CS244a: An Introduction to Computer Networks

Handout 7: Congestion Control

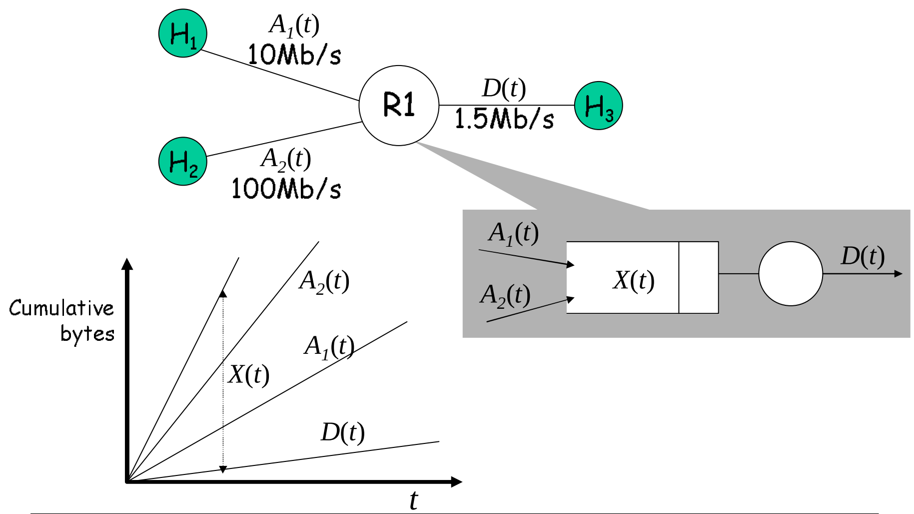
Nick McKeown
Professor of Electrical Engineering
and Computer Science, Stanford University

nickm@stanford.edu
http://www.stanford.edu/~nickm

Main points

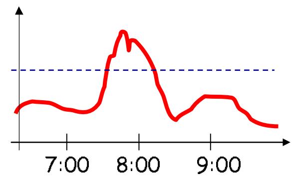
- Congestion is inevitable
- Congestion happens at different scales – from two individual packets colliding to too many users
- TCP Senders can detect congestion and reduce their sending rate by reducing the window size
- TCP modifies the rate according to “Additive Increase, Multiplicative Decrease (AIMD)”.
- To probe and find the initial rate, TCP uses a restart mechanism called “slow start”.
- Routers slow down TCP senders by buffering packets and thus increasing delay

Congestion

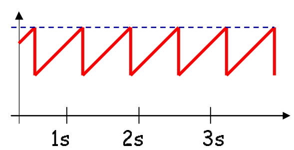


Time Scales of Congestion

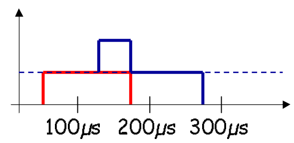
Too many users using a link during a peak hour



TCP flows filling up all available bandwidth

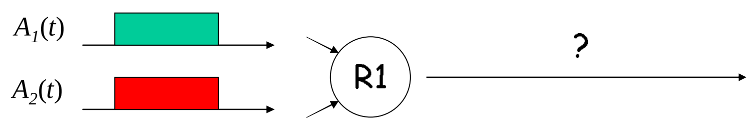


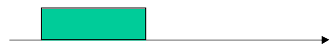
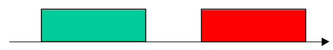
Two packets colliding at a router



Dealing with Congestion

Example: two flows arriving at a router



Strategy	
Drop one of the flows	
Buffer one flow until the other has departed, then send it	
Re-Schedule one of the two flows for a later time	
Ask both flows to reduce their rates	

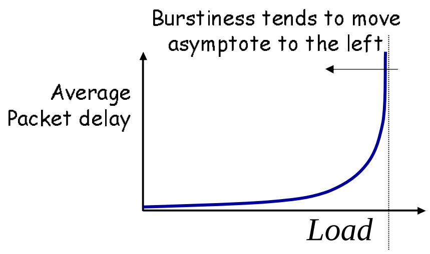
Congestion is unavoidable

Arguably it's good!

- We use packet switching because it makes efficient use of the links. Therefore, buffers in the routers are frequently occupied.
- If buffers are always empty, delay is low, but our usage of the network is low.
- If buffers are always occupied, delay is high, but we are using the network more efficiently.
- So how much congestion is too much?

