An overview and a proposal
Jan 24, 2007

Balaji Prabhakar
Stanford University
Outline

• A framework for congestion control research
  – Widely used in the academic world
  – Simulations, analysis

• Discussions of BCN and ECN

• Proposal: A simple scheme
  – Combining BCN with (F)ECN
A framework for congestion control

• Goals of congestion control scheme
  – High throughput, low latency/loss, fair, robust, and simple

• The steps in the framework
  1. Stability analysis: Need to ensure high utilization and non-oscillatory queues. The “unit step response” of the network.
     • If the switch buffers are short, oscillating queues can overflow (hence drop packets/pause the link) or underflow (hence lose utilization)
     • In either case, links cannot be fully utilized, throughput is lost, flow transfers take longer
  2. Dynamic (realistic) loading: Interested in flow transfer time
     – How quickly does network transfer flows/files?
  3. In addition to theory, extensive simulations of 1 and 2, usually using ns-2
TCP--RED: The prototypical control loop

TCP: Slow start + Congestion avoidance

Congestion avoidance: AIMD
No loss: increase window by 1;
Pkt loss: cut window by half

RED: Drop probability, $p$, increases as the congestion level goes up

$p$ vs $q_{avg}$

$min_{th}$ $max_{th}$
TCP--RED: Analytical model

TCP Control

RED Control

Time Delay

\[ W \]

\[ \frac{W}{2} \]

\[ 1/R \]

\[ N/R \]

\[ \int \]

\[ \text{LPI} \]
TCP--RED: Analytical model

Users: \[
\frac{dW_i(t)}{dt} = \frac{1}{RTT_i(t)} - W_i(t) \times \frac{W_i(t)p(t)}{RTT_i(t)}
\]

Network: \[
\frac{dq}{dt} \approx \sum_{i}^{N} \frac{W_i(t)}{RTT_i(t)} - C
\]

\[
p_{RED}(q_a) = \begin{cases} 
0 & \text{if } q_a < min_{th} \\
p_{max}\left(\frac{q_a - min_{th}}{max_{th} - min_{th}}\right) & \text{if } min_{th} \leq q_a < max_{th} \\
1 & \text{if } q_a \geq max_{th}
\end{cases}
\]

W: window size; RTT: round trip time; C: link capacity
q: queue length; q_a: ave queue length  p: drop probability

*By V. Misra, W. Dong and D. Towsley at SIGCOMM 2000
Accuracy of analytical model

- grp1: RED, 100Mbps, # of TCP flows: 600
- grp2: RED, 100Mbps, # of TCP flows: 1200
- grp3: RED, 100Mbps, # of TCP flows: 600
Delay at Link 1!!
Accuracy of analytical model

![Graph showing queueing delay over simulation time]

- Black line: scale = 1 (model)
- Red line: scale = 1 (ns)
Accuracy of analytical model
TCP--RED: Stability analysis

• “Linearize and analyze”
  – Linearize equations around the (unique) operating point
  – Analyze resultant linear, delay-differential equations using Nyquist or Bode theory

• End result:
  – Design stable control loops
  – Obtain control loop parameters: gains, drop functions, …
Instability of TCP--RED

- As the bandwidth-delay-product increases, the TCP--RED control loop becomes unstable

- Parameters: 50 sources, link capacity = 9000 pkts/sec, TCP--RED
- Source: S. Low et. al. Infocom 2002
Flow-level Models
Flow-level Models

• This type of traffic is more realistic: flows, of differing sizes, arrive at random times and are transferred through the network by the congestion management algorithms and transport protocols
  – Flow completion (transfer) time is the main quantity of interest: what is its mean? variance? how does it depend of flow sizes? on network topology, on round trip time, etc?
Flow-level models: Simulation

arrival rate: 60flows/sec
propagation delay: 50msec
# of packets/flow ~ Pareto

grp1

arrival rate: 60flows/sec
propagation delay: 100msec
# of packets/flow ~ Pareto

grp2

arrival rate: 60flows/sec
propagation delay: 150msec
# of packets/flow ~ Pareto

grp3

10Mbps 10Mbps

DropTail / RED
Layer 2 Congestion Control
BCN and (F)ECN

• BCN has been tested extensively in the previous framework
Some observations about ECN
ECN

• Stands for Explicit Congestion Notification (not to be confused with ECN from the Internet context)
  – Proposed by Prof Raj Jain at the Nov 2006 Dallas meeting

• It would be great to apply the previous framework to ECN, but…
  – We have only managed some simulations
  – And a basic control analysis

• However, I do have a couple of observations
  – They’re interesting, fundamental, and puzzling: need to understand more
The ECN scheme

- The main ideas are
  - switches estimate and advertise the current fair rate to the sources
  - sources transmit at this rate until the advertisement changes
  - each source has a switch on its path whose advertisement it obeys: the one which advertises the minimum rate
  - the key component is the rate estimation algorithm

