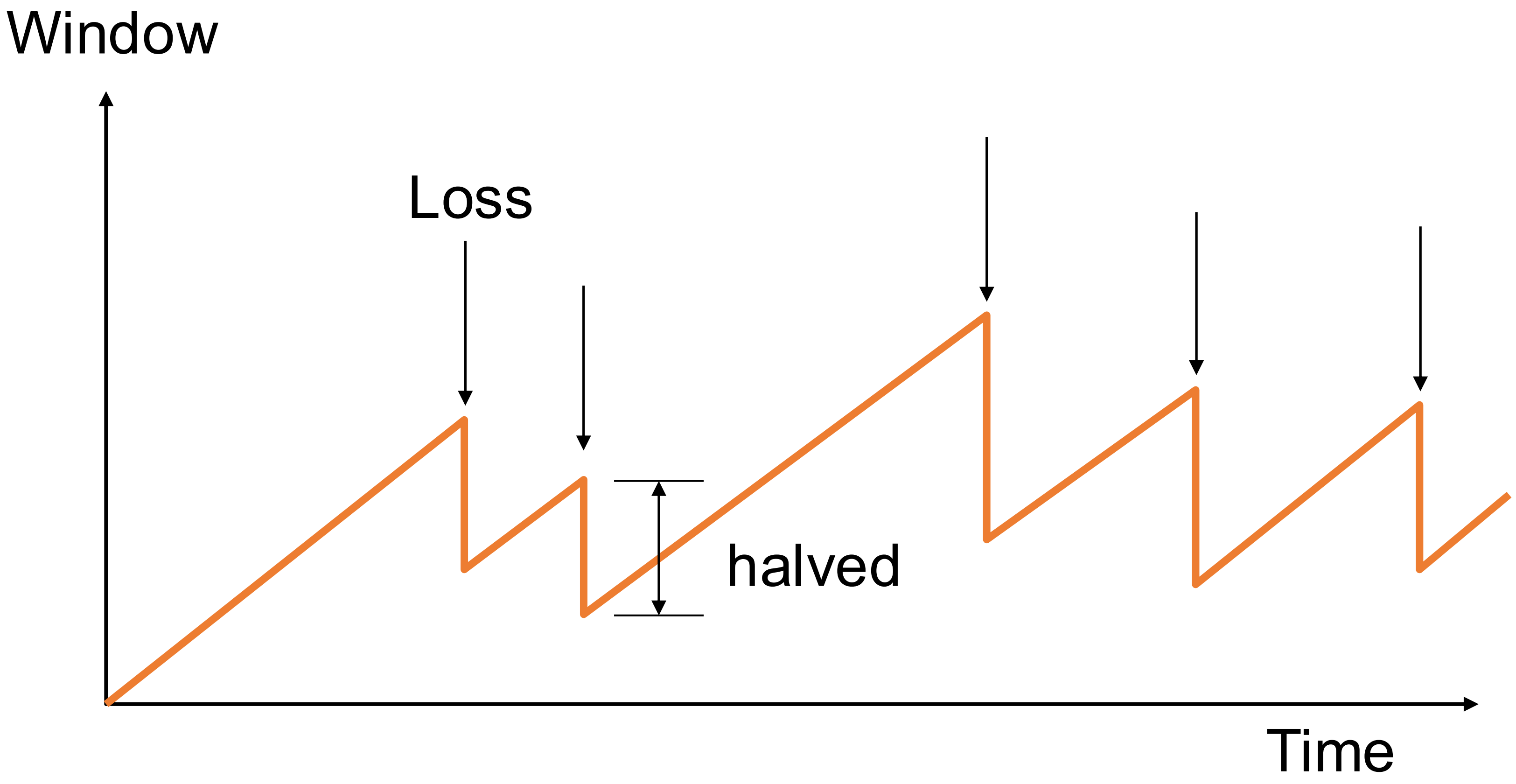
BBR Congestion Control

