CSC2209
Computer Networks

Congestion Control (in-the-middle + end-view)

Stefan Saroiu
Computer Science
University of Toronto
Project

• By now, you should have:
  – Send me an e-mail with your group’s members
  – Send me an e-mail with your group’s project topic

• Please do so by Wednesday at 9am
  – No extensions for next Tuesday’s project proposal
  – Submit by e-mail by 10am

• Several teams already approached me about their projects
  – I don’t like surprises…
Fair-Queuing

- FQ requires $O(n)$ state and $O(\log n)$ computation
  - Expensive to deploy in core (i.e., very fast) routers
- Any ideas on how to go about deploying FQ?
Idea: CSFQ

- Edge routers compute per-flow rate estimates + label packets w/ estimates.
- Core routers: FIFO queuing + keep no state, they employ probabilistic dropping based on packet labels and own aggregate traffic estimates.

Source: CSFQ, Stoica, Berkeley
Trends

• Networks are becoming very fast
  – Bandwidth-delay product is growing
• Implications: …
Trends

- Networks are becoming very fast
  - Bandwidth-delay product is growing

- Implications:
  - TCP is not-aggressive enough following a burst of congestion
    - Is this a problem?
  - Longer gap between congestion occurring and TCP’s reaction
    - Things are not tightly coupled
    - TCP becomes oscillatory with AQM
  - Fairness is becoming more important
    - At least that’s what the paper claims :-)

Idea?

- Why not controlling number of flows for cc?
  - Every RTT with no loss, initiate additional TCP flow?
  - Every RTT with loss, kill half of TCP flows?

- More flows makes everything more aggressive?
- Tighter control: not every TCP flow needs to experience a loss before backing off
- Problems?
Idea?

• Why not controlling number of flows for cc?
  – Every RTT with no loss, initiate additional TCP flow?
  – Every RTT with loss, kill half of TCP flows?

• More flows makes everything more aggressive?
• Tighter control: not every TCP flow needs to experience a loss before backing off

• Problems?
  – More state. Yuck.
  – But hold on… isn’t this just changing TCPs cc scheme?
XCP’s Gist Idea

- **Generalize ECN**
  - If plenty of spare bandwidth --> increase a lot
  - If little spare bandwidth --> increase little

- **Decouple fairness control from congestion control**
  - They should be done orthogonal
  - CC controls how much additional data to send
  - FC controls who should the additional data be allocated to
XCP: The protocol

After going thru the EC and FC, it finds ok to allow +10 for this flow

After going thru the EC and FC, it allows +5 only

RTT = XXXX
Congestion window = yyyy
Feedback = +10

RTT = XXXX
Congestion window = yyyy
Feedback = +10

RTT = XXXX
Congestion window = yyyy
Feedback = +5

Slide from Pun at USC and Shu at Rice
The protocol

- **Sender**
  - Fill in congestion information

- **Receiver**
  - Change rate according to feedback

- **Router**
  - Compute feedback
  - Operate on top of other dropping policy
  - Make decision every average RTT
  - Efficiency controller and Fairness controller
Efficiency Controller

- Maximize link utilization, minimizing drop rate and persistent queues.
- Look at aggregate traffic only, not individual flows.
- Aggregate feedback ...
Efficiency Controller

• Maximize link utilization, minimizing drop rate and persistent queues.
• Look at aggregate traffic only, not individual flows.
• Aggregate feedback $\phi = \alpha \cdot d \cdot S - \beta \cdot Q$
  
  $\alpha, \beta$ constant, $d$ average RTT, $S$ spare bandwidth, $Q$ persistent queue size
• Proportional to spare bandwidth
• Also want to drain the persistent queue
Fairness controller

• Convergence to fairness
  ➢ If $\phi > 0$, increase all flows with same Xput
  ➢ If $\phi < 0$, decrease all flows with same fraction of their Xput
• What if $\phi = 0$? Is this a problem?
Fairness controller

• Convergence to fairness
  ➢ If $\phi > 0$, increase all flows with same $Xput$
  ➢ If $\phi < 0$, decrease all flows with same fraction of their $Xput$

• What if $\phi = 0$? Bandwidth shuffling
  • $h = \max(0, \gamma \cdot y - |\phi|)$
  • $\gamma$ constant = 0.1, $y$ input traffic
  • At least 10% of traffic is redistributed using AIMD
Deployment Story

- Can benefit from CSFQ-like deployment
- TCP-friendly XCP
Is there congestion today? Where?
Is there congestion today? Where?

- Congestion is found typically on last-mile router
  - Core is believed to be very well-provisioned
  - Congestion/drops caused by broadband ISPs shaping Xfic
How Do You Design Congestion Control for Broadband?

• New assumptions
How Do You Design Congestion Control for Broadband?

• New assumptions
  – Endhost has full view on number of flows
    • No lagging response
    • Full information about congestion, but lack of understanding router policy

• Case for endhost congestion control scheme
  – Could be deployed on top of UDP