Congestion Control

Diya Joy and Peter Gunarso

Outline

- Congestion Control
  - Discussion: TCP
- QUIC: Successor to TCP?
  - Discussion: QUIC

Key Terms

- **TCP**: Transmission Control Protocol
- **Congestion**: Sudden drop in network performance
  - Happens when input rate > output rate
- **Congestion Control**: Techniques to manage congestion in a network
- **Equilibrium**: When input rate = output rate and windows are full of data
  - No congestion!

Equilibrium Challenges

1. Connection doesn’t get to equilibrium
2. Sender breaks equilibrium by sending too many packets
3. Equilibrium can’t be reached because of resource limits along path

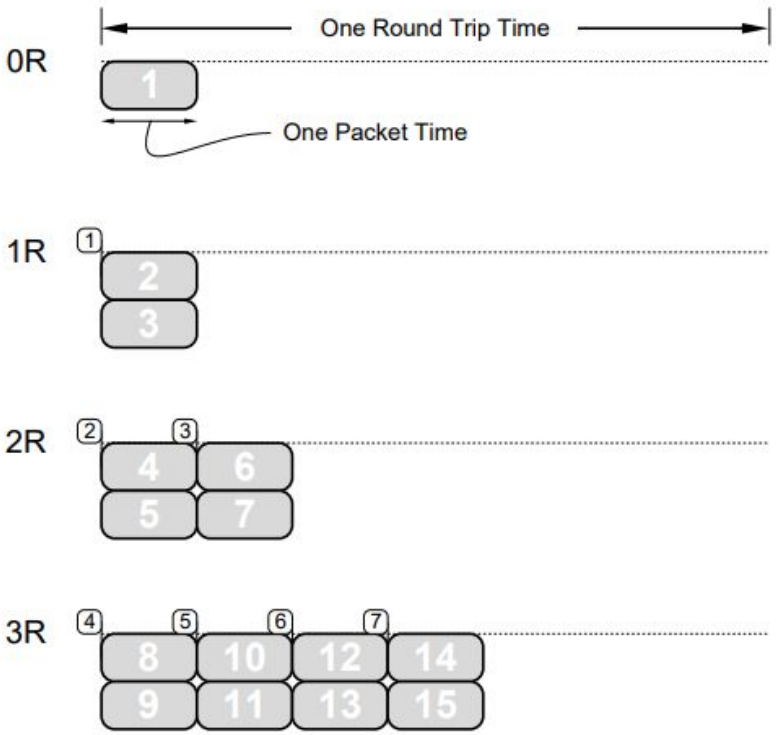
1. Getting to Equilibrium

- A simple TCP connection is “self-clocking”
  - Client sends new packets when it gets acks
  - Sender sends new acks when it gets packets
- Bootstrapping problem - how can we get to equilibrium

Slow Start

- Add a congestion window (cwnd)
- Start with a cwnd size of 1
- On each ack, increase cwnd by 1