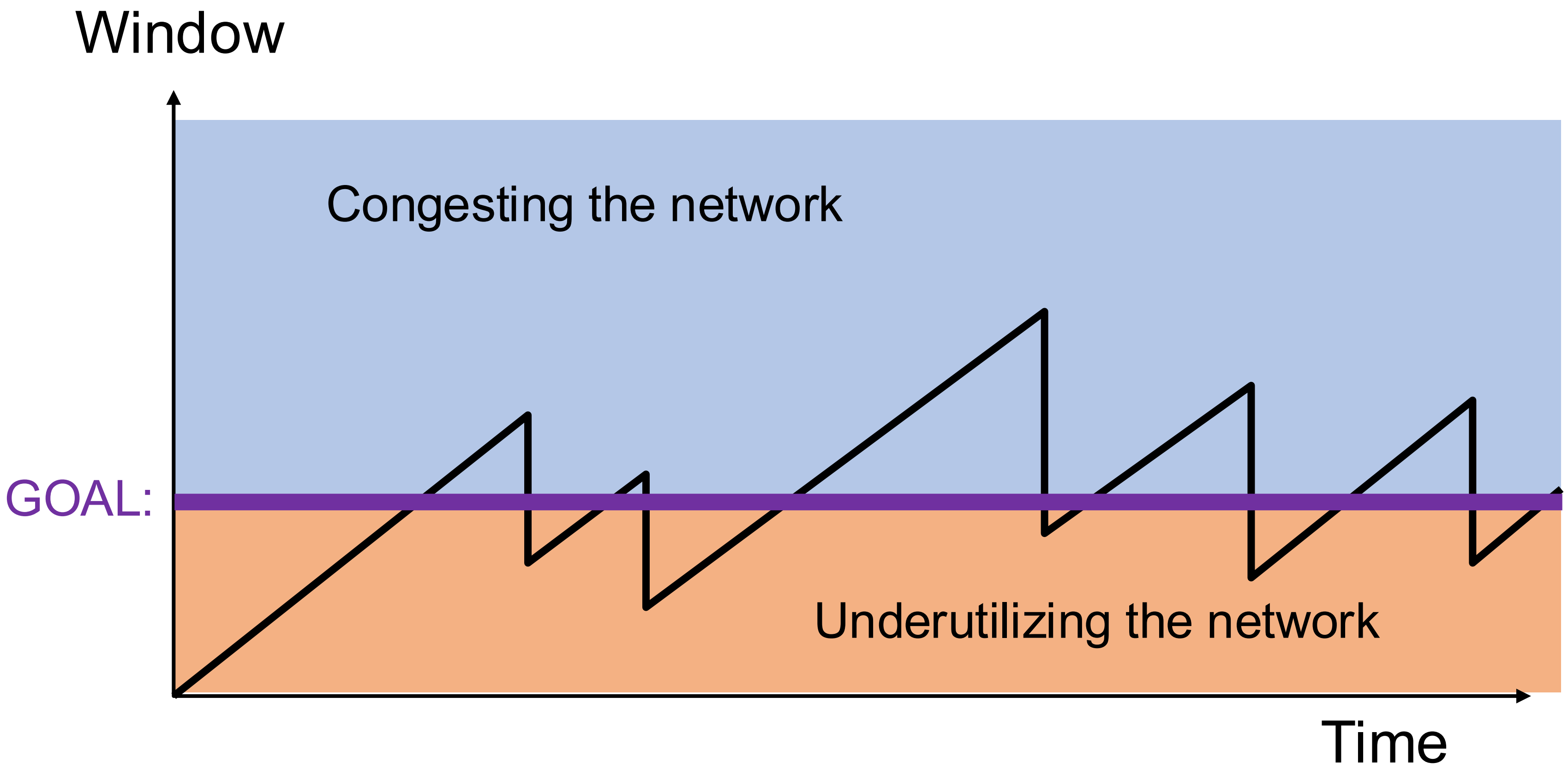
COS 316: Principles of Computer System Design
Lecture 8

