Computer Networking

Slide Set 1

Andrew W. Moore
Andrew.Moore@cl.cam.ac.uk

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/

Other textbooks are available.

Thanks

- Slides are a fusion of material from

  to 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.

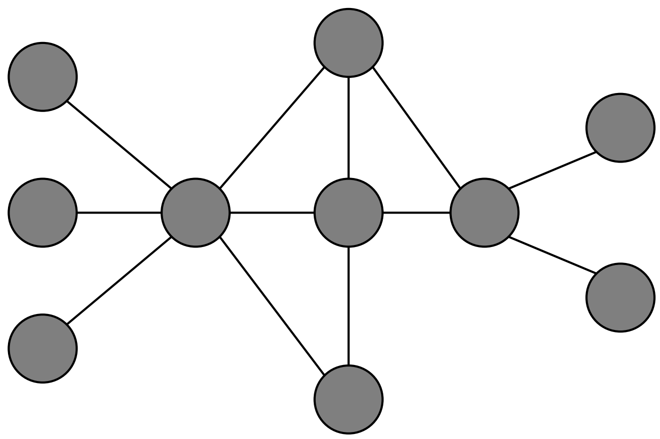
- Supervision material is drawn from

  Stephen Kell, Andy Rice, and the TA teams of 144 and 168

- Finally thanks to the fantastic past Part 1b students and Andrew Rice for all the tremendous feedback.

What is a network?

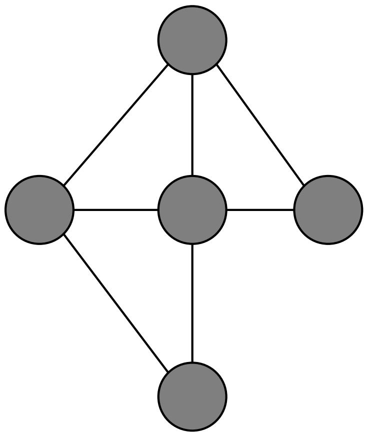
- A system of “links” that interconnect “nodes” in order to move “information” between nodes



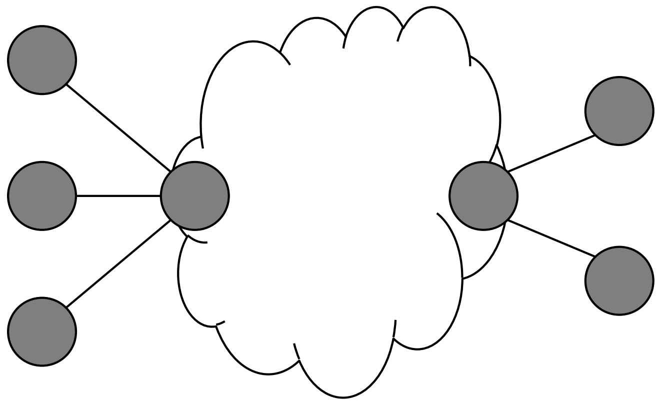
- Yes, this is all rather abstract

What is a network?

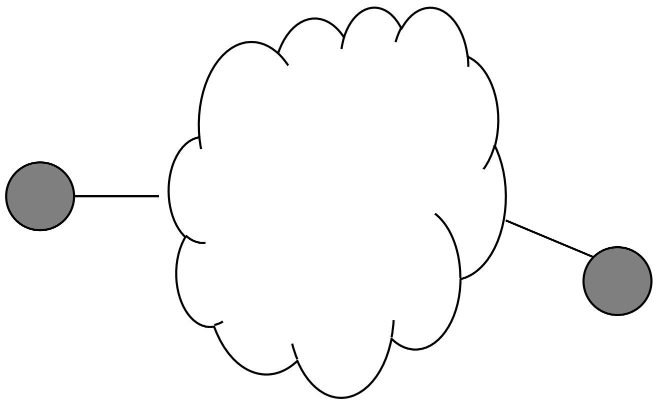
- We also talk about



or



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

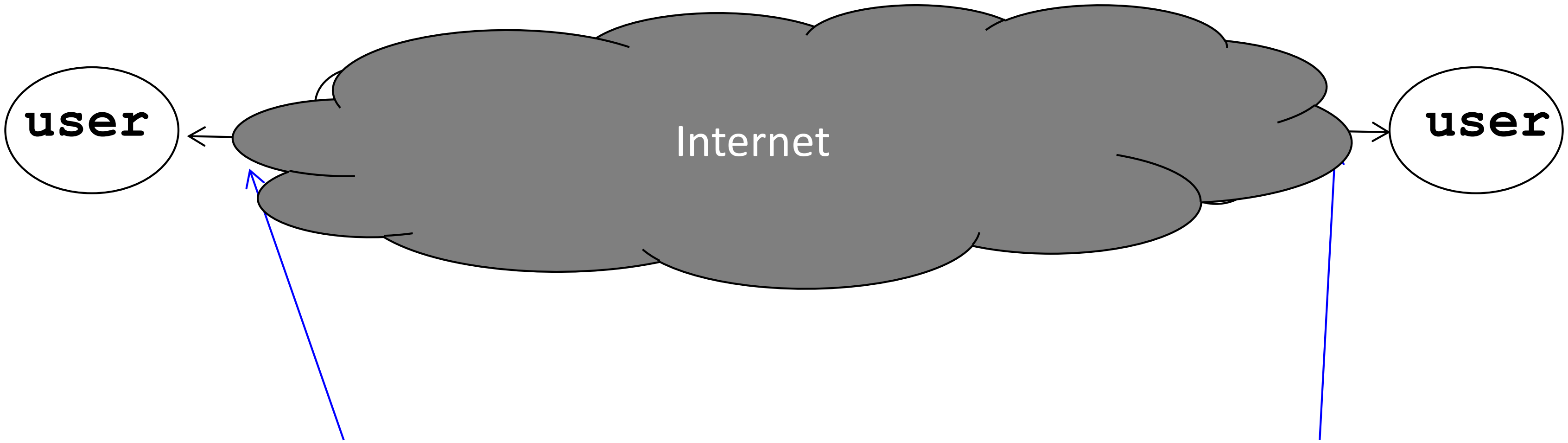
The Internet has transformed everything

- 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 populatio
- 1.8 Billion web sites
  - 34.5% of which are powere ... ess!
- 4.88 Billion smar ... % of population)
- 500 Mil ... day
- ... WhatsApp messages per day
- ... illion 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: 100bits/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:

- 56kilobits/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 componentsin 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) theoretical models
- "Working code" doesn't mean much
- Performance benchmarks are too narrow

An Engineered System

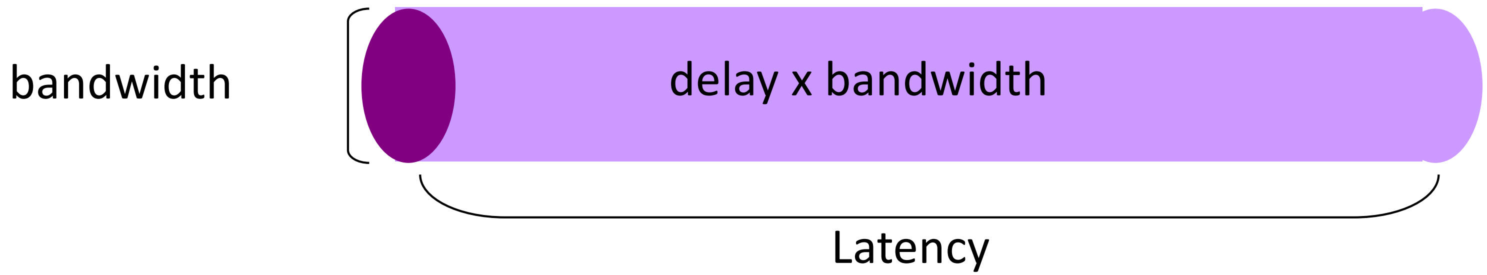
- Constrained by what technology is practical
  - Link bandwidths
  - Switch port counts
  - Bit error rates
  - Cost
  - …

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 ~ 10^6bits ~ 125KBytes

  17km * c = 56µs << 10ms

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

  750m * c = 56µs ≅ 30µs

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

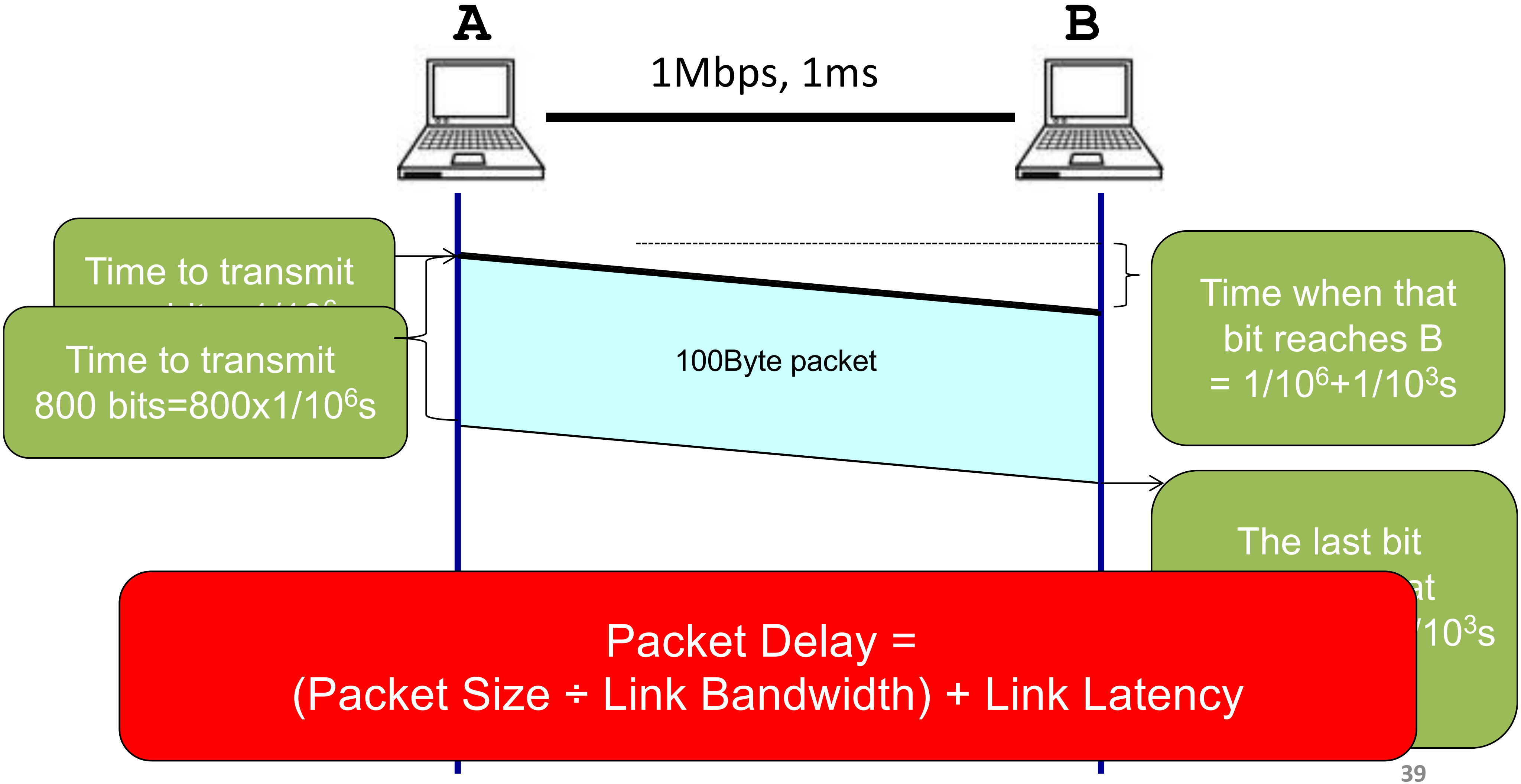
  9708km * c = 32ms << 140ms

- Intra Host:
  - BW~100Gbps
  - Latency~16nsec
  - BDP ~ 1600bits ~ 200Bytes

  25cm * c = 83ps << 16ns

Packet Delay

Sending a 100B packet from A to B?

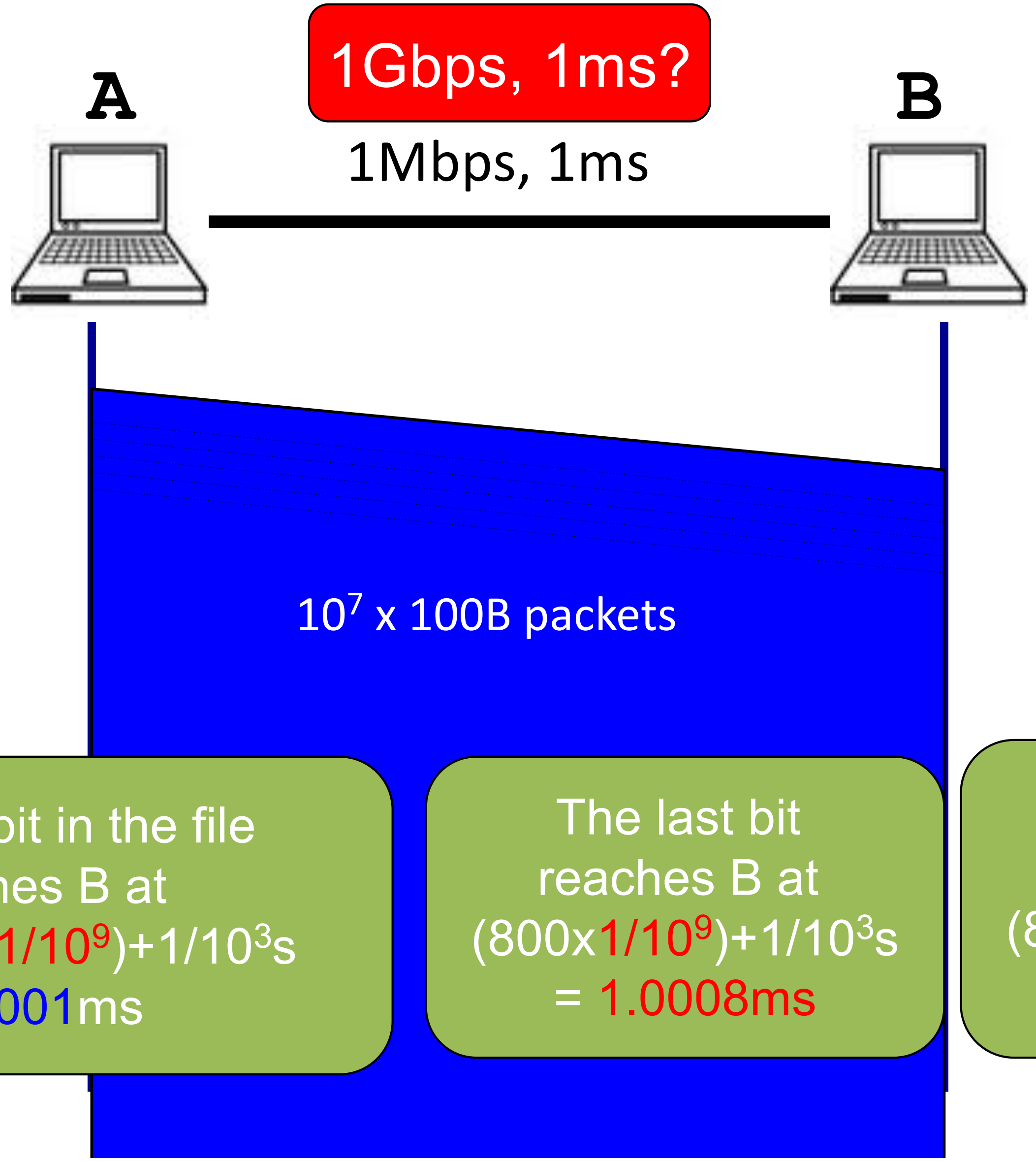


Packet Delay =
(Packet Size ÷ Link Bandwidth) + Link Latency

1GB file in 100B packets

ay

Sending a 100B packet from A to B?



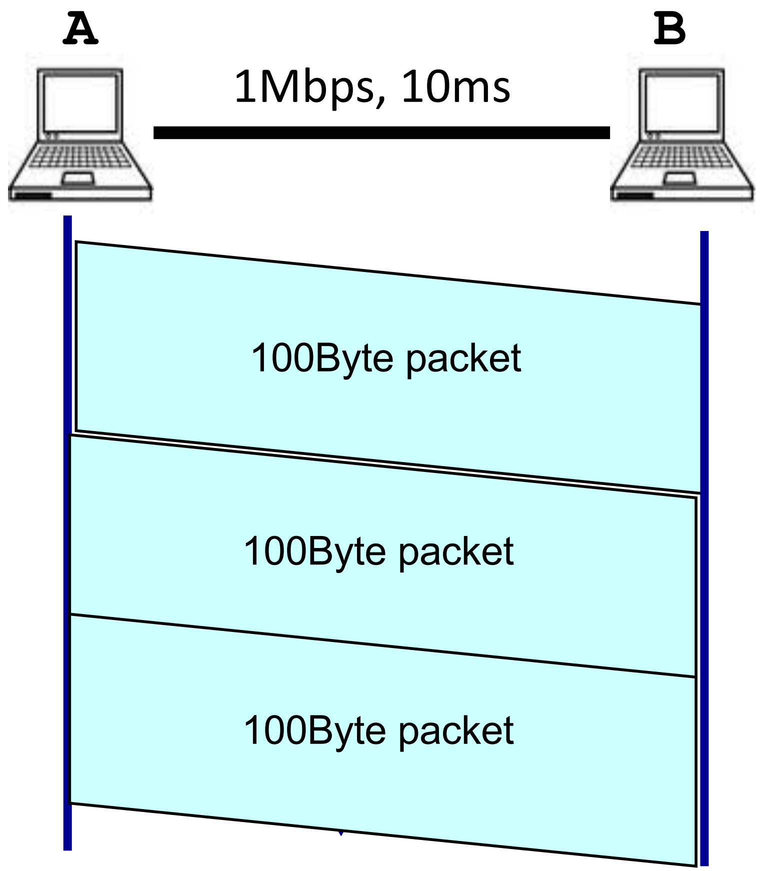
The last bit in the file reaches B at $(10^7 x 800 x 1/10^9) + 1/10^3 s$ = 8001ms

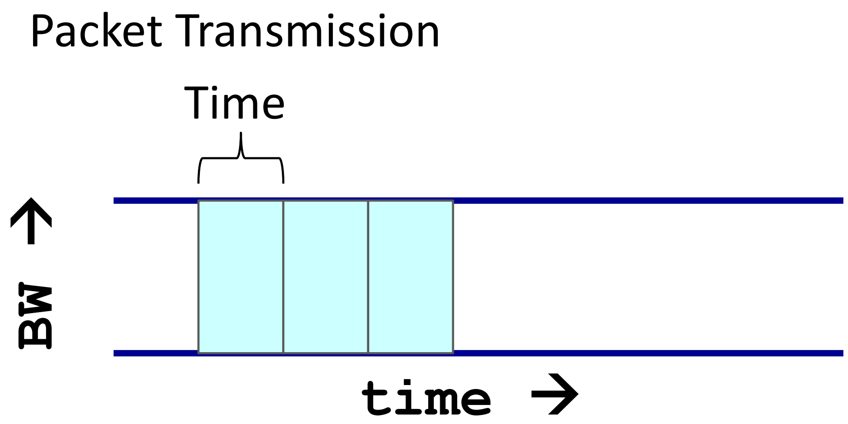
The last bit reaches B at $(800 x 1/10^9) + 1/10^3 s$ = 1.0008ms

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

Packet Delay: The "pipe" view

Sending 100B packets from A to B?

