Flow and Congestion Controls for Datacenter Networks - 2

L2 Congestion Control

A Converged Enhanced Ethernet-centric View of Physical and Virtual DCNs

Presented by Mitch Gusat
IBM Research, ZRL

HPSR'14 Vancouver, FCC Tutorial

IBM Research GmbH, Zurich Research Laboratory

Outline #2: Layer 2 Congestion Control

1. Why L2 CC?
    1. Solutions to HOL-blocking and saturation trees in lossless networks
2. DCN benchmarking
    1. DCN topologies, traffic scenarios, metrics
3. IBA: The CCA (Congestion Ctrl. Annex)
4. CEE: The .1Qau QCN (Quantized Congestion Notification)

Advanced / alternate QCN topics (+refs and reading lists)

1. Load balancing and adaptive routing
2. QCN with TCP/UDP
3. QCN-based traffic monitoring

What Problem Are We Trying to Solve?

Ethernet Congestion 'Pyramid'

Time* Constant	Solution Class	Sensor Type	Feedback Loop(s)
s	TCP	loss, RED, REM, .1au sensor (if available)	ACK, self-clocking window, ECN
ms	.1au CM	q, q', delay, N, rate…	ECN + Probing
us	LL-FC	Buffer	PAUSE / PPP

* HS duration = time_ct * (10s to 1000s)

High Order HOL-blocking & Saturation Trees Congestion in DCNs

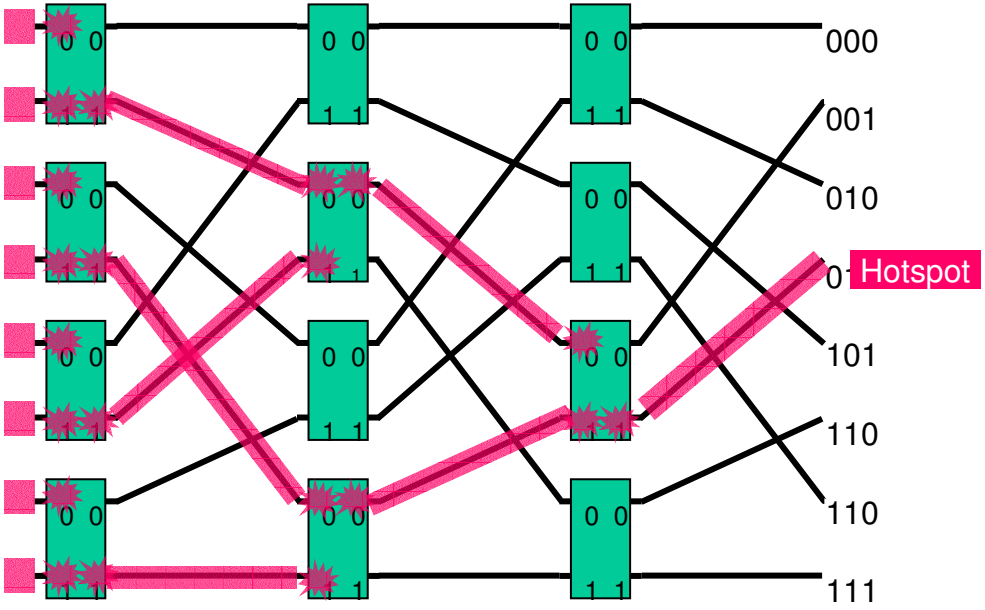
The Bad side of LL-FC (lossless links)

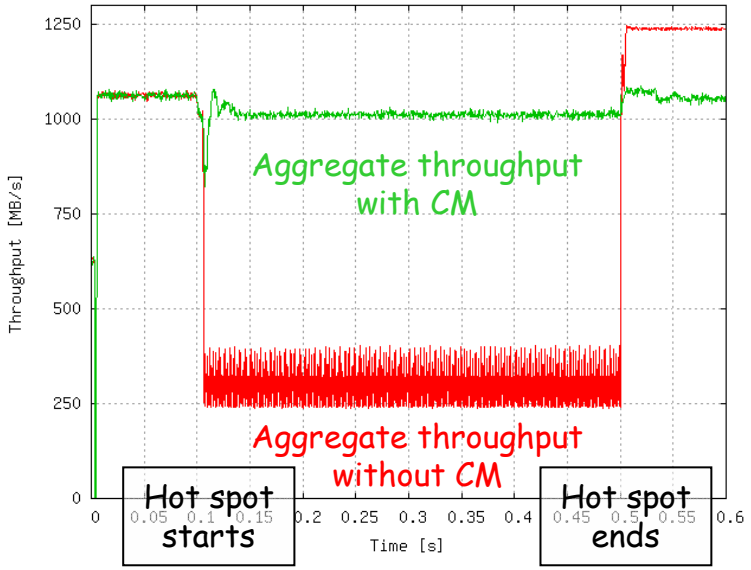
IBM Research GmbH, Zurich Research Laboratory

Lossless Link ➔ From HOL-blocking to Saturation Trees

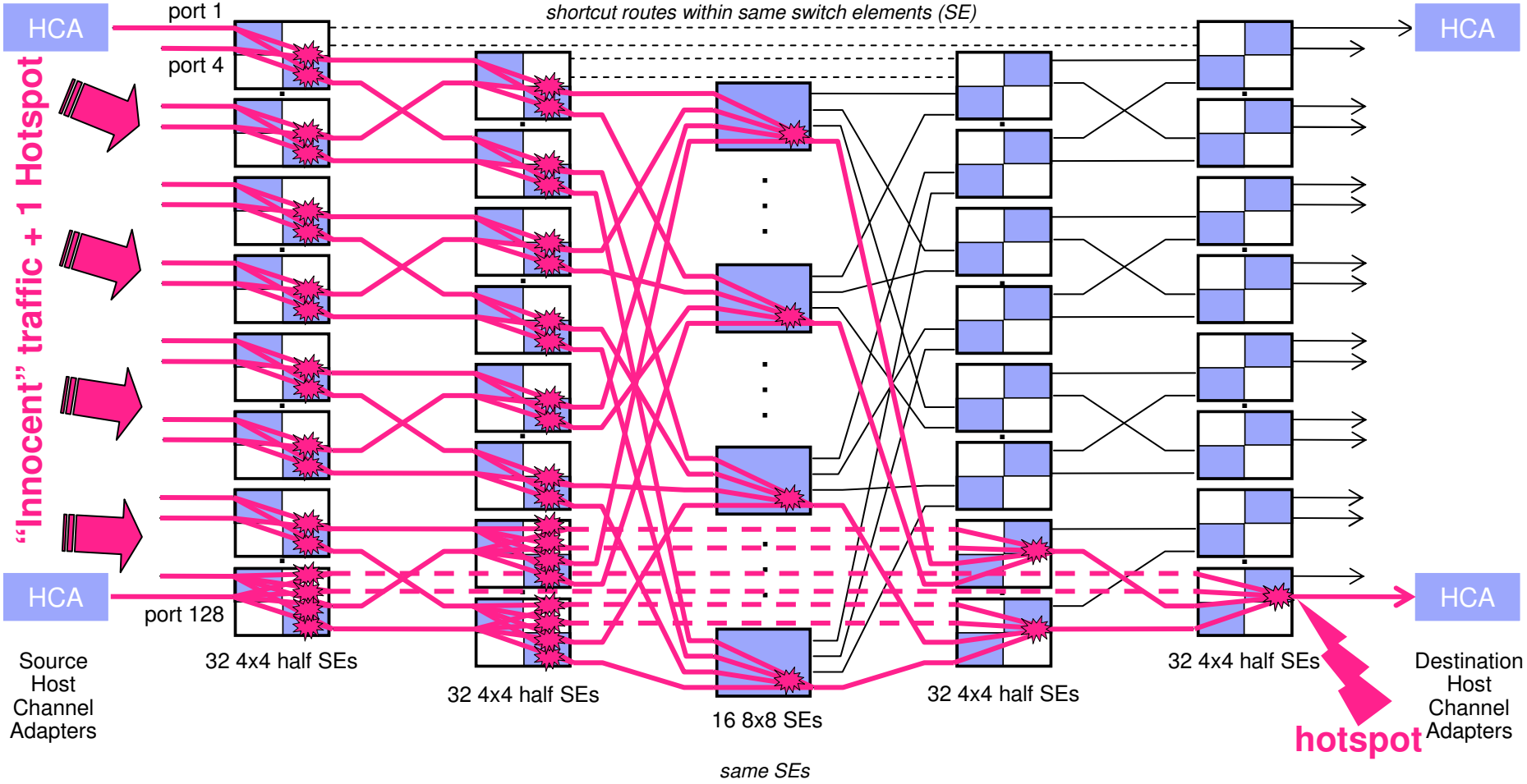
- Network congestion can lead to severe performance degradation
  - Lossy networks suffer from the "avalanche" effect: High load ➔ drops ➔ retransmissions ➔ increased load ➔ even more drops
  - Lossless networks suffer from *saturation tree congestion:* Link-level flow control (LL-FC) can cause congestion to roll back from switch to switch

- Congestion management (CM) is needed to deal with long-term (sustained) congestion
  - Dropping and LL-FC are ill suited to this task, dealing only with short-term (transient) congestion
  - Push congestion from the core towards the edge of the network