Wyatt Lloyd

TCP "Sawtooth"



TCP Sawtooth Misses the Mark



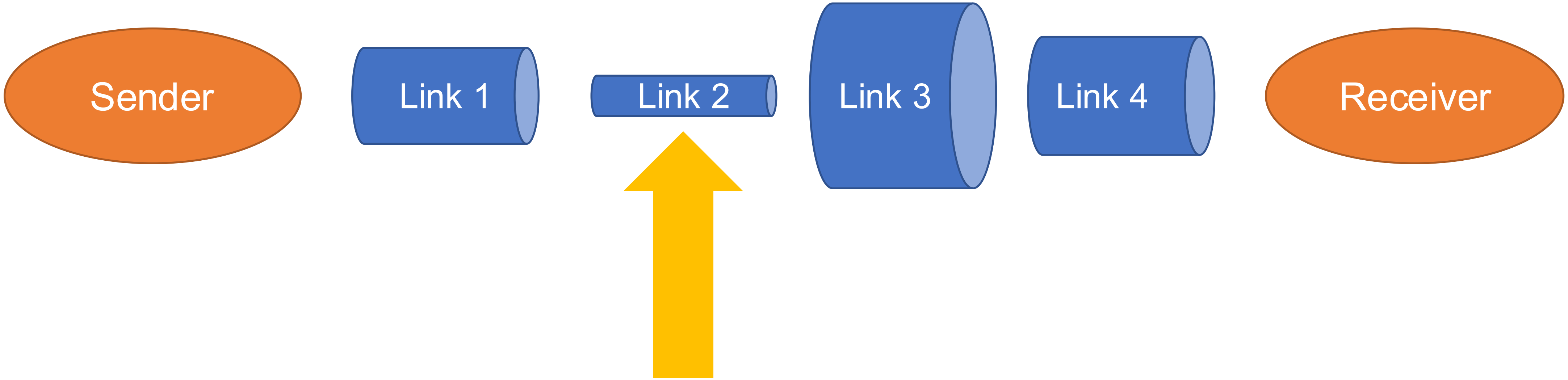
Can We Do Better?

- Yes! Researchers in academia and industry actively working on it for 35+ years and still going!
- 100s of congestion control schemes proposed…
- A couple of papers at every SIGCOMM…

