Computer Networks COL 334/672

Congestion Control

Tarun Mangla

Slides adapted from KR

Sem 1, 2024-25

Quiz: Moodle

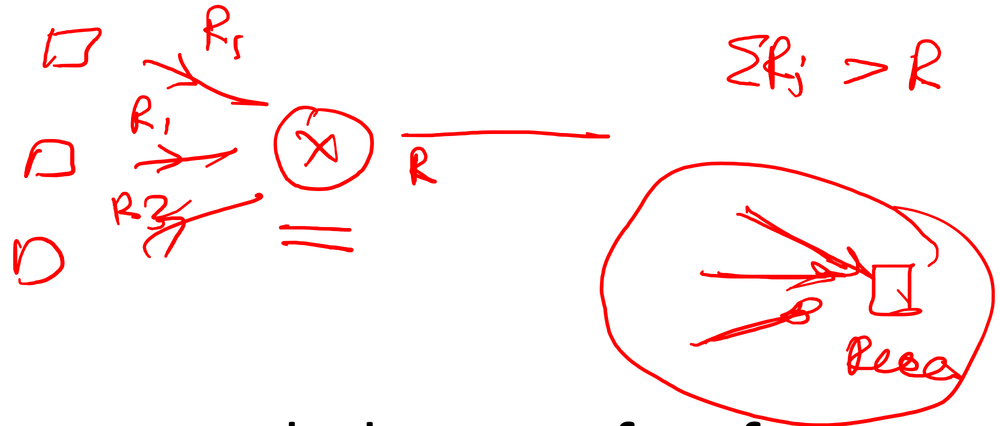
Password: tcp

Recap

- Transport Congestion Protocol
  - Connection establishment
  - Reliability
  - Flow control
  - **Congestion control**

Congestion Control

control sending rate



Congestion:

- informally: "too many sources sending too much data too fast for *network* to handle"
- manifestations:
  - long delays (queueing in router buffers)
  - packet loss (buffer overflow at routers)
- different from flow control!
- a top-10 problem!

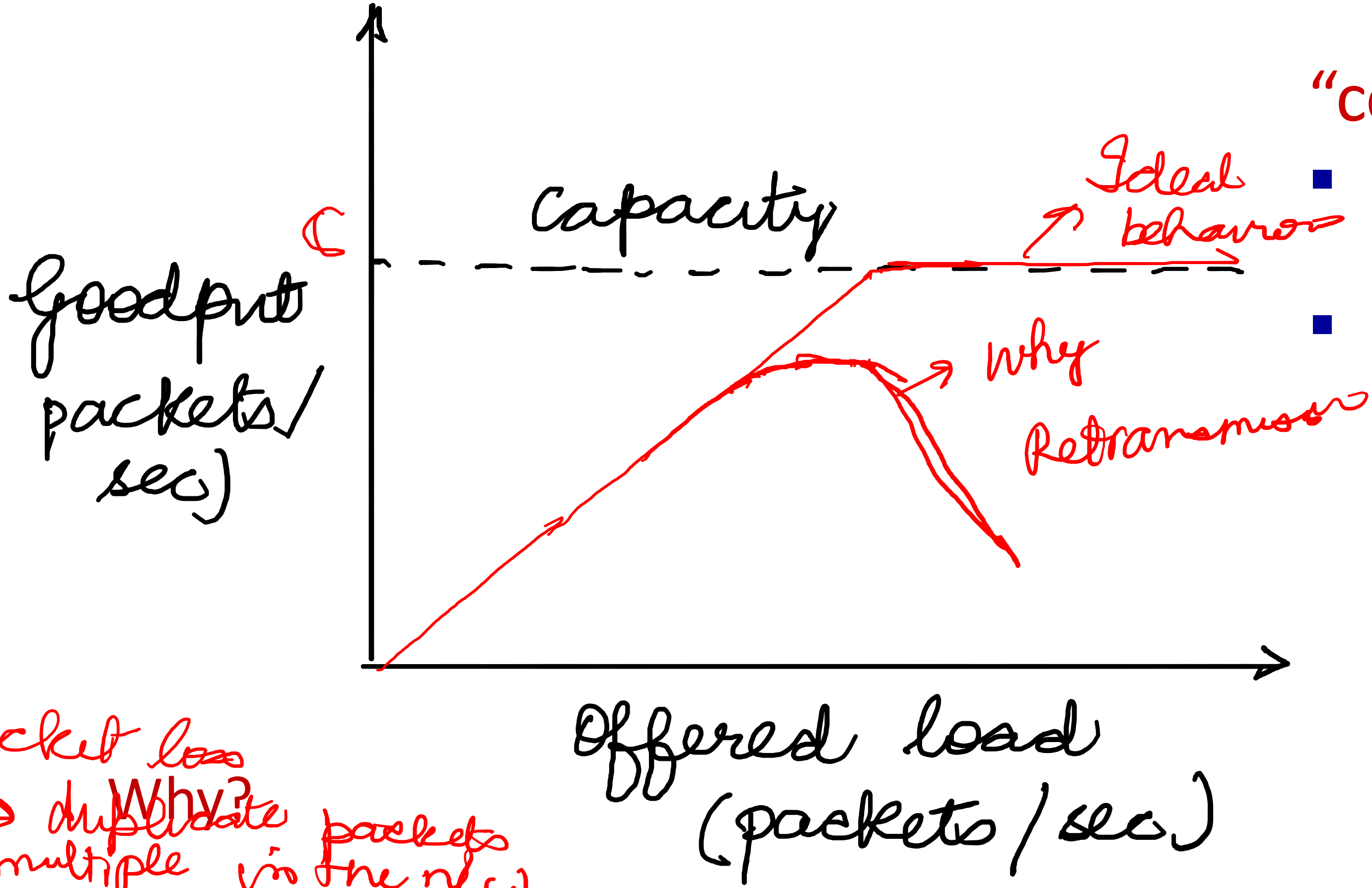
congestion control: too many senders, sending too fast

