TCP congestion control: Issues
Issues with current CC algorithms

Cellular networks vs Long trunk links

Current congestion control algorithms cannot deal with either...
Cellular networks: problems

- Highly dynamic: variable throughput, latency, and loss rates
- TCP interprets all such signals as congestion
- Ideal solution: distinguish between congestion and non-congestion signals
Long, trunk links: problems

- High bandwidth-latency product, low loss rate
- TCP takes a long time to ramp up! Especially for RTT-based variants
- Ideal solution: open a large sending window at time of connection
TCP ex Machina: Remy
Remy: design

- Offline optimization program to generate CC algorithms
- \((\text{network conditions, goal}) \rightarrow \text{action}\)
  - Lookup table that maps conditions to actions
Congestion signals

- $r_{\text{ewma}} \rightarrow$ Moving average of ACK intervals
- $s_{\text{ewma}} \rightarrow$ Moving average of sender ACK intervals (info in packets)
- $\text{rtt}_{\text{ratio}} \rightarrow$ Ratio of most recent RTT to the currently smallest RTT value
RemyCC in action

\[
\text{Rule}(r_{\text{ewma}}, s_{\text{ewma}}, \text{rtt\_ratio}) \rightarrow <m, b, \tau>
\]

- \(m\) multiplicative increase
- \(b\) additive increase
- \(\tau\) interval between outgoing packets
RemyCC in action: on ACK

- Generate $<m, b, \tau>$ tuple using network conditions and rules
- Send next packet if
  - Window size $>$ packet size
  - Last packet sent more than $\tau$ ago
Remy: rule table generation

(see Remy slides)
Computation complexity

- A few wall-clock hours to generate one algorithm
- Might be too long!
PCC Vivace: online-learning driven congestion control

Presenter: Yuxin David Huang
Why learn?

- Existing TCP congestion control methods cannot adapt to changing network conditions
- Performance is far from optimal
- Current computer-generated congestion control schemes (Remy, BRR) all lack in some way