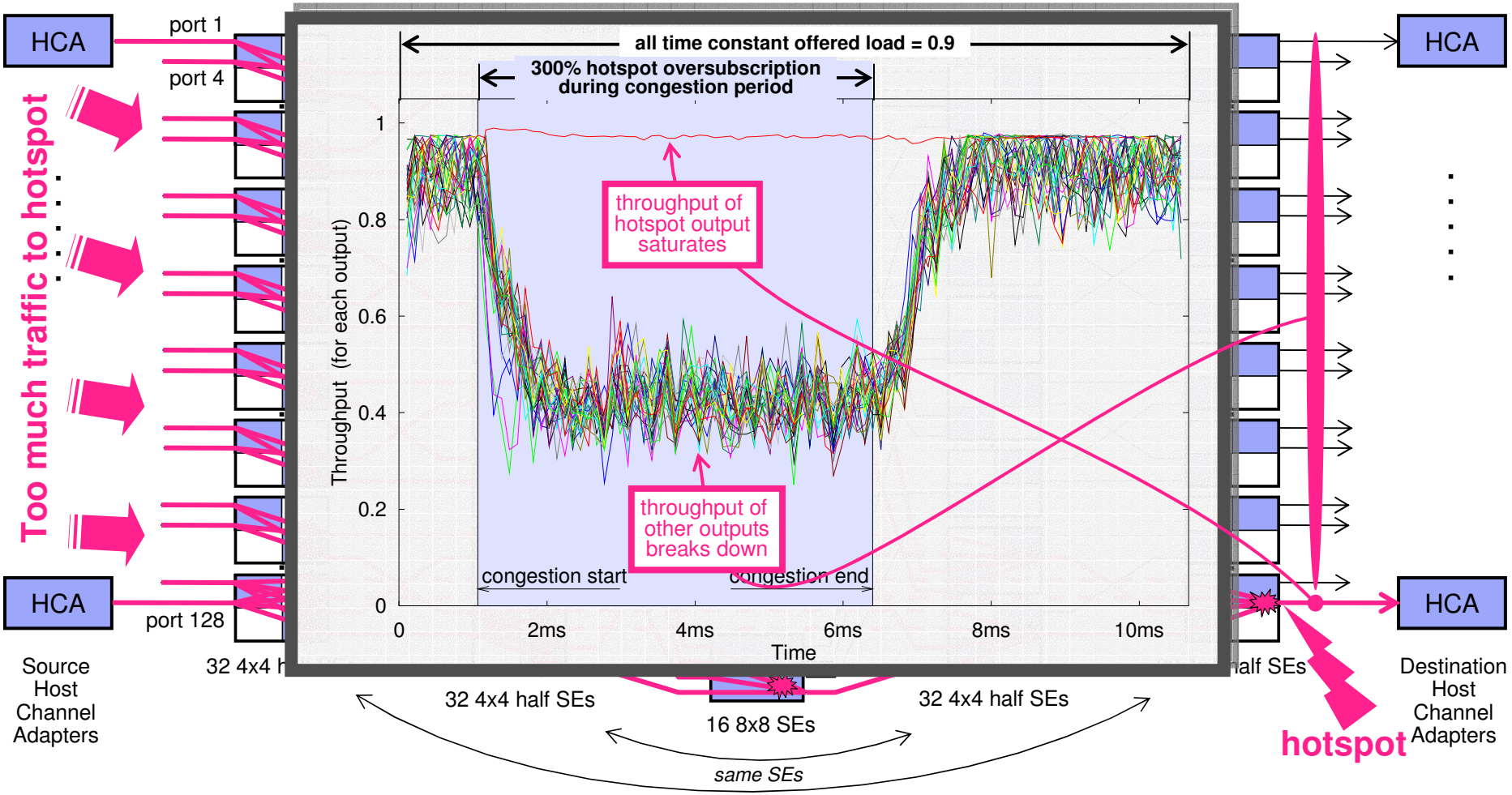




IBA: Hotspot Congestion in Lossless ICTNs => Saturation Trees



Effect: Global collapse of throughput caused by single hotspot



DCN L2 Benchmarking
DCN topologies, traffic scenarios, metrics

Adopted from IBA and 802 DCB

Aims of an Ideal CM Solution

- The ideal CM should provide:
  1. global work-conservation of all the links and switches
     -> even under severe congestion
  2. saturation tree containment/disentanglement
     -> un-blocking of innocent flows
  3. fast convergence to the max. *fair* aggregate T_{put}
  4. min. e2e latency of cold flows
  5. self-stabilization independent of the traffic pattern
     -> id est, no manual tuning

Metrics and Types of Congestion

- The three main issues of single-hotspot congestion
  1. *How severe is the HS?*
     the congestion factor (cf) at the root switch, or, overload of the bottleneck link
  2. *Where is the HS located – at the 'front' or the 'back' of the fabric?*
  3. *What is "my" (a source's) relative contribution to the HS?*
     - what % of the overload is mine?

- A simple answer yields 8 distinct classes of congestion:

  - 4 types of congestion
    - Mild, cf=1..2
    - Severe, cf >> 2 (tens or more)
    - Near (the sources): within 2-3 link RTTs
    - Remote: many RTTs away (large queuing delay)

  - 2 types of HS participation (HS Degree)
    - Low: a few flows
    - High: many flows (Ks, due to SDN / virtualization)

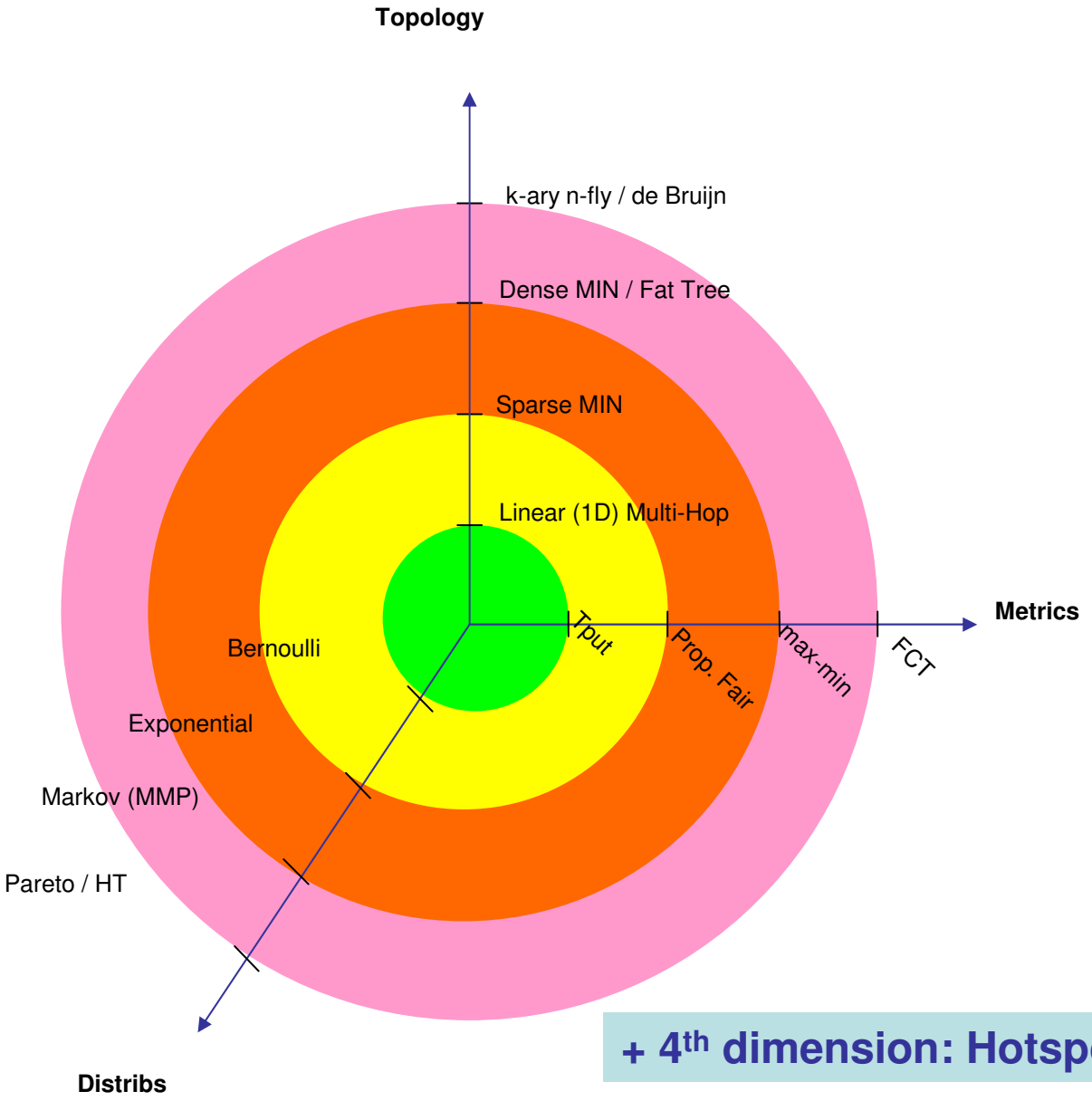
A Brief Overview of IBA and DCB/CEE Congestion Modeling

- Method

- Topology and hotspot types
  - Baseline: dumb-bells, parking lots, multi-hop
  - Practical: MINs (fat-trees)
  - HS types
    - many2one (IG, fan-in)
    - link capacity reduction (OG)

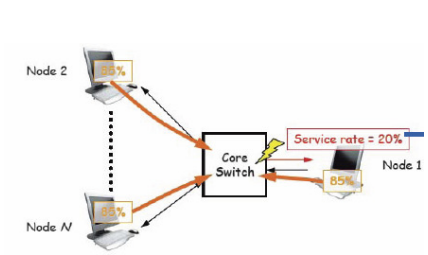
- New metric: Flow Completion Time (FCT) @ L2

