QCN: Quantized Congestion Notification

Rong Pan, Balaji Prabhakar, Ashvin Laxmikantha
Overview

• Description the QCN scheme
  – Pseudocode available with Rong Pan: ropan@cisco.com

• Basic simulations
  – Infinitely long-lived flows: stability of control loop
  – Dynamic flows: FCT

• Notes on FCT, heavy-tailed flow size distributions
**Reaction Point:** Where the rate of injection of a flow (or flows) is changed due to congestion signals; usually, the place where rate limiters reside.

**Congestion Point:** Where resources (buffers/links) exist and can be congested, and where congestion signals are generated; usually, switch buffers and the links they are attached to.

**Reflection Point:** Where congestion signals are reflected back to the source.

**Congestion Management Domain:** ReaP -- CPs -- RefP.
Summary of QCN

• Summary description of mechanisms at the Reaction Points, Congestion Points and Reflection Points.
  1. **Reaction Points**: Insert $F_b = 0$ in outgoing packets. When congestion message arrives: perform multiplicative decrease, fast recovery and active probing.
  2. **Congestion Points**: Compute $F_b$, overwrite $F_b$ in packet header, reflect *negative* $F_b$ values as “intermediate reflection points”.
  3. **Reflection Points**: Reflect $F_b$ values to ReaPs with a probability biased by the $F_b$ value in the packet.

• We also describe a number of useful enhancements; the details can be found in the pseudocode.
  1. Extra fast recovery: Helps a reaction point recover from bursty “dings”.
  2. Positive reinforcement: Helps a reaction point use $F_b=0$ values to increase rate.
  3. No queue-length capping: Helps a congestion point use current queue length, even though this may exceed $q_{eq}$. This is possible in QCN because $F_b$ is computed at the switch. It allows the switch to send high $F_b$ values when its queue size is large. This is similar to BCN-MAX.

• Feedback message: Packet generated by a reflection point for a reaction point and corresponds to a data packet sampled during congestion. Contains $F_b$ value and flow identification information.
• ReaP Dynamics
  – Starting rate for every flow equals 10Gbps
  – Insert Fb = 0 in outgoing packet
  – Insert a “flowid” into outgoing packet’s header (optional)

• Whenever a ReaP receives a feedback message
  – Decrease rate from R to Rnew = R(1 - IFbIxGd), where Gd is a gain
  – Perform “fast recovery” and “active probing”
  – Useful refinements: “extra fast recovery” and “positive reinforcement”
    • The way refinements are to be incorporated is detailed in the p-code
    • We give a high-level description at the end of the presentation
Fast Recovery and Active Probing

Time

Rate

R

Rd

Rd/2

Rd/4

Rd/8

Fast Recovery

Active Probing

Congestion message recd
Reaction Point (cont’d)

• **Fast recovery**
  – Let $R_d = R | F_b | G_d$ be the amount of rate decrease
  – Fast recovery proceeds in cycles, clocked by the “fast recovery timer”
  – Fast recovery timer
    • Resets upon receipt of negative $F_b$ message
    • Increments for every transmitted byte
  – Fast recovery cycles
    • The FR timer counts out cycles (every so many transmitted bytes)
    • Cycles numbered 0, 1, 2, 3, 4, 5; the maximum cycle number is 5
  – The transmission rate during cycle $k$ equals $R_{new} + R_d/2 + R_d/4 + \ldots + R_d/2^k$ as long as no further QCN messages are received
  – If a congestion message is received, then cut rate as before and restart fast recovery
  – At the end of fast recovery, the source moves to Active Probing

• **Active probing (multiplicative increase) always follows fast recovery**
  – Active probing is clocked by the same byte counting time as fast recovery
  – When timer expires, the current rate is changed to $R + R_i$, where $R_i$ is the increase amount
  – If a congestion message is received, cut rate as before, perform fast recovery and then active probing
Congestion Point

- At the CP
  - Sample packets with probability $p$
  - Compute: $F_b = -[q_{off} + w q_{delta}]$
  - Overwrite
    - If $F_b < 0$, and if $F_b$ value is smaller (more negative) than $F_b$ value in the packet header, then overwrite $F_b$ value in packet header with computed $F_b$ value
  - Reflect
    - If $F_b < 0$, then reflect $F_b$ value probabilistically back to the source with a bias which increases with $F_b$
    - Set the “frame-reflected” bit
Reflection Point

- The end reflection point (in the 3-point architecture)
  - If the incoming frame has the “frame-reflected” bit set, do nothing
  - Elseif $F_b < 0$, reflect $F_b$ value to source with a probability which increases with $|F_b|$
  - Else $F_b = 0$; reflect this back to source with some small probability, say $P_0$
Refinements