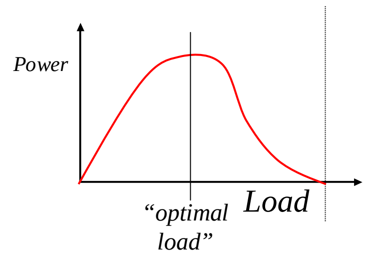
Load, delay and power

Typical behavior of queueing systems with random arrivals:



A simple metric of how well the network is performing:

$$Power = \frac{Load}{Delay}$$



Options for Congestion Control

1. Implemented by host versus network
2. Reservation-based, versus feedback-based
3. Window-based versus rate-based.

TCP Congestion Control

- TCP implements host-based, feedback-based, window-based congestion control.
- TCP sources attempts to determine how much capacity is available
- TCP sends packets, then reacts to observable events (loss).

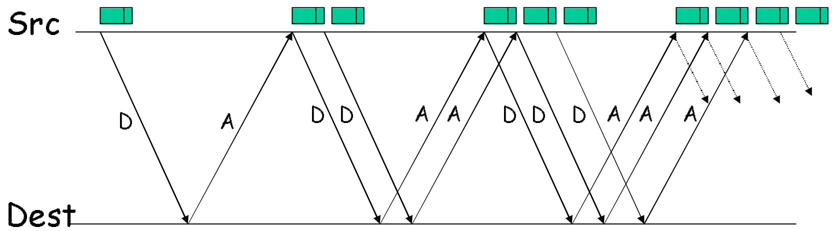
TCP Congestion Control

- TCP sources change the sending rate by modifying the window size:

  $$\text{Window} = \min\{\underbrace{\text{Advertized window}}_{\text{Receiver}}, \underbrace{\text{Congestion Window}}_{\text{Transmitter (“cwnd”)}}\}$$

- In other words, send at the rate of the slowest component: network or receiver.
- “cwnd” follows additive increase/multiplicative decrease
  - On receipt of Ack: cwnd += 1
  - On packet loss (timeout): cwnd *= 0.5

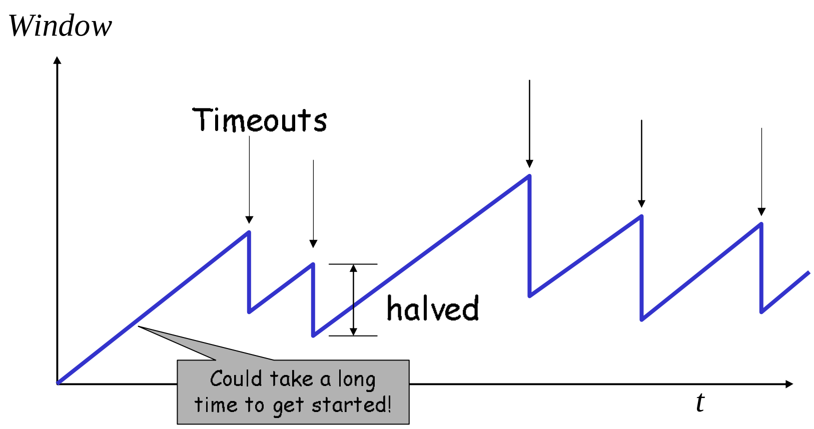
Additive Increase



Actually, TCP uses bytes, not segments to count:
When ACK is received:

$$cwnd += MSS\left(\frac{MSS}{cwnd}\right)$$

Leads to the TCP "sawtooth"