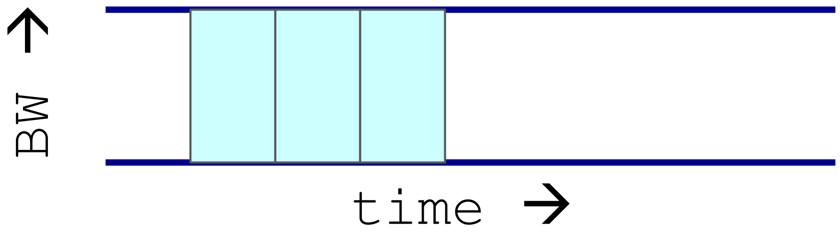




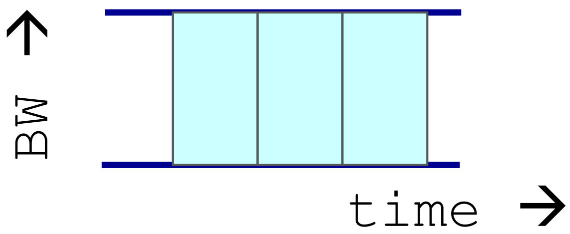
Packet Delay: The “pipe” view

Sending 100B packets from A to B?

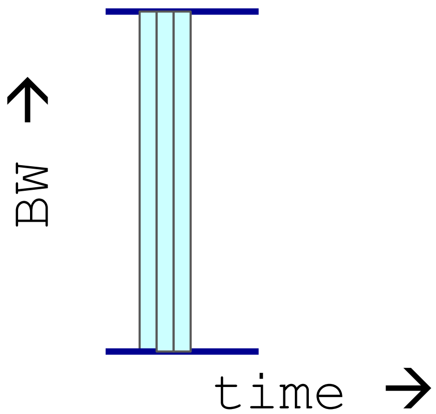
1Mbps, 10ms (BDP=10,000)



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

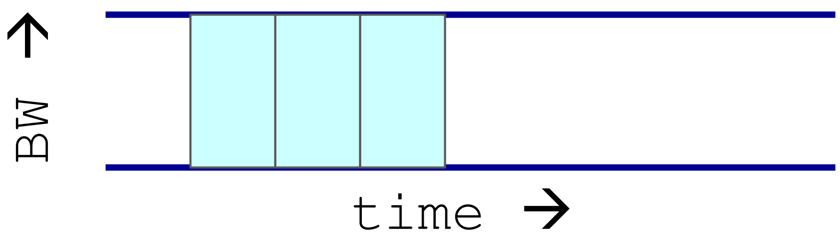


10Mbps, 1ms (BDP=10,000)



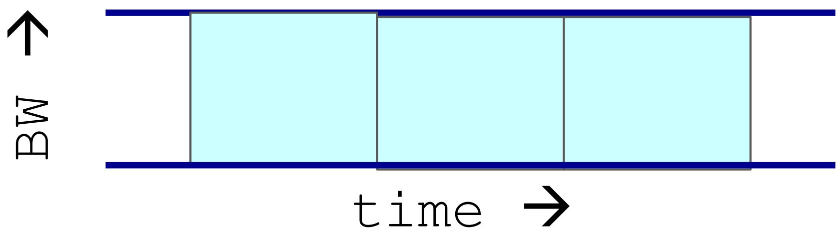
Packet Delay: The “pipe” view
Sending 100B packets from A to B?

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

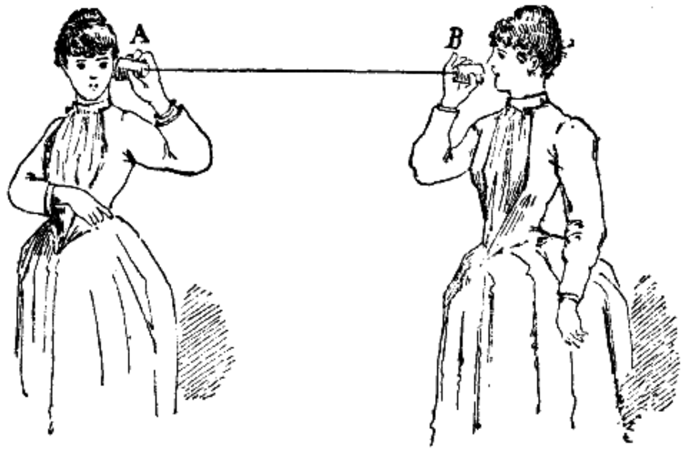


What if we used *200Byte packets??*

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

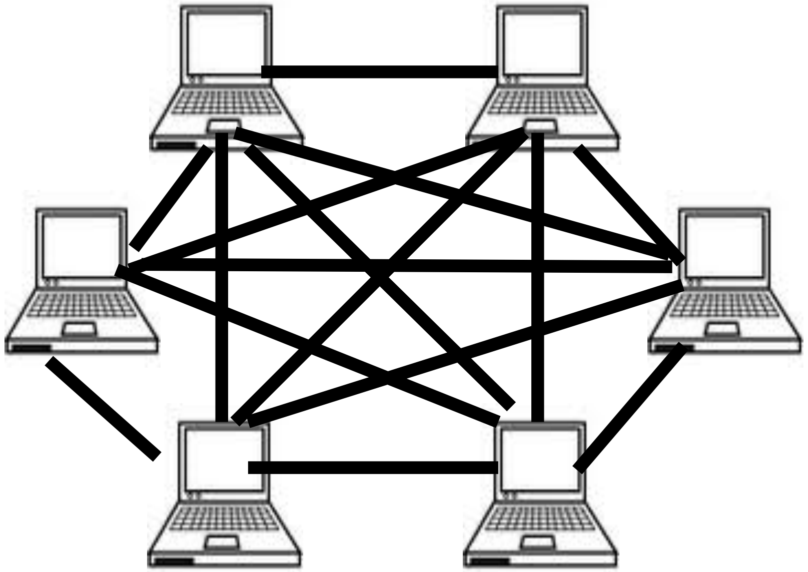


Recall Nodes and Links



What if we have more nodes?

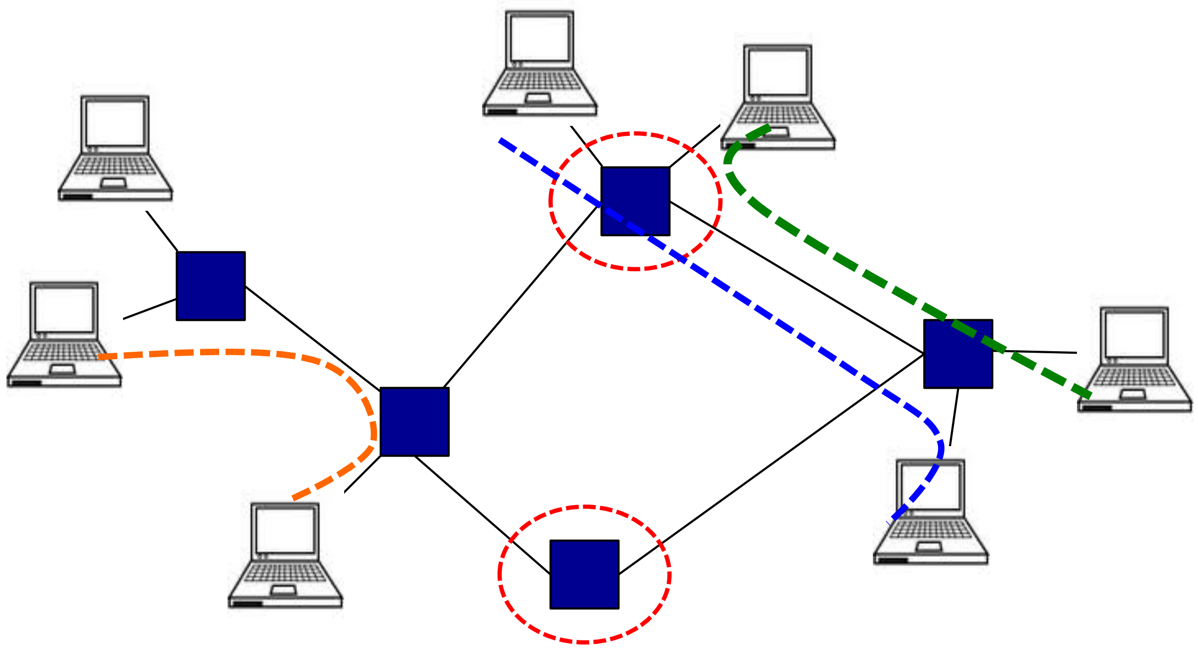
One link for every node?



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

Solution: A switched network

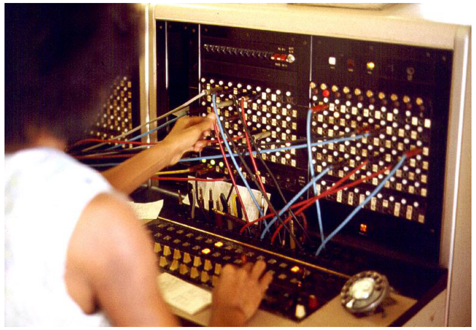
Nodes share network link resources

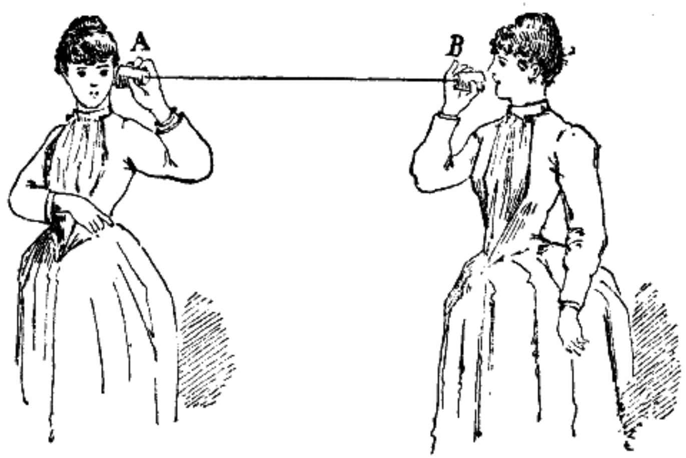


How is this sharing implemented?

Two examples of switched networks

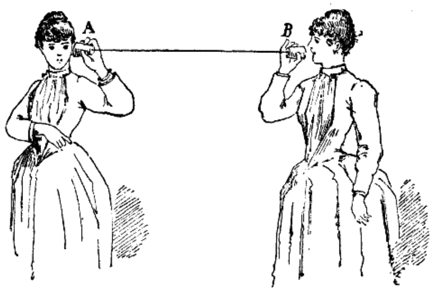
- Circuit switching (used in the *POTS*: Plain Old Telephone system)



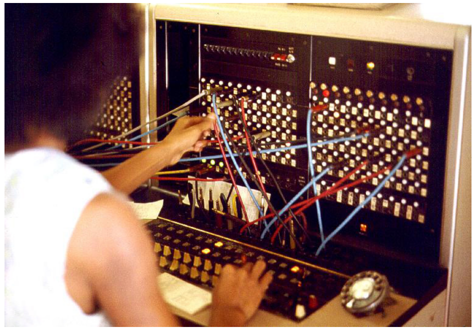


- Packet switching (used in the Internet)