Today: BBR Congestion Control

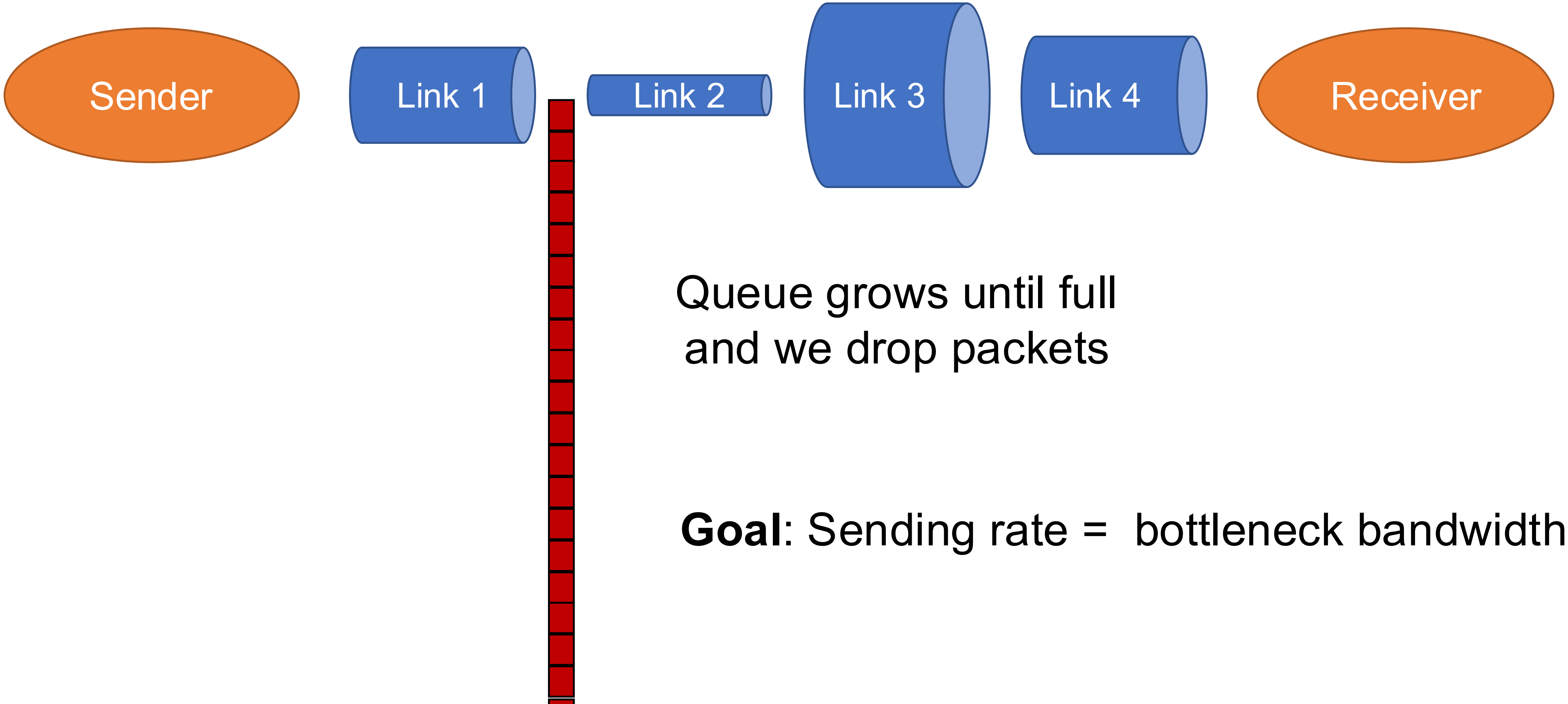
- BBR: <u>b</u>ottleneck <u>b</u>andwidth and <u>r</u>ound-trip propagation time

Bottleneck Bandwidth



Bottleneck link

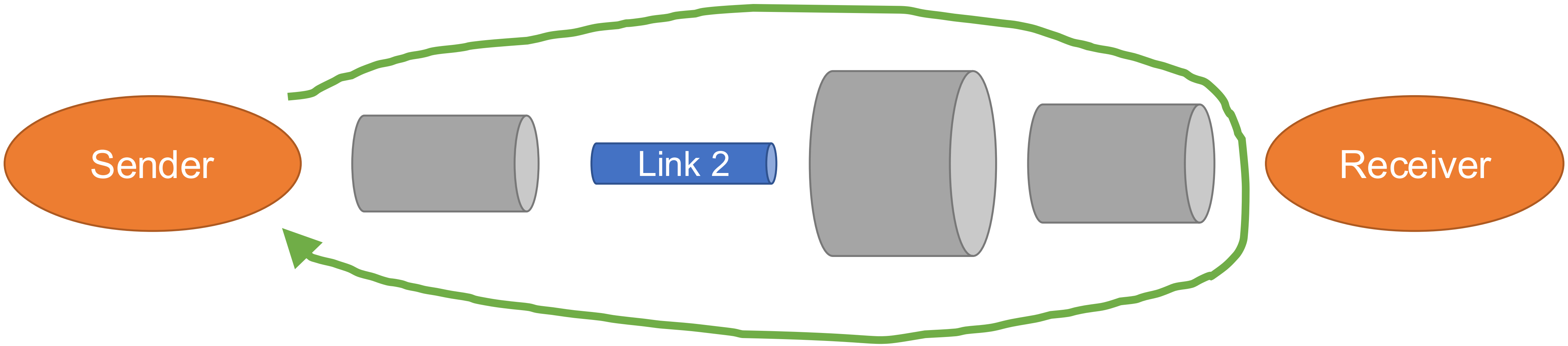
Send at > Bottleneck Bandwidth



Queue grows until full
and we drop packets

Goal: Sending rate = bottleneck bandwidth

Bandwidth Delay Product (BDP)