flow control: one sender too fast for one receiver

The Case of Congestion Collapse

could send at $\boxed{\text{Rwnd}}$

- Early TCP protocol in 1980s used fixed size window
  - The focus was mainly on providing reliability
- As network load grew, goodput reduced



Why?

Packet loss → duplicate multiple packets in the n/w

"costs" of congestion:

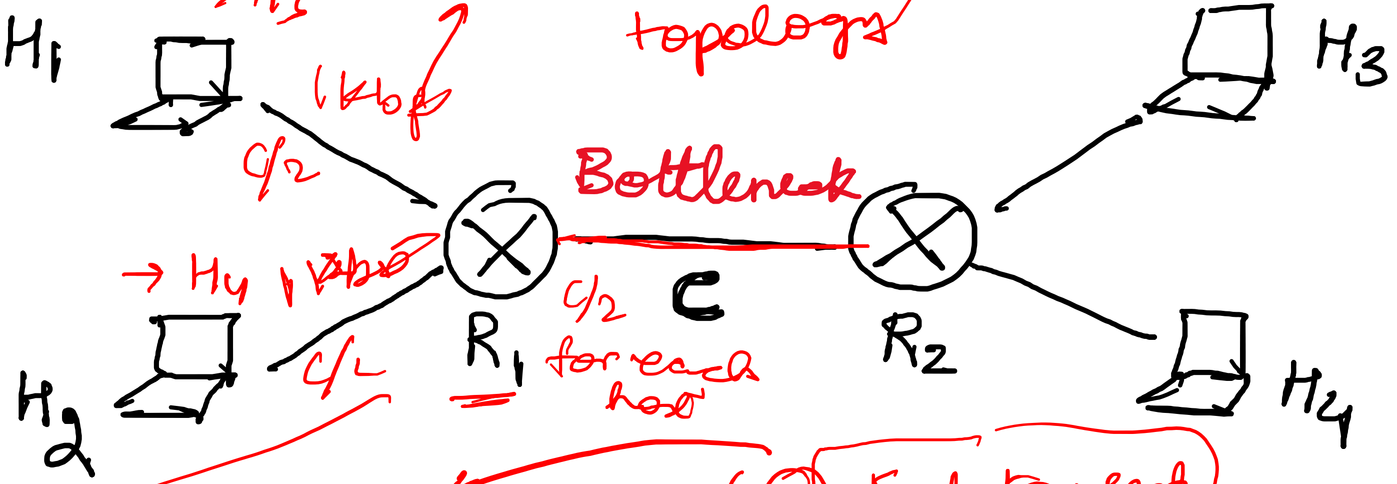
- more work (retransmission) for given receiver throughput
- unneeded retransmissions: link carries multiple copies of a packet
  - decreasing maximum achievable goodput

Need mechanisms for congestion control

How Would You Design a Congestion Control Algorithm?