Circuit switching



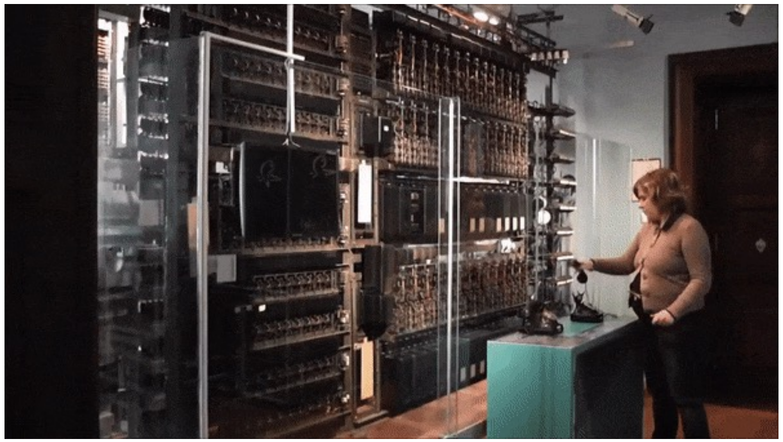
Telephone



Exchange

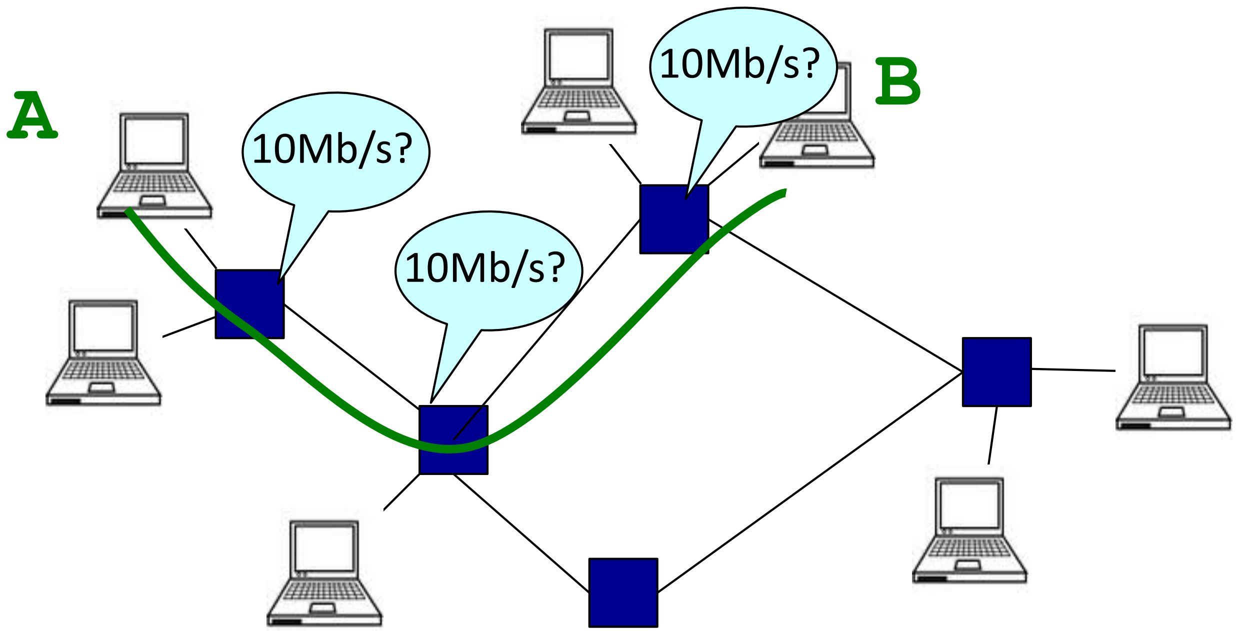
Exchange

Telephone



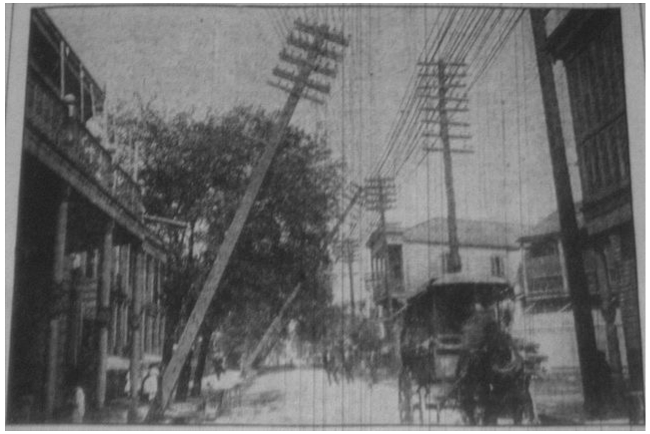
Circuit switching

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

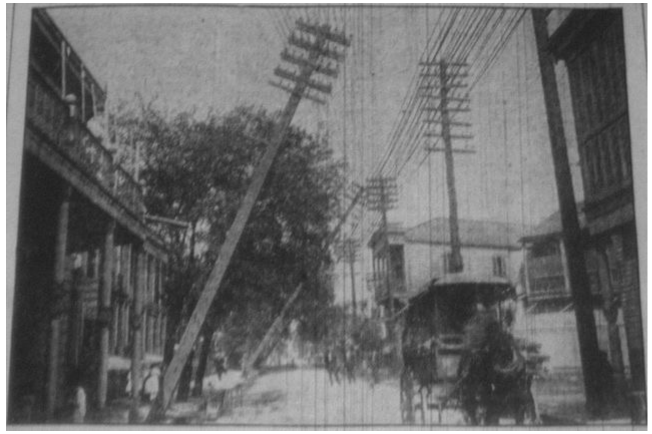
Multiplexing



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

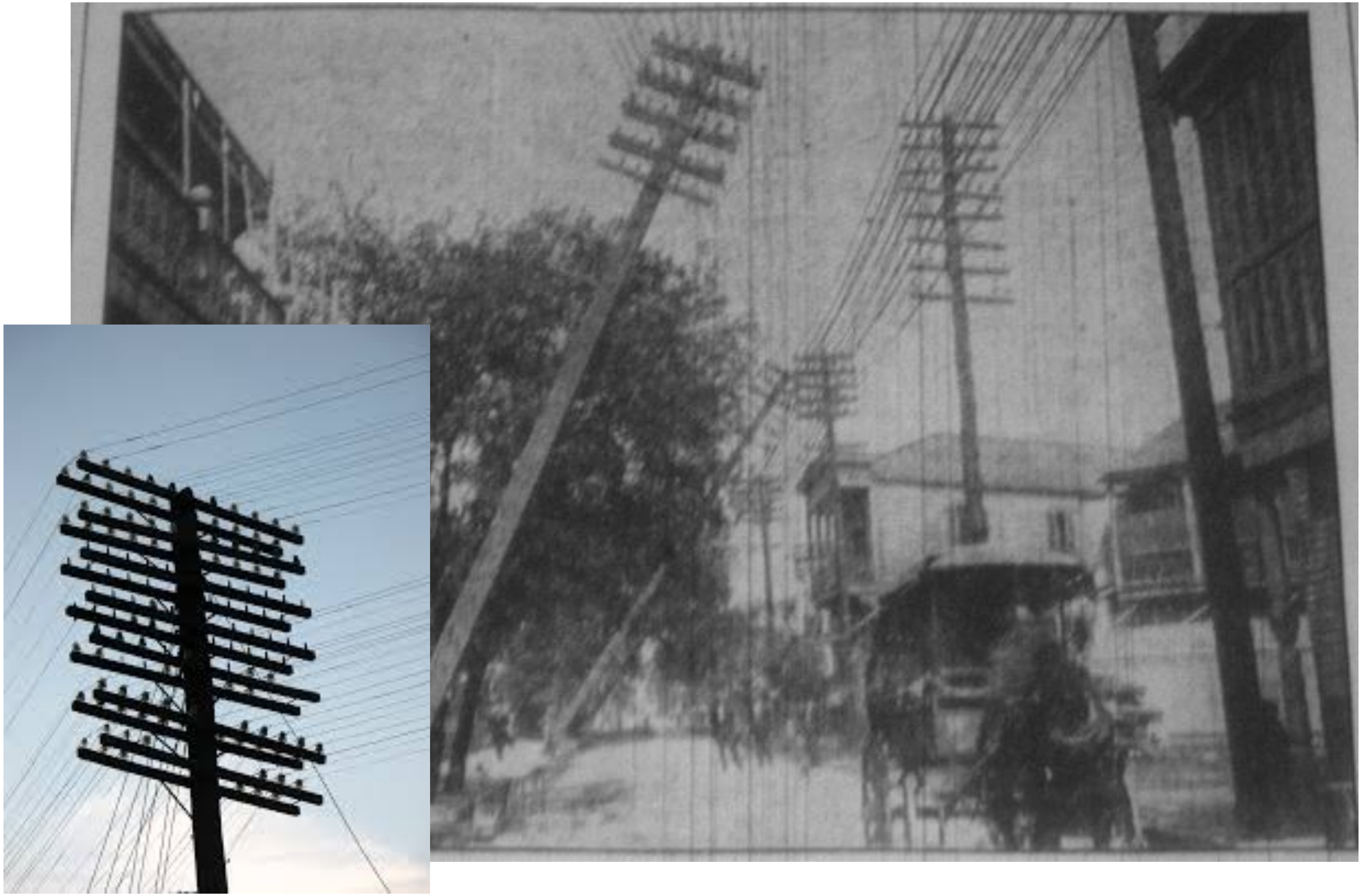
Multiplexing



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 Multiplexing

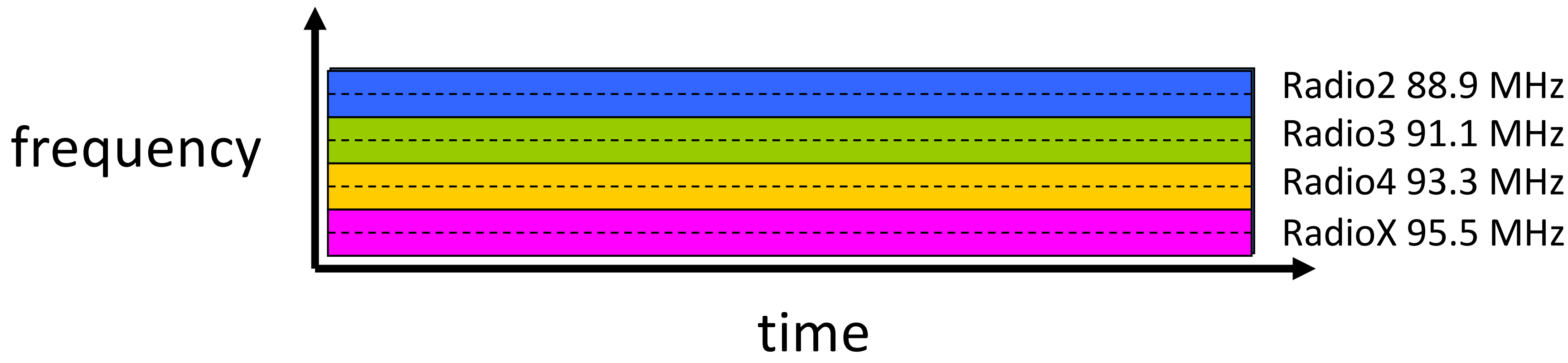


Sharing Circuit Switching: FDM and TDM

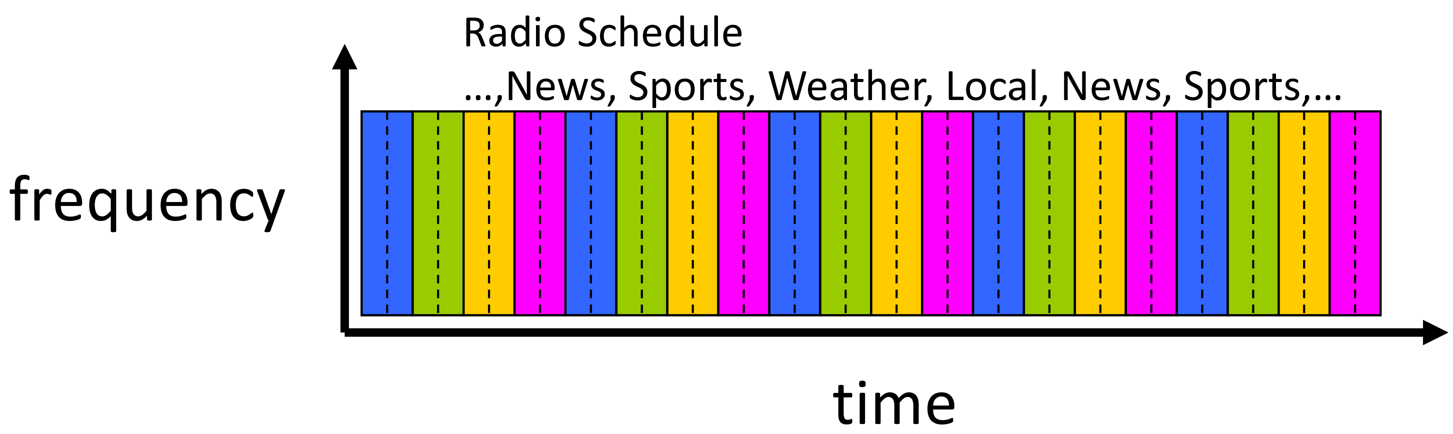
Example:

4 users

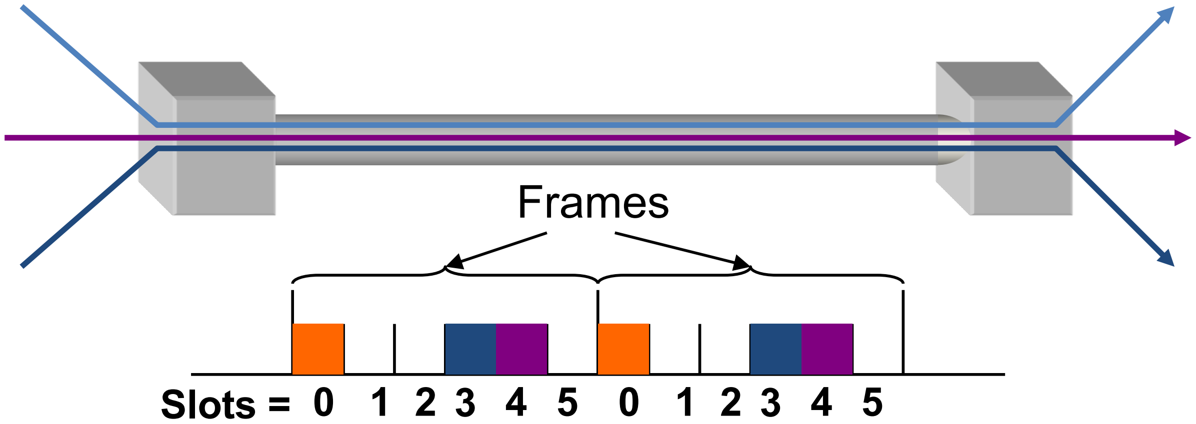
Frequency Division Multiplexing



Time Division Multiplexing

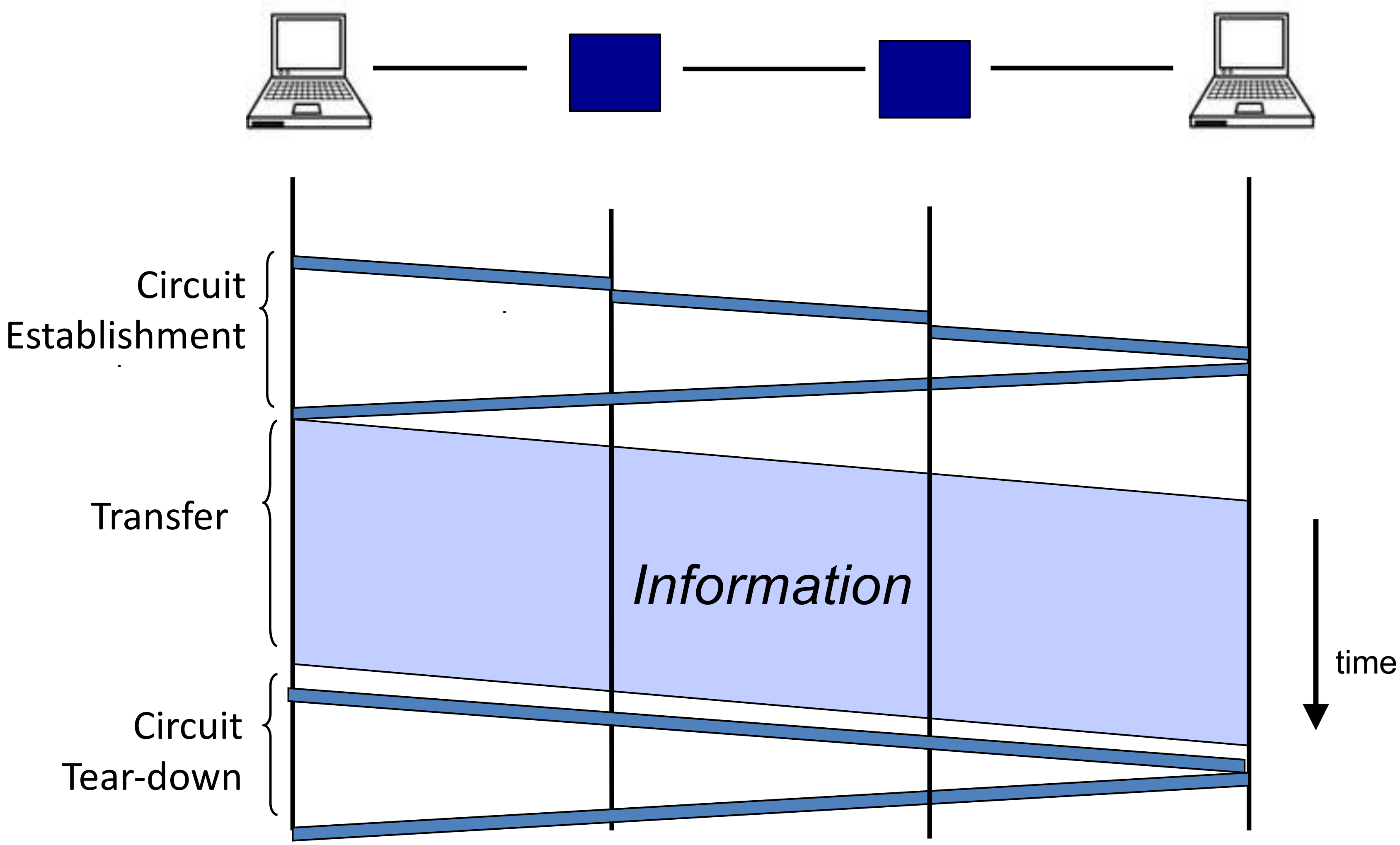


Time-Division Multiplexing/Demultiplexing



- 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

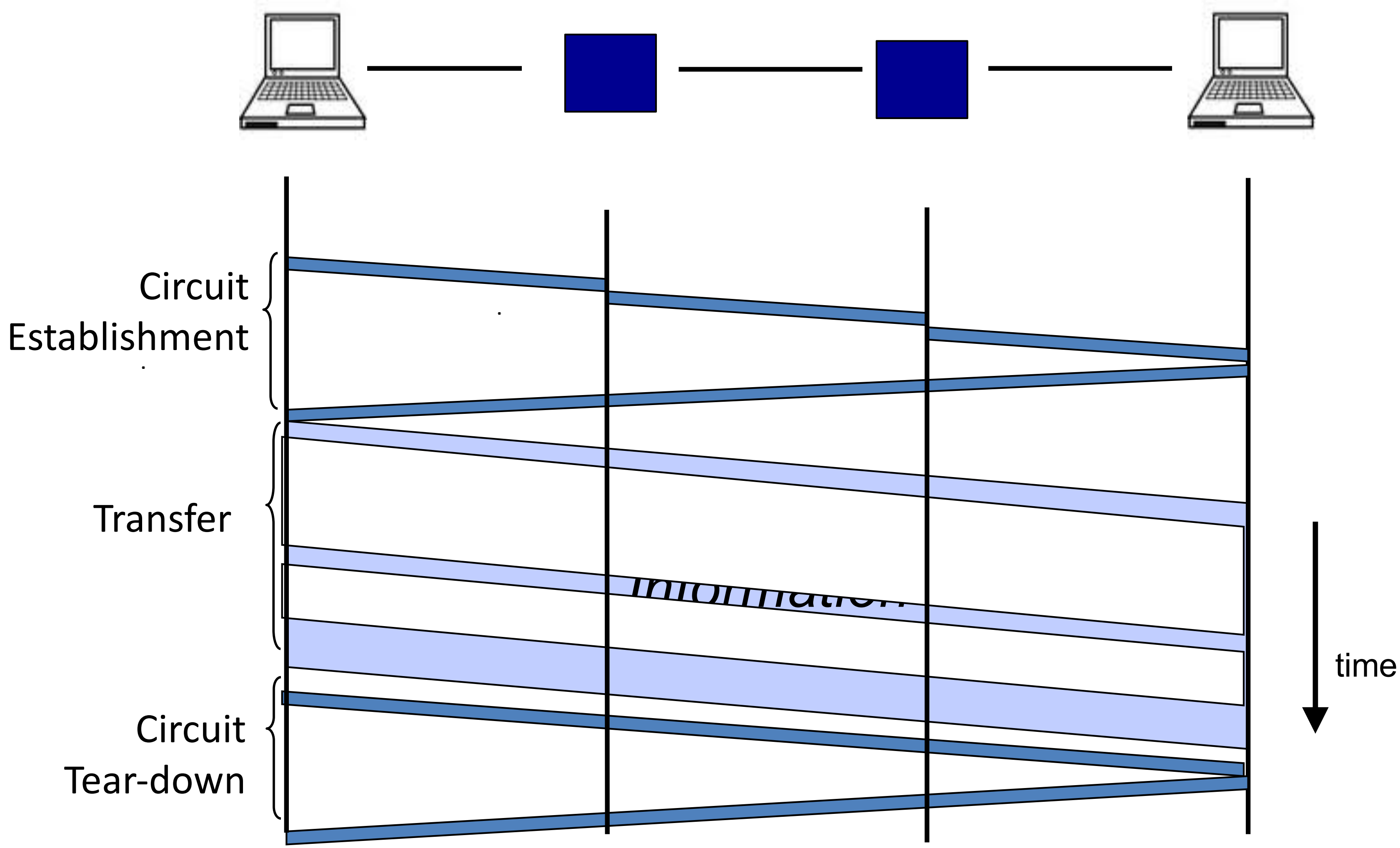


Circuit switching: pros and cons

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

- Cons

Timing in Circuit Switching

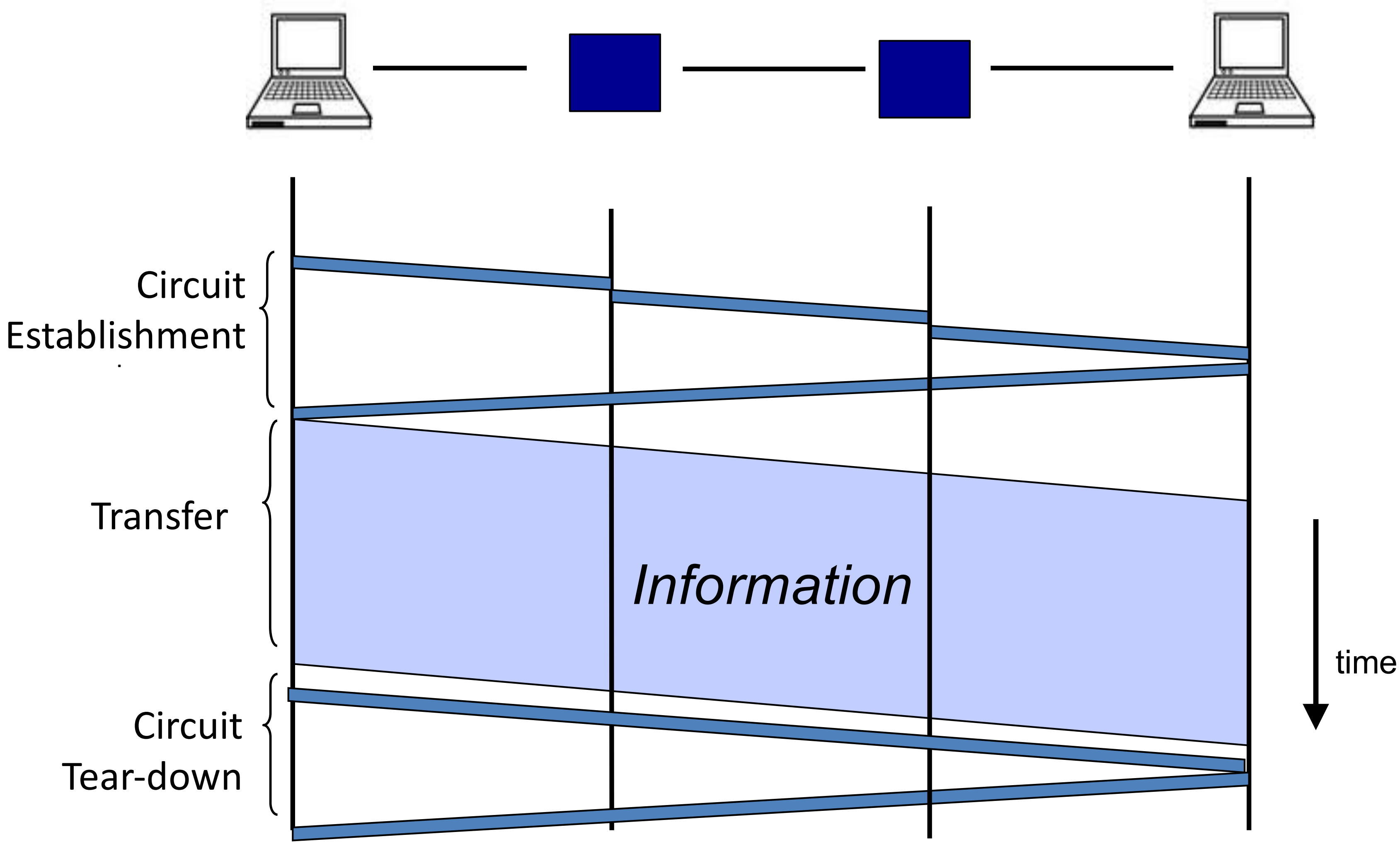


Circuit switching: pros and cons

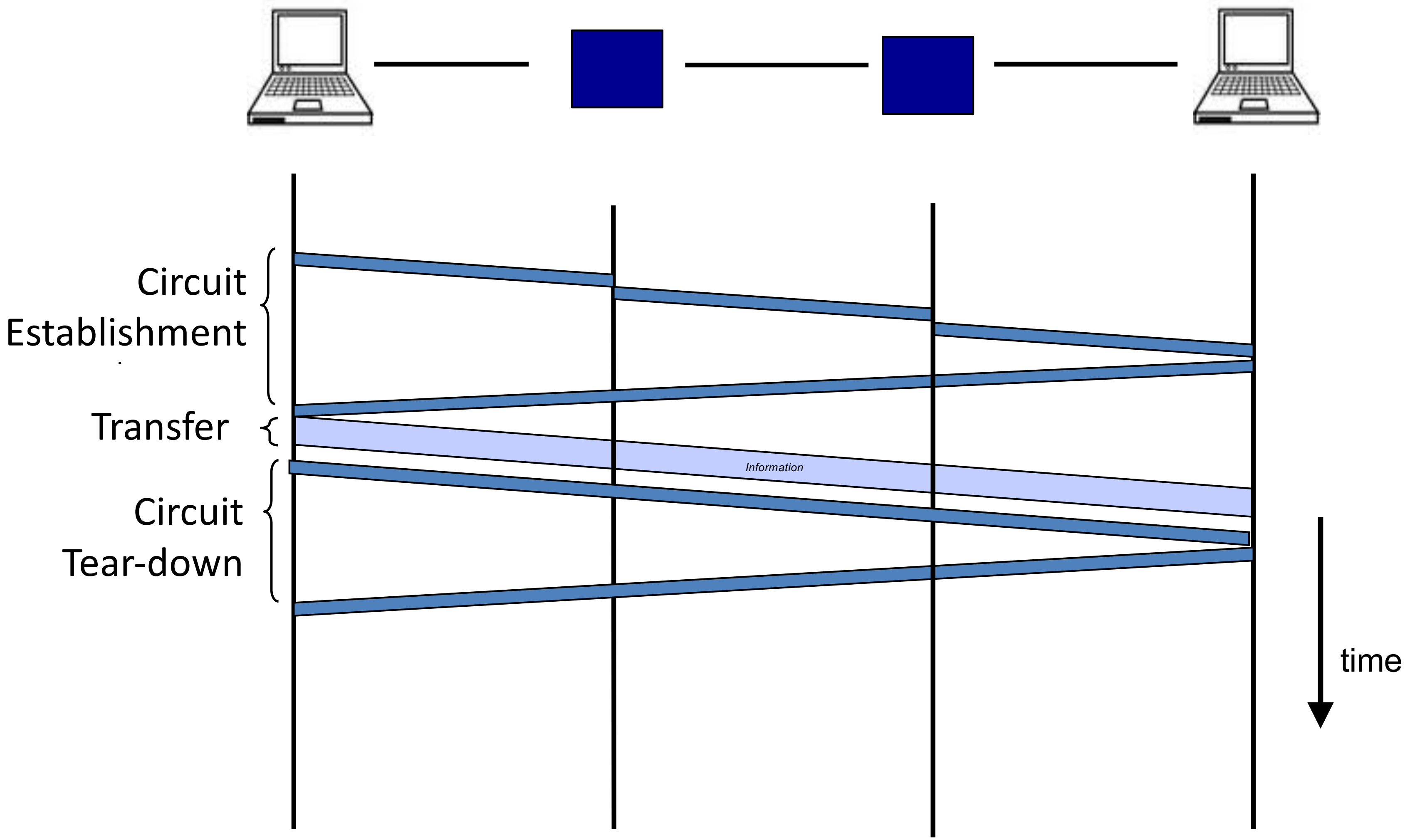
- Pros
  - guaranteed performance
  - fast transfer (once circuit is established)

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

Timing in Circuit Switching



Timing in Circuit Switching

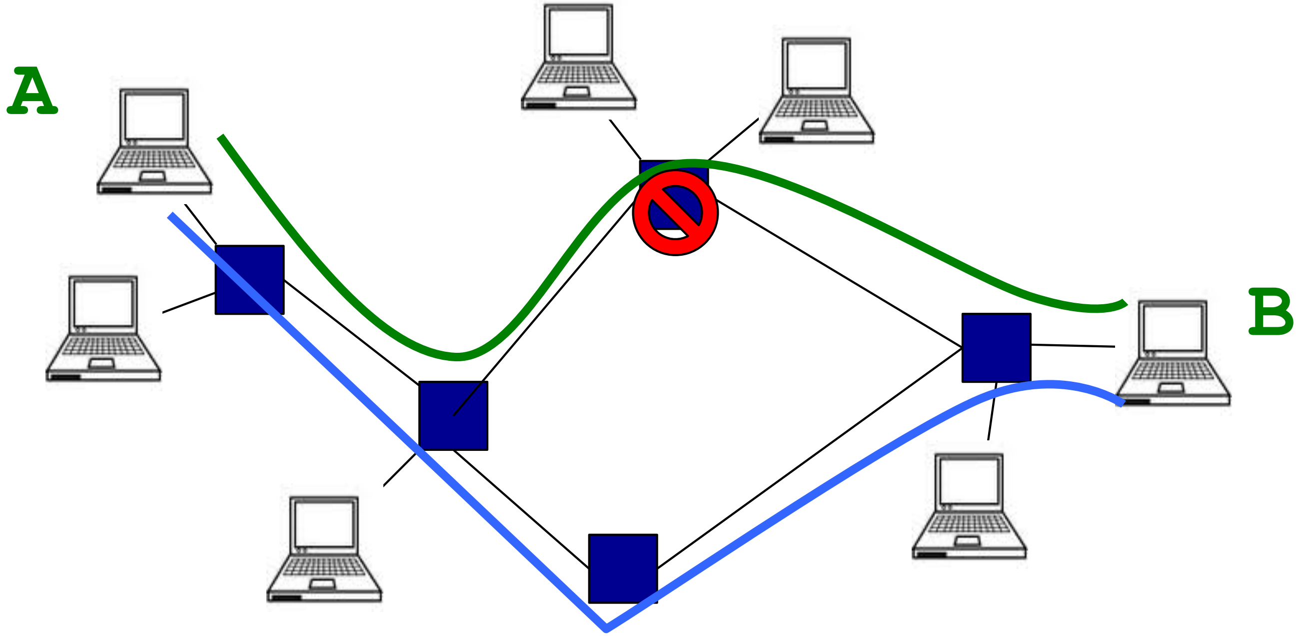


Circuit switching: pros and cons

- 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 slots/sec
  - 500 msec to establish end-to-end circuit

Let’ s work it out!

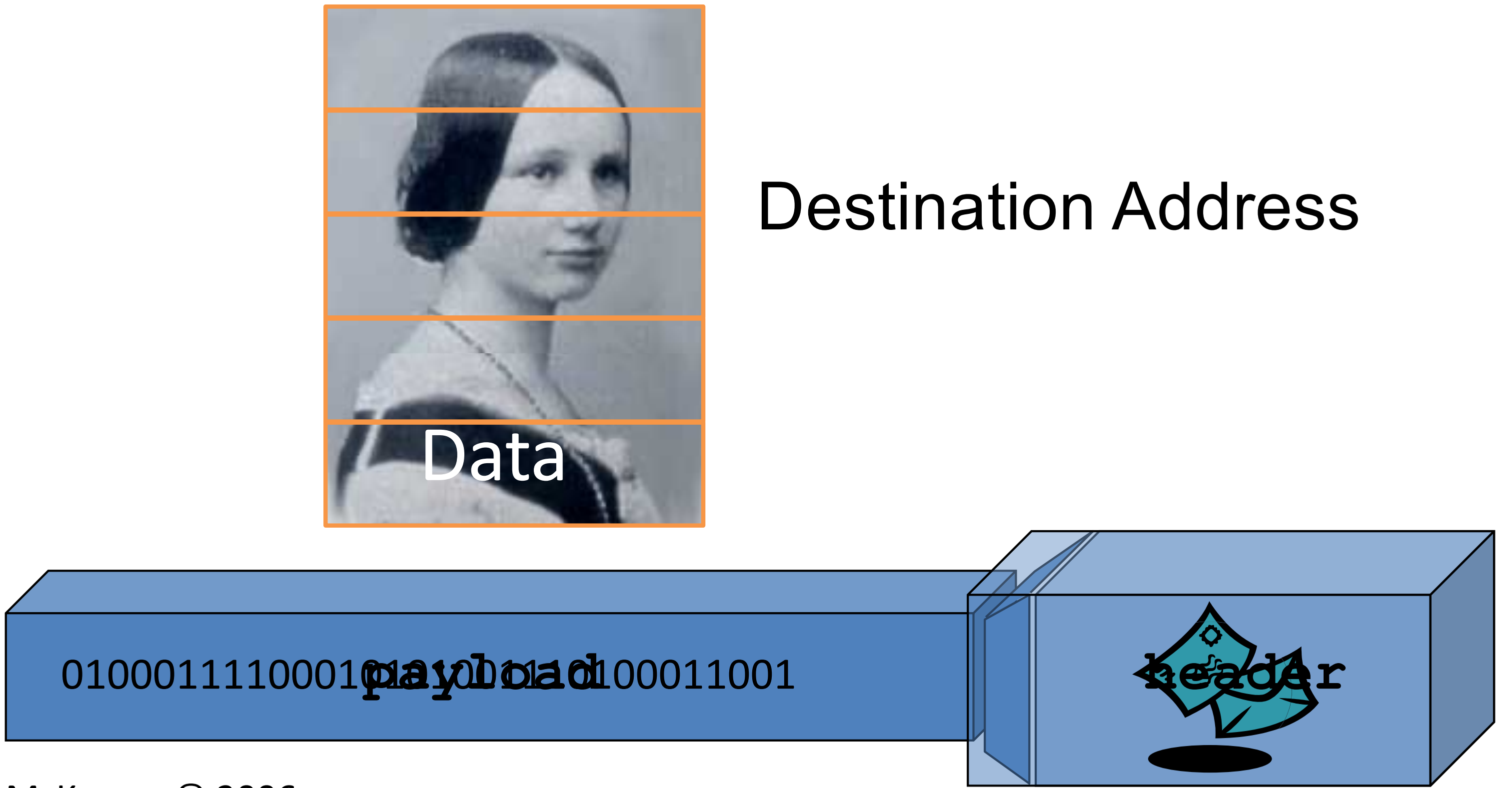
Two examples of switched networks

- 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"*



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)
    - Do I acknowledge this? Who signed for it?
    - Were the contents ok?

Packet Switching

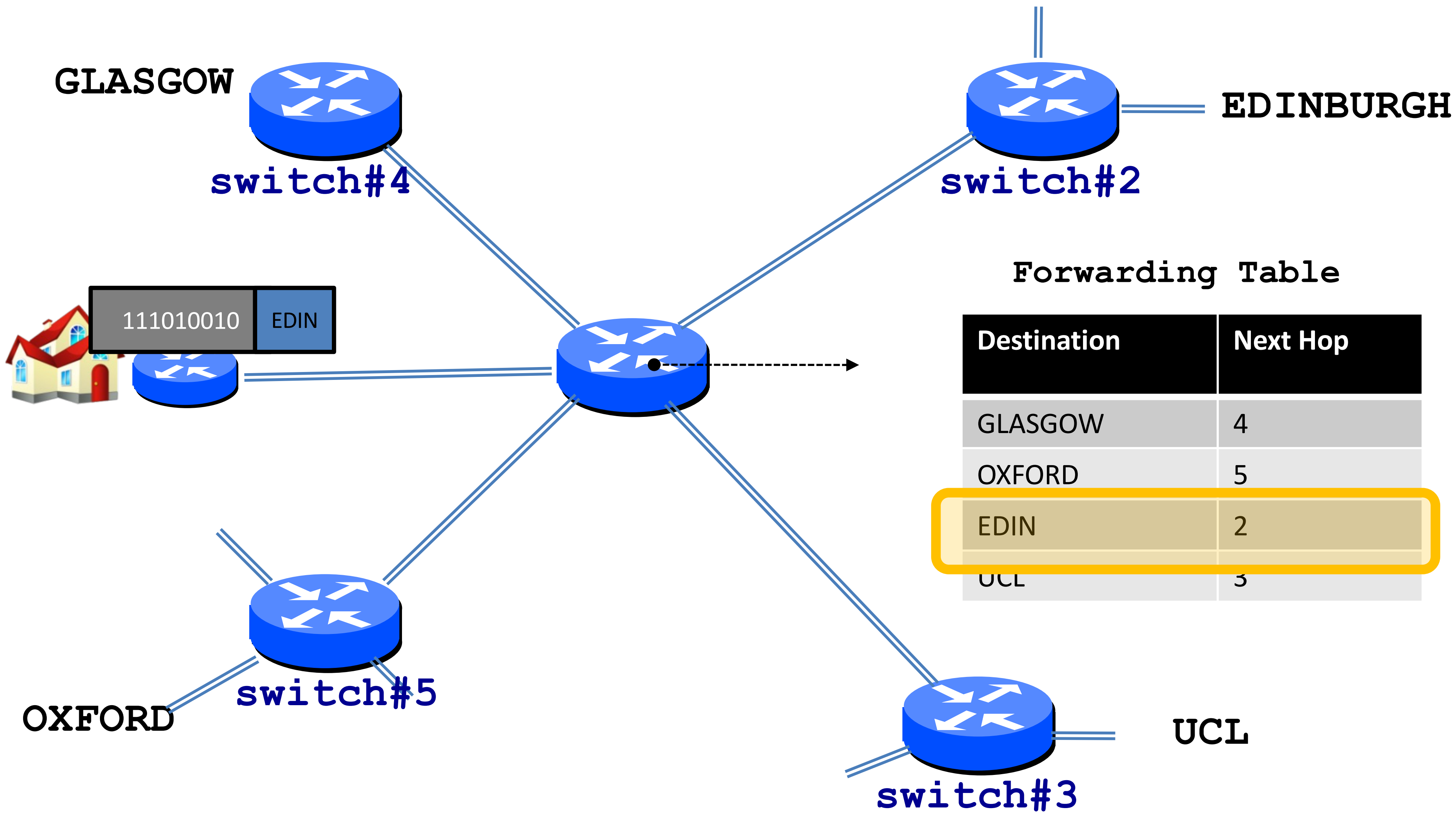
- Data is sent as chunks of formatted bits (Packets)
- Packets consist of a “header” and “payload”
- Switches “forward” packets based on their headers