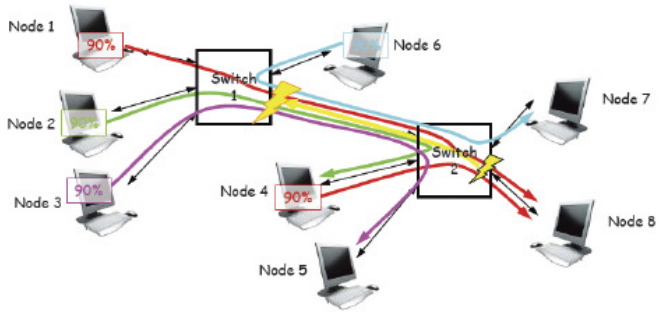
Proposed Method: CM Benchmarking Sphere

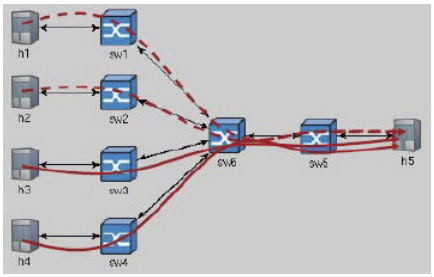


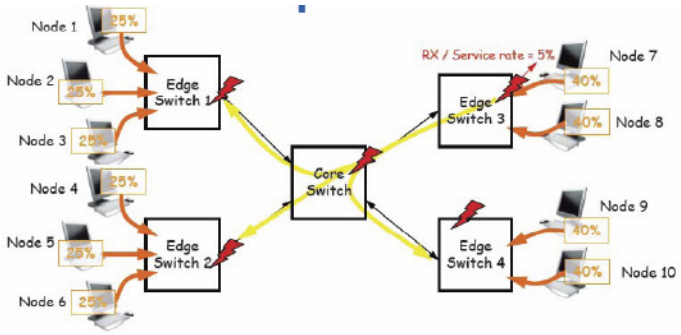
+ 4th dimension: Hotspot Primitives

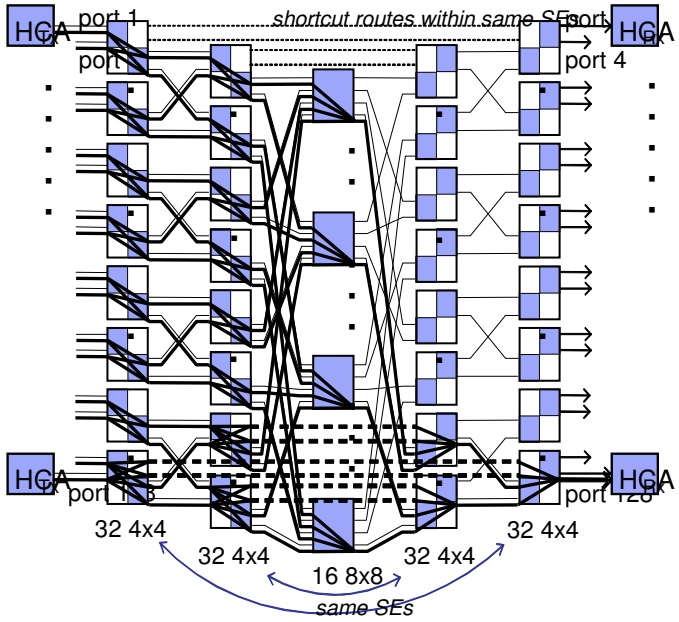
Topology: From Single-Stage thru Sparse MINs to Fat-trees and k-ary n-cubes







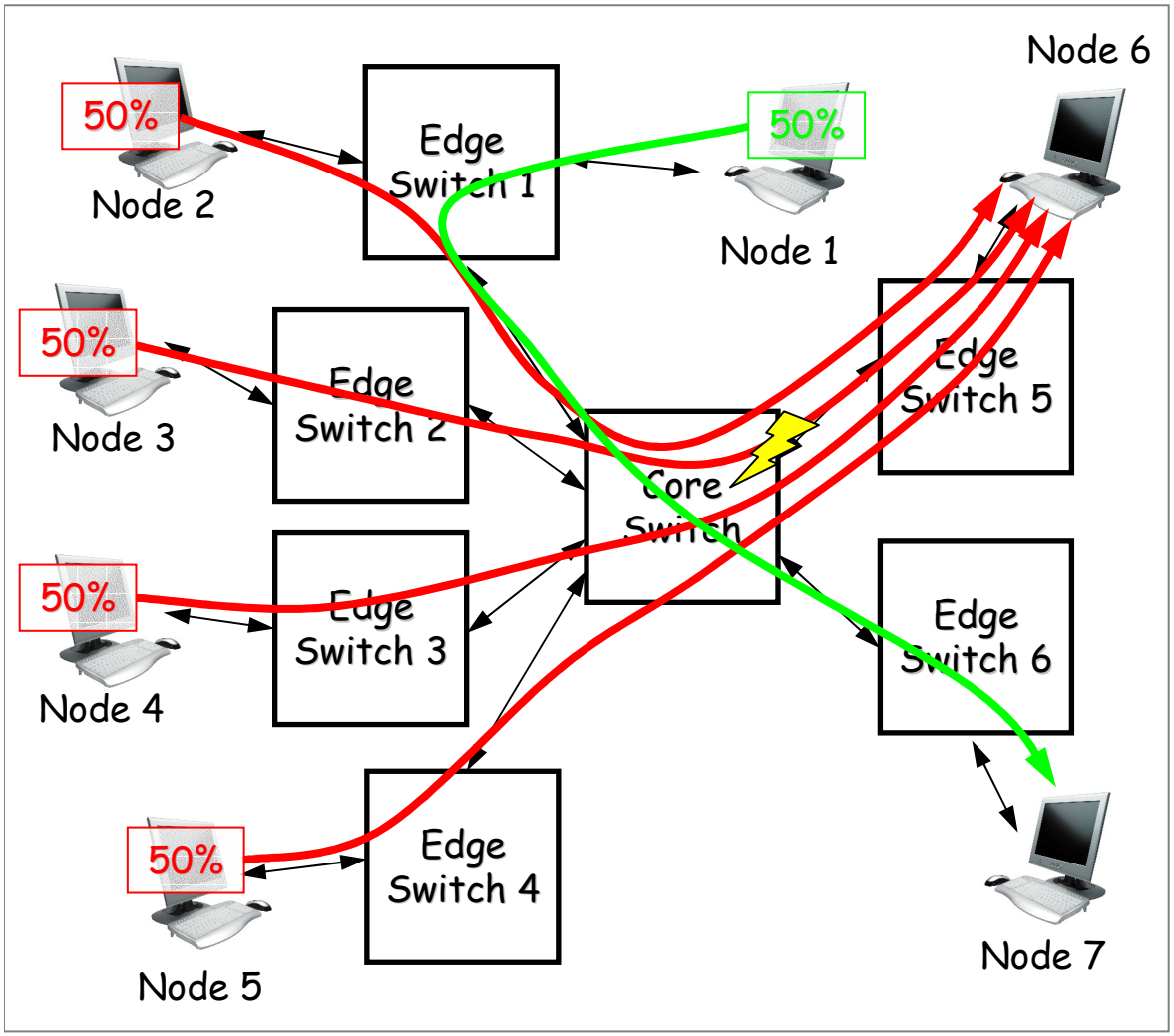




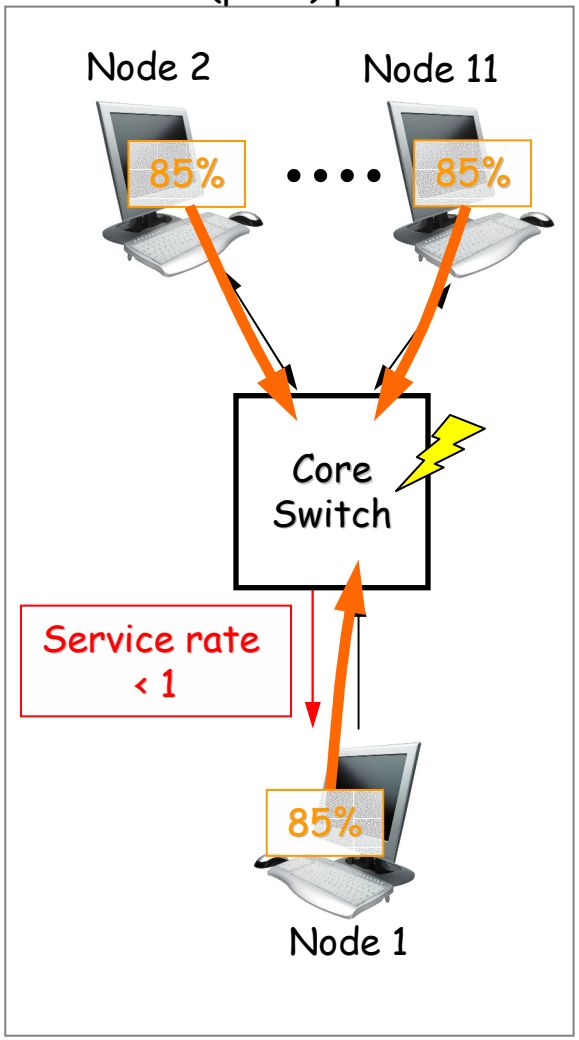
- From single hop and dumbbells (unidim. topo graphs) to 2D nets: a step up in realism (and complexity)
  - sim runtimes grow (super/sub)-linear

