaNet gatew

Crossing the Atlantic Ocean

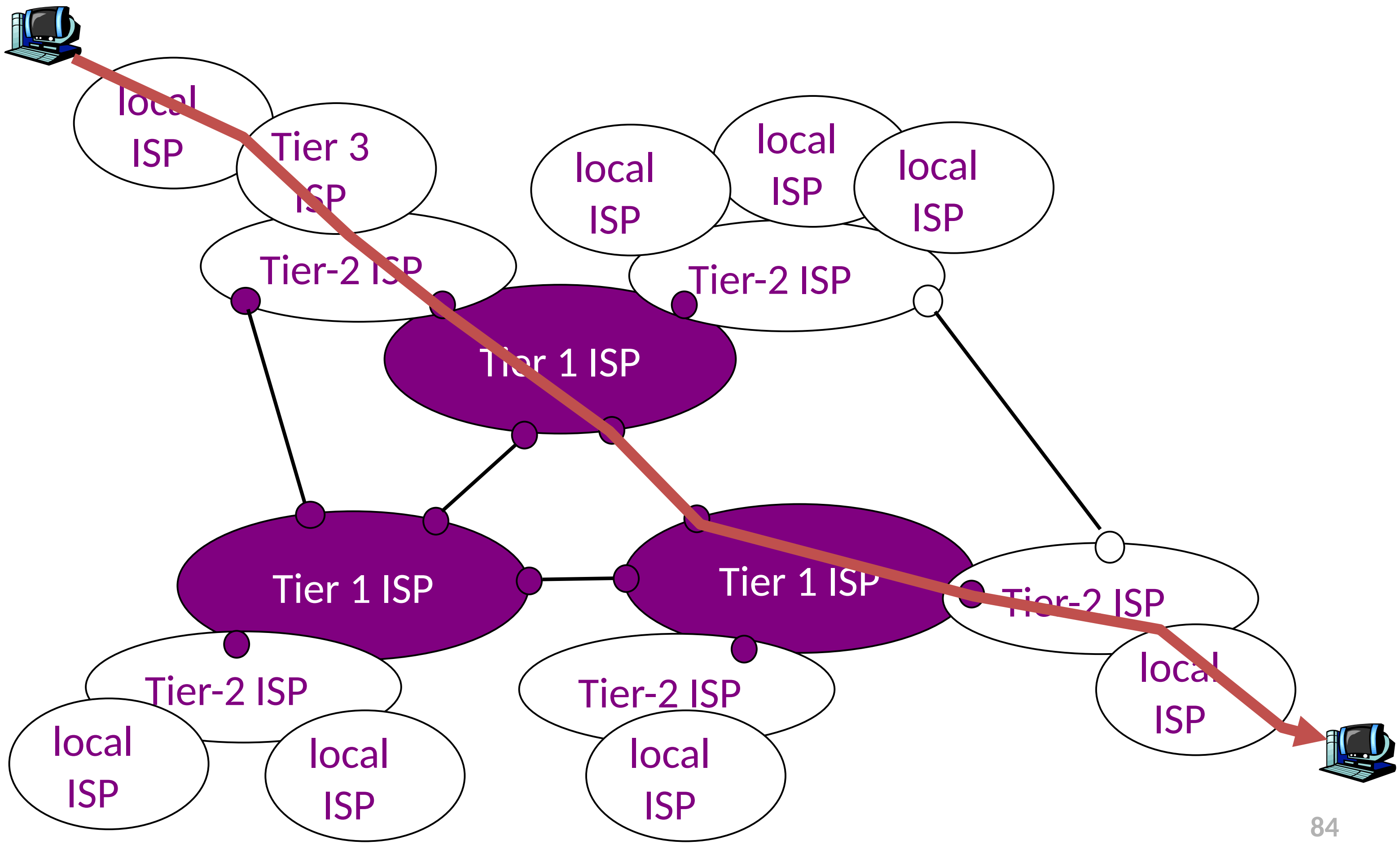
Crossing the Pacific Ocean

* means no response (probe or reply lost, router not replying)

