Flow Control; Congestion Control

Lecture 16

http://www.cs.rutgers.edu/~sn624/352-F24

Srinivas Narayana

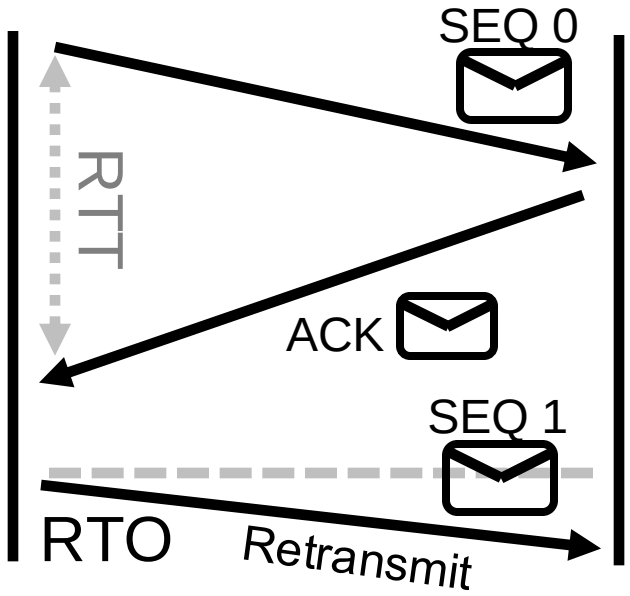
RUTGERS
UNIVERSITY | NEW BRUNSWICK

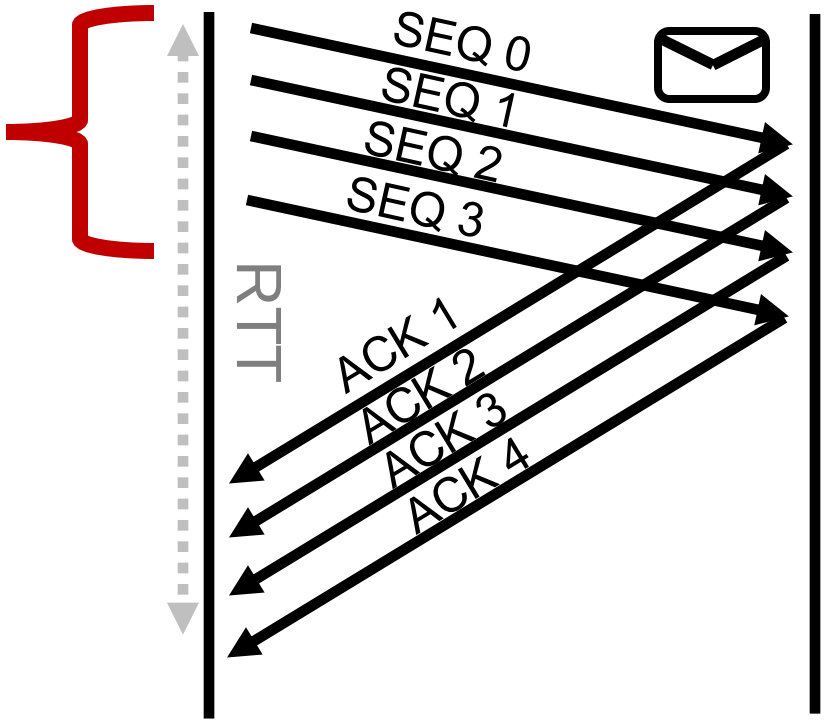
= window size

Proportional to throughput

How much data to keep in flight?