Round trip propagation delay (RTProp)

Bandwidth Delay Product = RTProp * Bottleneck Bandwidth

Data in Flight vs. Bandwidth Delay Product

- Data in flight = un-acknowledged data

- If data in flight > bandwidth delay product?
  - Queue before bottleneck grows

- If data in flight < bandwidth delay product?
  - Can’t fill bottleneck at all time => underutilization

- **Goal**: Data in flight = BDP = RTProp * bottleneck bandwidth

BBR’s Two Goals

- Sending rate = bottleneck bandwidth
- Data in flight = BDP = RTProp * bottleneck bandwidth

- High-level technique:
  - Estimate bottleneck bandwidth
  - Estimate RTProp
  - Pace sending to bottleneck bandwidth
  - Run experiments to test if bottleneck bandwidth or RTProp change
    - Still constrain overall data in flight to be BDP

Estimating Bottleneck Bandwidth

- Take a measurement between every send and ack:
  - `bandwidth_estimate = Δdelivered / Δt`
- Can never send faster than bottleneck bandwidth
- Bottleneck bandwidth = max estimate in last *N* seconds
  - (*N* = 10)

Estimating Round Trip Propagation Delay

- Take a measurement between every send and ack:
  - `RTprop_estimate = time_at_ack – time_at_send`

- Can never receive ack faster than Rtprop

- RTprop = min estimate in last *N* seconds
  - (*N* = 10)

Pacing Sending

- Goal: send at bottleneck bandwidth rate
- Send a packet every `packet_size / bottleneck bandwidth`
  - `e.g., 1500B/40Mbps = 1500B/5MBps = 1 packet / 300µs`

```
if (now >= nextSendTime)
     …
     nextSendTime = now + packet.size / BtlBw_estimate
```

Run Experiments

- Is there more bandwidth available?
  - Try sending extra data
    - Same time to ack => no queue => extra bandwidth available!
    - Longer to ack => queue grew => no extra bandwidth available
  - Compensate by sending less data to keep inflight data < BDP
    - Experiment increases queue, compensation drains them