Figure 2: The Chronology of a Slow-start



Slow Start

Figure 3: Startup behavior of TCP without Slow-start

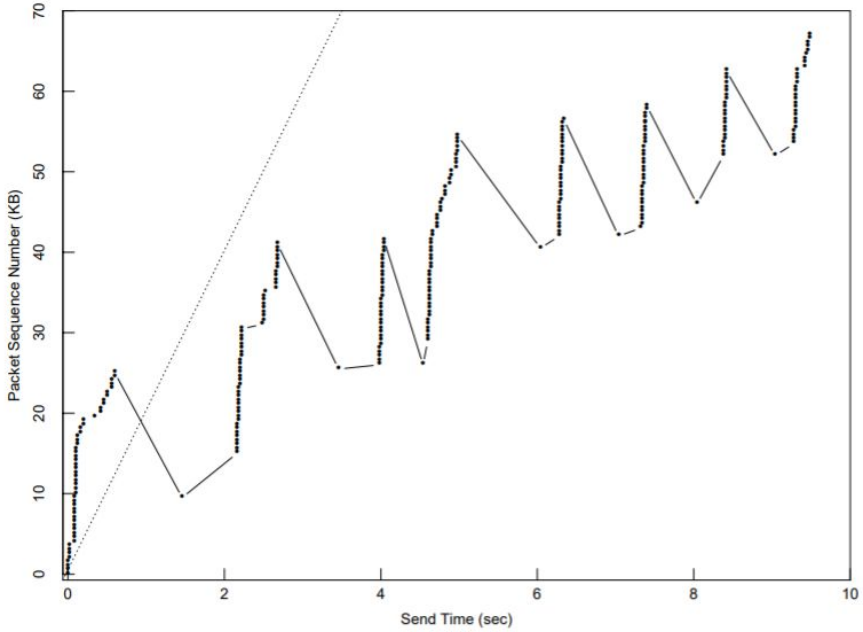
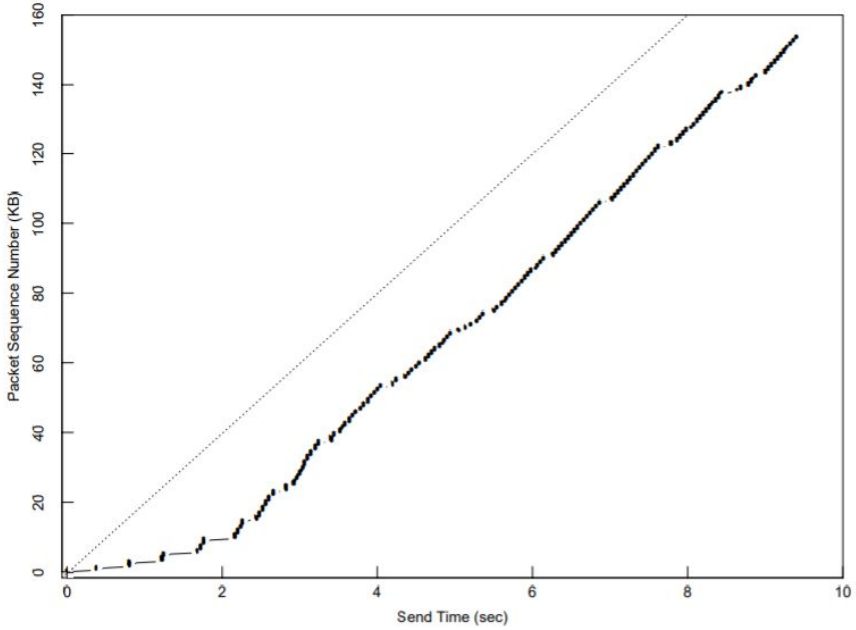


Figure 4: Startup behavior of TCP with Slow-start



2. Conservation at equilibrium

- Problem: packets that were delayed but will eventually arrive will be retransmitted, making the congestion worse

- TCP Solution: $R \leftarrow \alpha R + (1-\alpha)M$
  - Retransmit timer interval = βR
- Proposed solution: Estimate variance instead of using a fixed β value
  - Retransmit timer interval = mean + variance

RTT Estimation

Figure 5: Performance of an RFC793 retransmit timer

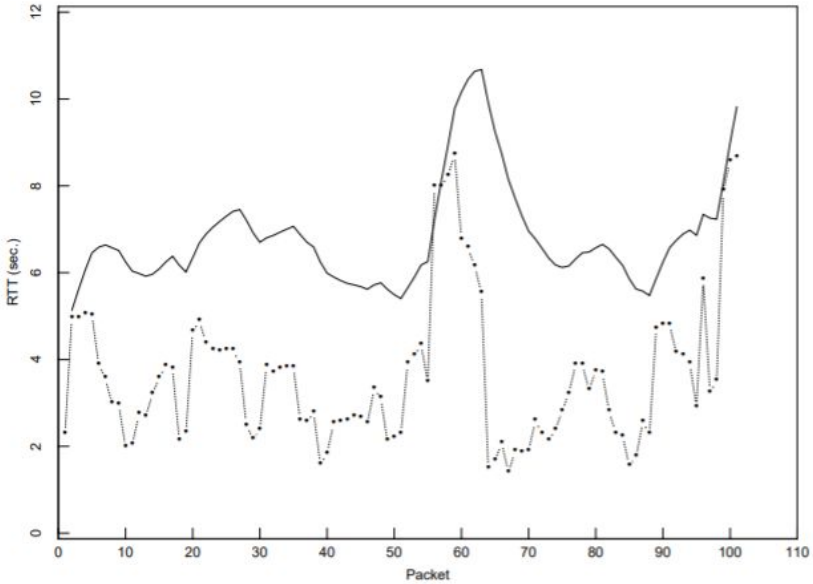
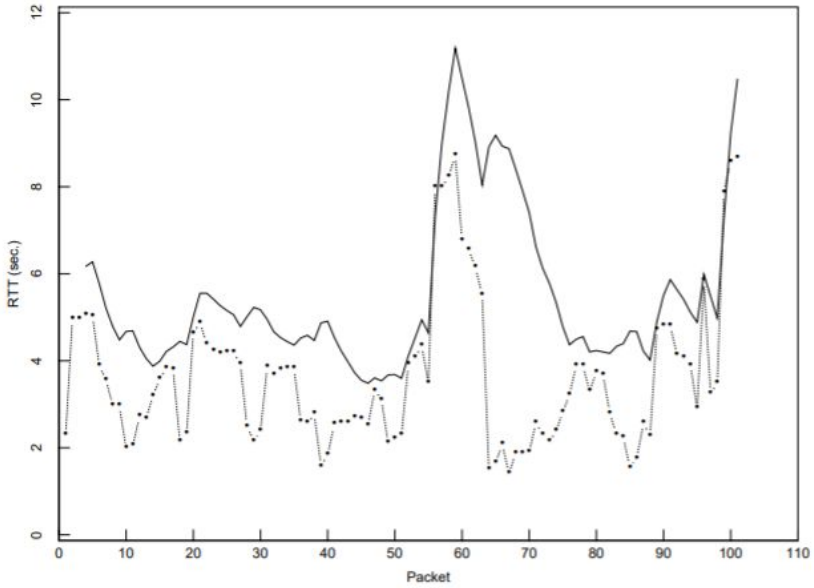