Stop and Wait



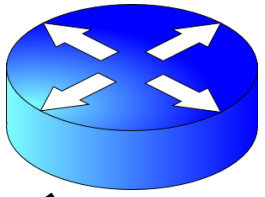


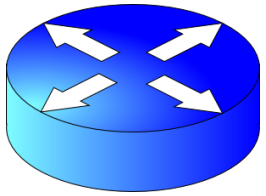
Pipelined Reliability

sender

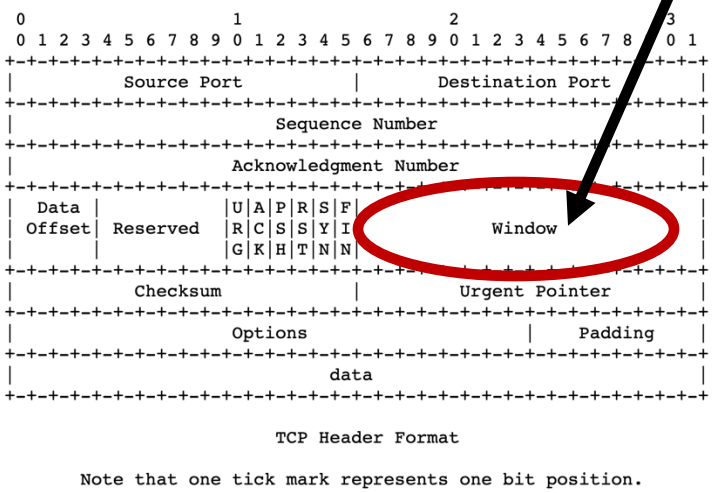
Multiple locations for bottlenecks

Congestion Control





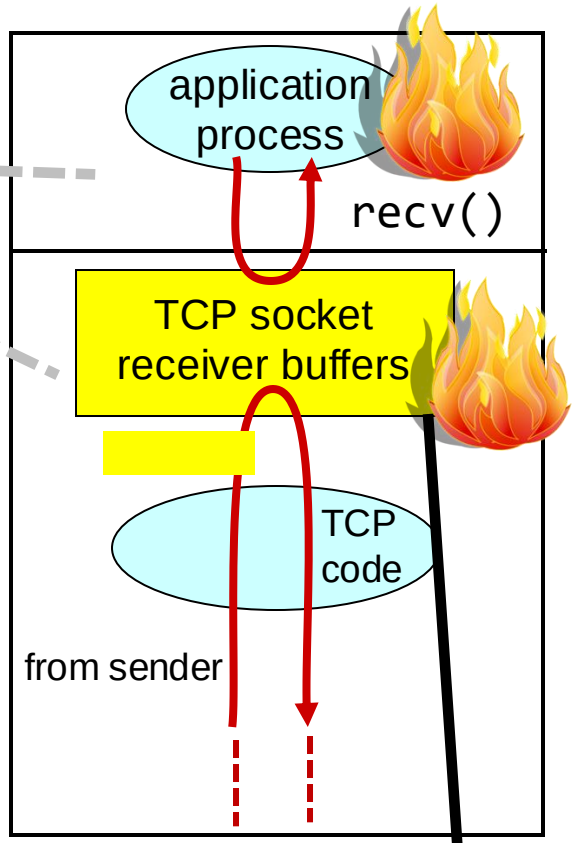
What's the bottleneck? How to adapt how much data to keep in flight?



TCP Header Format

Note that one tick mark represents one bit position.

Flow Control: Receiver informs sender free buffer over time



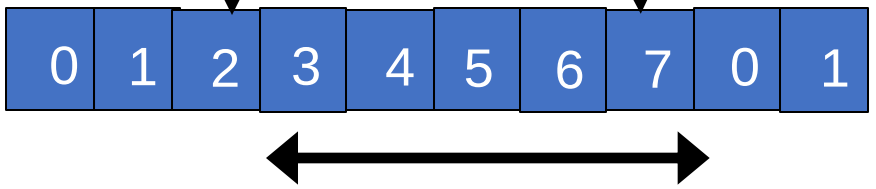
Flow Control

receiver

Last cumulative ACK'ed seq #

Last transmitted seq #

Sender's view:

