QCN: Quantized Congestion Notification
An Overview

Rong Pan, Balaji Prabhakar, Ashvin Laxmikantha

IEEE 802.1 Interim Meeting

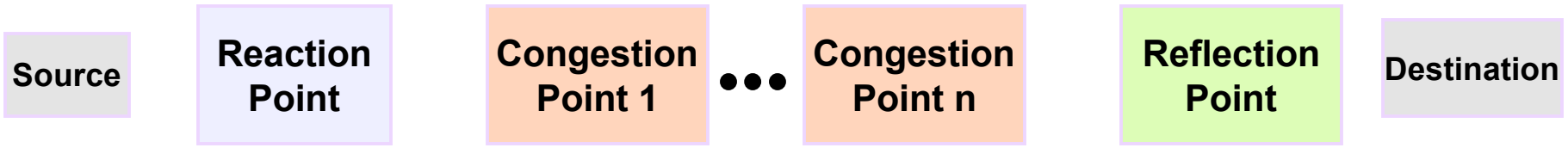
May 29, 2007

Geneva

Outline of presentation

- Overview of QCN
  - Including a discussion of options and choices
  - Implementation, deployment
- Discussion of simulations
- Rong Pan's presentation
  - Details of QCN
  - Simulation results

Congestion management loop components

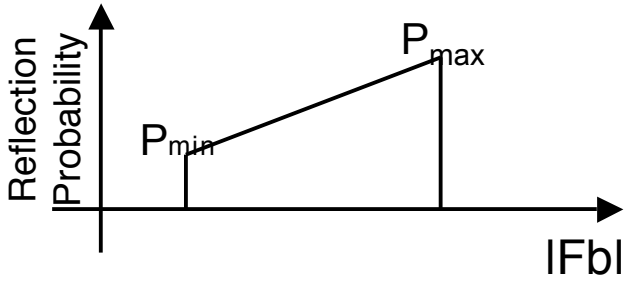


- **Reaction Point:** Where the rate of injection of a flow (or flows) is changed due to congestion signals; usually, the place where rate limiters reside.
- **Congestion Point:** Where resources (buffers/links) exist and can be congested, and where congestion signals are generated; usually, switch buffers and the links they are attached to.
- **Reflection Point:** Where congestion signals are reflected back to the source.
- **Congestion Management Domain:** ReaP -- CPs -- RefP.

Basic QCN

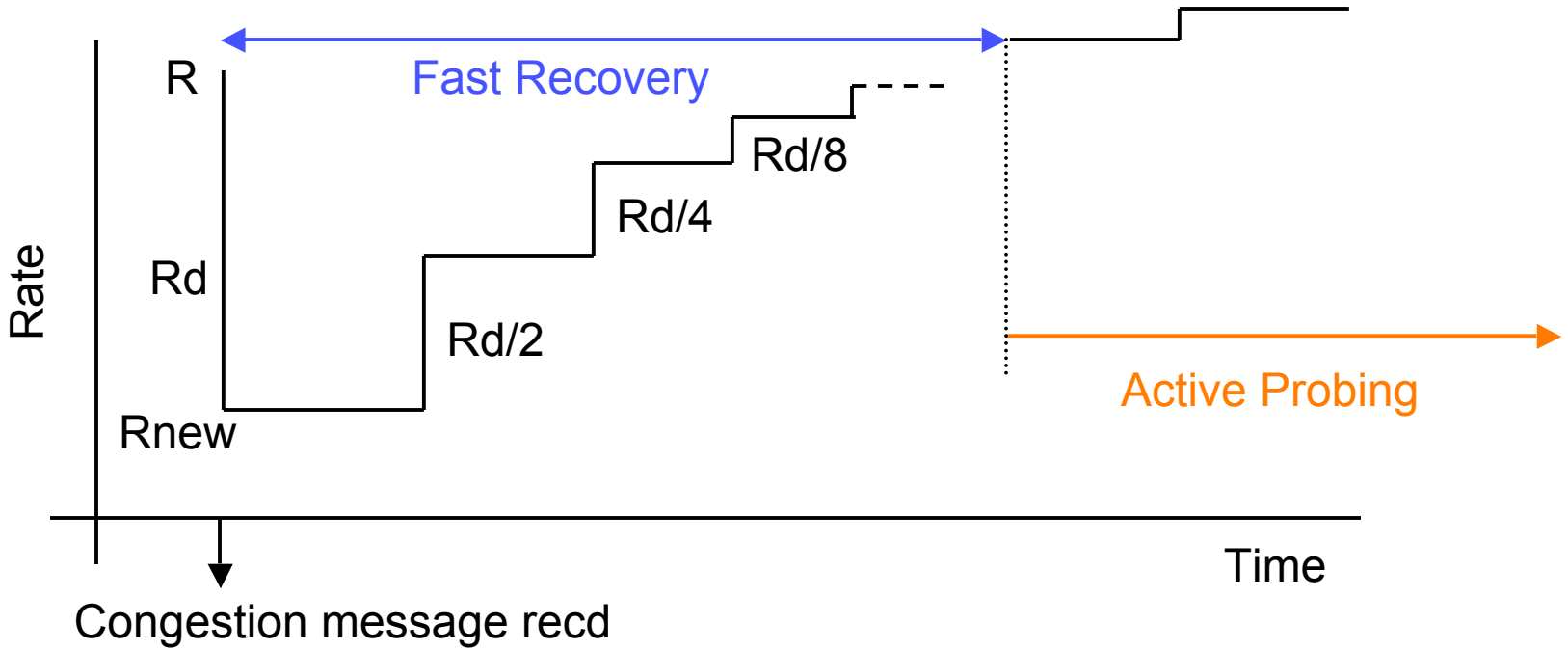
- 2-point architecture: Reaction Point -- Congestion Point
  1. **Congestion Points:** Sample packets, compute feedback (Fb), quantize Fb to 6 bits, and reflect only *negative* Fb values back to Reaction Point with a probability proportional to Fb.