Hotspot Primitives: IG and OG

Input-generated hotspot ~ external disturbance



Output-generated HS ~ internal (plant) perturbation



FCT as Congestion Metric @ L2

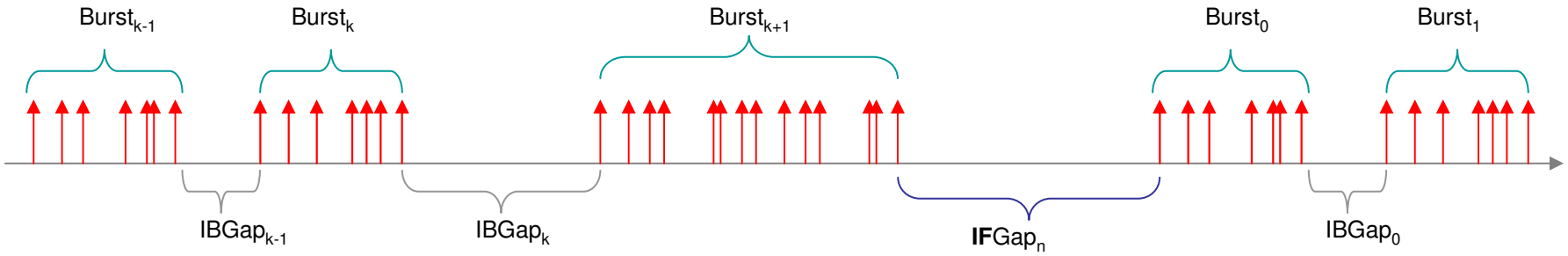
- FCT was recently proposed by Stanford Univ. for CM [refs]
  - "*FCT is an important – arguably the most important - performance metric for the user*" [N. Dukkipati, N. McKeown "Why Flow-Completion Time is the Right metric for Congestion Control and why this means we need new algorithms"]
  - FCT is being de-facto adopted also in .1au simulation results from Stanford, Cisco and ZRL
  - Characterizes CM performance from an User's perspective

- FCT: intriguing, yet difficult metric... It elicits precise
  1. Flow definition
  2. Completion definition
  3. Benchmarking measurement method

  ...none of which trivial !

What is a Flow on L2? You get what you measure...

- I) Assuming precise definition of "flow", measuring FCT results with PAUSE=On is un-ambiguous according to Case #1

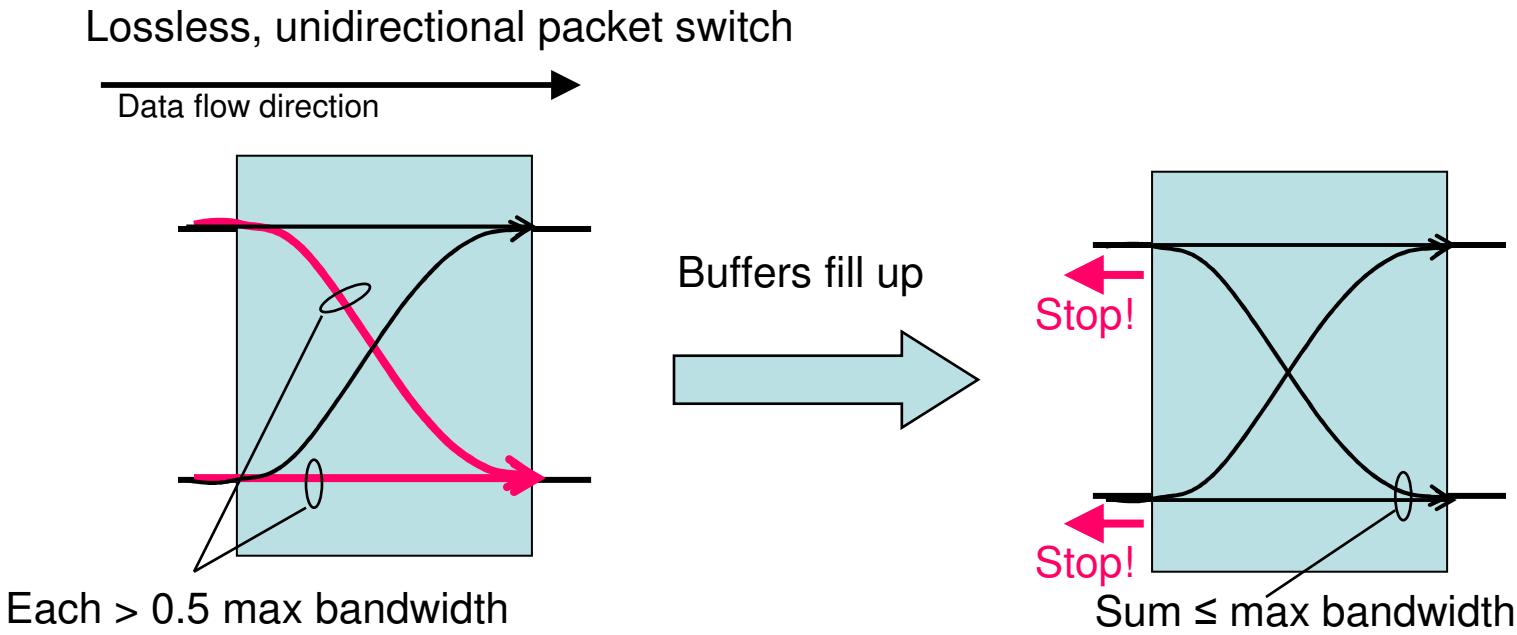


- II) However, with PAUSE = Off, FCT *also* requires definition of "completion"
  - flows entirely received w/o any loss
  - flows entirely received w/ some loss
  - flows partially received
  - flows not arrived yet at destination...

- How do we count for these?
- Traffic-driven
  - to get good Tput, just drop all small flows (mice)
  - to get good latency, just drop all large flows (elephants)
- We need an agreed upon FCT approach to fully capture the relevant statistics

IBA CCA

Source of the Problem*: 1 + 1 > 1

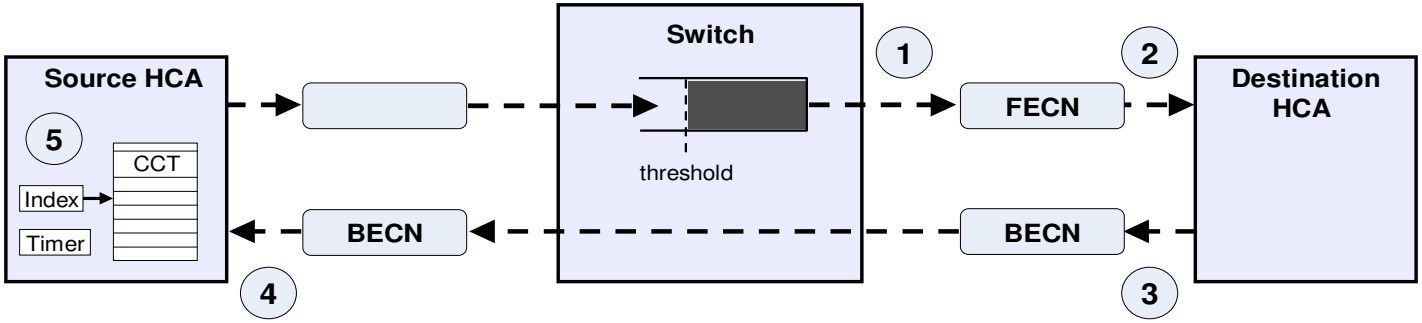


- Without global flow control that allocates bandwidth on every network link before starting transmission , the situation shown above can happen:
  - "Backpressure" to prior stage, e.g., by not sending back flow-control credits.
- This is the way link-level flow control is supposed to work.
- The problem:
  - In a multistage network, this can propagate.
  - Blocked (hot) flows usually prevent other (cold) flows from advancing, thus increasing their latency and wasting link bandwidth.

* As formulated by Greg Pfister in '03.

Spring'03: Modeling the IBA CCA

- IBTA's CM (aka CCA) already shaped



1. Load Sensor (LS): Q-occupancy;
2. Feedback (Fb): FECN; binary; single closed loop –Fb;
3. Source response function (SRF):
   1. down-rate ~ FECN IA;
   2. up-rate = timer-based self-recovery

Solution derived from TCP/ECN seems OK, but not quite…

- "Partial" [solution] because it leaves several unanswered questions:
  - How, exactly, does a switch decide it's congested?
  - How much does a source reduce its injection rate per BECN received?
    - For how many BECNs in a row?
  - How does a source go about re-increasing its injection rate when the congestion stops?
    - And how did it know that congestion stopped?

Few years and many sims later: Tuning Guidelines for IG Congestion

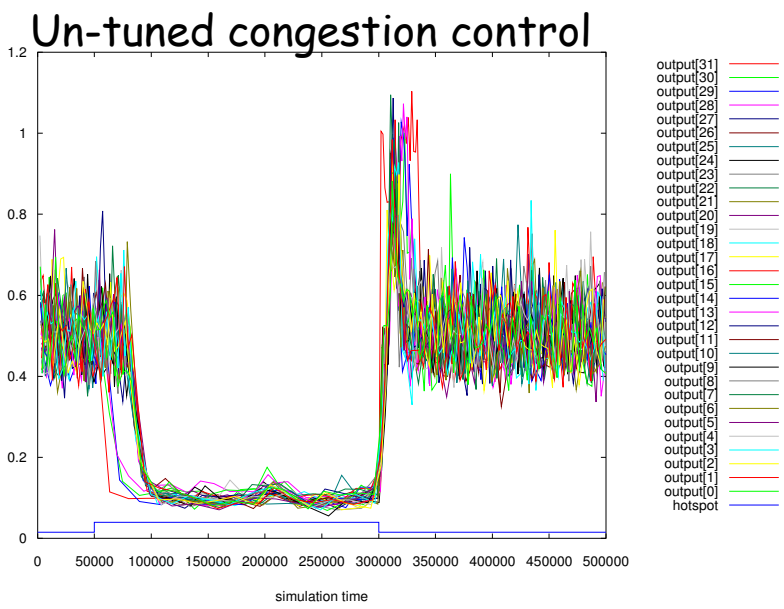
- N = network size, i.e., number of ports
- HSD = maximum hot spot degree
  - Worst case = N, but can be less if system partitioning known.
  - Absolute time (μsec) values are w/ reference to our model's RTT of 2-20 μsec. (w/o congestion)

