TCP Congestion Control -- Overview

Kameswari Chebrolu

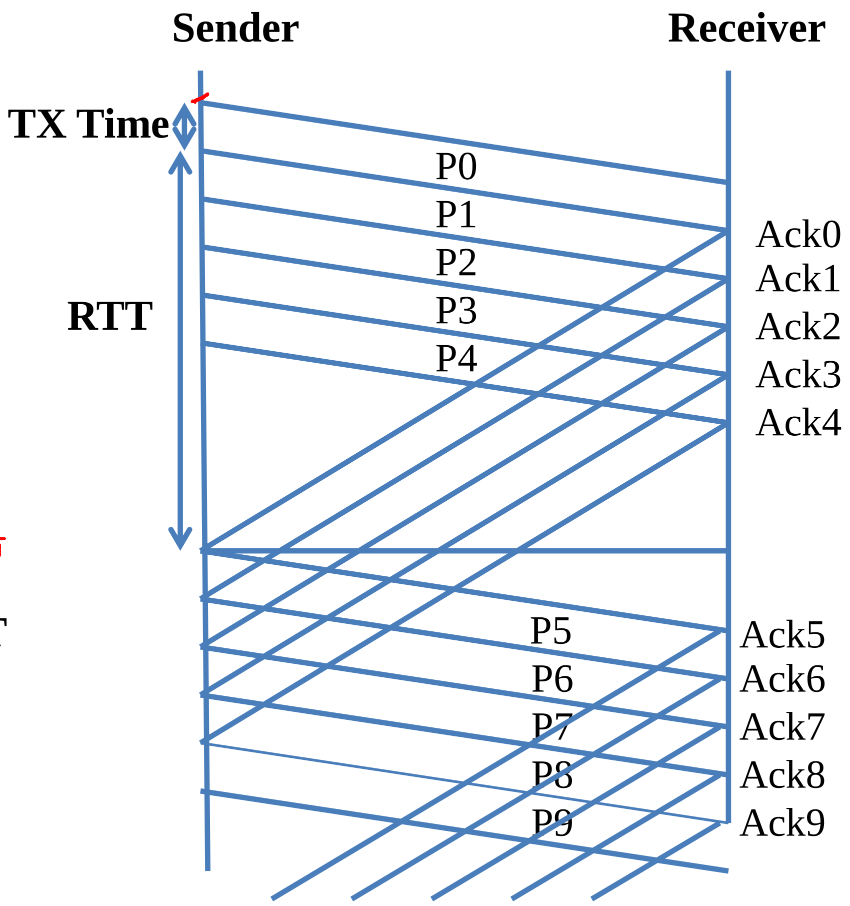
Seminal Paper: Congestion Avoidance and Control
by Van Jacobson and Michael J. Karels

Recap: TCP Services

- Multiplexing/Demultiplexing
- Reliable point-to-point data transfer
- Full-duplex
- Congestion control
- Flow control

Sliding window Protocol

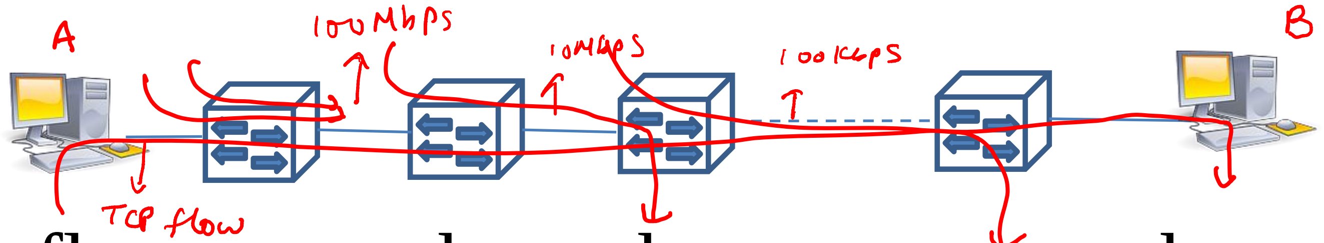
Recap: Sliding Window



wind. size pkts

Throughput ~= ($\boxed{W}$ * MSS)/RTT

Congestion Control: Problem Statement



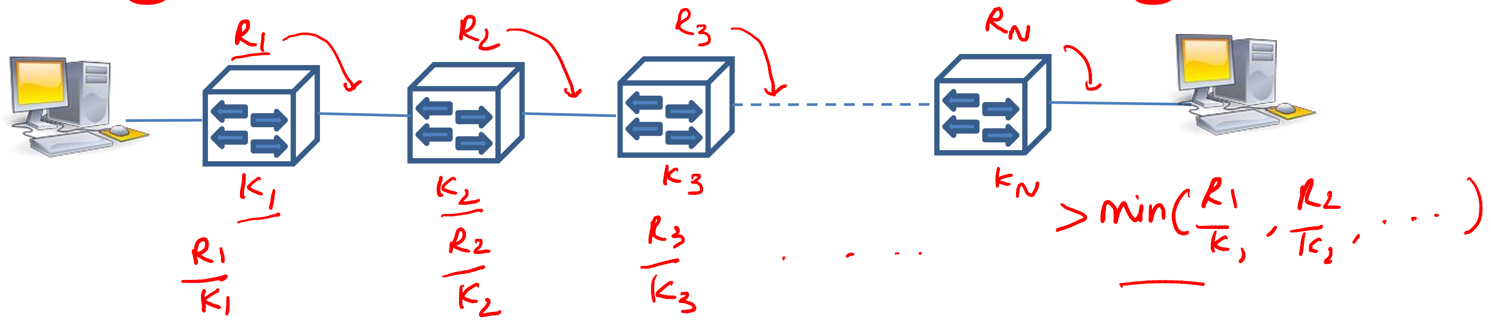
- Many flows pass through a router; number varies with time

fairness

blast

- Flows can be TCP or UDP
- The link capacities of the routers are different
- End Result: Throughput achieved by a given flow function of many factors

Congestion Control: Challenge



- Need to estimate W (of sliding window) such that each flow gets its fair share
  - Estimate small → underutilization; Estimate large → Congestion
- W will vary over time
- Congestion Control: Preventing sources from sending too much data too fast and thereby 'congest' the network

Sliding Window Protocol

- Roughly, idea translates to the following:
- View network as a pipe
- Determine the capacity of the pipe (Bandwidth-delay product)
- Fill the pipe with data
- As you remove one packet form the pipe, add another
  - ACKs help clock out data (Self Clocking)

3 Steps

- Getting to Equilibrium
- Conservation at equilibrium
  - Don't put new packet unless old one is removed
- Adapting to Path Dynamics

Summary

- Congestion Control is a complex problem
- Need to implement it in the context of the sliding window protocol
  - Self clocking is a useful feature
  - Need to determine and adapt W (window size) such that you don’t underutilize bandwidth or congest the network
- Ahead: Actual details