A switch looks at the header and

immediately decides which physical port

In a switch: address maps to port

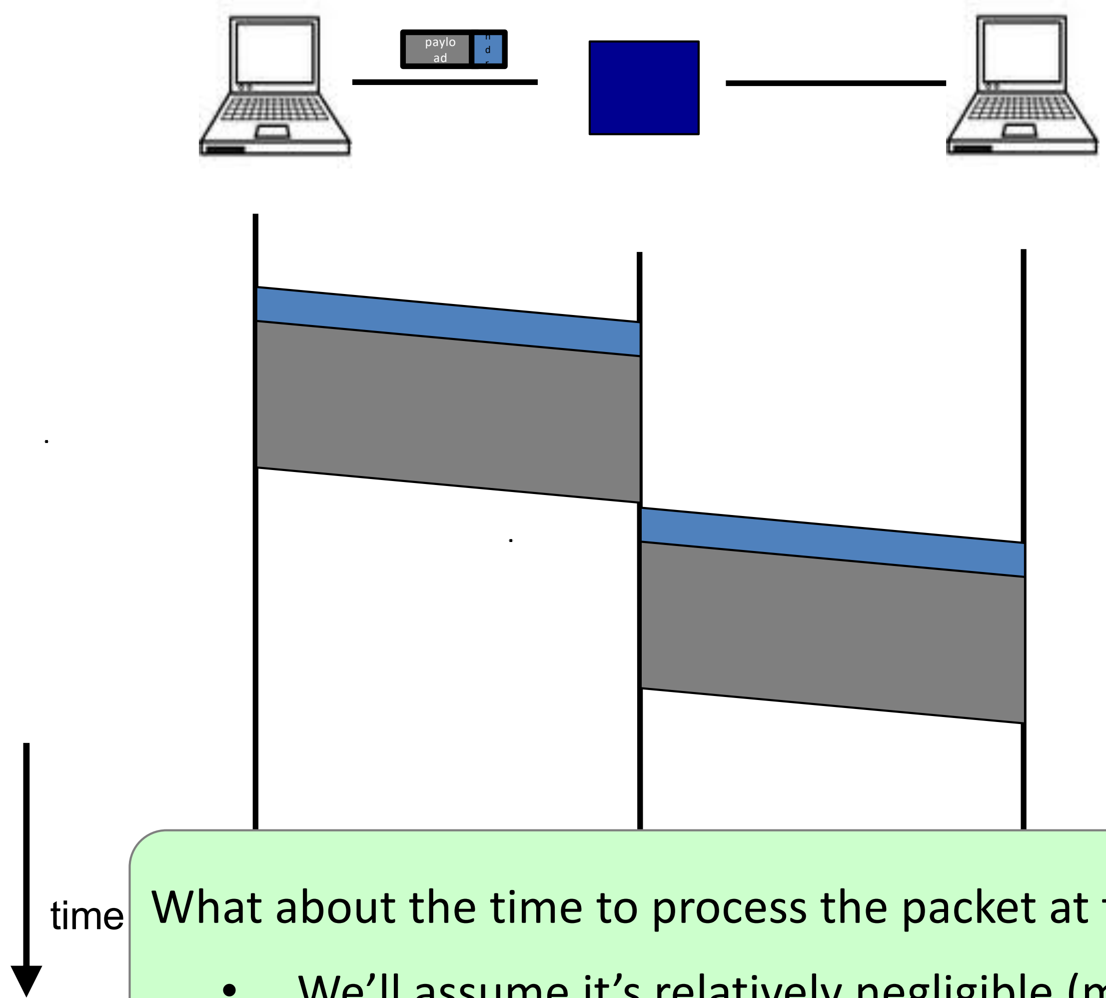
Switches forward packets



Forwarding Table

Destination	Next Hop
GLASGOW	4
OXFORD	5
EDIN	2
UCL	3

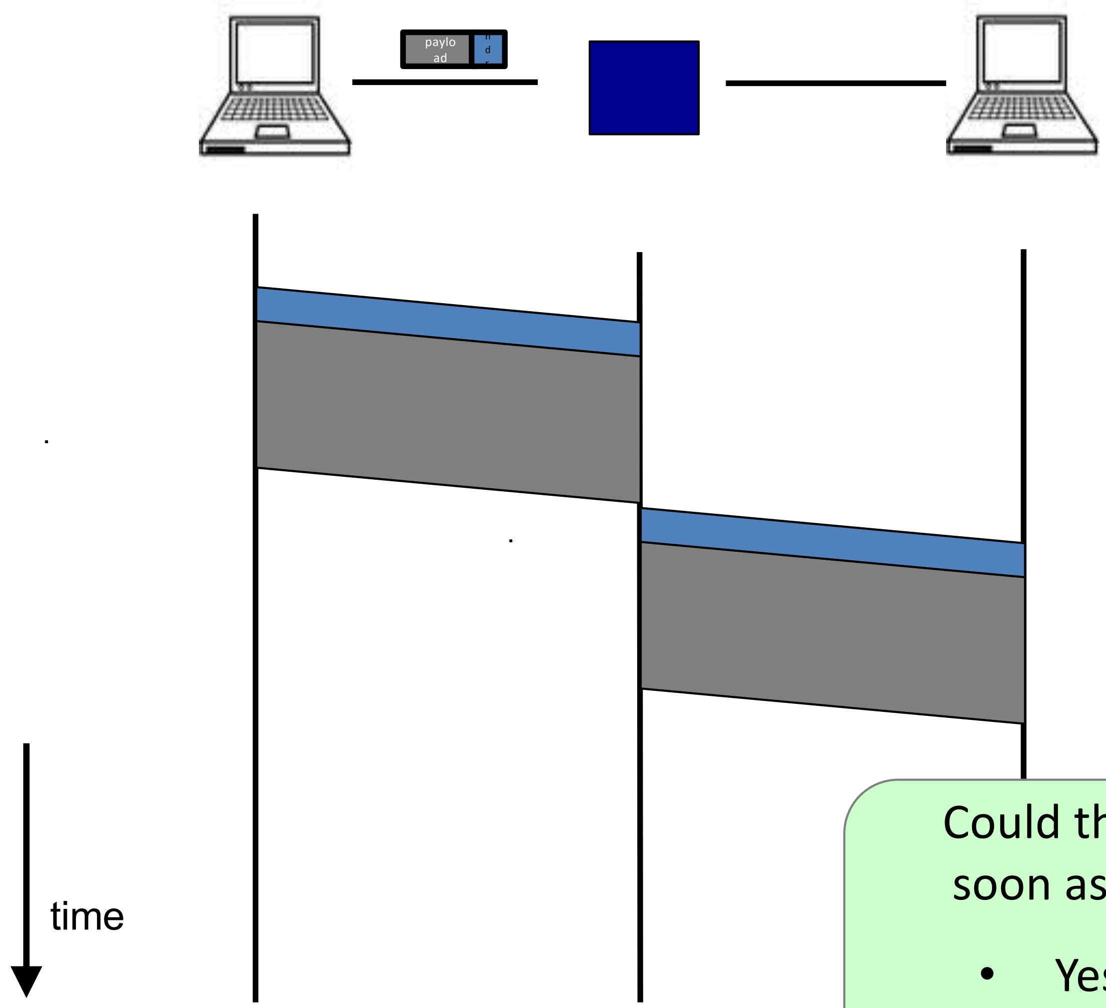
Timing in Packet Switching



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

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

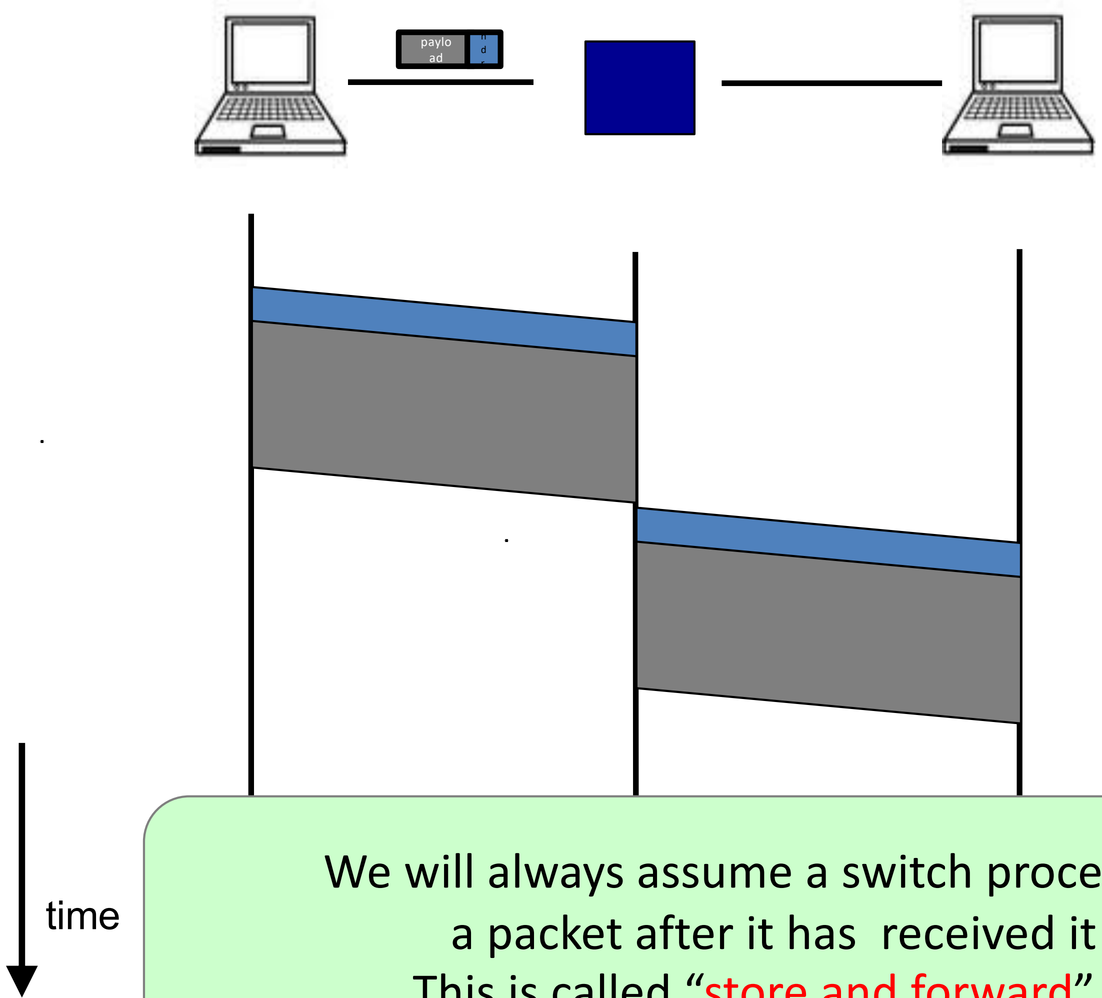
Timing in Packet Switching



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

- Yes! This would be called a “cut through” switch

Timing in Packet Switching



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

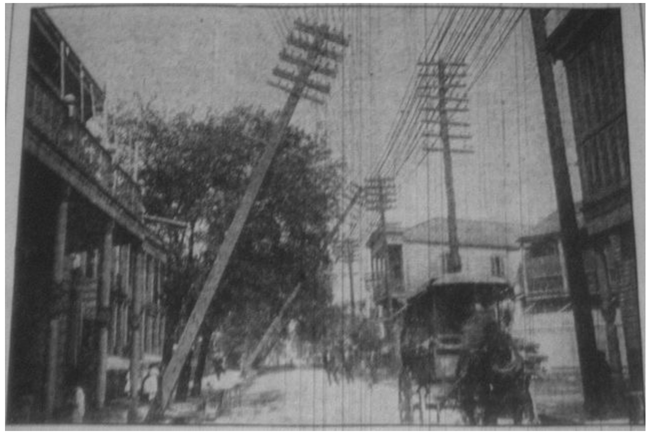
Packet Switching

- 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)

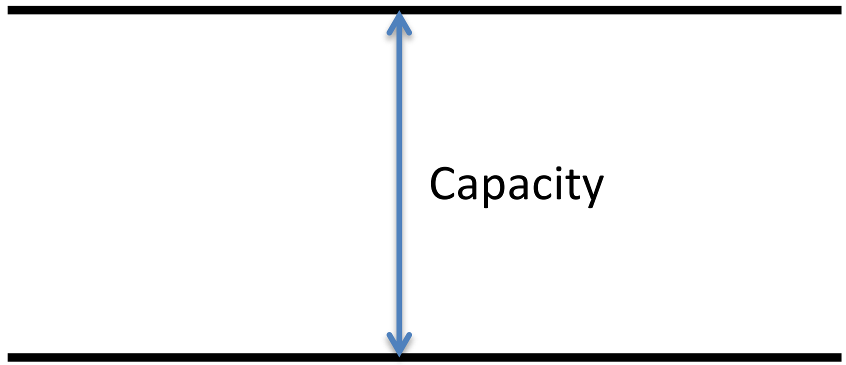
Multiplexing



Sharing makes things efficient (cost less)

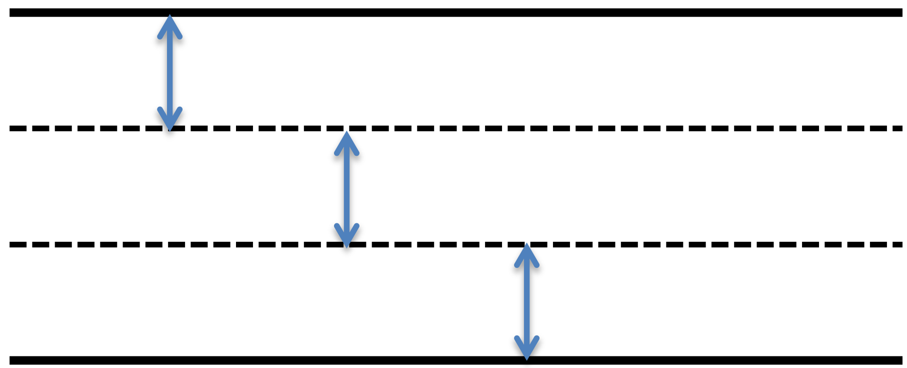
- One airplane/train for 100's of people
- One telephone for many calls
- One lecture theatre for many classes
- One computer for many tasks
- One network for many computers
- One datacenter many applications

Three Flows with Bursty Traffic



When Each Flow Gets 1/3rd of Capacity

Frequent Overloading



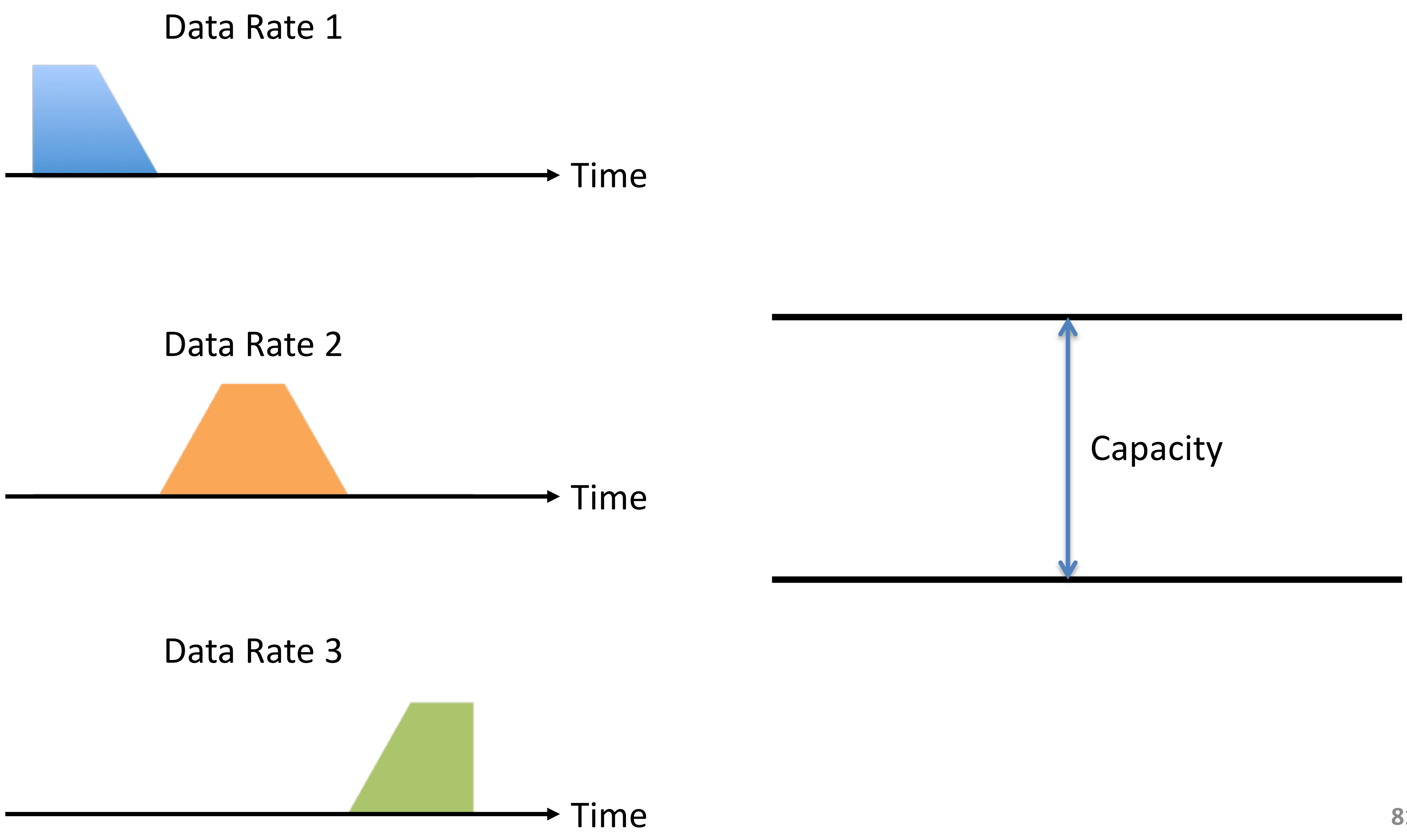
When Flows Share Total Capacity

No Overloading

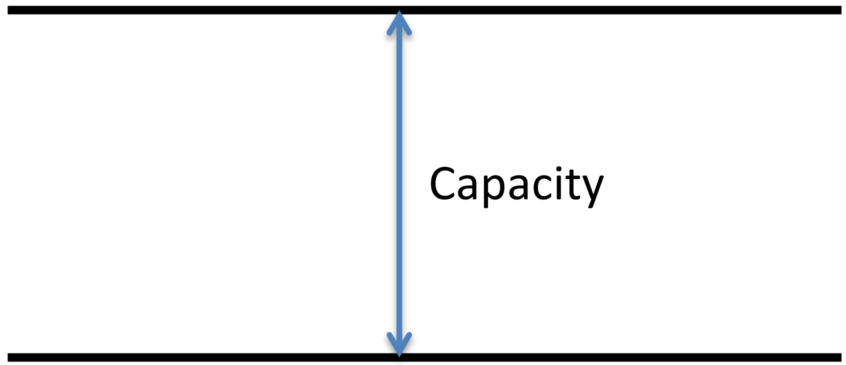
Statistical multiplexing relies on the assumption that not all flows burst at the same time.

Very similar to insurance, and has same failure case

Three Flows with Bursty Traffic

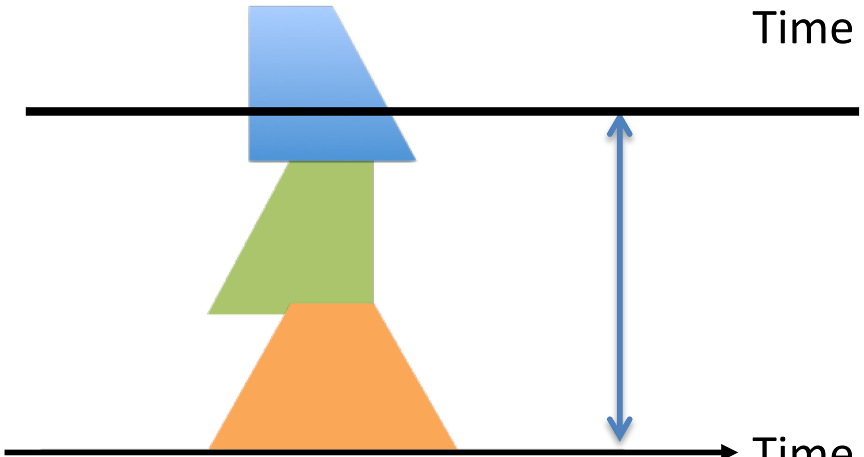


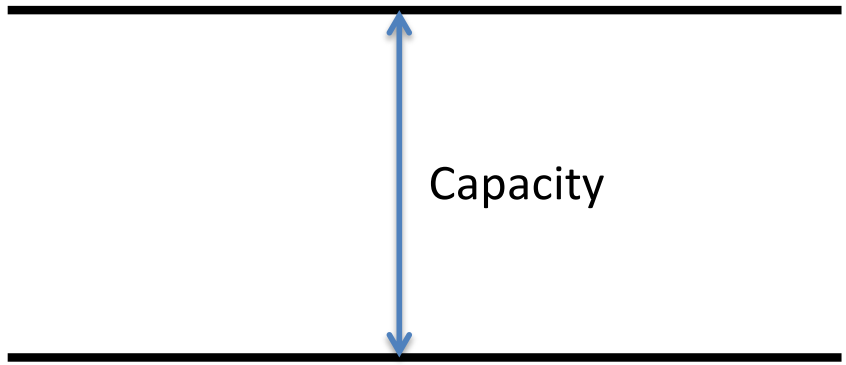
Three Flows with Bursty Traffic



Three Flows with Bursty Traffic

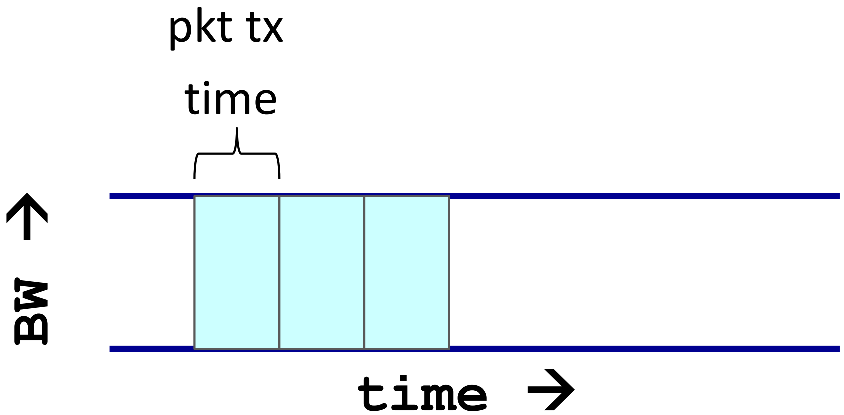
Data Rate 1+2+3 >> Capacity



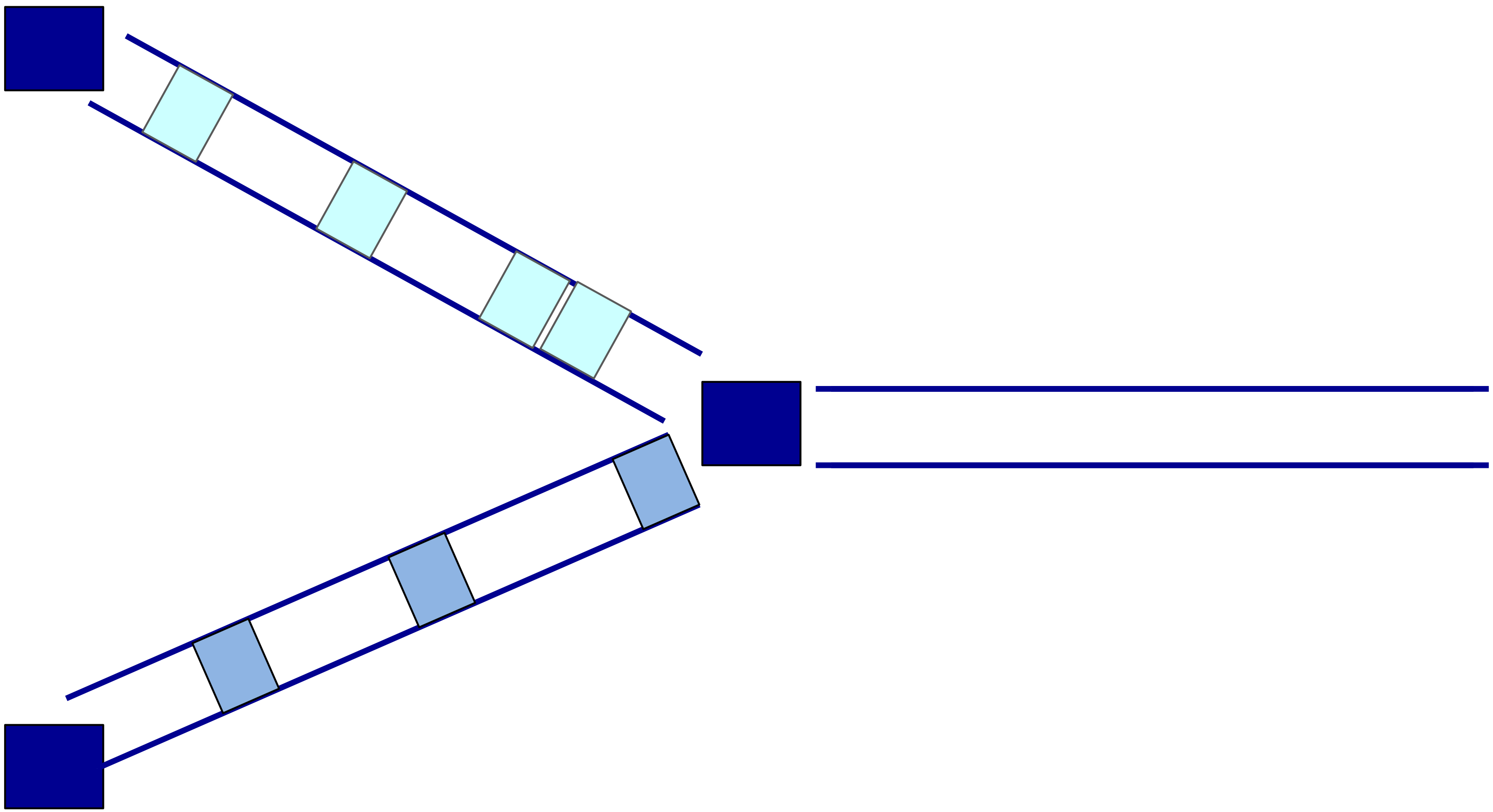


What do we do under overload?