Figure 6: Performance of a Mean+Variance retransmit timer

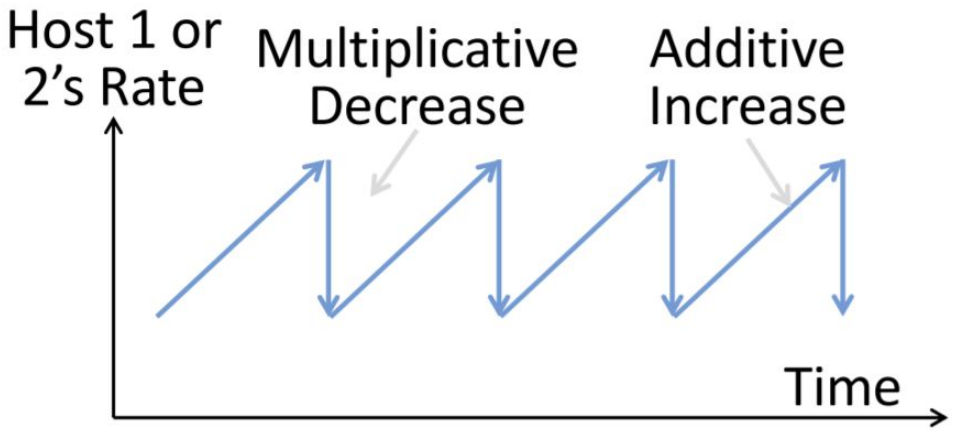


3. Adapting to the path

- Timeouts probably come from congestion in the network
  - Packet loss becomes a good signal for congestion
  - No packet loss is a signal for no congestion
- Can we avoid congestion by reacting to a lost packet?

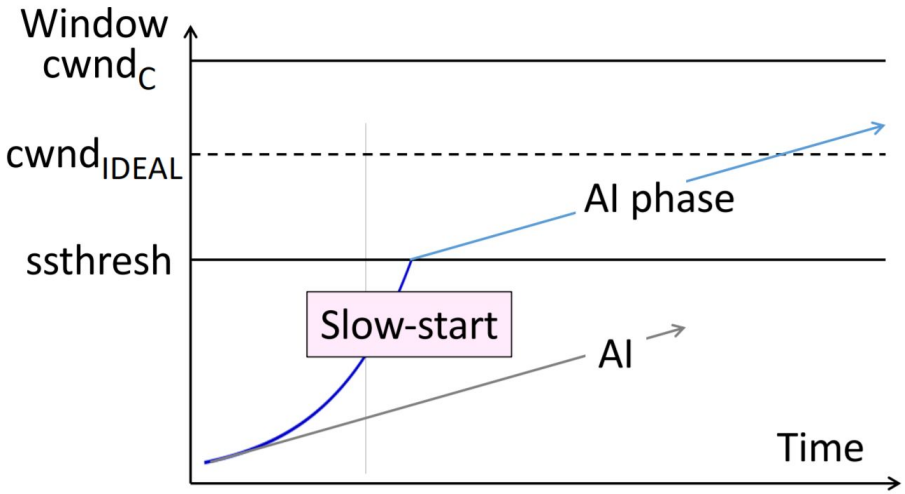
AIMD

- Additive increase, multiplicative decrease
  - Increase incrementally, decrease exponentially