300ms RTT, 150ms one way Internet, 59.4ms by photon, 42ms by neutron

Internet structure: network of networks

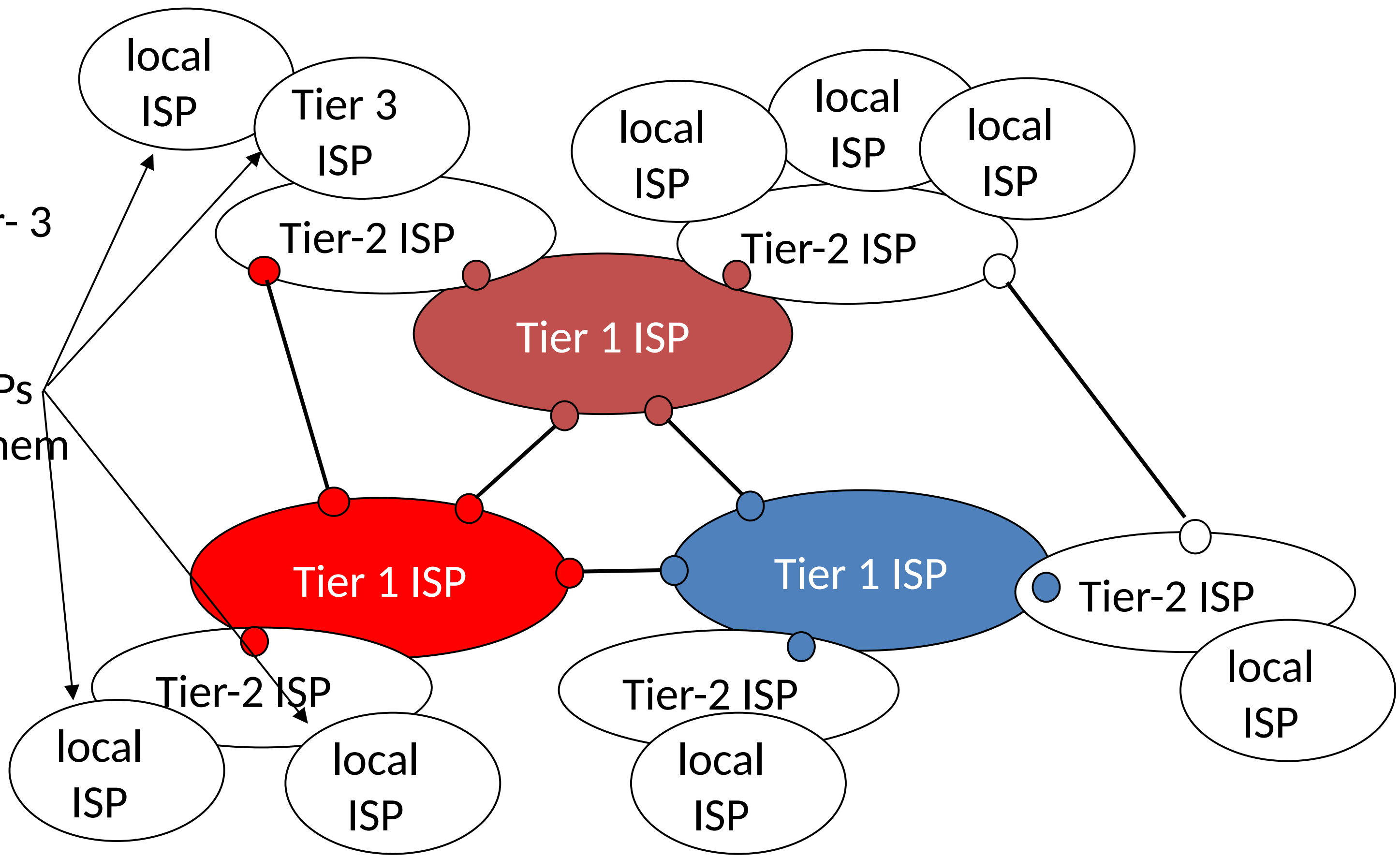
- a packet passes through many networks!