1. **Extra-fast Recovery:** Suppose a source gets multiple congestion messages in a burst, driving its rate down by a lot. Say that the rates of decrease are $R_{d1}$, $R_{d2}$ and $R_{d3}$. Fast recovery only uses the *last* amount of decrease $R_{d3}$. Using $R_{d1}+R_{d2}+R_{d3}$ (or even $\max(R_{d1}, R_{d2}, R_{d3})$) improves performance. This is done *only* during the first cycle (cycle 0) of Fast Recovery. Negative messages received in subsequent cycles restart Fast Recovery.

2. **Positive Reinforcement:** If a Reaction Point receives an $F_b=0$ message, it immediately advances to the next cycle of Fast Recovery or Active Probing. I.e. it changes the timer value and transmission rate appropriately.

3. **No queue-length capping:** $F_b$ is computed as $F_b = - [(q-q_{eq}) + w \cdot q_{\text{delta}}]$, where $q$ is not constrained to be between $-q_{eq}$ and $2q_{eq}$. Removing this restriction allows the switch to compute more negative $F_b$ values if the buffer is congested beyond $2q_{eq}$. Of course, these are quantized to the maximum negative value of $F_b$, should they exceed the range of $F_b$. This behavior correctly signals the “congested buffer” condition similarly as BCN-MAX, while retaining “linearity.”

- Note: we have tried not to introduce extra parameters or behaviors so as to avoid extra specification and tuning.
Basic Simulations
1. Infinitely long-lived flows
   • Simultaneous starts
   • Staggered starts

2. Dynamic heavy-tailed flows
   • Flow completion times for long flows, short flows
   • Losses
Parameters and settings

• Infinitely long-lived flows: simultaneous starts
  – Single link, 6 flows on at 10 Gbps at time 0
  – Link delay (RTT): 40 microseconds
  – Gd = 1/128
  – w = 2
  – Ri = 3 Mbps
  – Sampling function = linearly increases with IFbl from 1--25%
  – Reflection probability = 0 and 2.5%

• Staggered starts: staggered starts
  – Single link, 6 flows on 500 microseconds apart
  – Same parameters as above
Simultaneous, reflection probability = 0
Queue length; # Pkts dropped = 449
Simultaneous, reflection probability = 0 Rate
Simultaneous, reflection probability = 2.5%
Queue length; #Pkts dropped = 462
Simultaneous, reflection probability = 2.5% Rate
Staggered, reflection probability = 0
Queue length; #Pkts dropped = 11
Staggered, reflection probability = 0
Rate
Staggered, reflection probability = 2.5%  
Queue length; #Pkt s dropped = 8
Staggered, reflection probability = 2.5% Rate
Unit step response vs FCT

- Historically, congestion control research has considered the performance of a scheme under infinitely long-lived flows
  - This gives the unit step response of the scheme
  - Very useful for control-theoretic analysis and hence for picking the parameters for the stability of the control loop
  - But, it does not capture dynamic situation of flows arriving and departing (which is the actual situation)
  - It does not have a notion of “load” which can be increased; it is always at 100% load
  - It does not capture flow completion time (FCT), a quantity users care about

- The recent literature takes a 2-step approach
  - First study scheme under infinitely long-lived flows
  - After picking parameters and ensuring stability of control loop, consider FCT
  - This is consistent with CPU performance under “workloads” consisting of files and brings the role of algorithms into focus
  - Key metric: FCT and the related quantity “slowdown”
  - Slowdown for job or flow of size $x = \text{FCT of flow} / x = 1 / \text{Bdwdth given to the flow}$
Dynamic flows: FCT and Drops

- **Workload**
  - IPC traffic: Mean = 5 KB (uniform distribution)
  - Data traffic: Pareto, shape 2, mean 100 KB
  - Parameters (Gd, w, etc): same as before
  - Reflection probability = 0, 2.5 and 5%
Completion Time of Short Flows

C=10Gbps, IPC traffic mean =5KB (Uniform), Data traffic = 100KB (Pareto: Shape 2.0), Delay = 40 mus
Completion Time of Long Flows

C=10Gbps, IPC traffic mean =5KB (Uniform), Data traffic = 100KB (Pareto: Shape 2.0), Delay = 40 mus
Drops

C=10Gbps, IPC traffic mean = 5KB (Uniform), Data traffic = 100KB (Pareto: Shape 2.0), Delay = 40 mus