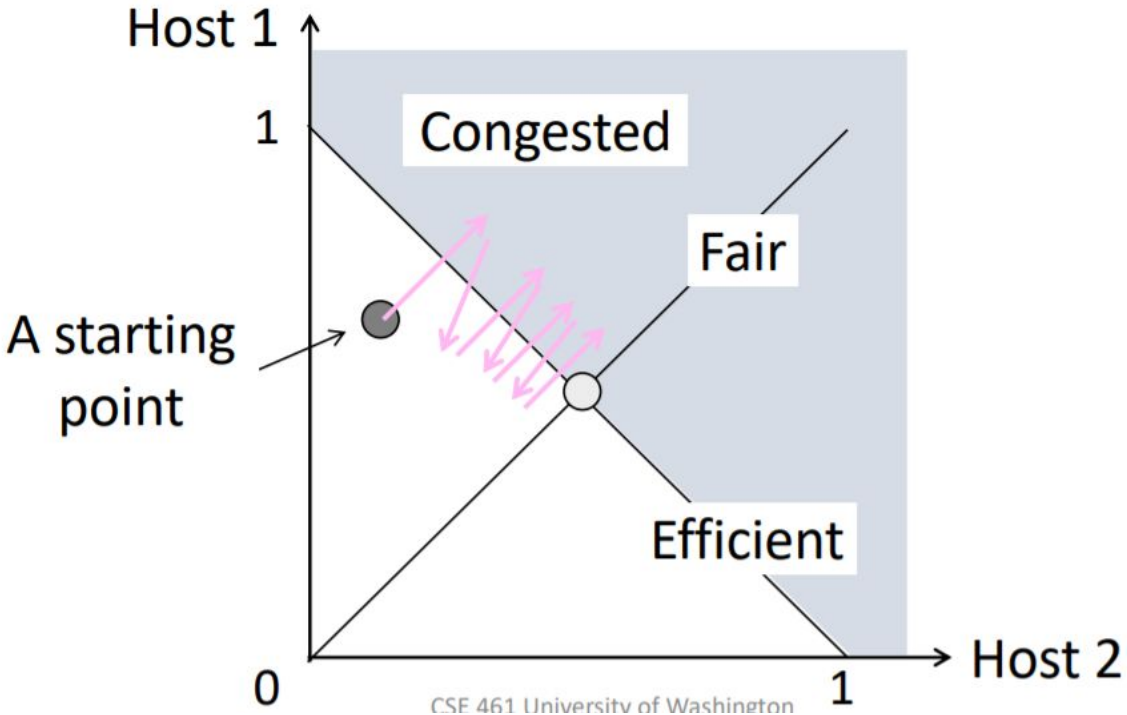


AIMD

- Should be implemented in tandem with slow start



AIMD - Fairness



Summary

- Congestion control is all in implementation details
- We aim for equilibrium (packets in = packets out)
- Challenges and Solutions
  - Getting to Equilibrium → Slow Start
  - Maintaining Equilibrium → RTT Estimation
  - Adapting to Network Conditions → AIMD

https://tinyurl.com/cse550-congcon

Discussion Question:

TCP is the standard in internet communication. What are some areas of improvement? Which situations would these improvements be useful in?

QUIC

The TCP Killer?

QUIC

- Protocol built on top of UDP
- Made to be equivalent to TCP with lower latency
- Supported by Chrome, Chrome derivatives, and Firefox