- Rate estimation scheme: consider N sources passing through a link of capacity C at a switch
  - Time is slotted, each slot is T secs long
  - During slot k, the advertised rate is $r_k$. Ideally, $r_k = C/N$
  - The rate of arrivals during slot k is $A_k$
  - $q_k$ is the queue size at the end of slot k
  - Let $f(q_k)$ be an decreasing function of the queue size
  - $r_k$ is then recursively estimated as follows (new version has some enhancements)
The ECN scheme

\[ r_{k+1} = \frac{r_k}{\rho_k}, \text{ where } \rho_k = \frac{A_k}{C} \frac{1}{f(q_k)} \]

Let \( g(q_k) = \frac{1}{f(q_k)} \); then \( g() \) is a decreasing function of the queue-size.

Now, we get

\[ r_{k+1} = r_k \frac{C}{A_k} g(q_k). \tag{1} \]

Another equation we can write down is

\[ r_{k+1} = r_k + C - A_k - g(q_k) = r_k - (A_k - C) - g(q_k) \tag{2} \]

What is the difference between (1) and (2)?
Well...

- Eqn (1) is multiplicative, eqn (2) is linear in
  - $A - C$, which is approximately equal to rate of change of queue
  - $g(q)$ is linearly increasing in $q$ when $f(q)$ is hyperbolic!

- In other words
  - ECN feeds back the state (which is queue-size and its derivative) multiplicatively while BCN feeds it back linearly

- Multiplicative feedback isn’t common in control theory
  - In fact, the Internet controllers PI and REM are also linear in the state
  - Thus, these well-studied controllers they are almost identical to BCN

- Multiplicative feedback needs to be better understood
  - Being non-linear, it is susceptible to measurement noise in rate estimation and packet sampling, and to instability under delay
  - At is stage, we need to crack open a couple of differential equations --:
  - But, we did some ns-2 simulations of ECN to test its sensitivity
Simulations of ECN

- Using ns-2
  - New rate averaging enhancement included
  - New and increased measurement interval = 1 msec
  - Hyperbolic drop function; values from Prof Jain’s Nov presentation
  - Scenario: from Prof Jain’s on/off loading model in Nov presentation
ECN with smaller $r_0$

Rate (Gbps)
delay = 10e-6 seconds, Starting rate = 200Mbps

Queue (pkts)
delay = 10e-6 seconds, Starting rate = 200Mbps
BCN in same scenario and bigger delays
BCN queue depths
BCN individual rates
What happened to ECN’s control loop?

- The nonlinearity has some serious consequences (thanks Rong Pan and Ashvin Lakshmikantha)

- It makes $q_{eq}$ a parameter of the control loop!!
  - That is, the bigger $q_{eq}$ is, the more stable it is!
  - This is not true of BCN (or other Internet controllers like PI and REM)
  - And is entirely because ECN multiplies state, while BCN and the others add

- If this is true, we should be able to increase $q_{eq}$ in the previous setup and stabilize ECN
Throwing buffers to buy stability
About fairness

- Fairness is a key metric, along with high throughput and low backlogs
  - There is always a higher price to pay for fairness in terms of algorithm complexity. Why?

- Consider example below: 2 links, each with capacity = 1
Complexity and fairness

- From J. Mo and Walrand (1998):
Other issues

• Measurement interval: Can’t be long or short!
  – Gone up to 1 msec from 30 musecs in Nov 2006
  – Short interval: Noisy estimation hurts stability
    • Rate estimation is noisy, long interval helps convergence
    • Can’t signal too many sources (30 musecs = 30 1500B pkts)
  – Long interval: Not responsive, need buffers to store changes
    • Rate estimation is accurate, but can’t be very responsive
    • New sources will get old rate for 1 msec; switch needs to absorb extra pkts with bigger buffers

• Need 32 bits to signal rate in fine detail
  – Cannot give flows one of, say, 16 or 32 levels
  – Because every flow needs to send at exactly the same rate; rate differences are not allowed!
  – Quantization will lead to less total arrival rate at one level and to higher rate at the next one up

• Possible security issue: Network advertising rate explicitly on bottleneck links invites attacks!
Summary on ECN

• Nonlinear feedback of state is very uncommon
  – In this case leads to serious control problem: stability needs big buffers
  – This is not true of BCN (or other Internet schemes like REM and PI)

• Max-min fairness is complex whichever way you try to do it
  – No distributed, low communication overhead algorithm known to date
  – Equivalent to per-flow work

• Measurement interval cannot be chosen painlessly

• Need detailed rate signaling capability, a 4 or 5 bit signal is not sufficient

• Possible security issue: Network advertising rate explicitly on bottleneck links invites attacks!
A proposal: Combining BCN and (F)ECN
Proposal: A Simple Algorithm