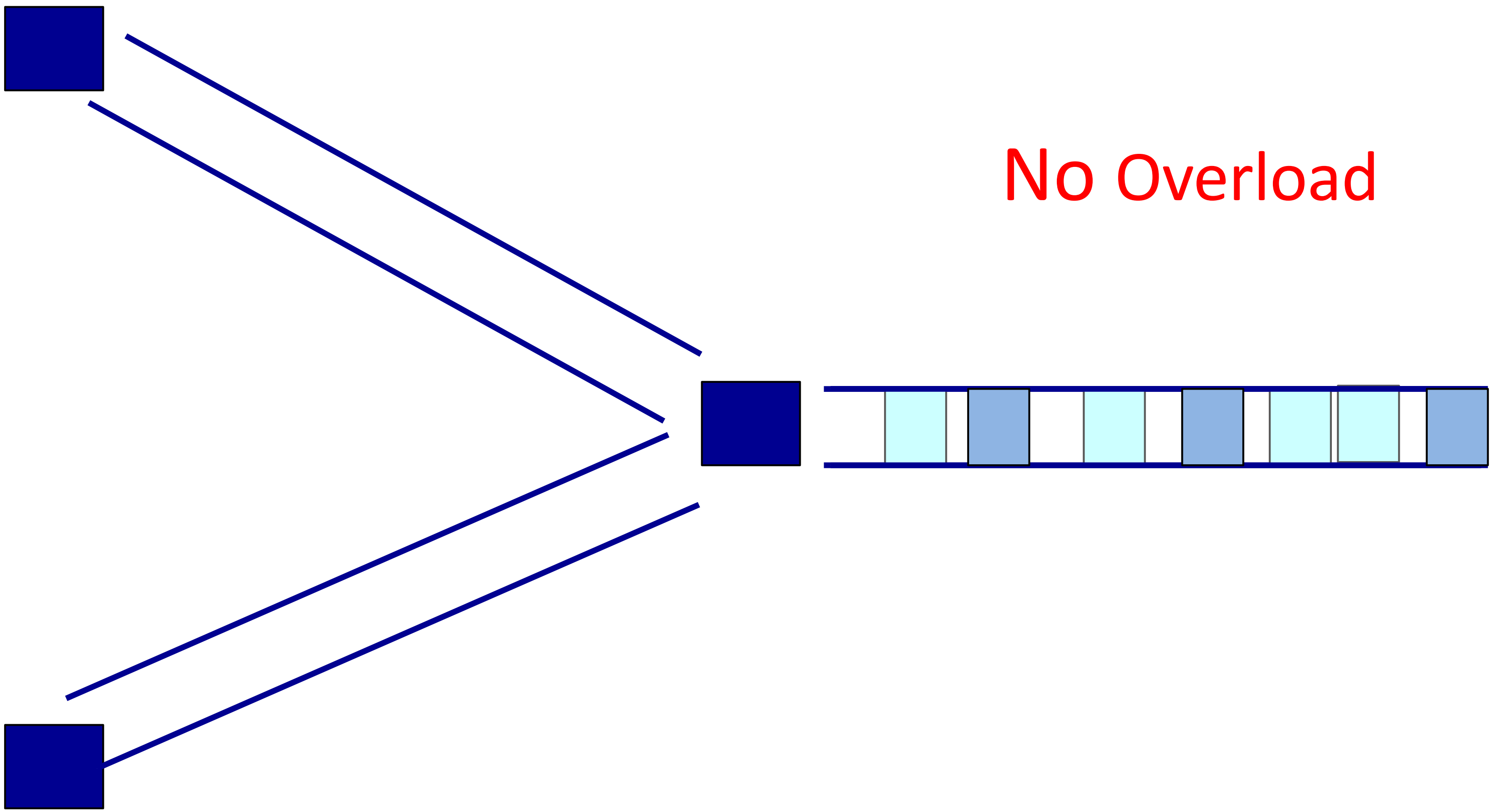
Statistical multiplexing: pipe view



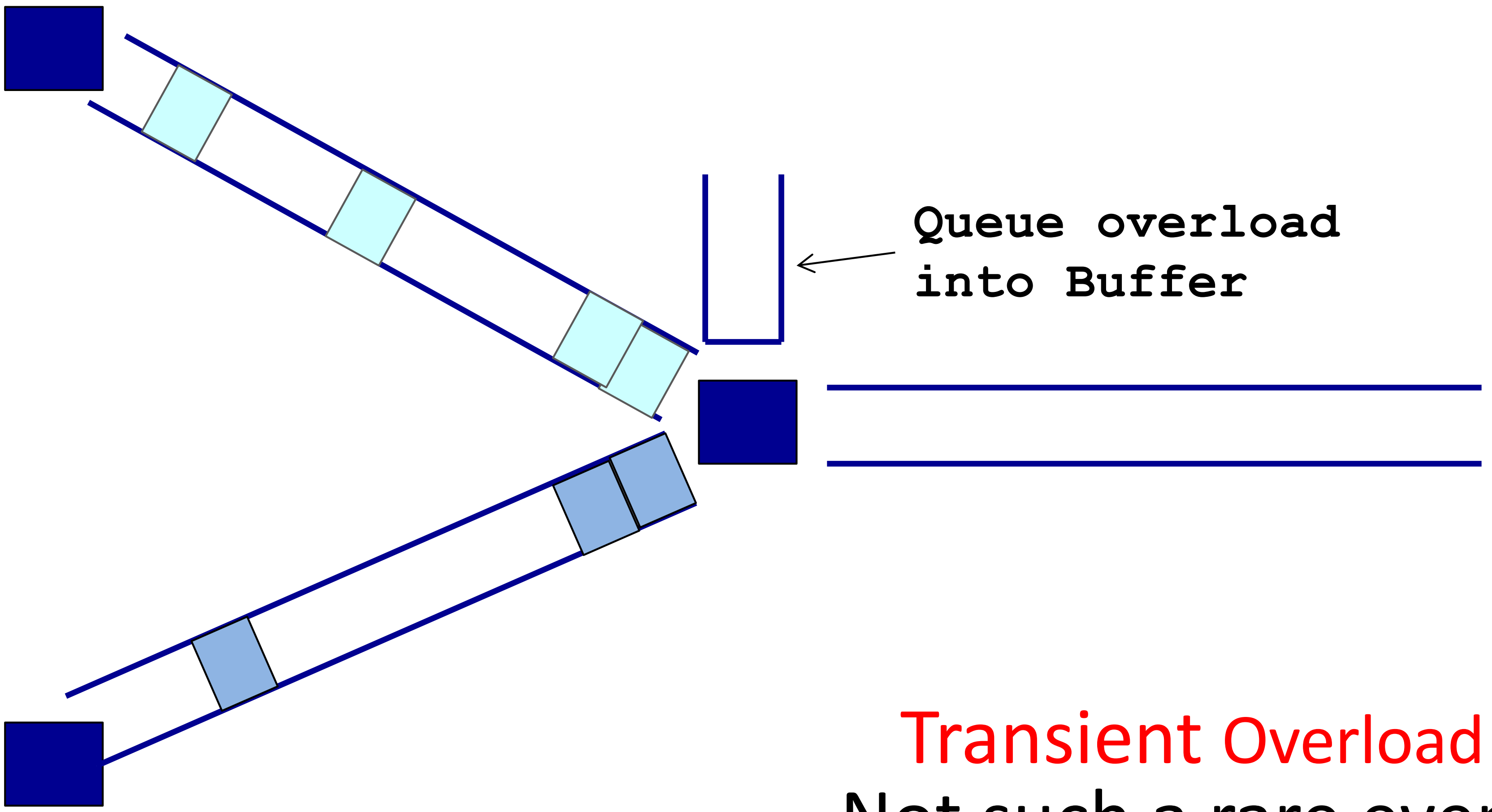
Statistical multiplexing: pipe view



Statistical multiplexing: pipe view

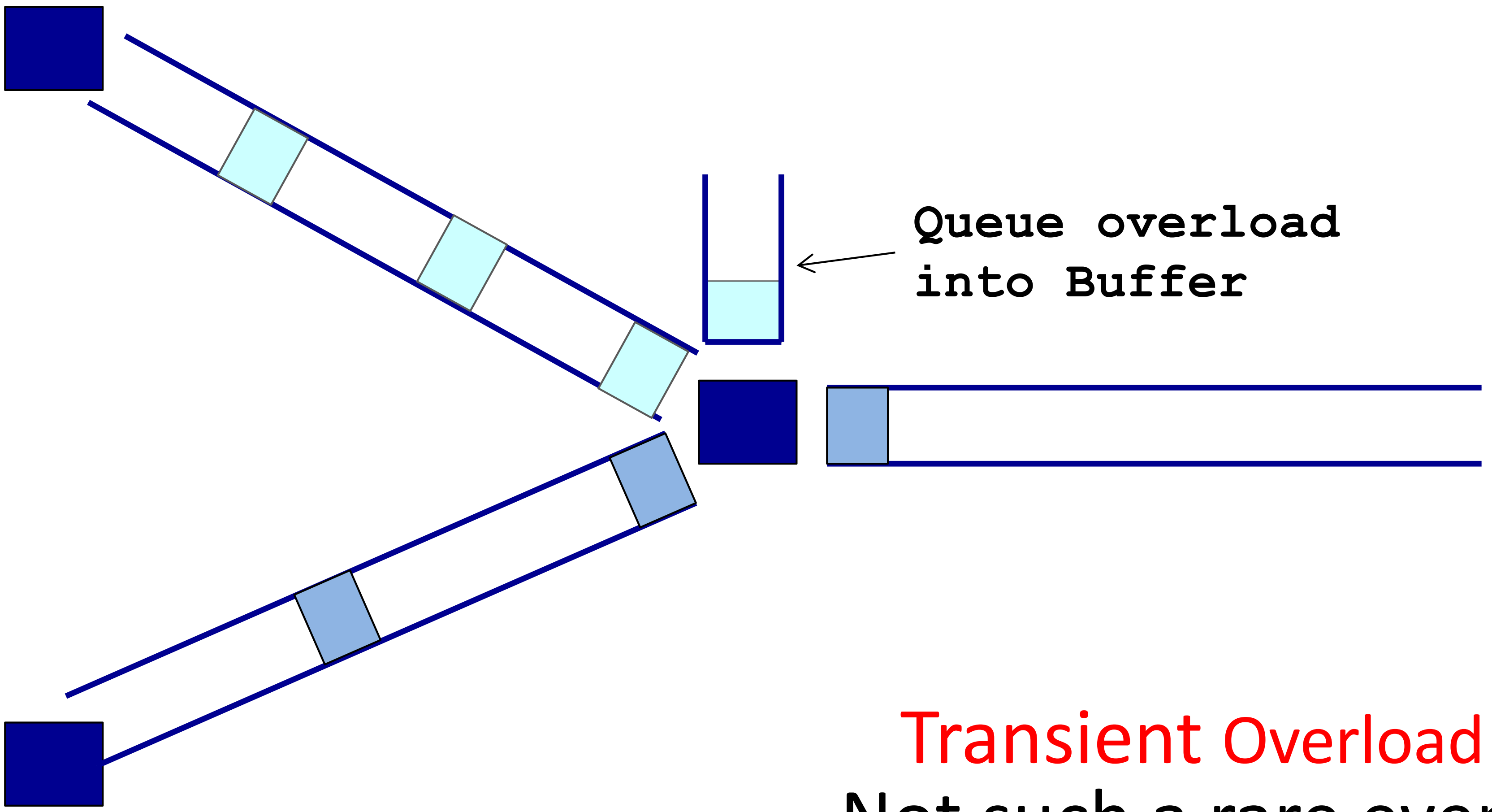


Statistical multiplexing: pipe view



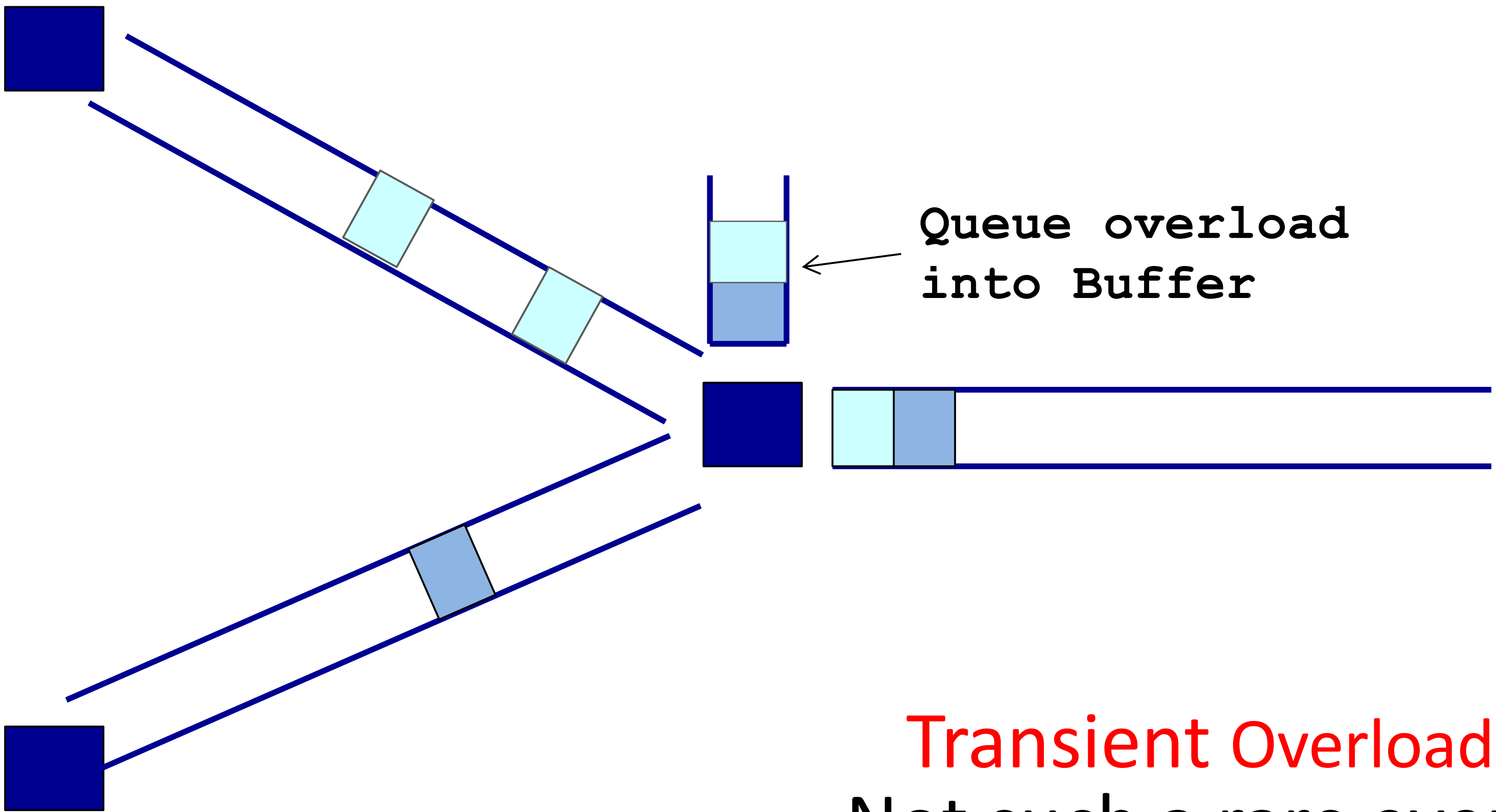
Transient Overload
Not such a rare event

