TCP Congestion Control

Sonia Fahmy
Department of Computer Sciences
Purdue University
fahmy@cs.purdue.edu
http://www.cs.purdue.edu/homes/fahmy/

Overview

- Current Internet
- Congestion control versus avoidance
- Slow start and congestion avoidance
- Timer estimation
- Fast retransmit and recovery
- NewReno, SACK, FACK
- TCP Vegas
- Link characteristics

Current Internet

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Applications in TCP (in bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP traffic (in bytes)</td>
<td>TCP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Applications in TCP (in bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP</td>
</tr>
</tbody>
</table>

TCP at popular web servers

- HTTP
- FTP
- SMTP
- NNTP
- Telnet

Congestion Control/Avoidance

- Congestion control:
  - Recover from zero throughput and infinite delay
  - Between knee and cliff

- Congestion avoidance:
  - Operate near high throughput, low delay point
  - Operate at knee (maximum power) of curves

Goals

- Optimize throughput and delay
- Minimize loss probability
- Minimize control overhead
- Distributed control
- Minimize parameter dependence on network characteristics
- Low parameter sensitivity

TCP Congestion Control

Go back N
Self clocking (ACKs)
Delayed ACKs

Slow Start (exponential)
Congestion Avoidance
Additive Increase
Multiplicative Decrease

Saw-tooth pattern
RTT and Timeout

- Smoothed RTT (SR) is estimated using a low pass filter.
- Assume M is latest RTT sample (non-retransmitted).
- \( V = \beta \text{abs}(SR - M) + (1-\beta)V \) \( \beta = \frac{1}{4} \)
- \( SR = \alpha M + (1-\alpha)SR \) \( \alpha = \frac{1}{8} \)
- \( \text{RTO} = SR + 4V \)
- Coarse granularity: 4V is rounded to 500ms increment ⇒ inefficiency
- Karn’s algorithm: Double RTO when timer expires for retransmitted packet

Fast Retransmit and Recovery

Duplicate ACKs start fast retransmit and recovery
Fast recovery terminates with a partial or full ACK

NewReno, SACK, FACK

- NewReno:
  - Only a full ACK or timeout terminates fast recovery; partial ACKs deflate window
- SACK:
  - Sender retransmits “holes” only. Sends when pipe (outstanding data) < cwnd
- FACK:
  - Keep track of forward-most data ACKed and retransmitted data outstanding
  - Keep outstanding data within 1 of cwnd

TCP Vegas

- Expected throughput = window size/base (min) RTT
- Diff = Actual throughput – expected throughput
- If Diff < \( \alpha \) (2), linear increase in next RTT
- If Diff > \( \beta \) (4), linear decrease in next RTT
- Slow start increases every other RTT
- Go to congestion avoidance when actual throughput is lower (by \( \gamma \)) than expected throughput
- Retransmit after one dupack if large RTT. Same for non dupack after retransmission

Wireless/Satellite Links

- Loss may not be due to congestion
  - Asymmetry
  - High error rate
  - Low speed links
  - TCPsat
    - High bandwidth-delay product
- Solutions: MTU discovery, FEC, T/TCP, large IW, delayed ACK, ssthresh estimation, rwnd, byte counting, ELN, ECN, multiple connections, segment pacing, header/payload compression, persistent connections, ACK: CC, filtering, compaction, scheduling, backpressure

TCP Flavors Summary

- Tahoe (slow start, congestion avoidance, RTT)
- Problem: inefficiency waiting for a timeout, reducing CWND to 1
- Fast retransmit (3 duplicate acks): RFC 2581 (some Tahoe)
- Fast recovery (go to ssthresh, do congestion avoidance)
- New Reno (SIGCOMM 96, RFC 2582) better recovery from multiple losses
- Vegas (better congestion avoidance) JSAC 95
- SACK (retransmit only lost segments) RFC 2018, CCR 7/96
- FACK (SACK+better fast recovery)
Key Points

- Congestion control is a very active and important research area
- Lots of work on fine tuning, e.g., initial window, Idle periods, limited transmit, SACK, etc.
- Work on RED, ECN and equation-based TFCC
- Work on sharing info across sessions
- See RFCs 2018, 2309, 2414, 2415, 2416, 2481, 2488, 2525, 2581, 2582, 2760, internet drafts, papers on Mark Allman’s web page (design and evaluation) and Ramakrishnan and Zhang tutorials

Thank You!

Questions?