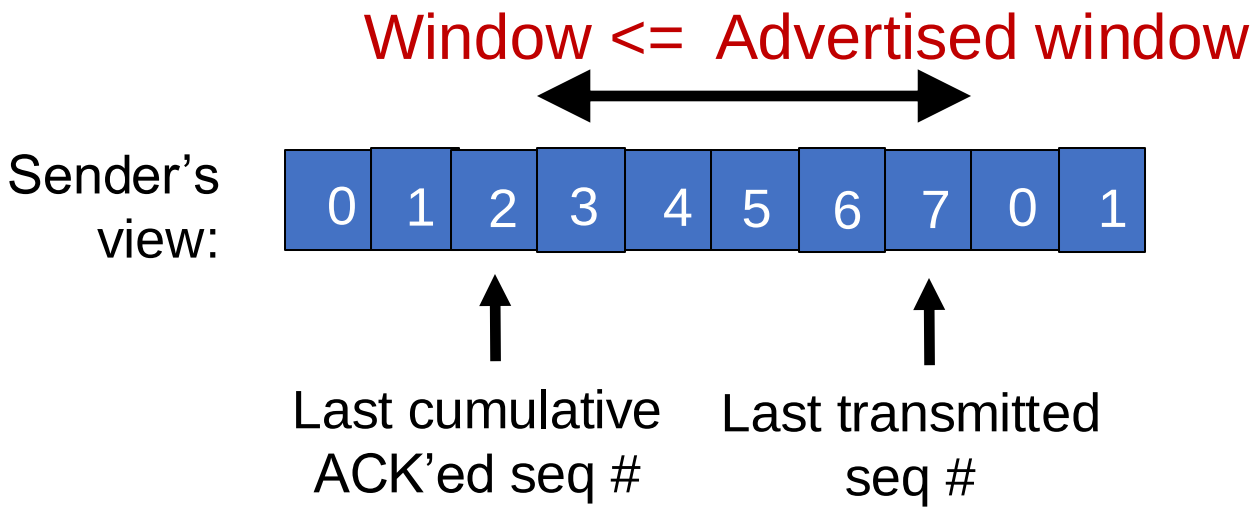


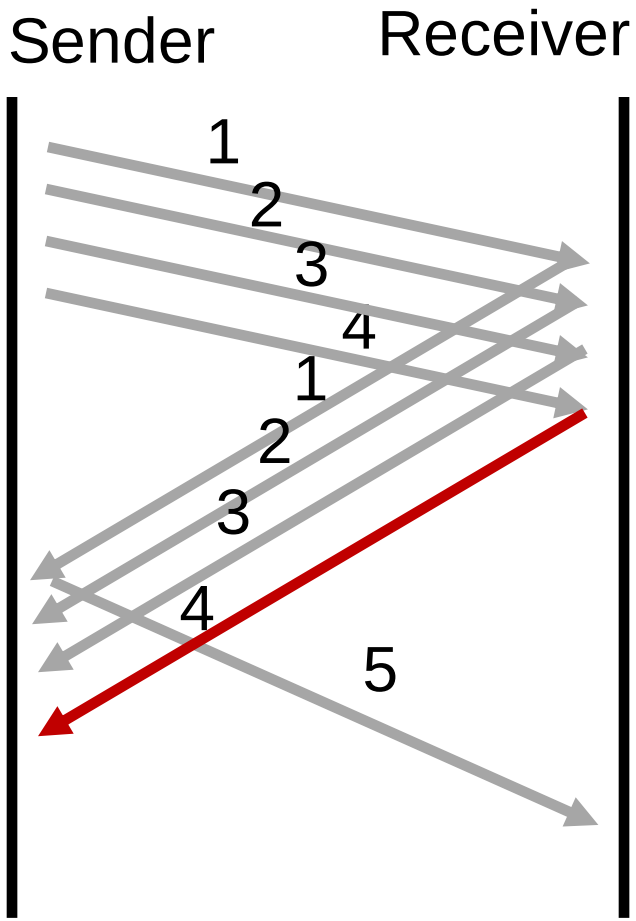
How to size this buffer?