Dumb-bell topology

control sending rate



→ H3 1 Kbps C/2

→ H4 1 Kbps C/L

C/2 for each host

control sending rate

① Routers implement or assist in CC / Network assisted

② End-to-end manner

② Centralized controller vs Distributed

harm : CC-B does not harm existing CC-A any more than A already harms itself

Distributed, end-to-end manner

metrics

④ Minimize latency

① Decrease packet loss

② Maximize link utilization → tput

③ Fairness : Equal share of b/w

Jain's fairness index

$$\frac{(\sum x_i)^2}{n \sum x_i^2}$$

1 → fair

1/n → highly unfair

Max-min fairness

① Each host get no more than its need

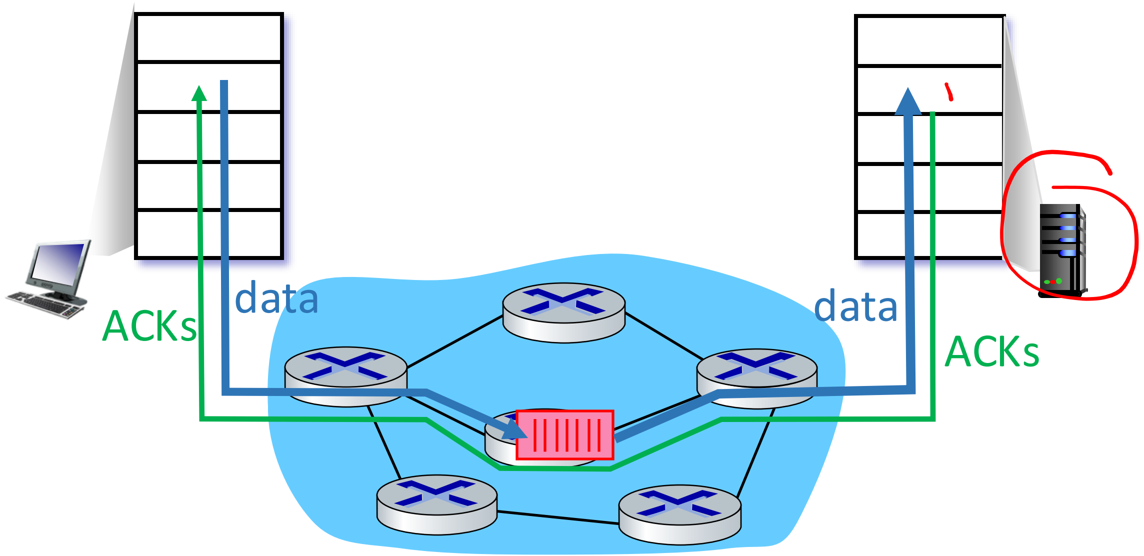
② Maximize the minimum allocation

③ This is true even if you remove that min & adjust the resource

Approaches towards congestion control

End-end congestion control:

- no explicit feedback from network
- congestion *inferred*
- **approach taken by TCP**



Network-assisted congestion control:

- routers provide *direct* feedback to sending/receiving hosts with flows passing through congested router
- may indicate congestion level or explicitly set sending rate
- E.g., TCP ECN, ATM, DECbit protocols

Metrics for evaluation of a Congestion Control Algorithm?

- Maximize the throughput
- Minimize the end-to-end delay
- Stability across a range of operating scenarios
- Flows should receive fair-share of bandwidth

common requirement

End-to-end congestion control

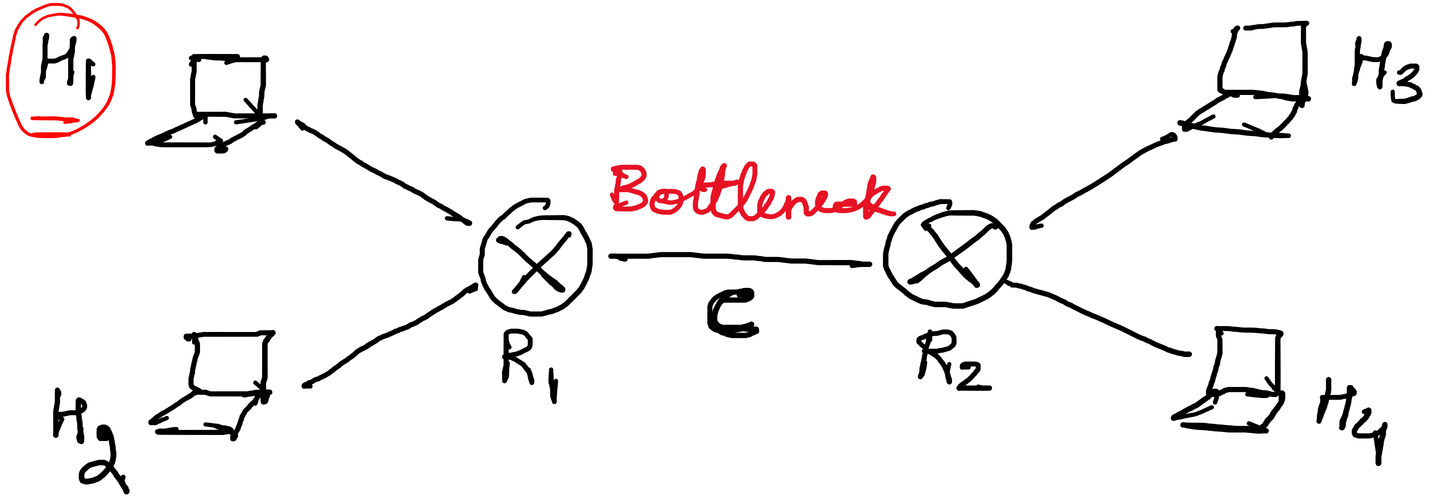
→ rate-based: send at some average rate R.