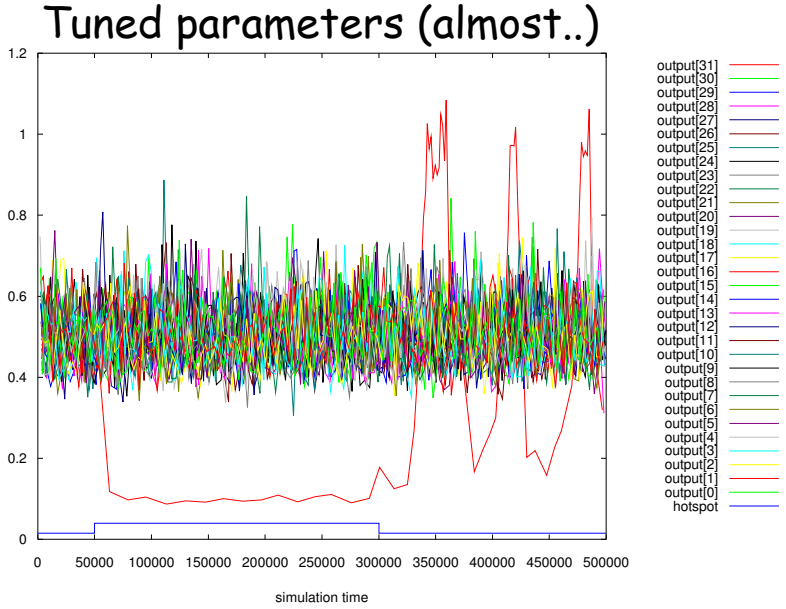
Item	**Value (for QDR IBA fabric with CCA)**
IPD Table size	128
Switch threshold for congestion detection	90% of queue capacity, with hysteresis
IPD table index increment	Min(1/6•N, 1/2•HSD)
Max IPD value	2/3 HSD μsec.
Recovery timer	10 μsec.

* See HPIDC'06 and HOTI publications on IBA CCA.

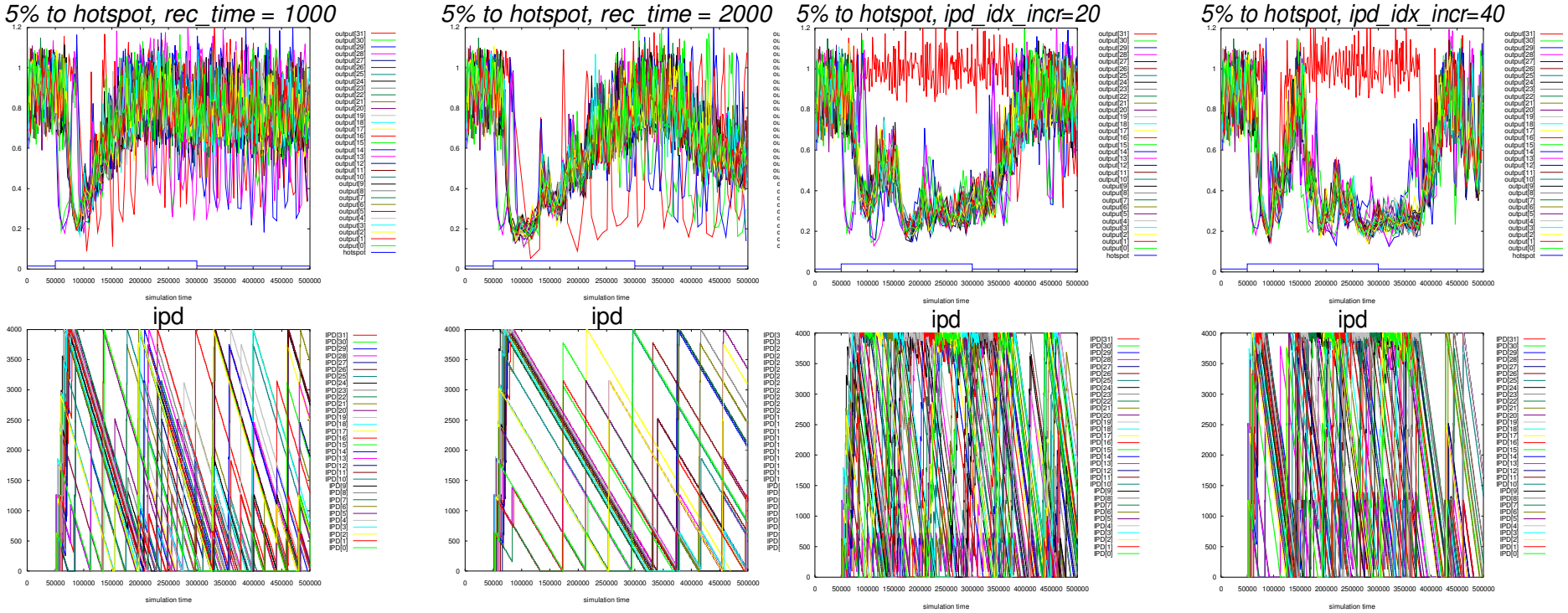
Does it work?

- Qualified "yes" => needs tuning
  - easy for small fabrics w/ simple traffic, hard for others...
- Param *tuning* required per (1) fabric architecture and (2) traffic
- Narrow stability: CCA sensitivity to (1,2) and params





High degree and severe HS at 90% load: Instability



- 3-stage 128-port network
  - HS degree = 128 (each source contributes 5% hot to HS + 90% unif. bgnd. load)
  - switch threshold *sw_th* = 90% of switch input buffer size
  - maximum IPD table contents *max_ipd* = 4000 (= 84μs)
  - IPD table index increment *ipd_idx_incr* = {20,40}
  - IPD recovery time *rec_time* = {500 (= 10.5μs), 1000, 2000}

FECN Issue in Lossless DCNs

- Q: Is the BECN interarrival frequency a reliable (error) signal?
- A: Not always - from a control systems point of view.

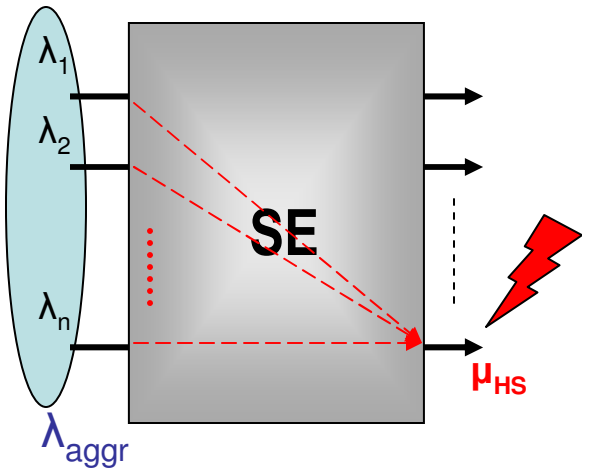
- **Hypothesis**: The FECN/BECN stream is *not a meaningful indicator* of congestion.

Proof by contradiction

1. A useful congestion signal is proportional to the problem at hand.

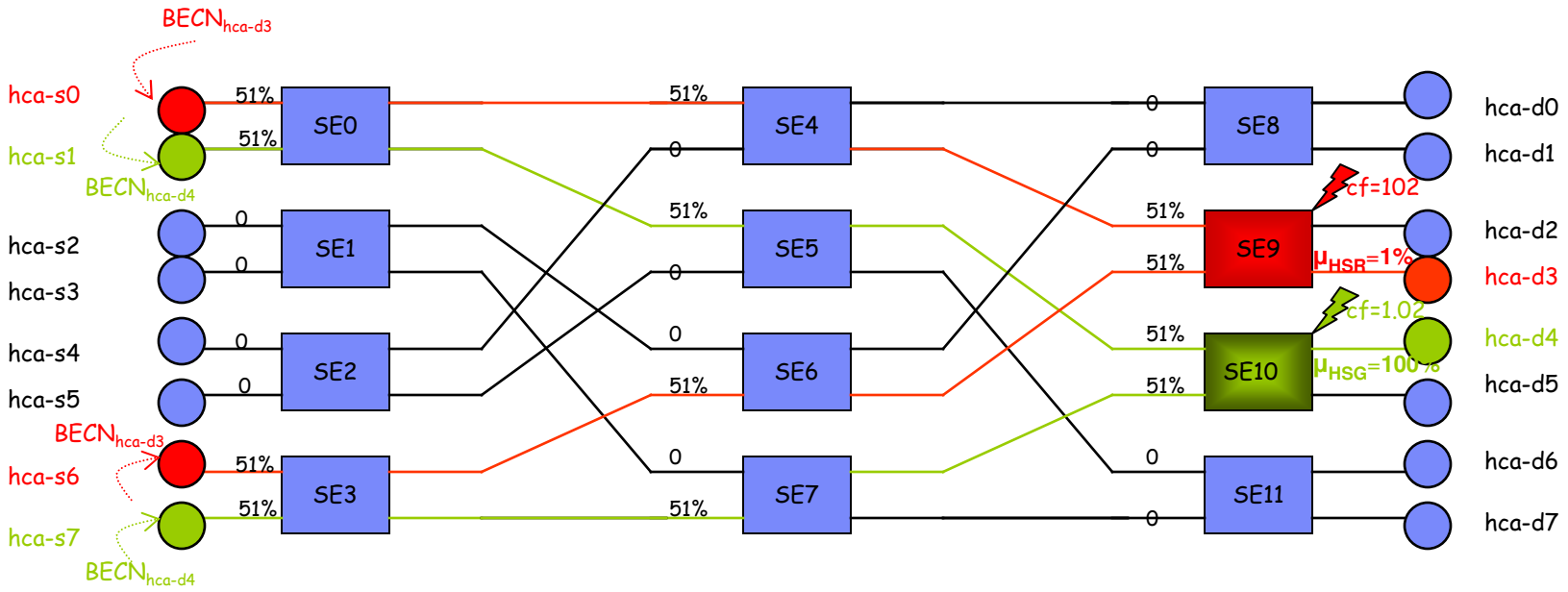
   Useful metrics that characterize the problem:
   - ***congestion factor*** : ***cf*** = λ_{aggr} / μ_{HS} , where
     $\lambda_{aggr} = \sum \lambda_i$ at hotspot (HS) site, and,
     μ_{HS} = *service rate of the HS*, i.e.,
     the drain rate of the bottleneck link;
   - *cf* = {*mild, moderate, severe, extreme*}
   - other factors: distance, contribution