$$Fb = -(q_{off} + w\, q_{delta})$$
$$= -(\text{queue offset} + w.\text{rate offset})$$



  2. **Reaction Points:** Transmit regular Ethernet frames. When congestion message arrives: perform multiplicative decrease, fast recovery and active probing.
     - Fast recovery similar to BIC-TCP: gives high performance in high bandwidth-delay product networks, while being very simple.

Fast Recovery and Active Probing



Basic QCN: Outcomes/results

- Easy to deploy, light resource requirement
  - No header modifications, no tags, **immediately deployable**.
  - Can work with a *single* rate limiter.
    - Alias all flows which have received negative feedback onto the rate limiter. RL becomes “meta-flow” with fast recovery + active probing ensuring good performance.
    - The algorithm is well-defined; i.e. does not rely on the existence of multiple rate limiters for correctness of specification since it has no tags or probes.

- Quantizing Fb simplifies implementation
  - Fb value used to index into a small table to find the decrease factor.
    - No potentially expensive hardware resources needed for computations.
  - Lookup table also makes the scheme **easily reconfigurable** (if Fb --> Rate relation changes), a useful workaround.

QCN: 3-point architecture

- ReaP--CP--RefP
  - Allows signaling Fb=0 values to ReaP, which indicate *lack* of congestion. Only the RefP can do this without the use of RP-->CP association tags.
  - When a ReaP receives an Fb=0 signal, it just skips to the next cycle of Fast Recovery or Active Probing; i.e. it increases the rate appropriately and it restarts the byte counter
    - Simple behavior, no increase gains or parameters.
  - Two flavors of signaling
    - In-band: Using packet headers
    - Out-of-band: Using probe packets (as in E2CM and FECN)
- In-band signaling
  - In the pseudocode released, we showed how the 6-bit Fb field in the packet header can be modified at the switch for sampled packets and how reflection occurs at CP and RefP.
  - A probe version of this scheme can also be done.

Simplifying signaling further

- Note that
  - To maintain low drops while allowing sources to come on at 10 Gbps, we need negative Fb values to be signaled backward; the forward path has a larger delay.
  - To grab extra bandwidth, it is useful to signal Fb=0. We can employ forward signaling to do this without tags.

- Therefore, we propose
  - All Fb-negative signals generated probabilistically by CPs
  - RefP reflects only Fb=0 signals
  - This elegantly extends the 2-point architecture to the 3-point architecture
  - As we will see in the simulations, it also performs excellently

- Two concrete signaling methods based on this proposal are…

Signaling in the 3-point architecture

1. Use probe packets, say 1 in K packets from the source
   - Probe enters network with a single Fb0-bit set to 0 and passes through the CPs
   - If a CP has Fb < 0 value, it sets the Fb0-bit to 1
   - When RefP receives a probe
     - If Fb0-bit is set to 1, do nothing
     - Else, reflect probe with small probability (e.g. 1-3%)

2. Using the DE (Discard Eligible) bit in the packet header
   - DE bit set to 0 when packet leaves source
   - If a CP samples the packet
     - If DE bit is set to 0 and CP sends Fb-negative message for this packet, set DE bit to 1
     - If DE bit is set to 1, do nothing (specifically, don't send Fb-negative message)
   - When RefP receives a packet
     - If DE bit is set to 1, do nothing
     - Else send Fb=0 signal to source with small probability (e.g. 1-3%)

About the pseudocode

- The pseudocode is complete, but it is important to note that
  - Some points pertaining to signaling (e.g. use probe packets or headers?) are not yet finalized in the p-code because they are under discussion.
  - The Fb field in the packet header may not be needed if we use the DE bit or probes.
  - Parts of the p-code will be affected by decisions on above points (e.g. overwriting Fb field in packet header).
    - The performance of the algorithm does not depend on these decisions which are signaling-related.
  - Finally, the p-code continues to be edited because of user feedback. We will post updates periodically.

Simulations

- Have performed basic simulations
  - Infinitely long-lived flows: stability of control loop
  - Dynamic flows: FCT
  - Baseline simulations

- More simulations, which study relationship of performance with limited number of rate limiters is for further work. This is v.useful to understand and an important implementation consideration.

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
- The study of dynamic flows and FCT has a firm intellectual basis, extensively used; I'll give a tutorial soon and discuss concrete steps with Mitch, et al