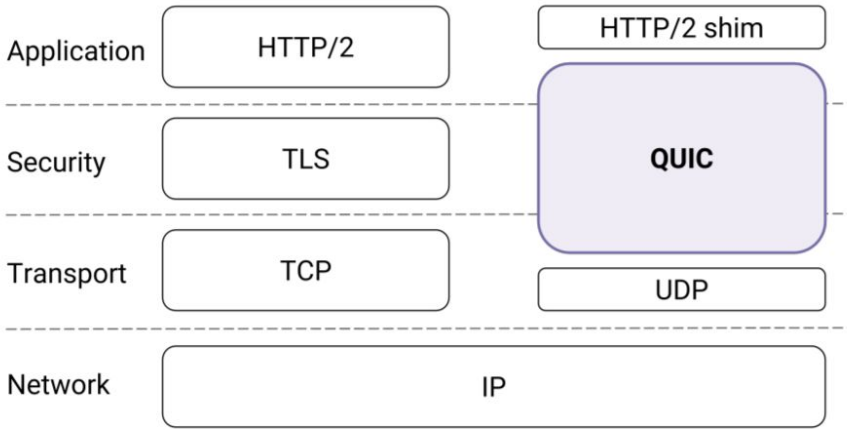
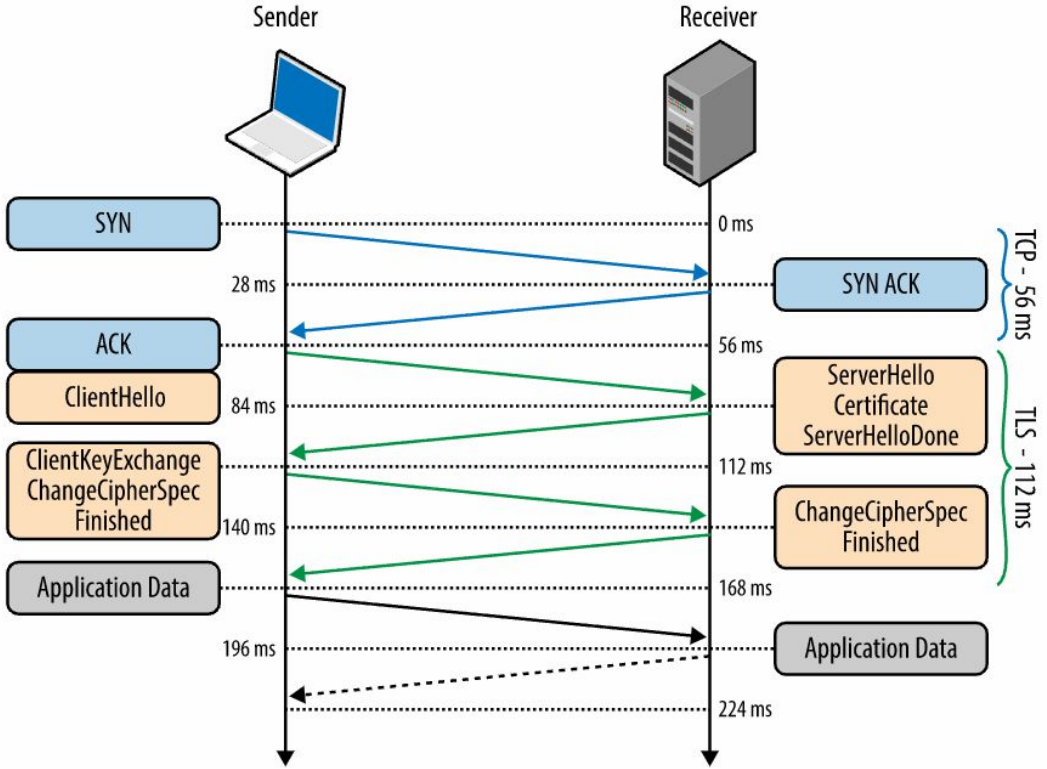


Figure 1: QUIC in the traditional HTTPS stack.

Where can TCP be beaten?