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

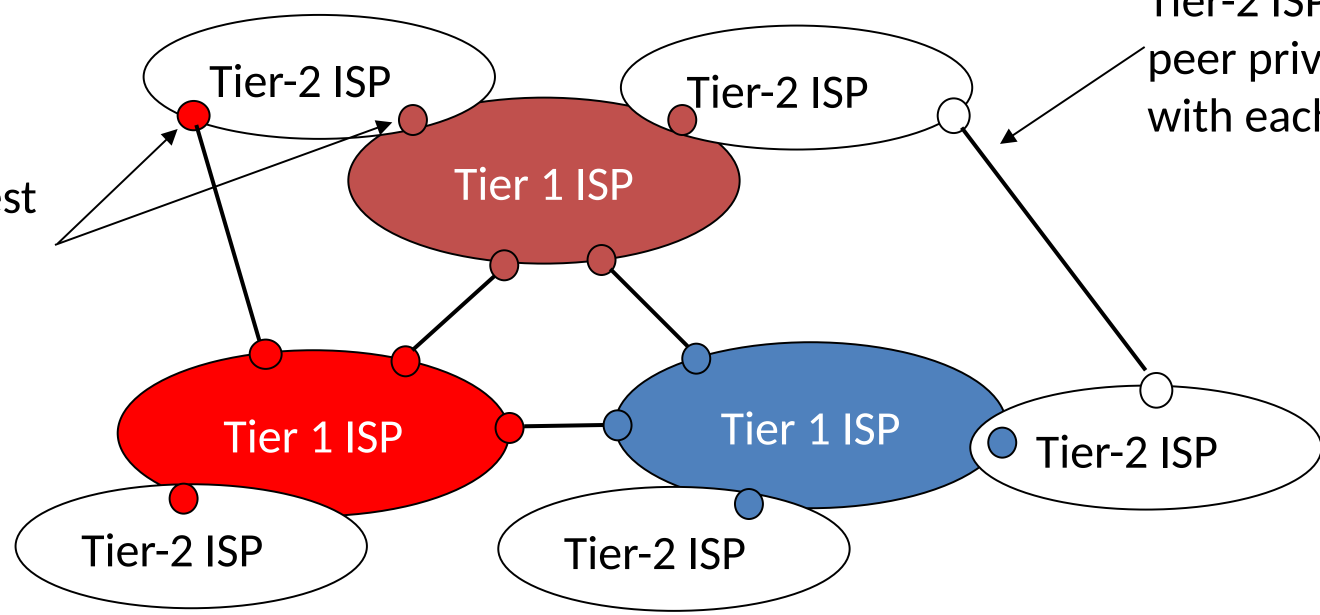


Internet structure: network of networks

- “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

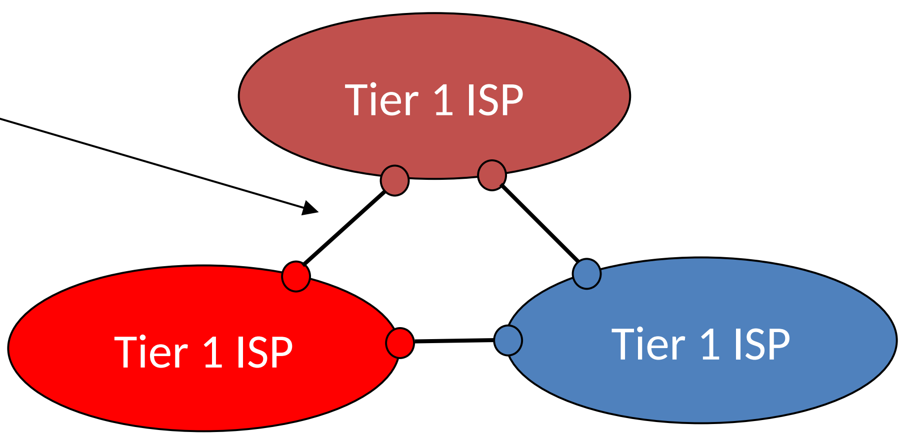


Tier-2 ISPs also peer privately with each other.