Statistical multiplexing: pipe view



Transient Overload
Not such a rare event

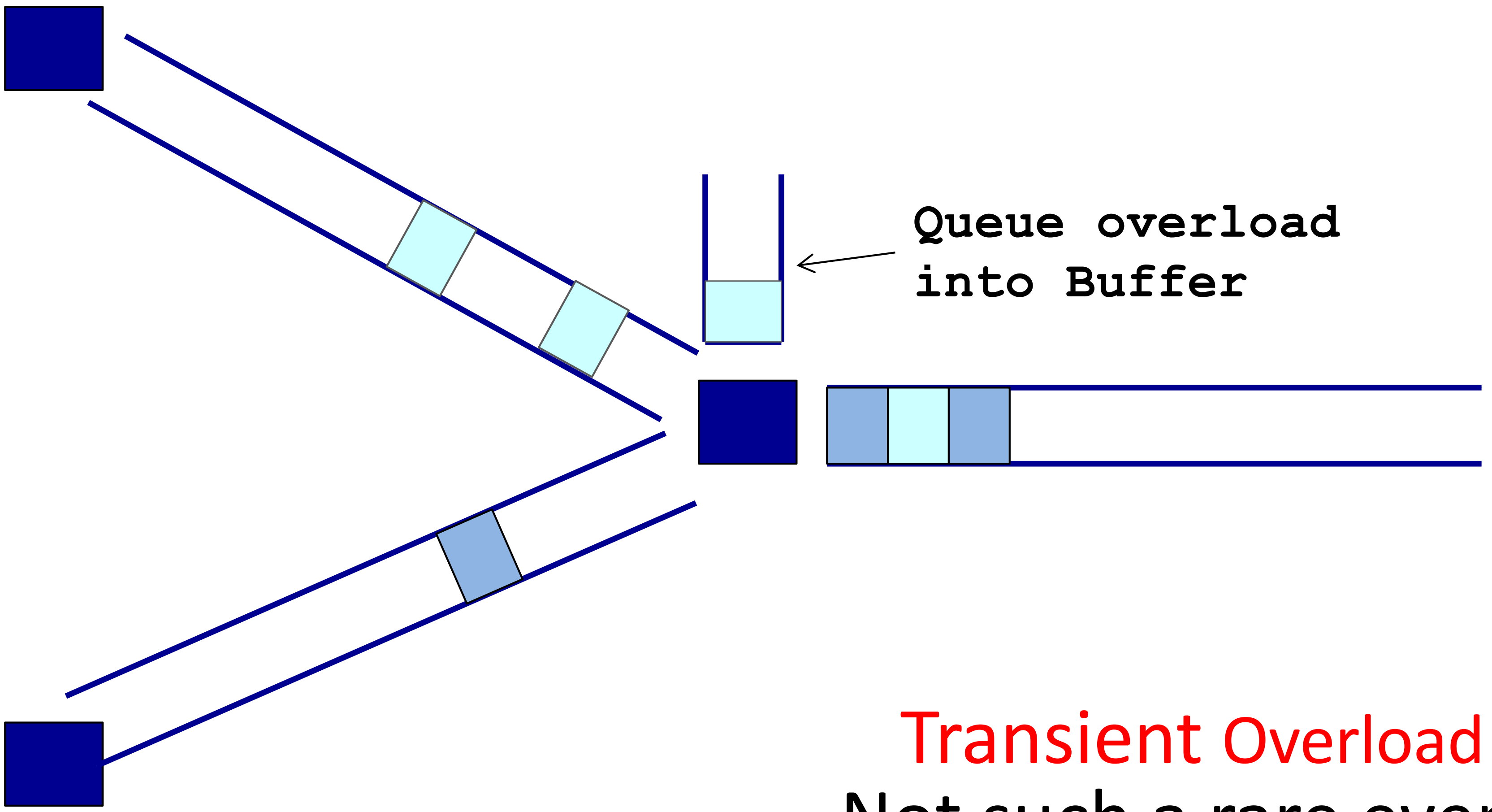
Statistical multiplexing: pipe view



Transient Overload

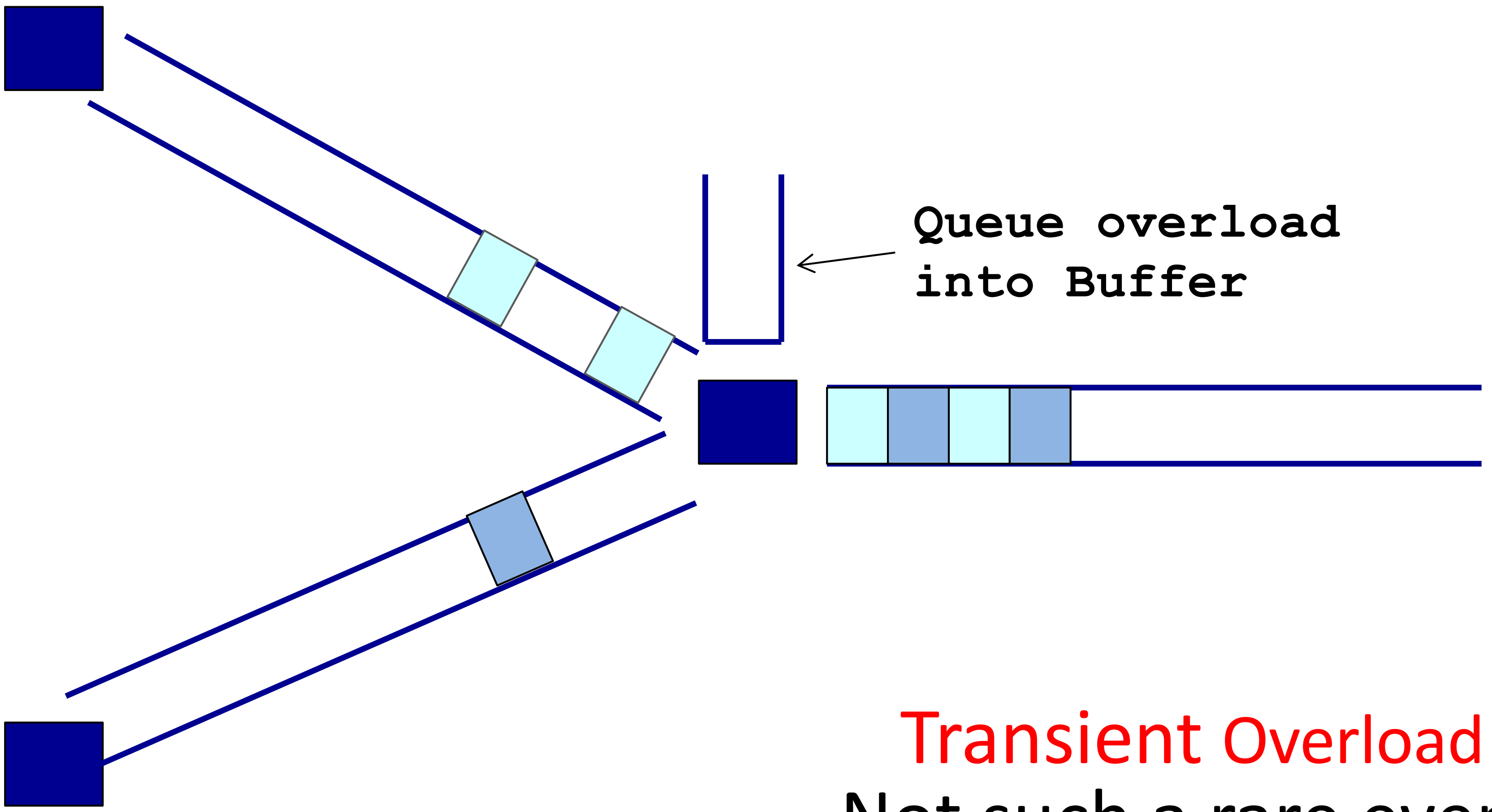
Not such a rare event

Statistical multiplexing: pipe view



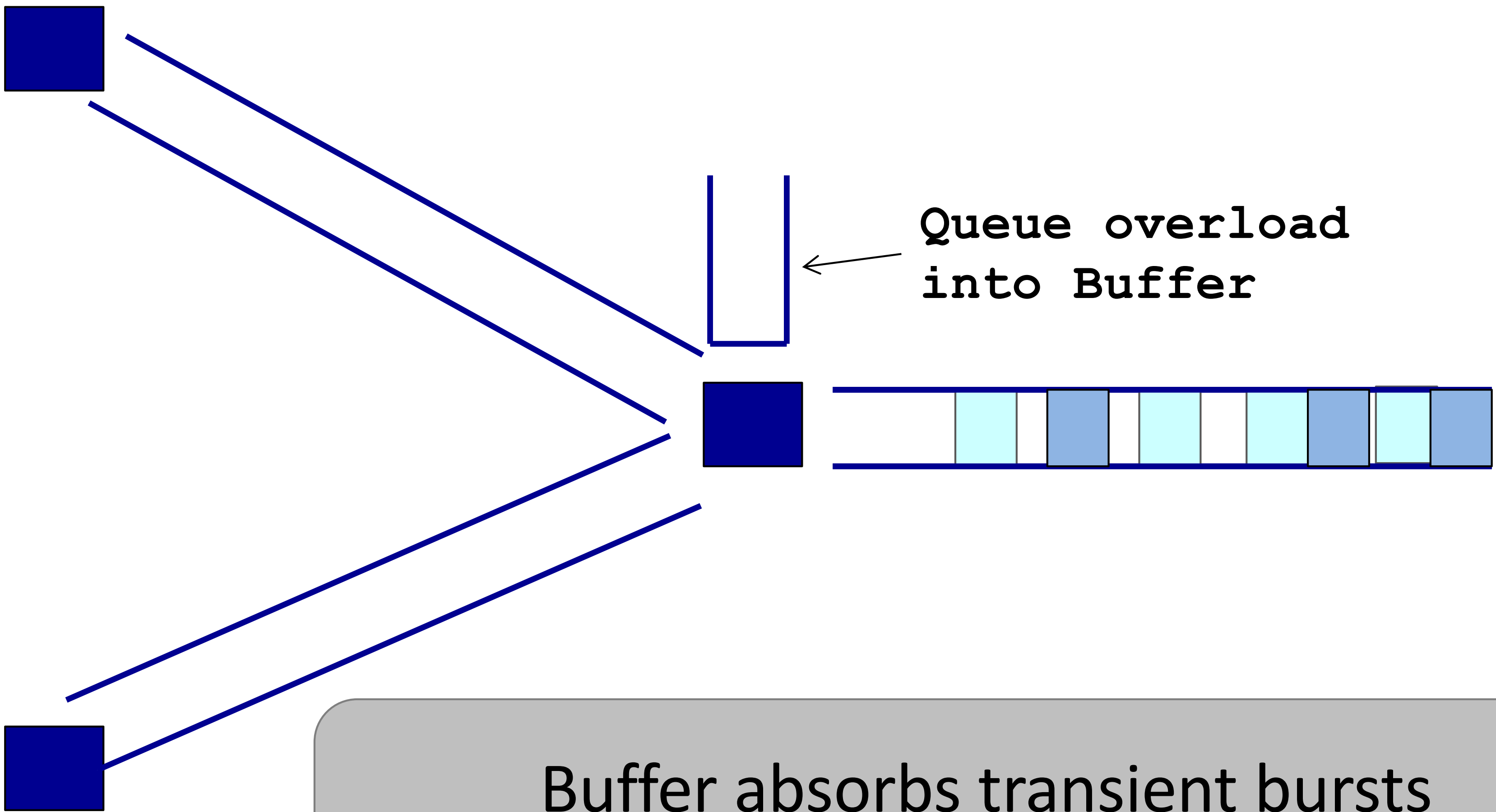
Transient Overload
Not such a rare event

Statistical multiplexing: pipe view



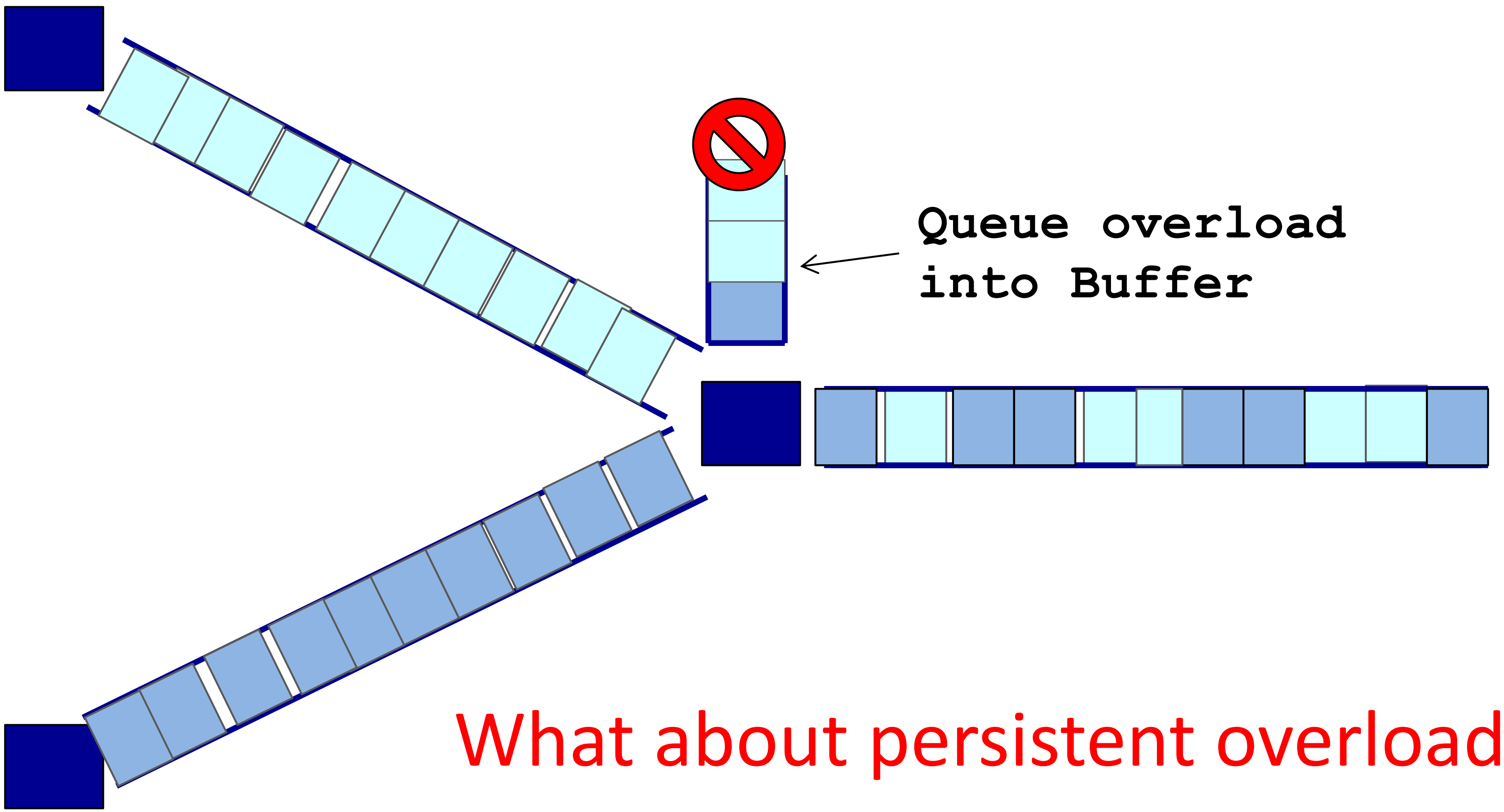
Transient Overload
Not such a rare event

Statistical multiplexing: pipe view



Buffer absorbs transient bursts
But *NOT* additional capacity

Statistical multiplexing: pipe view



What about persistent overload?

Will eventually 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 (*)

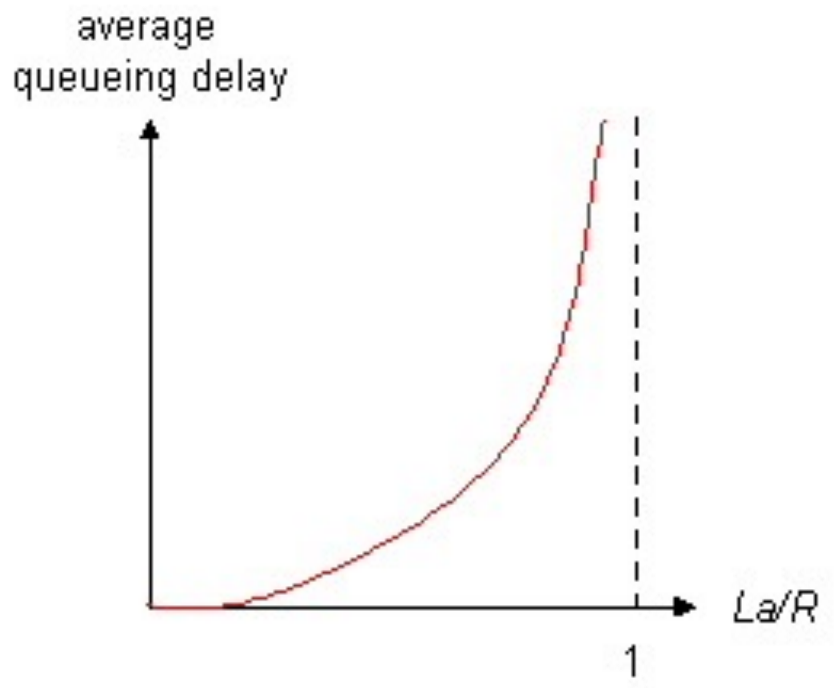
- Queuing delay caused by "packet interference"

- 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

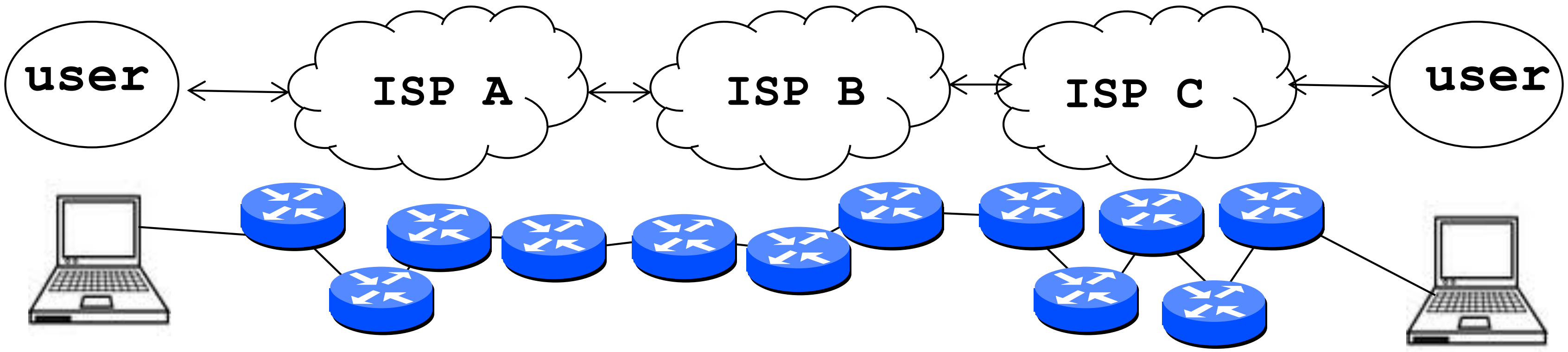


traffic intensity = La/R

- 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: rio.cl.cam.ac.uk to people.eng.unimelb.edu.au

(tracepath on winows is similar)

Three delay measurements from rio.cl.cam.ac.uk to gatwick.net.cl.cam.ac.uk

```
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  * * *
```