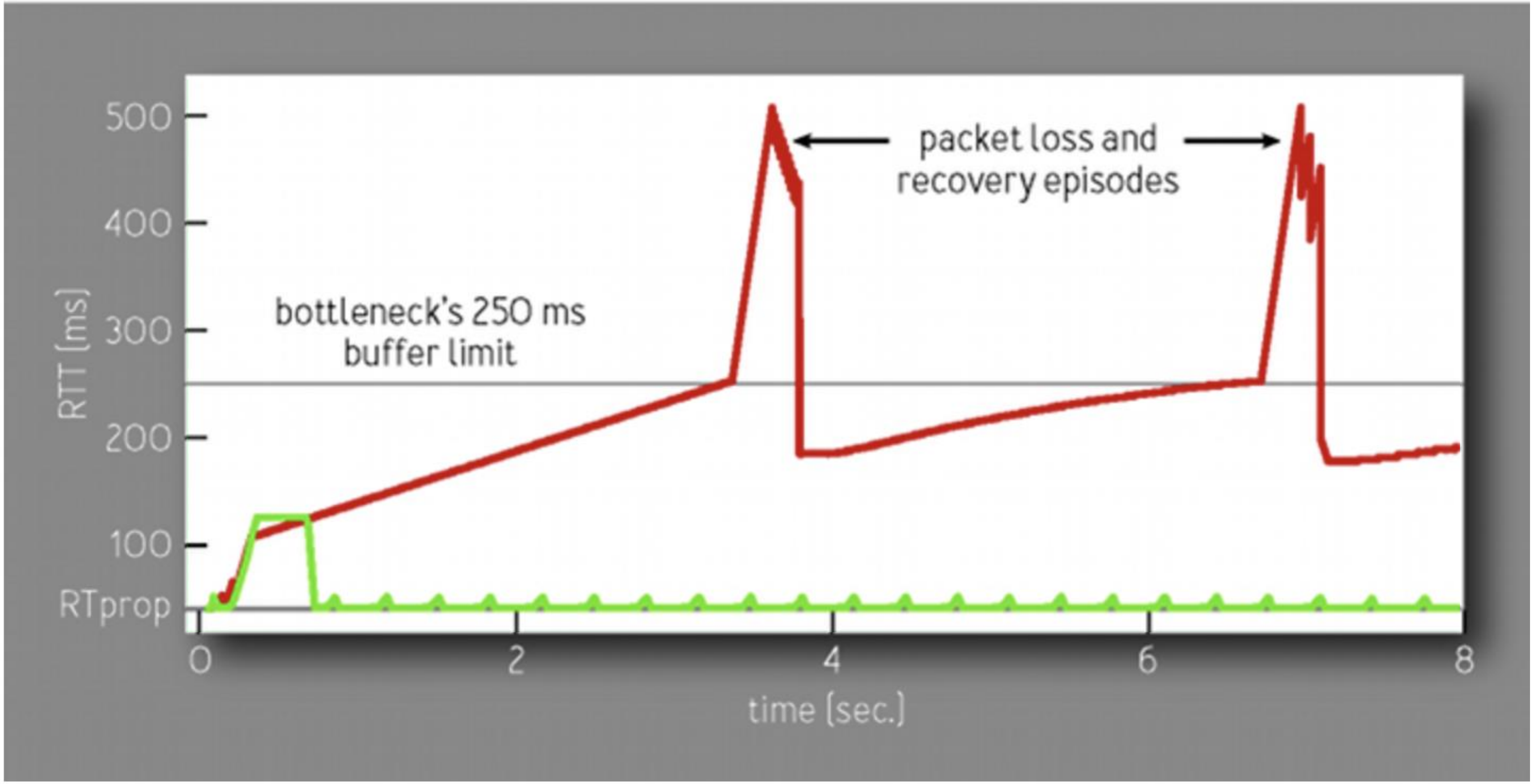
- Is RTprop shorter?
  - Try sending very little data to avoid queuing

BBR High Level Review

- Estimate bottleneck bandwidth with max estimate
- Estimate RTProp with min estimate
- Pace sending to bottleneck bandwidth rate
- Run experiments to test if bottleneck bandwidth or RTProp change
  - Still constrain overall data in flight to be BDP