- Connection Handshakes
- Head-of-line delay
- Retransmission Ambiguity

TCP/TLS Handshakes



QUIC Handshakes

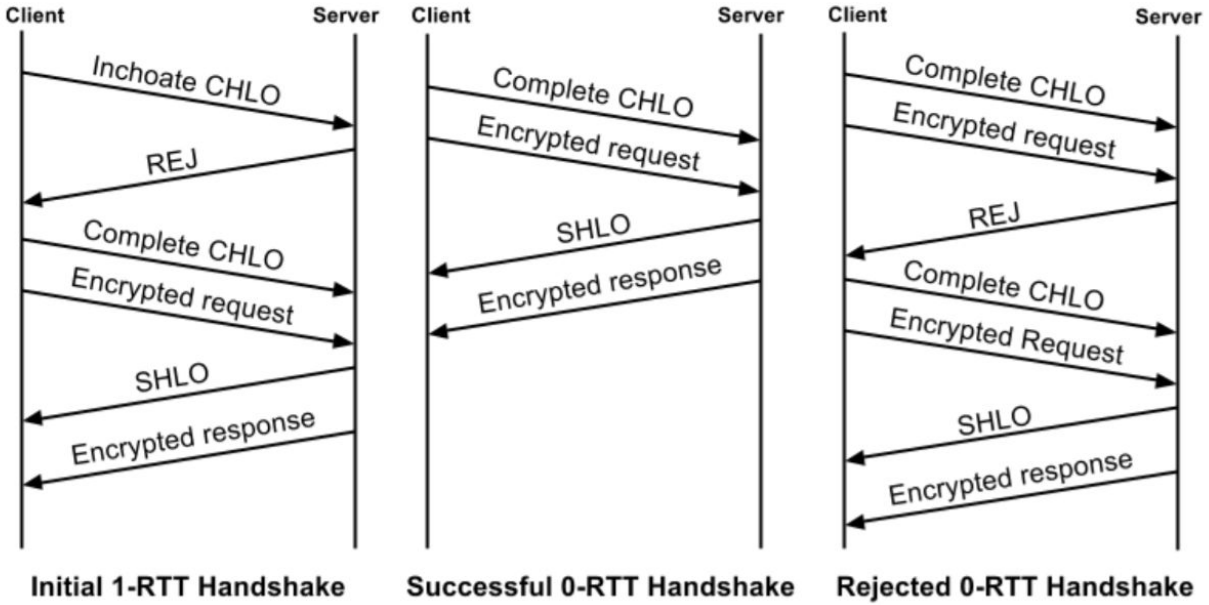


Figure 4: Timeline of QUIC's initial 1-RTT handshake, a subsequent successful 0-RTT handshake, and a failed 0-RTT handshake.

QUIC Handshake Latency

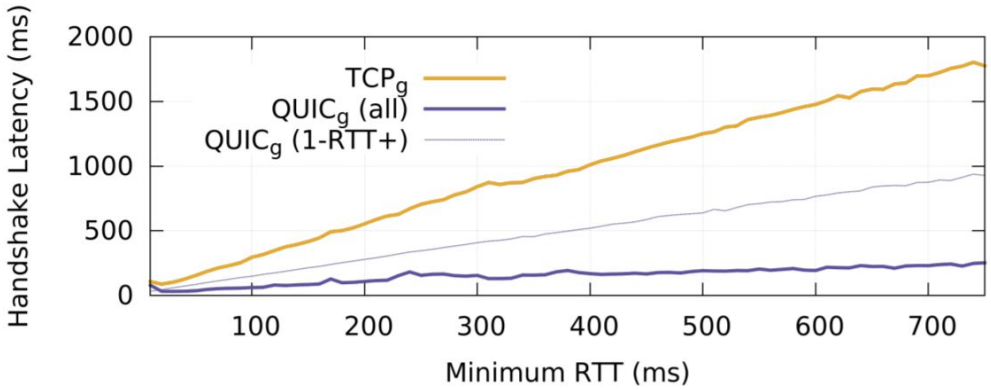


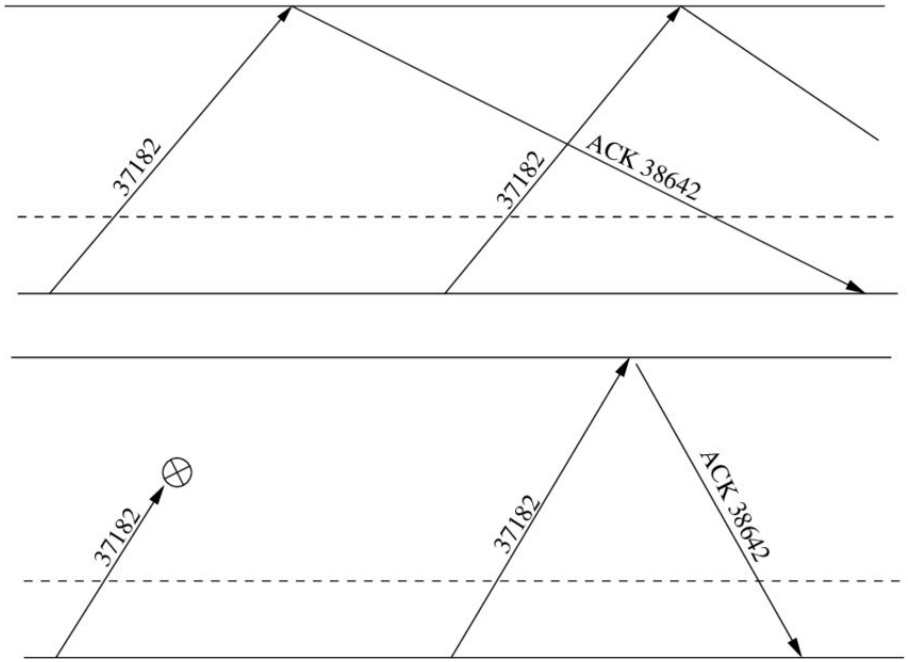
Figure 7: Comparison of handshake latency for $QUIC_g$ and TCP_g versus the minimum RTT of the connection. Solid lines indicate the mean handshake latency for all connections, including 0-RTT connections. The dashed line shows the handshake latency for only those $QUIC_g$ connections that did not achieve a 0-RTT handshake. Data shown is for Desktop connections, mobile connections look similar.