Direct London-Perth

Australian link

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

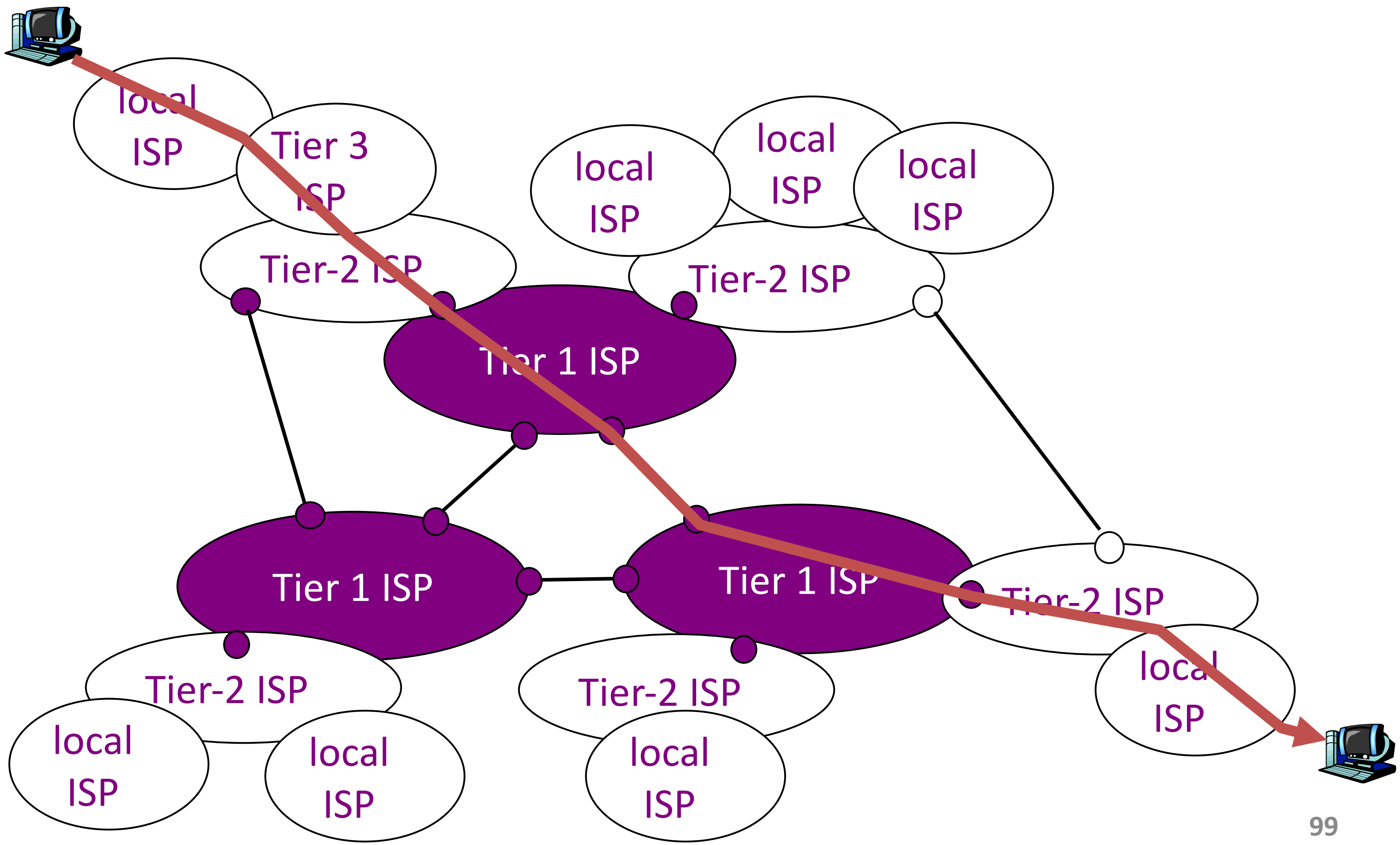
traceroute: rio.cl.cam.ac.uk to www.caida.org

```
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 (c-hi.b-jc.net.cam.ac.uk)  0.353ms  0.346ms  0.338ms
 5  131.111.7.217 (ips-out.b-jc.net.cam.ac.uk)  0.582ms  0.441ms  0.397ms
 6  146.97.41.37 (ae0.lowdss-ban1.ja.net)  2.754ms  2.648ms  2.701ms
 7  146.97.35.245 (ae26.lowdss-sbr1.ja.net)  2.852ms  2.728ms  2.738ms
 8  146.97.33.25 (ae30.erdiss-sbr2.ja.net)  5.412ms  5.177ms  4.474ms
 9  146.97.33.21 (ae31.londpg-sbr2.ja.net)  8.408ms  8.213ms  8.293ms
10  62.40.125.57 (janet-bckp.mx1.lon2.uk.geant.net)  9.199ms  9.140ms  9.108ms
11  62.40.98.64 (ae2.mx1.lon.uk.geant.net)  10.119ms  9.818ms  9.756ms
12  62.40.124.45 (internet2-gw.mx1.lon.uk.geant.net)  95.065ms  95.962ms  95.434ms
13  163.253.1.120 (fourhundredge-0-0-0-0.4079.core2.ashb.net.internet2.edu)  152.834ms  153.562ms  154.448ms
14  163.253.1.139 (fourhundredge-0-0-0-1.4079.core2.clev.net.internet2.edu)  154.008ms  153.800ms  154.429ms
15  163.253.2.17 (fourhundredge-0-0-0-2.4079.core2.eqch.net.internet2.edu)  155.463ms  154.863ms  154.334ms
16  163.253.1.66 (fourhundredge-0-0-0-18.4079.core1.eqch.net.internet2.edu)  153.802ms  153.600ms  154.553ms
17  163.253.1.206 (fourhundredge-0-0-0-1.4079.core1.chic.net.internet2.edu)  154.783ms  154.926ms  154.796ms
18  163.253.2.29 (fourhundredge-0-0-0-1.4079.core2.kans.net.internet2.edu)  152.851ms  152.414ms  154.916ms
19  163.253.1.250 (fourhundredge-0-0-0-1.4079.core2.denv.net.internet2.edu)  155.571ms  155.047ms  154.572ms
20  163.253.1.169 (fourhundredge-0-0-0-3.4079.core2.salt.net.internet2.edu)  153.369ms  153.824ms  154.321ms
21  163.253.1.114 (fourhundredge-0-0-0-8.4079.core1.losa.net.internet2.edu)  153.786ms  153.549ms  154.839ms
22  137.164.26.200 (hpr-lax-agg10--i2.cenic.net)  152.552ms  153.465ms  152.493ms
23  137.164.25.89 (hpr-sdg-agg4--lax-agg10-100ge.cenic.net)  154.682ms  154.604ms  154.752ms
24  137.164.26.43 (hpr-sdsc-100ge--sdg-hpr3.cenic.net)  167.094ms  154.553ms  154.627ms
25  192.12.207.46 (medusa-mx960.sdsc.edu)  154.854ms  154.646ms  156.379ms
26  192.172.226.122 (proxy.caida.org)  154.581ms  154.390ms  154.477ms
```

A little more interesting because each hop resolves to a name (caida is in San Diego)

Internet structure: network of networks

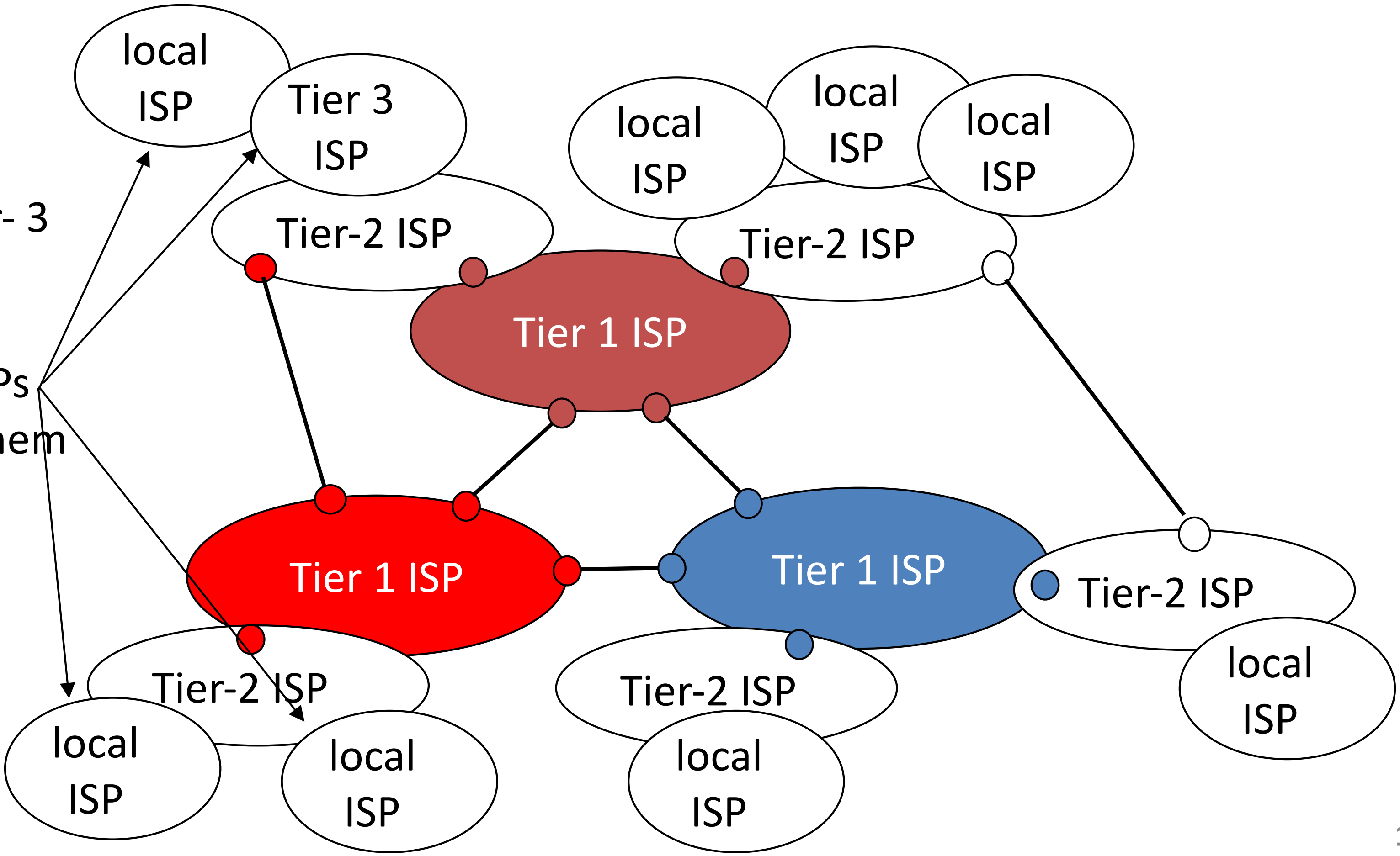
- a packet passes through many networks!



Internet structure: network of networks

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

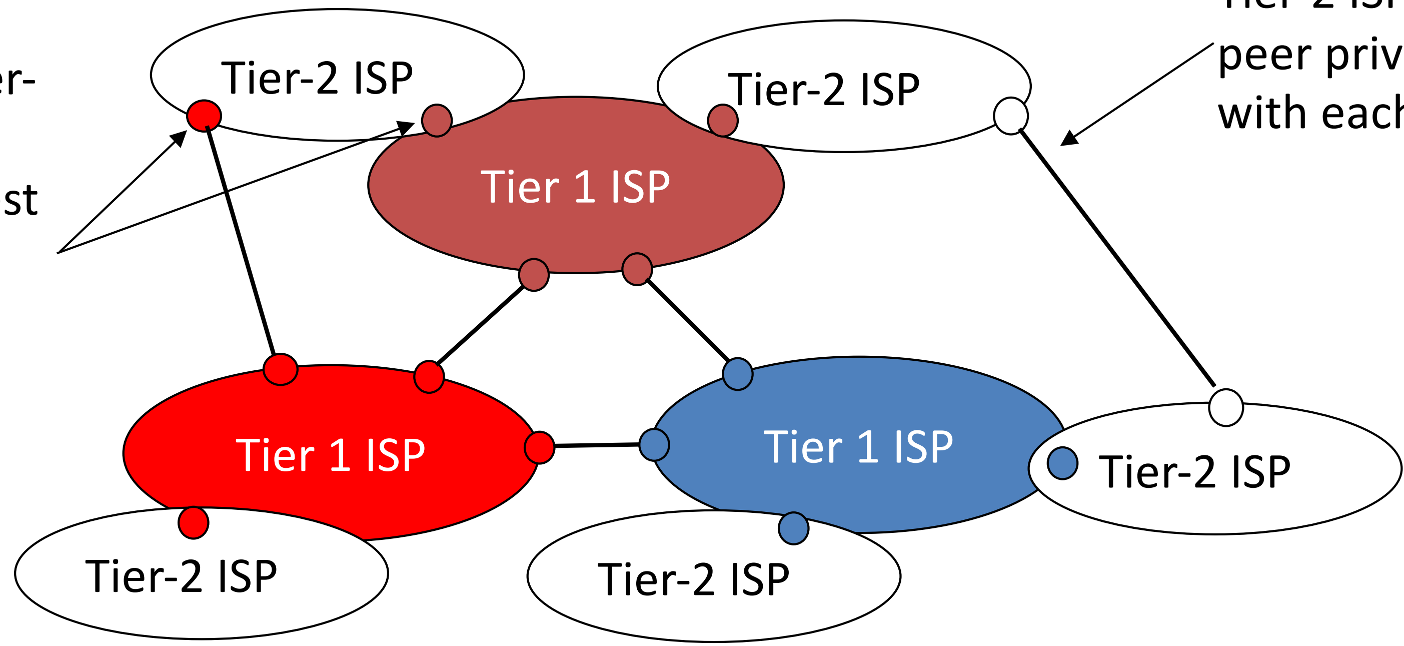
Local and tier- 3 ISPs are *customers* of higher tier ISPs connecting them to rest of Internet



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

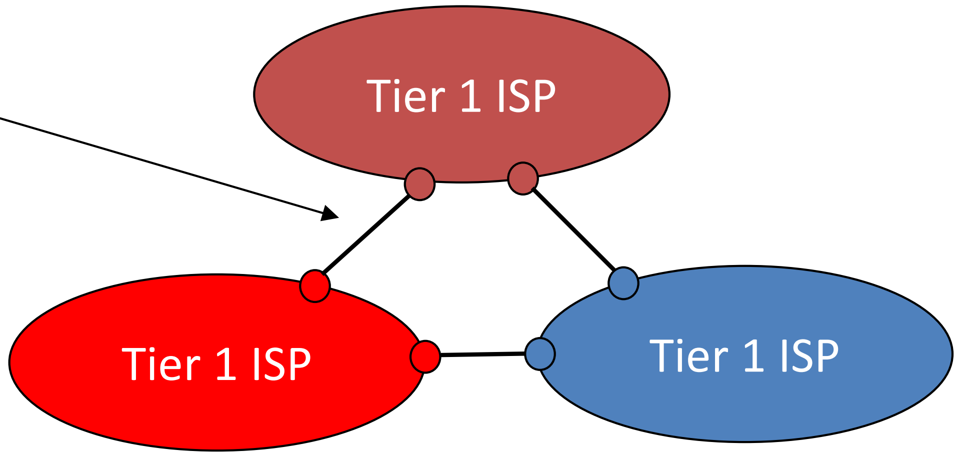


Tier-2 ISPs also peer privately with each other.

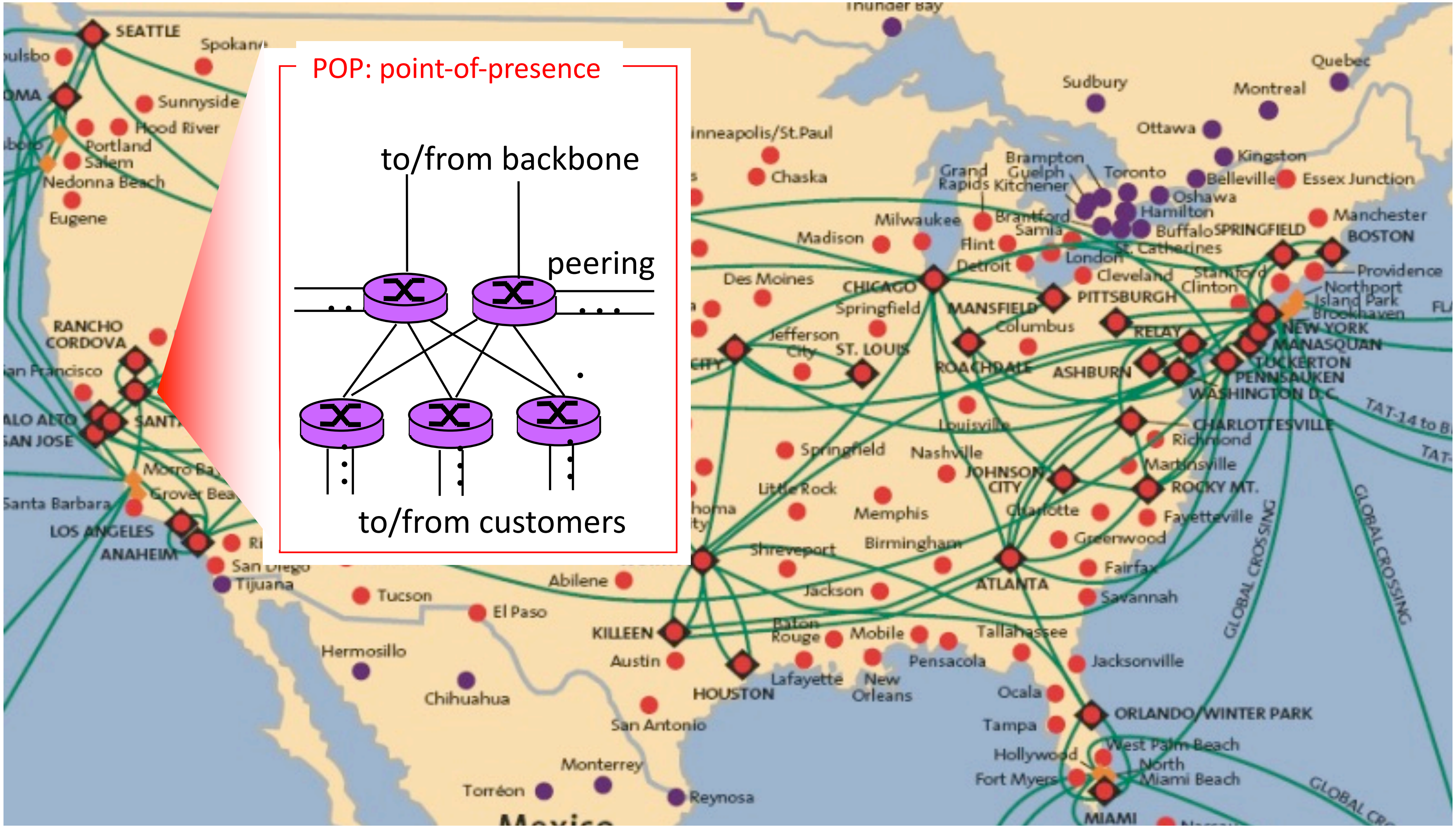
Internet structure: network of networks

- 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 providers interconnect (peer) privately



Tier-1 ISP: e.g., Sprint



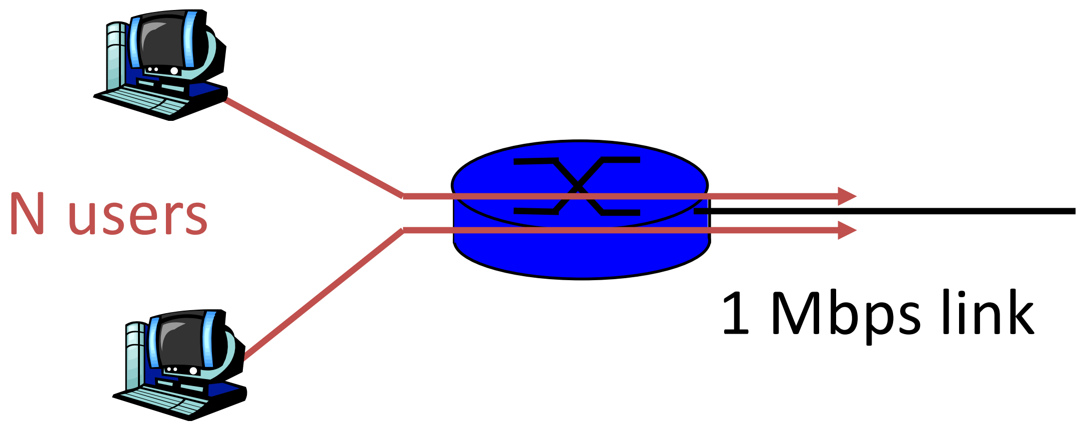
Packet Switching

- 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

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

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

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

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

where $p = 0.1$:

$$P(u \leq 10) = 0.99958$$

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

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