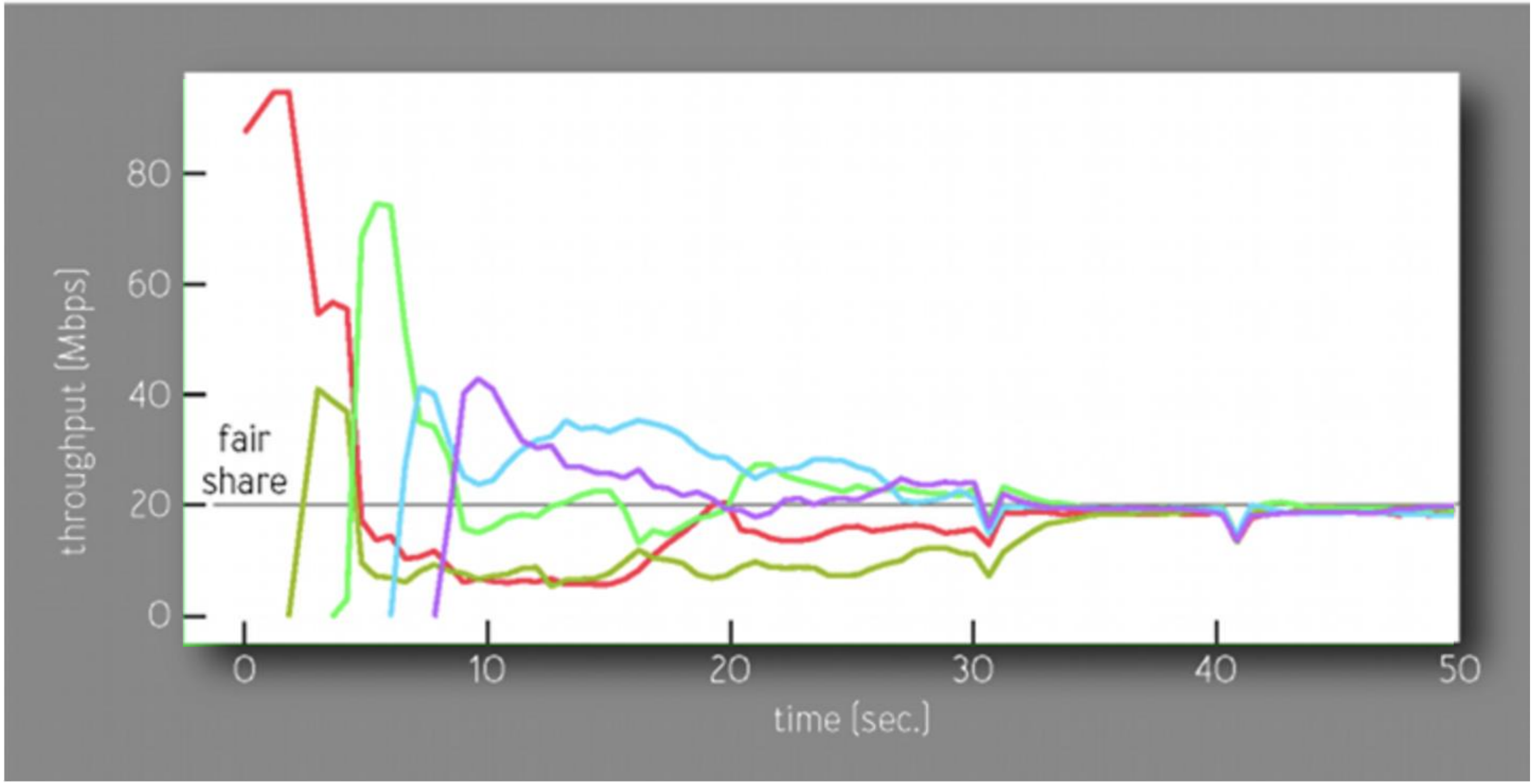
BBR's Latency? [fig 5 from queue paper]

FIGURE 5: FIRST 8 SECONDS OF 10-MBPS, 40-MS CUBIC AND BBR FLOWS



BBR's Throughput? [fig 5 from queue paper]

FIGURE 6: THROUGHPUTS OF 5 BBR FLOWS SHARING A BOTTLENECK



BBR in Practice

- In Linux since 2016
  - `sysctl net.ipv4.tcp_congestion_control=bbr`

- BBR is used for Google’s internal traffic
  - Inside a datacenter
  - Between Google datacenters
- BBR is used for Google’s external traffic
  - Google.com, YouTube
- BBR has *some* adoption outside Google
  - 8% of most popular 20K websites [Mishra et al. SIGCOMM ‘24]
  - e.g., Amazon.com, primevideo

BBR Conclusions

- Congestion is inevitable
  - Internet does not reserve resources in advance
  - BBR in TCP estimates the most traffic it can send without increasing congestion
    - Runs experiments to push the envelope
- Congestion can be handled
  - BBR sender limits traffic to the bandwidth delay product (congestion window)
- Running in practice!