2. One can construct counter-examples
   (when the FECN/BECN signal is not
   proportional to *cf*, aka the severity
   of congestion)



cf = λ_{aggr} / μ_{HS}

An Example to the Contrary: Two Simultaneous Hotspots



- Each hotspot, whether red or green, is *exclusively* fed by its 2 sources, each injecting at 51% (λ=0.51)
  => λ_{aggr} = .51 + .51 = 1.02 > 1 => hotspot / saturation tree (one red, one green)

- Due to the different drain* rates of hotspotted links, the respective congestion factors are
  $cf_R = \lambda_{aggr} / \mu_{HSR}$ = **1.02 / 0.01 = 102 (severe)**
  $cf_G = \lambda_{aggr} / \mu_{HSG}$ = **1.02 / 1 = 1.02 (mild)**

- Since the red congestion is 100x more severe than the green one, while their respectives sources are identical (in intensity, distance, topology, etc.), the corresponding BECN frequencies should be directly *proportional to the severity.*

- Expectation: $f_{BECNhca\text{-}d3} \sim 100 \times f_{BECNhca\text{-}d4}$
- Reality: $f_{BECNhca\text{-}d3} \sim (1/100) \times f_{BECNhca\text{-}d4}$

} Due to the FECN dependency on μ_{HS}, the discrepancy is *arbitrarily* large, e.g. O(10,000)

=> no proportionality. qed.

IBA CCA Parting Thoughts

- 1st generation of L2 CM schemes for DCNs
- With few exceptions (?), CCA has yet to deployed 10 yrs later
- As it is, the basic scheme is only punctually stable
  - Primal: switch load sampler exposed to the FECN 'issue'
  - Dual: non-linear SRF reaction in the end node.
- In the absence of a principled Rev. 2, CCA mandates laborious tuning per each workload.

QCN =
Congestion Point (CP) +
Congestion Notification Message (CNM) +
Reaction Point (RP)

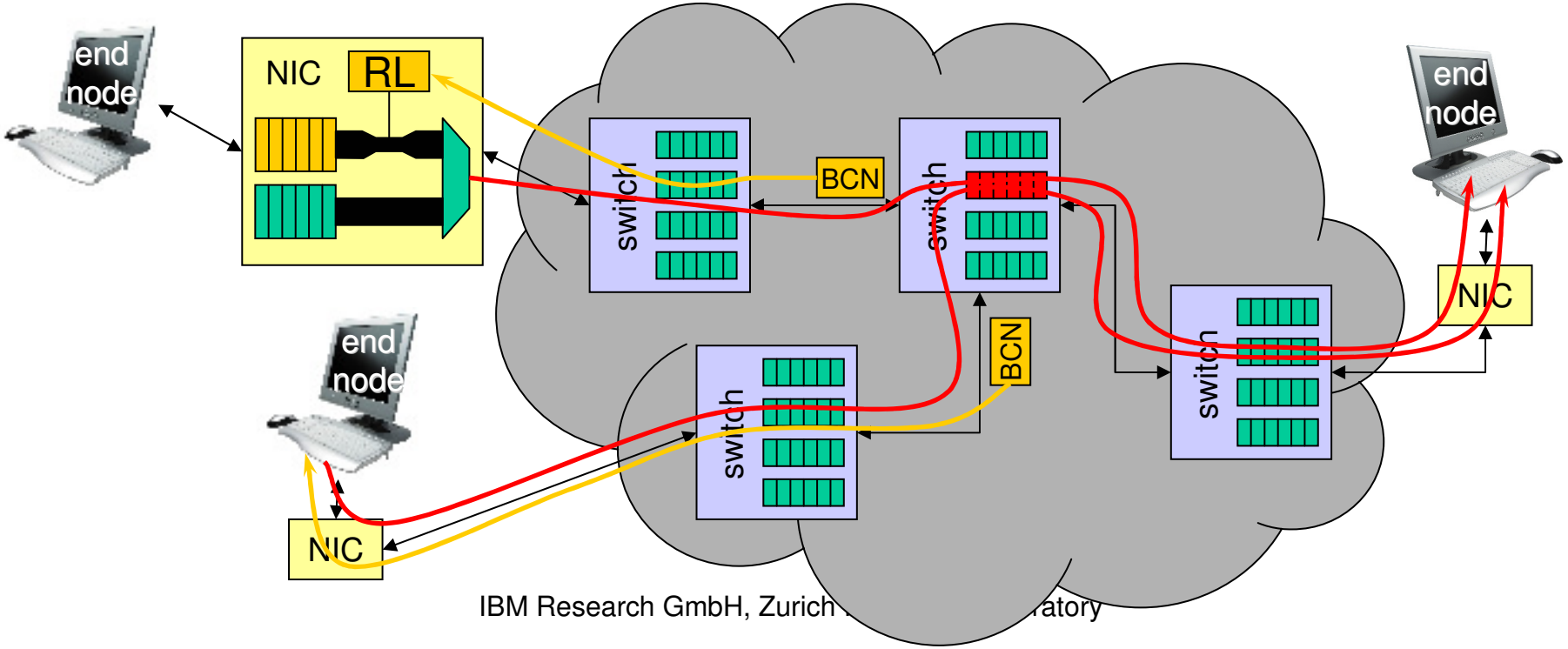
101 Intro to 802 DCB QCN

Ethernet Congestion Management Framework

1. Congestion point (CP)
   - Sampling: Observe metric indicating congestion level
   - Derive feedback value (e.g. by applying PID operations)

2. Feedback channel
   - Convey congestion notifications from CP to sources of "offending" traffic
   - Notifications contain congestion information, incl. a feedback value

3. Reaction point (RP)
   - Use rate limiters (RL) at the edge to shape flows causing congestion
   - Adjust rates based on feedback values received from congestion points



CNM Frame Format and Exemplary CNA Architecture

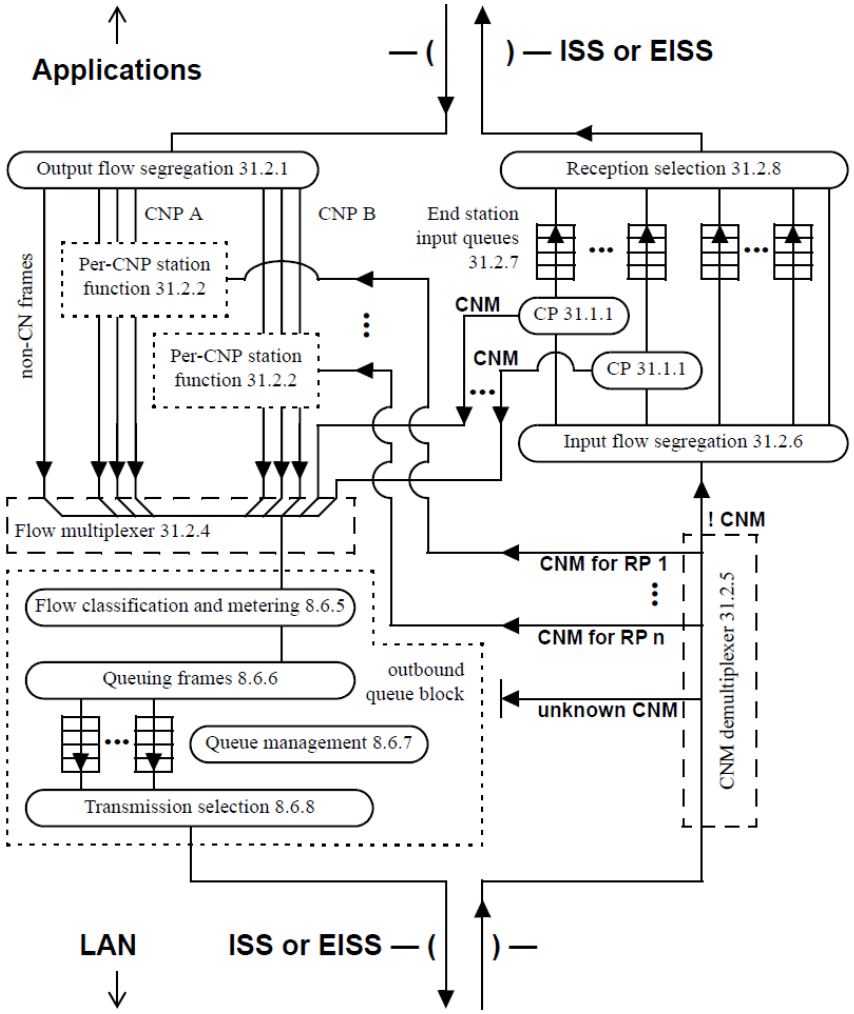


Figure 31-2—Congestion aware queue functions in an end station

Table 33-5—Congestion Notification Message PDU

	Octet	Length
Version	1	4 bits
ReservedV	1, 2	6 bits
Quantized Feedback	2	6 bits
Congestion Point Identifier (CPID)	3	8
cnmQOffset	11	2
cnmQDelta	13	2
Encapsulated priority	15	2
Encapsulated destination MAC address	17	6
Encapsulated frame length	23	2
Encapsulated frame	25	0—64