Window <= Advertised window

TCP flow control

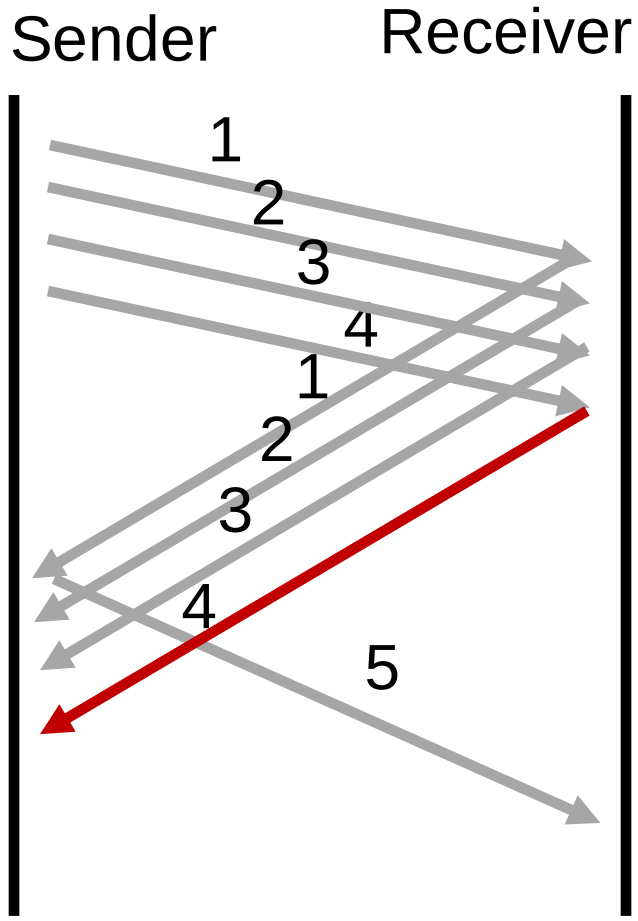
- If receiver app is too slow reading data:
  - receiver socket buffer fills up
  - => advertised window shrinks
  - => sender's window (sending rate) reduces
  - => sender's socket buffer fills up
  - => sender process put to sleep upon send()

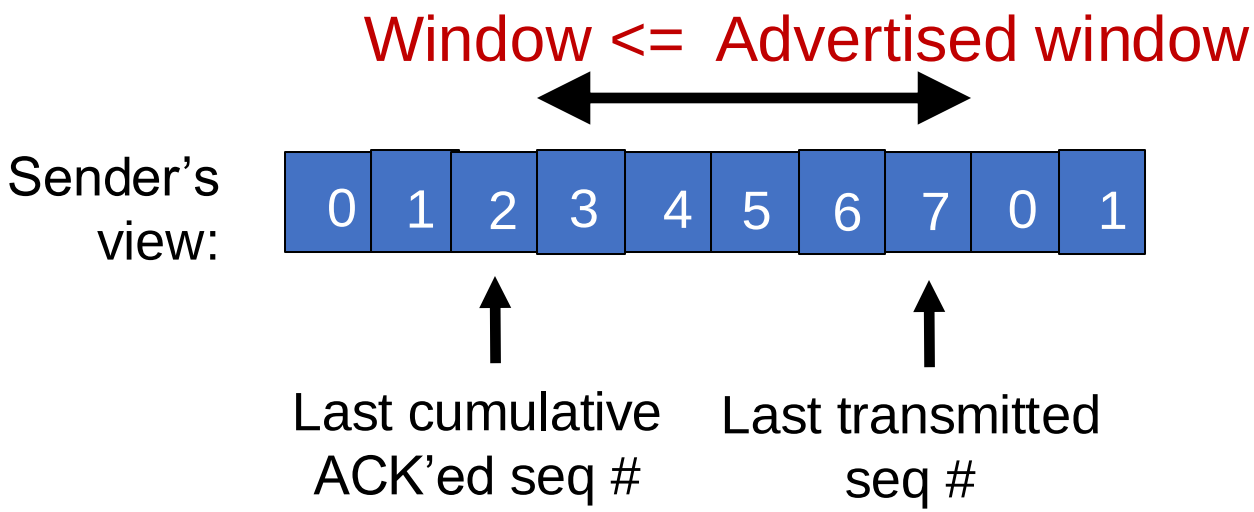




TCP flow control

Flow control matches the sending process's write speed to the receiving process's read speed.





Sizing the receiver's socket buffer

- Operating systems have a default receiver socket buffer size
  - Listed among `sysctl -a | grep net.inet.tcp` on MAC
  - Listed among `sysctl -a | grep net.ipv4.tcp` on Linux

- If socket buffer is too small, sender can't keep too many packets in flight ➔ lower throughput

- If socket buffer is too large, too much memory consumed per socket

- How big should the receiver socket buffer be?