- Use BCN’s control loop
  - Proven to be stable
  - Extensive work on REM and PI which are exactly like BCN (see below) in the Internet context, shows their stability and low backlogs

- BCN generates extra signaling traffic
  - Hence sampling probability is kept at 1%; this can go up to 10% and improve responsiveness by a lot
  - But, if forward signaling is possible, or another means of signaling more frequently can be found, then we can send less information per signal

- Main ideas
  - Compress and quantize BCN signals at switch: a 4-bit quantization works great
  - This multi-bit signal can be trivially looked up in a table at the source and generates source’s reaction (rate decrease/increase)
  - Let source increase rate multiplicatively and let switch only send decrease signals
Details of the simple algorithm

- Need a name…
  - DCN? For Distributed Congestion Notification
    - D is between B and FE
    - Deccan is part of India I’m from --:
  - QCN? For Quantized Congestion Notification
    - Quicken

- Recall: In the current BCN
  - The CP sends: Qoff and Qdelta
  - The RP:
    - Computes \( F_b = -(Q_{off} + w \cdot Q_{delta}) \)
    - If \( F_b > 0 \), then \( R \leftarrow R + G_i F_b R_u \)
    - If \( F_b < 0 \), then \( R \leftarrow R (1 + G_d F_b) \)
  - Note: only \( F_b \) is used in the rate computations! No need to send \( Q \) and \( Q_{delta} \)
  - \( F_b \) is exactly the quantity used by REM and PI to mark packets at router, instead of the RED drop function

- So, let switch compute \( F_b \) (very easy, esp because \( w \) is a power of 2, usually \( w = 2 \))
- Quantize \( F_b \) to one of 4 or 5 bit levels and send to source
Details of the simple algorithm

• QCN: control algorithm
  – Switch
    • On sampled packets switch computes Fb (very easy, esp because w is a power of 2, usually w = 2)
    • Switch quantizes Fb to one of 4 or 5 bit levels and send to source
  – Source
    • Reacts appropriately by using Fb to index a lookup table
    • Periodically (when timer expires) increases its rate multiplicatively

– Notes
  • All parameters chosen already, as in WG discussions
  • Quantization can be uneven (nonuniform quantization): more decrease levels, different spacing, etc
  • Simulations show that 4-bit quantization is nearly similar to full signaling
Why not send increase signals?

- Switch signals only rate decreases, source performs multiplicative rate increases.

This has a few benefits:

1. It gets rid of the sampling bias problem; i.e. no rate increases to already large flows
2. More importantly, it gets rid of the RP--CP association; if no CP is going to send an RP rate increase messages, then there is no need for the RP to store the id of last CP which signaled a decrease or to send this id out on packet headers.
3. Finally, there is a reduction in signaling traffic.
   - Note: we may still want to keep 1 or 2 increase signals because a switch can more quickly utilize its links
Performance of simple version

• Theoretically, neither feature affects the stability of the system; the stability margin is lowered a little, not the stability property
  – Because feedback is linear, quantization noise moves the poles by a small amount depending on the granularity of quantization; thus, the stability margin is slightly affected, not the stability itself.

• Simulation evidence: The following tests have been done till now (and will be exhibited in the next few slides).
  1. Davide Bergamasco has tried out, on his simulator, a 6-bit quantized version of BCN on the baseline scenario discussed in the WG. The performance is nearly indistinguishable; the quantized version is slightly wiggly.
  2. Ashvin has generated plots comparing the 5-bit quantized version to BCN for “on/off inputs.”
  3. Abdul has compared the 5-bit quantized version to BCN using flow-level models.
  – Grand conclusion: The simple version compares v.favorably.
Baseline scenario: 6-bit quantization
On/off sources: 5-bit quantization
Flow-level models: 5-bit quantization

- Simulation setup
  - Hyper-exponential with mean of 50 packets
  - SF: Short flows -> Mean size: 20 pkts
  - LF: Long flows -> Mean Size: 320 pkts
  - 10% Long flows
  - Sampling rate: 0.03
  - Single link, IEEE parameters
  - FCT measured in milliseconds
Ave flow completion time
FCT ave for long and short flows

![Graph showing FCT (milliseconds) vs Load for long and short flows. The graph compares long full bcn, short full bcn, long quantized bcn, and short quantized bcn.]
With no switch signaled increases
With no switch signaled increases
Conclusions

• Thanks for listening
  – Thanks again to Rong Pan, Ashvin Lakshmikantha, Abdul Kabbani, and Davide Bergamasco

• Overviewed Internet research
  – Fairly substantial, vibrant literature

• L2 Congestion Control
  – Presented some work on BCN
  – Some observations about ECN
  – Proposed QCN, combines BCN and (F)ECN

• Welcome your feedback