The 802.1Qau Approach

- Detection
  - Congestion point = output queue
  - Sample output queue length every *n* bytes received
  - Equilibrium length Q_{eq}
  - *Compute offset:* $Q_{off} = Q_{eq} - Q_{now}$
  - *Compute delta:* $Q_{delta} = Q_{old} - Q_{now}$
  - *Compute feedback:* $F_b = Q_{off} - W*Q_{delta}$
    $F_{b,max} = (1+2W)*Q_{eq}$

- Signaling
  - CP sends notification frame directly to source of sampled frame (BCN)
    - Source address = switch MAC
    - Destination address = source MAC of sampled frame
    - Feedback: Qoffset and Qdelta, F_b , or quantized F_b
    - Congestion point ID (depending on scheme)

- Reaction
  - Instantiate rate limiter; separately enqueue rate-limited flows
  - Reduce rate limit multiplicatively when $F_b < 0$
  - Increase rate limit additively when $F_b > 0$
  - Release rate limiter when limit returns to full link capacity

Ethernet Congestion Manager: ECM

- Explicit positive & negative feedback, multi-dimensional
  - BCN carries Q_{offset} and Q_{delta}
  - Fast recovery once congestion disappears

- Congestion point ID (CPID) signaling
  - Associate RL with CPID of most recent negative feedback
  - Filter out false positives at RP, i.e., if CPIDs do not match
  - Tag rate-limited frames with CPID to prevent generation of false positives by non-critical CP

- AIMD rate control
  - $F_b < 0 : \quad R := max(R*(1+G_d*F_b), R_{min})$
  - $F_b > 0 : \quad R := min(R + G_i*R_u, R_{max})$
  - G_i = increase gain, G_d = decrease gain, R_{min} = minimum rate R_{max} = line rate

QCN: Differences with ECM

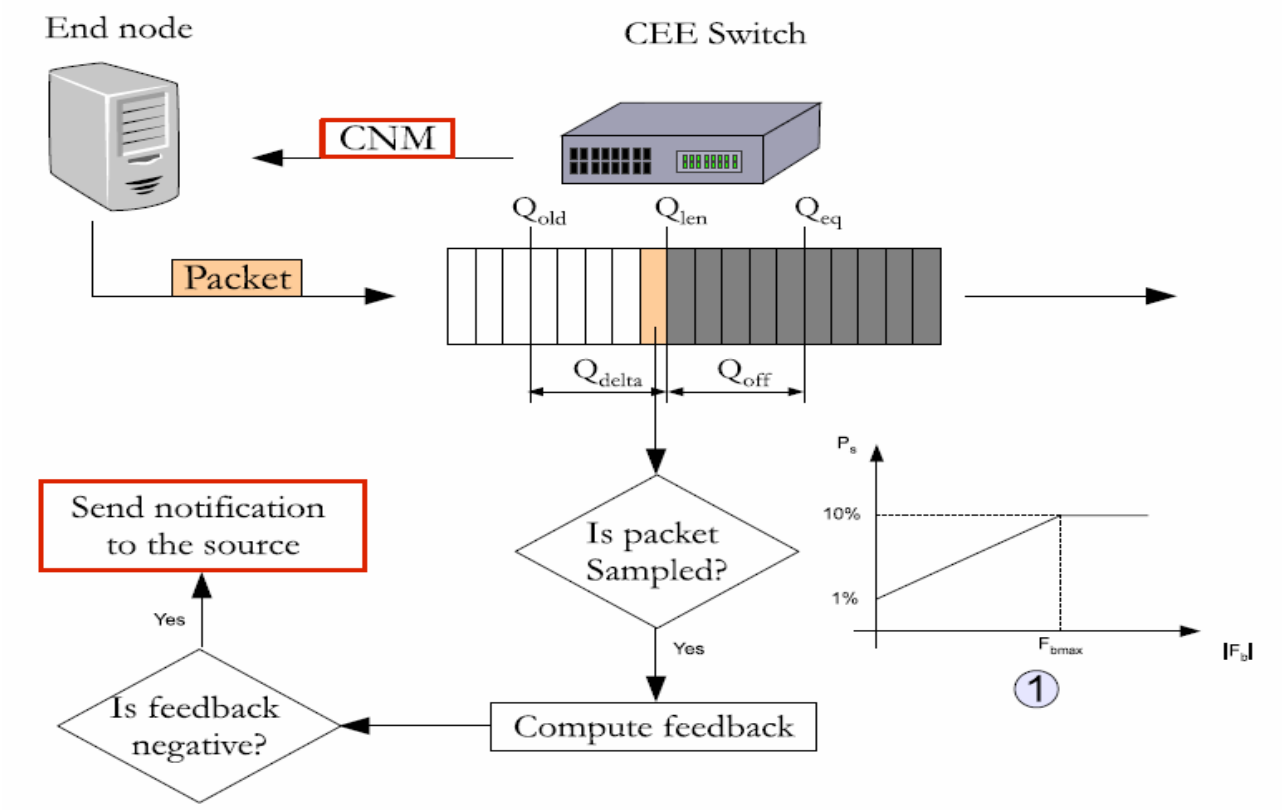
- Quantized feedback
  - F_b quantized with respect to $F_{b,max}$ using 6-8 bits
  - Lookup table instead of multiplication at RP
- Uni-dimensional feedback
  - BCN carries quantized F_b only
- Feedback-dependent sampling interval
  - Larger F_b ➔ smaller interval and vice versa
- Absence of explicit positive feedback
  - Autonomous rate increases based on amount of bytes sent
    - Two phases: Extra Fast Recovery (EFR), Active Increase (AI); increase rate at end of step *i* as follows:
      - EFR: Binary increase dependent on last rate decrease R_d: $R := R + R_d/(2^{i+1})$
      - AI: Linear increase by fixed amount: $R := R + R_i$
      - HAI: Hyperactive Increase (CUBIC-like grab for additional Bw)
    - Negative feedback received: go to EFR step 0 and remember (accumulate) rate decrease R_d
    - After *n* bytes sent, go to next step
    - After 5 steps of EFR, go to AI... and then to HAI
  - No need for CPID signaling or tags

QCN CP Mechanism

A) On packet arrival:

Oueue_length = queue occupancy level
Queue_delta = Queue_length – Queue_old
Queue_offset = Queue_eq – Queue_length

FB = feedback (Queue_delta, w, Queue_offset) (FB = Queue_offset + w * Queue_delta)
Ps = F(FB) → according to QCN sampling probability function
b = rand() / RAND_MAX (uniformly distributed in [0,1))
If (b > P) then sample packet