Sizing the receiver's socket buffer

- Case 1: Suppose the receiving app is reading data too slowly:
- No amount of receiver buffer can prevent low throughput (for a long-lived connection).
- Flow control matches throughput to the receiving app's (low) speed

Sizing the receiver's socket buffer

- Case 2: Suppose the receiving app reads sufficiently fast *on average* to match the sender's writing speed.
  - Assume the sender desires a window of size W.
  - The receiver must use a buffer of size at least W. Why?
- Captures two cases:
- (1) When the first sequence #s in the window are dropped
  - *Selective repeat*: data in window buffered until the "hole" within the window can be filled by the sender. Advertised window reduces sender's window
- (2) When the sender sends a burst of data of size W
  - The receiver may not keep up with the *instantaneous* rate of the sender
- Set receiver socket buffer size > desired window size

Summary of flow control

- Keep memory buffers available at the receiver whenever the sender transmits data
- Buffers needed to hold for selective repeat, reassembling data in order, and until applications can read data
- Inform available buffer to sender on an ongoing basis, with each ACK
- Function: match sender speed to receiver speed

- Correct socket buffer sizing is important for TCP throughput
  - Throughput = window size / RTT <= receiver socket buffer / RTT

Info on (tuning) TCP stack parameters

- https://www.ibm.com/support/knowledgecenter/linuxonibm/liaag/wkvm/wkvm_c_tune_tcpip.htm

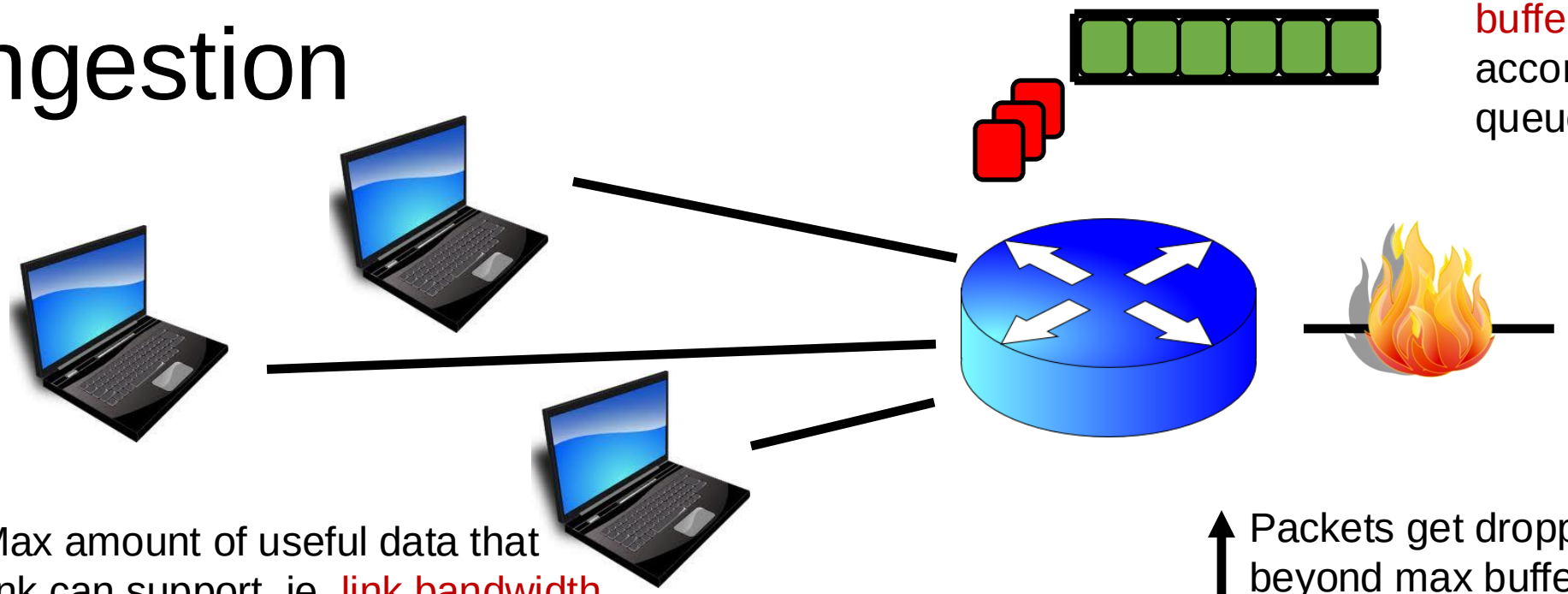
- https://cloud.google.com/solutions/tcp-optimization-for-network-performance-in-gcp-and-hybrid