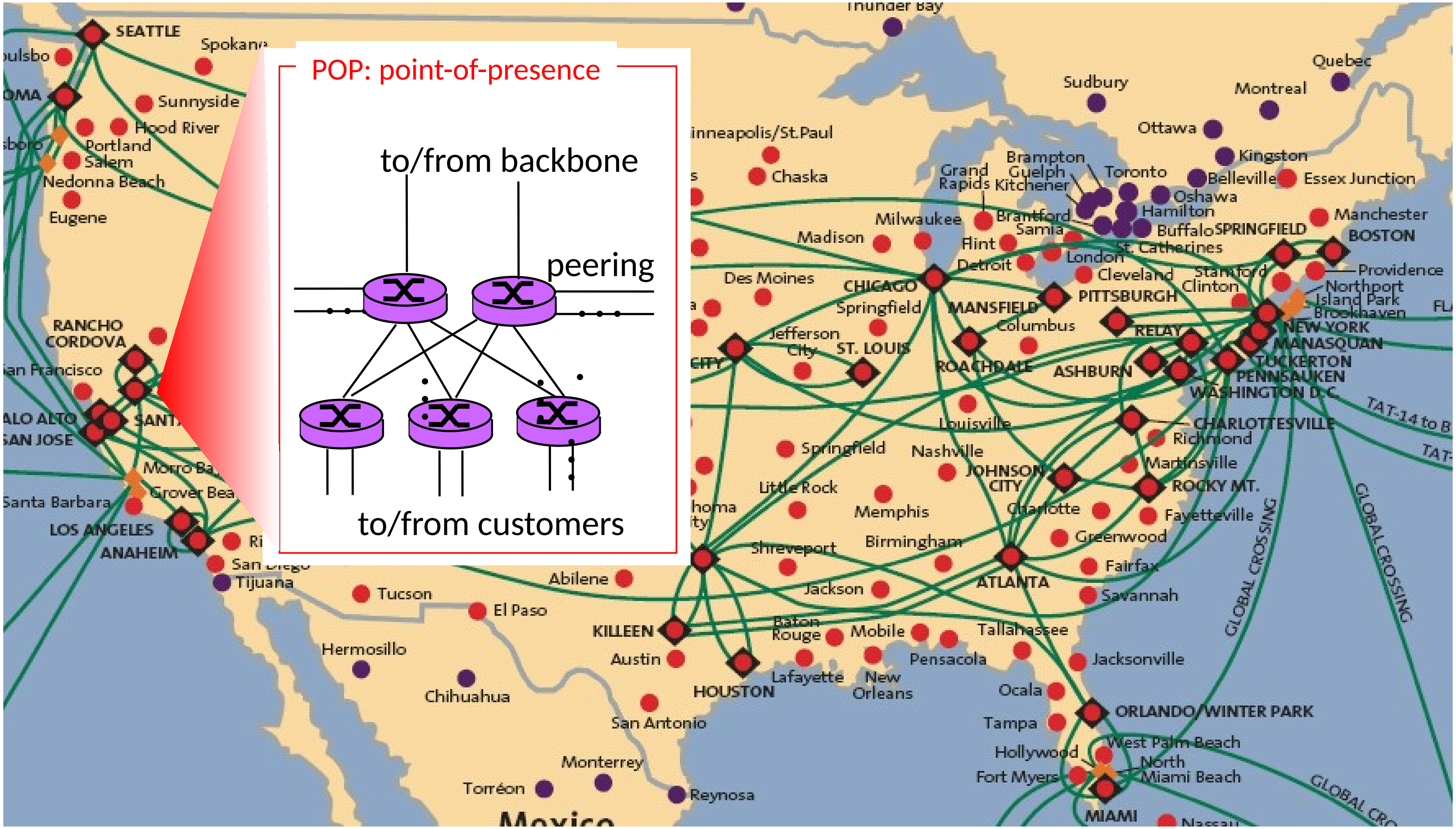
Internet structure: network of networks

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

Tier-1 providers interconnect (peer) privately



Tier-1 ISP: e.g., Sprint



Connectionless Packet Switching

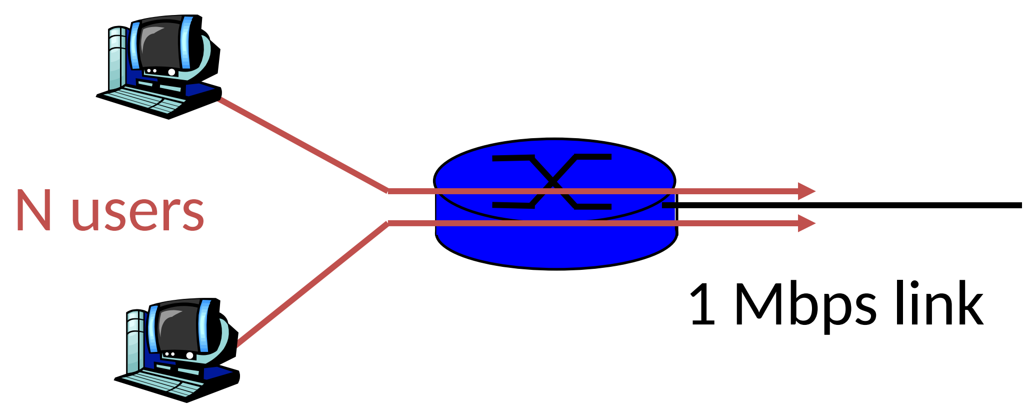
- Data is sent as chunks of formatted bits (Packets)
- Packets consist of a "header" and "payload"
- Switches "forward" packets based on their headers
- Each packet travels independently
- No link resources are reserved in advance. Instead, packet switching depends on statistical multiplexing
  - allows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

$$P(U=k) = \binom{n}{k} p^k (1-p)^{n-k}$$

$$\left[\therefore P(U \le K) = \sum_{k=0}^{K} \binom{n}{k} p^k (1-p)^{n-k} \right] \therefore \left[P(U>K) = 1 - \sum_{k=0}^{K} \binom{n}{k} p^k (1-p)^{n-k} \right]$$

for $n = 35$, $K = 10$

$$P(U \le 10) = \sum_{k=0}^{10} \binom{35}{k} p^k (1-p)^{35-k}$$

where $p = 0.1$:

$$P(U \le 10) = 0.99958$$

$$\therefore P(U > 10) = 0.00042$$

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