CS 514: Computer Networks
Lecture 5: Router-Assisted Resource Allocation

Xiaowei Yang
xwy@cs.duke.edu
Review: a fundamental question

- Who gets to send at what speed?
Design Space

- Router-based vs. Host-based
- Reservation-based vs. Feedback-based
- Window-based vs. Rate-based
Review

• Approach 1: End-to-end congestion control
  – e.g. TCP congestion control
    • No congestion, increase window size by one per RTT
    • Congestion, decrease window size by half per RTT
Model the system as a linear control system

\[ x_i(t + 1) = \begin{cases} 
  a_i + b_i x_i(t) & \text{if } y(t) = 0 \Rightarrow \text{Increase}, \\
  a_D + b_D x_i(t) & \text{if } y(t) = 1 \Rightarrow \text{Decrease}. 
\end{cases} \]

- Four sample types of controls
- AIAD, AIMD, MIAD, MIMD
Phase space

\[ X_2 = \frac{W_2}{RTT} \]

\[ X_1 = \frac{W_1}{RTT} \]
TCP congestion control is AIMD

- Problems:
  - Each source has to probe for its bandwidth
  - Congestion occurs first before TCP backs off
  - Unfair: long RTT flows obtain smaller bandwidth shares
Macroscopic behavior of TCP

• Throughput is inversely proportional to RTT:

\[
\frac{\sqrt{1.5 \cdot MSS}}{RTT \cdot \sqrt{p}}
\]

• In a steady state, total packets sent in one sawtooth cycle:
  – \( S = w + (w+1) + \ldots (w+w) = 3/2 \ w^2 \)

• the maximum window size is determined by the loss rate
  – \( 1/S = p \)
  – \( w = \frac{1}{\sqrt{1.5p}} \)

• The length of one cycle: \( w \ast RTT \)

• Average throughput: \( 3/2 \ w \ast MSS / RTT \)
Why is congestion control still relevant

• The Mice & Elephant phenomenon of Internet traffic
  – Many small flows
  – But a few large flows sent most of the byte
Today

• Router assisted congestion control

• What can routers do?
  – Queue/Buffer management
  – Signaling
  – Scheduling

• Active queue management (AQM)
  – Random Early Detection (RED)
  – Proportional Integral Enhanced
Queuing mechanisms

- Router-enforced resource allocation
- Default: DropTail
  - First come first serve (FIFO)
Drop-tail queues

Losses due to buffer overflow
- De-facto mechanism today

+ Very simple to implement
- Filled buffers (large delay)
- Synchronizes flows

Credit to Mohammad Alizadeh, MIT
How large should a router buffer be?

- TCP cuts its window size by half when a packet drop occurs
- Single TCP
  - Bandwidth * Delay
- Synchronized TCP flows
  - Bandwidth * Delay_avg
- Desynchronized N TCPs (N >> 1), e.g. backbone routers
  - BDP/sqrt(N)
Figure 7: The probability distribution of the sum of the congestion windows of all flows passing through a router and its approximation with a normal distribution. The two vertical marks mark the boundaries of where the number of outstanding packets fit into the buffer. If sum of congestion windows is lower and there are less packets outstanding, the link will be underutilized. If it is higher the buffer overflows and packets are dropped.
When there is a low degree of multiplexing

• Bufferbloat: large buffers lead to high latency
• “Comments on Buffer Bloat” by Mark Allman
Random Early Detection

• Random early detection (Floyd93)
  – Goal: operate at the “knee”
  – Problem: very hard to tune (why)

• RED is generalized by Active Queue Management (AQM)
Sally Floyd. “Her work on congestion control,” a colleague said, helped keep the internet “working for everyone.”  Carole Leita
High level idea

• A router measures average queue length using exponential weighted averaging algorithm:
  - \( \text{AvgLen} = (1 - \text{Weight}) \times \text{AvgLen} + \text{Weight} \times q \)
\textbf{RED algorithm}

- If \( \text{AvgLen} \leq \text{MinThreshold} \)
  - Enqueue packet
- If \( \text{MinThreshold} \leq \text{AvgLen} < \text{MaxThreshold} \)
  - Calculate dropping probability \( P \)
  - Drop the arriving packet with probability \( P \)
- If \( \text{MaxThreshold} \leq \text{AvgLen} \)
  - Drop the arriving packet
Limitations of RED

• Too many parameters. Hard to tune

• Does not directly control latency
  – High speed vs low speed links
Proportional Integral Controller Enhanced (PIE)

A Lightweight Control Scheme to Address the Bufferbloat Problem

RFC 8033
What is PIE?

- Directly control latency
- Randomly drops an incoming packet when congestion occurs
- Detecting congestion based on the derivative of queuing latency
- Parameters are chosen via control theory analysis
Fig. 3. Overview of the PIE Design. The scheme comprises three simple components: a) random dropping at enqueuing; b) latency based drop probability update; c) dequeuing rate estimation.

PIE: A lightweight control scheme to address the bufferbloat problem by Rong Pan et al.
Proportional Integral (PI) Algorithm

**Goal:** Drive error to zero

\[ e(t) = q(t) - q_{\text{ref}} \quad ("\text{error"}) \]

\[ q(t) \quad q_{\text{ref}} \]

**Every** \( T \):

\[ p(t) \leftarrow p(t - T) + \alpha (q(t) - q_{\text{ref}}) \quad ("\text{Integral control"}) \]

Credit to Mohammad Alizadeh, MIT
Proportional Integral (PI) Algorithm

Every T:

\[ p(t) \leftarrow p(t - T) + \alpha (q(t) - q_{\text{ref}}) + \beta (q(t) - q(t-T)) \]

Credit to Mohammad Alizadeh, MIT
Example: Varying TCP Traffic Intensity on 10Mbps Link

Credit: Rong Pan (Cisco)
PIE vs. RED Queuing Latency

Credit: Rong Pan (Cisco)
PIE deployment

• In October 2013, CableLabs' Data-Over-Cable Service Interface Specification 3.1 (DOCSIS 3.1) specification [DOCSIS_3.1] mandated that cable modems implement a specific variant of the PIE design as the active queue management algorithm.
Summary

• Active queue management
  – Random Early Detection
  – Proportional Integral Controller

• Next
  – Router signaling
    • Marking, Explicit messages