Playing around with socket buffer sizes

- iperf –s ; iperf –c localhost –i 1
- ping localhost
- sudo tc qdisc add dev lo root netem delay 100ms
- sudo sysctl net.ipv4.tcp_rmem # min, default, max
- Default buffer size 128KB; change e.g., 2.56MB by using
  - sudo sysctl net.ipv4.tcp_rmem="4096 2621440 6291456"
- Clean up and restore to defaults
- sudo tc qdisc del dev lo root netem
  - If needed:
  - sudo sysctl net.ipv4.tcp_rmem="4096 131072 6291456"

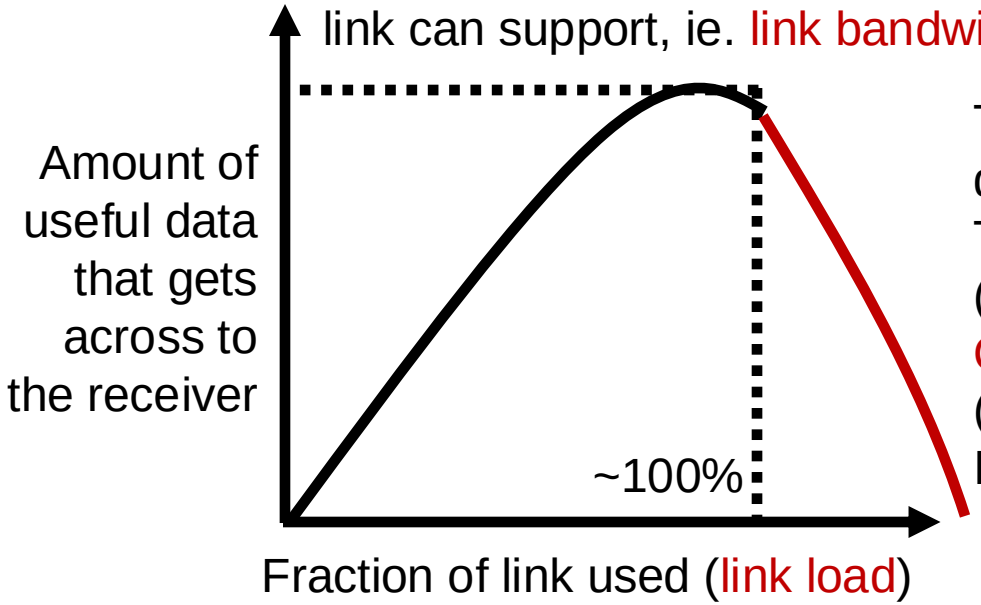
Congestion Control

Congestion

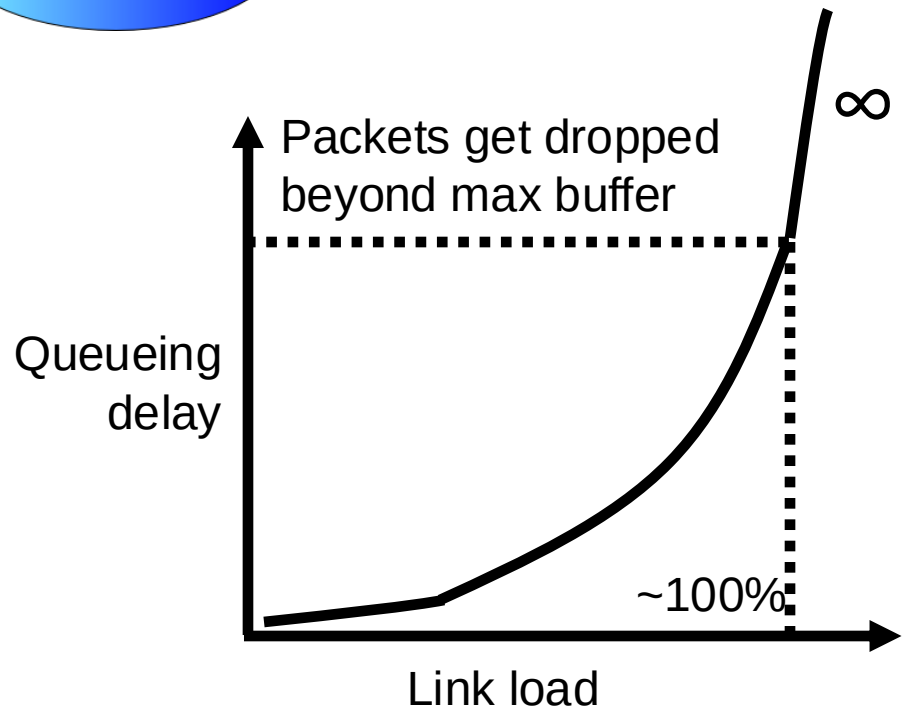


Routers have buffers which accommodate queued packets.