→ window-based min (rwnd, cwnd)

↓

used by TCP

$\boxed{\text{cwnd}}$



Fundamental questions

cwnd

- How to determine the available ~~capacity~~ bandwidth for a flow at any time? ↑ cwnd
- How to detect congestion? packet loss ✓ increase in RTT values
- How to react to congestion once detected? ↓ cwnd

TCP congestion control:

- *approach:* senders can increase sending rate until packet loss (congestion) occurs, then decrease sending rate on loss event
- Many functions possible
- *Class of linear* control algorithms:
  - $w_{i+1} = aw_i + b$
  - a and b are different in case of increase and decrease
- Additive increase, multiplicative decrease (AIMD)
  - $w_{i+1} = w_i + b$
  - $w_{i+1} = aw_i$

TCP congestion control: AIMD

- Additive increase, multiplicative decrease (AIMD)
  - *Increase: $w_{i+1} = w_i + b$, Decrease: $w_{i+1} = aw_i$*

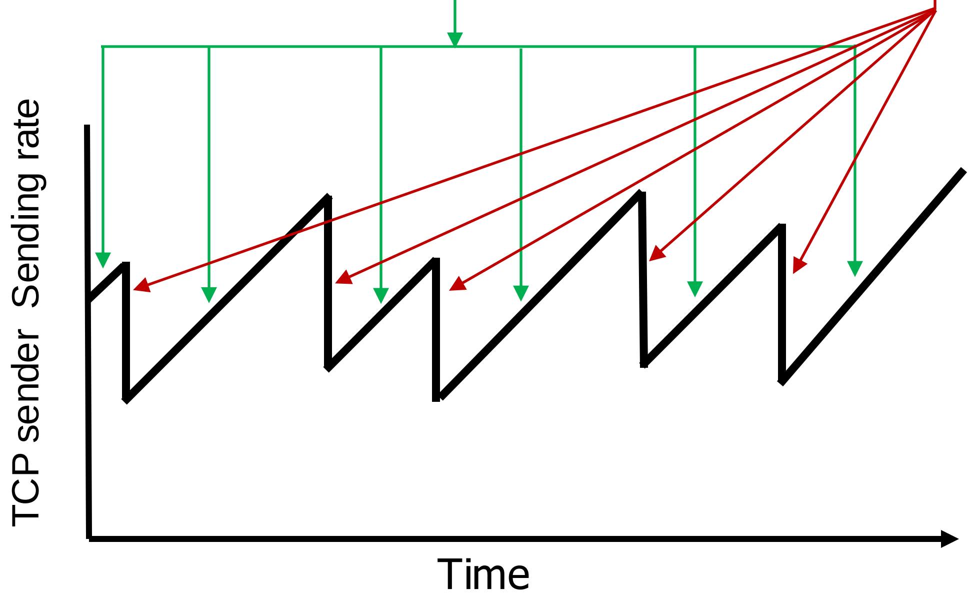
Additive Increase

increase sending rate by 1 maximum segment size every RTT until loss detected

Multiplicative Decrease

cut sending rate in half at each loss event

AIMD sawtooth behavior: *probing* for bandwidth



Why was AIMD chosen?

TCP fairness

Fairness goal: if *K* TCP sessions share same bottleneck link of bandwidth *R*, each should have average rate of $\boxed{R/K}$