Stream Multiplexing

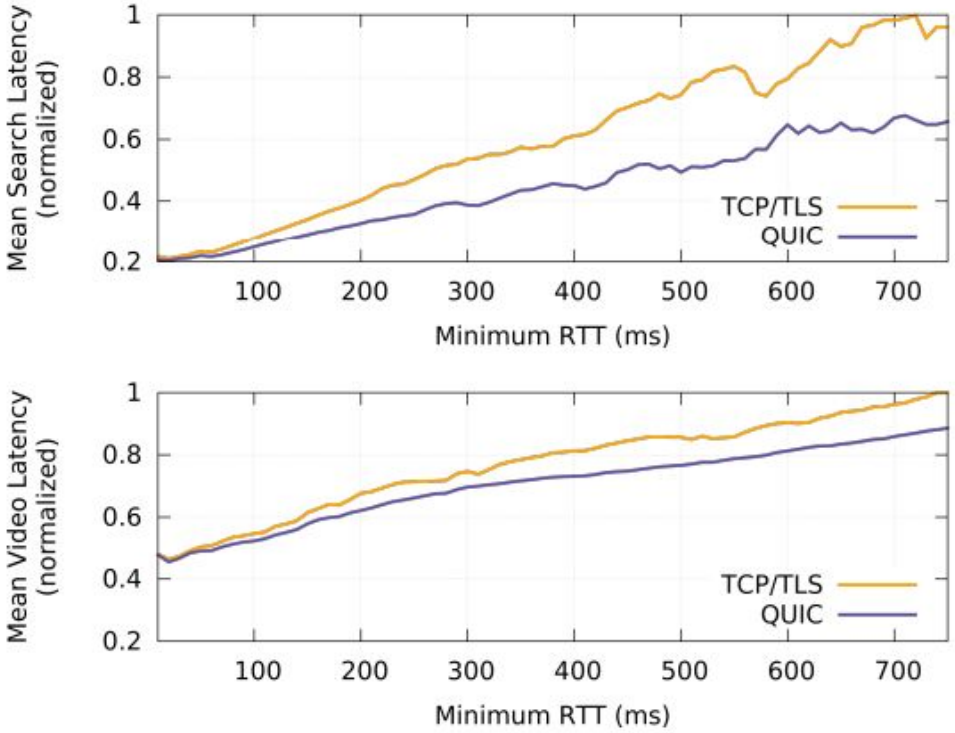
- Stream: Reliable bidirectional bytestream
  - TCP uses one stream per connection
- QUIC supports multiple streams per connection
  - Avoids head-of-line blocking in other streams when a packet is lost

Loss Recovery

- Retransmission Ambiguity
  - An issue in RTT estimation
  - Stems from TCP use of seq num
    - Conflates data ordering and packet transmission
- Case 1:
  - ACK sent for original packet, long RTT
- Case 2:
  - ACK sent for retransmission, short RTT



QUIC Performance



More QUIC Features

- Granular Flow Control
  - Stream Level
  - Connection Level
- Completed authenticated and mostly encrypted packets
- Bring-your-own congestion control
  - TCP congestion control methods still apply

https://tinyurl.com/cse550-congcon

Discussion Question:

Is QUIC *really* the TCP killer?

Things to consider:

- TCP is very well established
- Any downsides of QUIC?
    - Any downsides of building on UDP?