Max amount of useful data that link can support, ie. link bandwidth

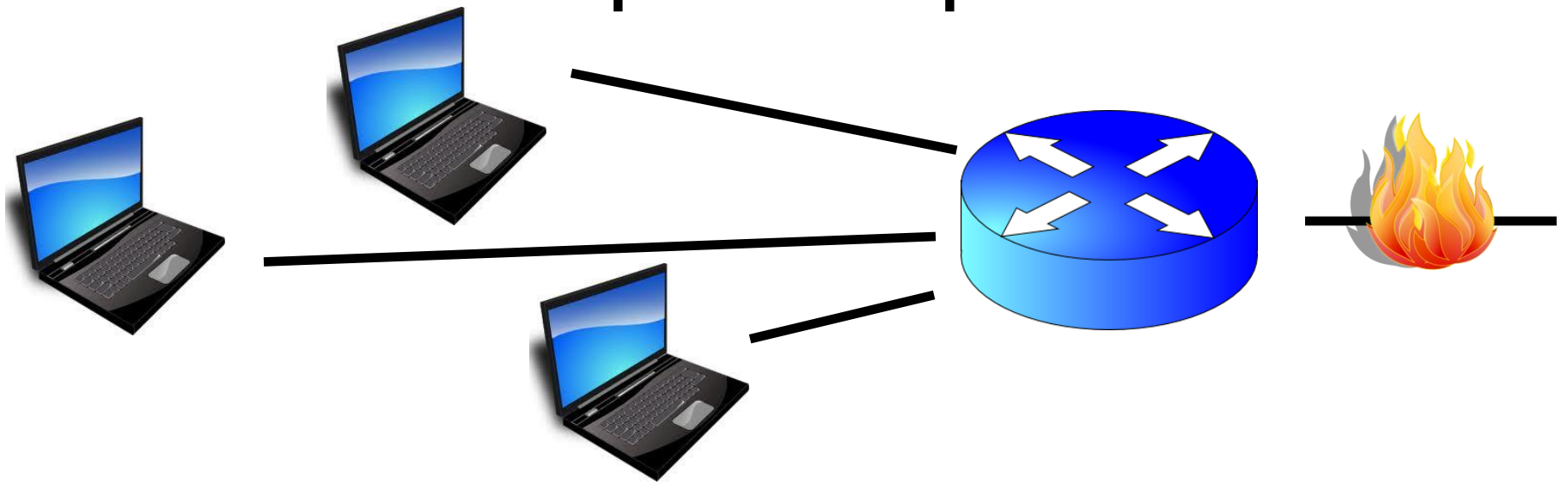


Too many retransmissions due to packet drops!
The amount of useful (fresh) data plummets.
Congestion collapse (occurred for real on the Internet in the last 80s)



https://en.wikipedia.org/wiki/Network_congestion#Congestive_collapse

How should multiple endpoints share net?



- It is difficult to know where the bottleneck link is
- It is difficult to know how many other endpoints are using that link
- Endpoints may join and leave at any time
- Network paths may change over time, leading to different bottleneck links (with different link rates) over time