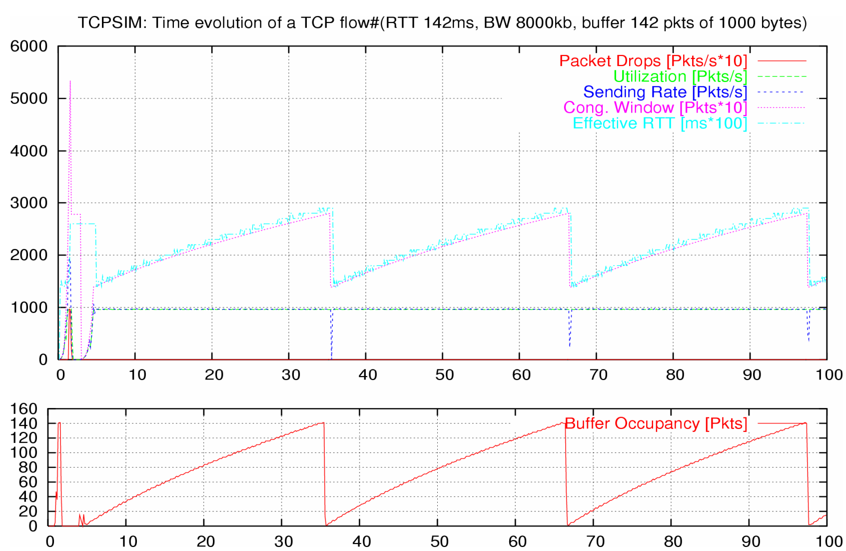


TCP Sending Rate

- What is the sending rate of TCP?
- Acknowledgement for sent packet is received after one RTT
- Amount of data sent until ACK is received is the current window size W
- Therefore sending rate is R = W/RTT

- Is the TCP sending rate saw tooth shaped as well?

TCP and buffers



TCP and Buffers

- For TCP with a single flow over a network link with enough buffers, *RTT* and *W* are proportional to each other
- Therefore the sending rate $R = W/RTT$ is constant (and not a sawtooth)
- But experiments and theory suggest that with many flows:

$$R \propto \frac{1}{RTT\sqrt{p}}$$

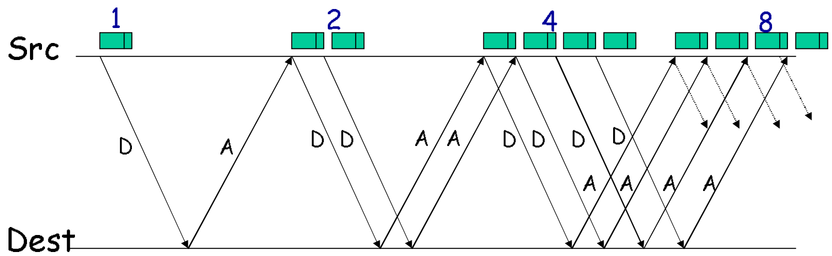
Where: p is the drop probability. You'll see this in a problem set.

- TCP rate can be controlled in two ways:
  1. Buffering packets and increasing the *RTT*
  2. Dropping packets to decrease TCP's window size

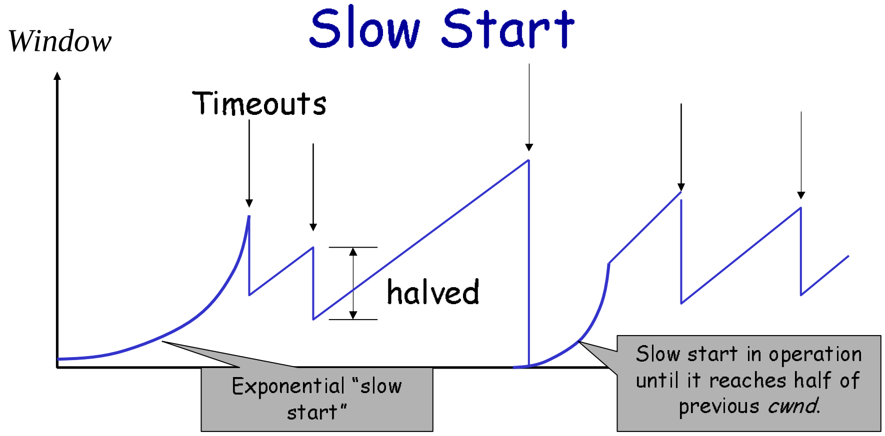
"Slow Start"

Designed to find the fair-share rate quickly at startup or if a connection has been halted (e.g. window dropped to zero, or window full, but ACK is lost).

How it works: increase cwnd by 1 for each ACK received.



Slow Start



Why is it called slow-start? Because TCP originally had no congestion control mechanism. The source would just start by sending a whole window's worth of data.

Congestion control in the Internet

- Maximum window sizes of most TCP implementations by default are very small
  - Windows XP: 12 packets
  - Linux/Mac: 40 packets
- Often the buffer of a link is larger than the maximum window size of TCP
  - A typical DSL line has 200 packets worth of buffer
  - For a TCP session, the maximum number of packets outstanding is 40
  - The buffer can never fill up
  - The router will never drop a packet