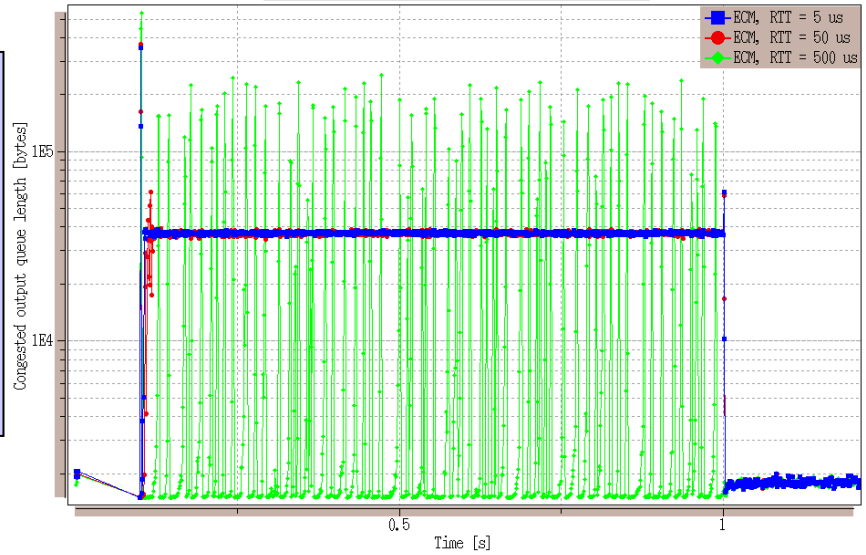


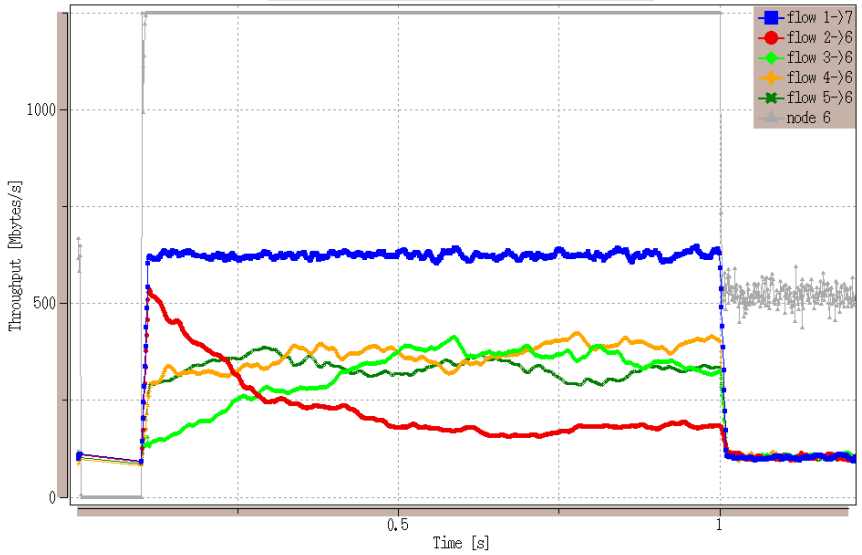
IG Hotspot Performance

Queue length

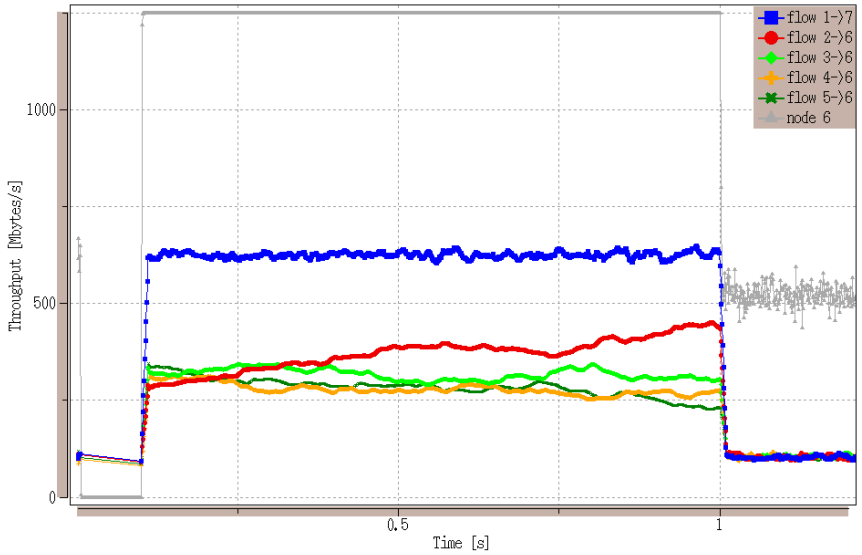
Throughput

ECM





QCN

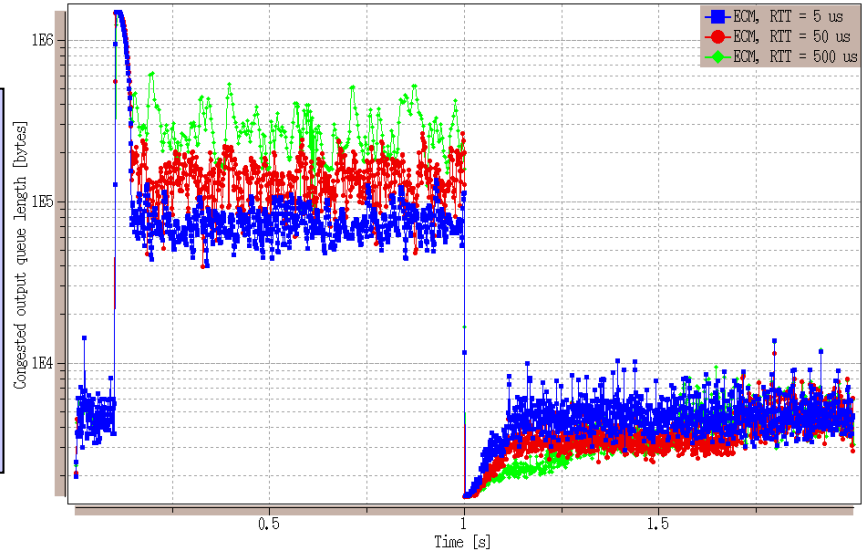


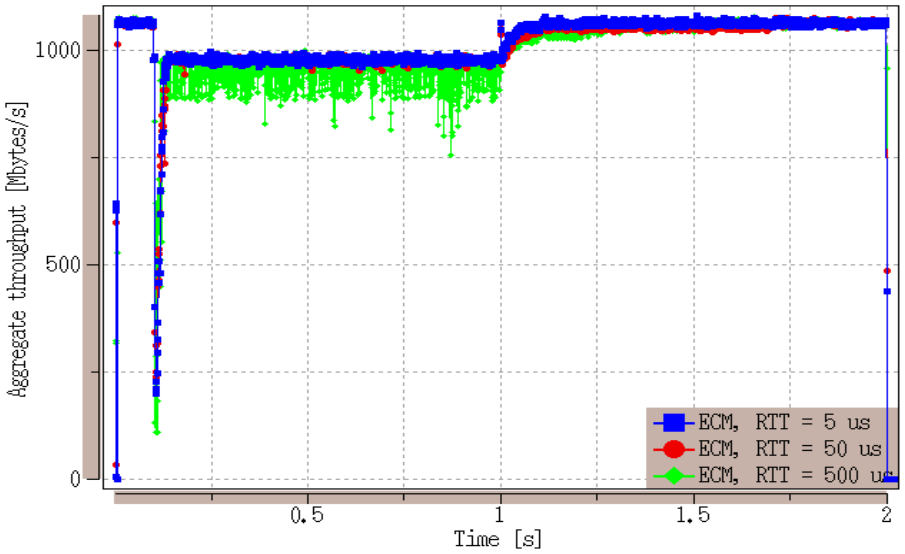
OG Hotspot Performance

Queue length

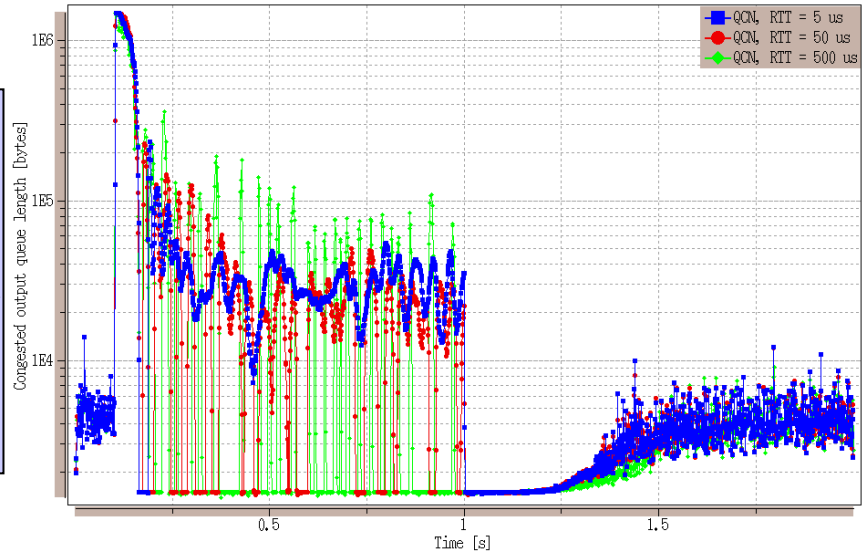
Aggr. throughput

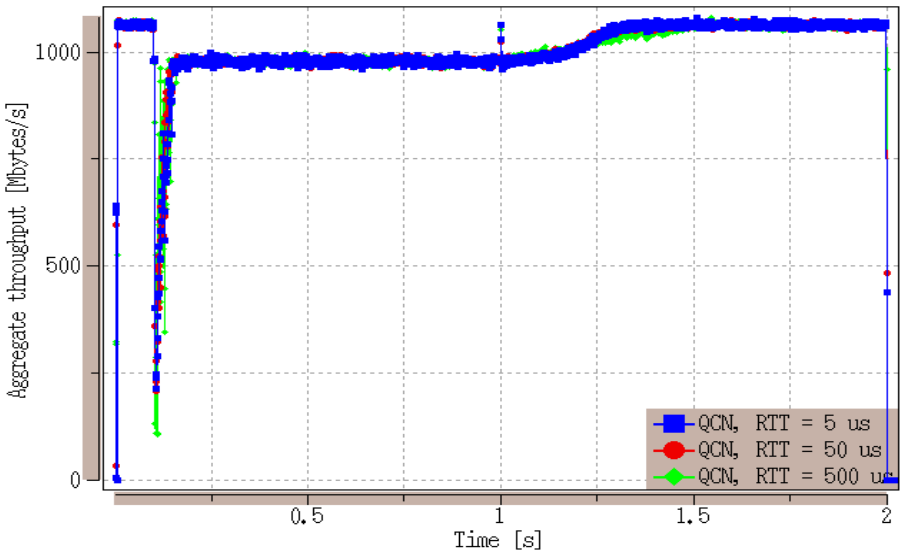
ECM





QCN





Room for Improvement

- Both schemes exhibit poor fairness

- ECM
  - Steady-state queue level and stability sensitive to additive increase gain in relation to target rate
  - Higher overhead

- QCN
  - Lack of spatial load information (congestion point ID) ➔ DONE (2010)
    - Required for adaptive routing and load balancing in many HPC and DC applications
  - Absence of positive feedback
    - Strong queue oscillations, especially with long RTT
    - Slow recovery
    - Improving recovery speed requires complex autonomous recovery schemes
  - Consolidated feedback
    - Loss of separate offset and delta (degree reduction)
    - F_b calculation in switch required

QCN Preliminary Conclusions

- Ethernet CM framework is based on
  - Pushing congestion to the network edge
  - Controlling the length of the switch output queues
    - Sampling switch output queue lengths, computing feedback based on queue offset and first derivative
    - Explicit backward congestion signaling from switch to traffic source
    - Rate limiting sources at network edge, adjusting rate based on feedback

- Basic principles have been shown to work
  - However, further performance improvements are needed
    - Both ECM and QCN have weak spots in terms of stability, performance and fairness

Thoughts on QCN

L2 vs L4 Congestion Detection and Marking

Alternative Schemes

IBM Research GmbH, Zurich Research Laboratory

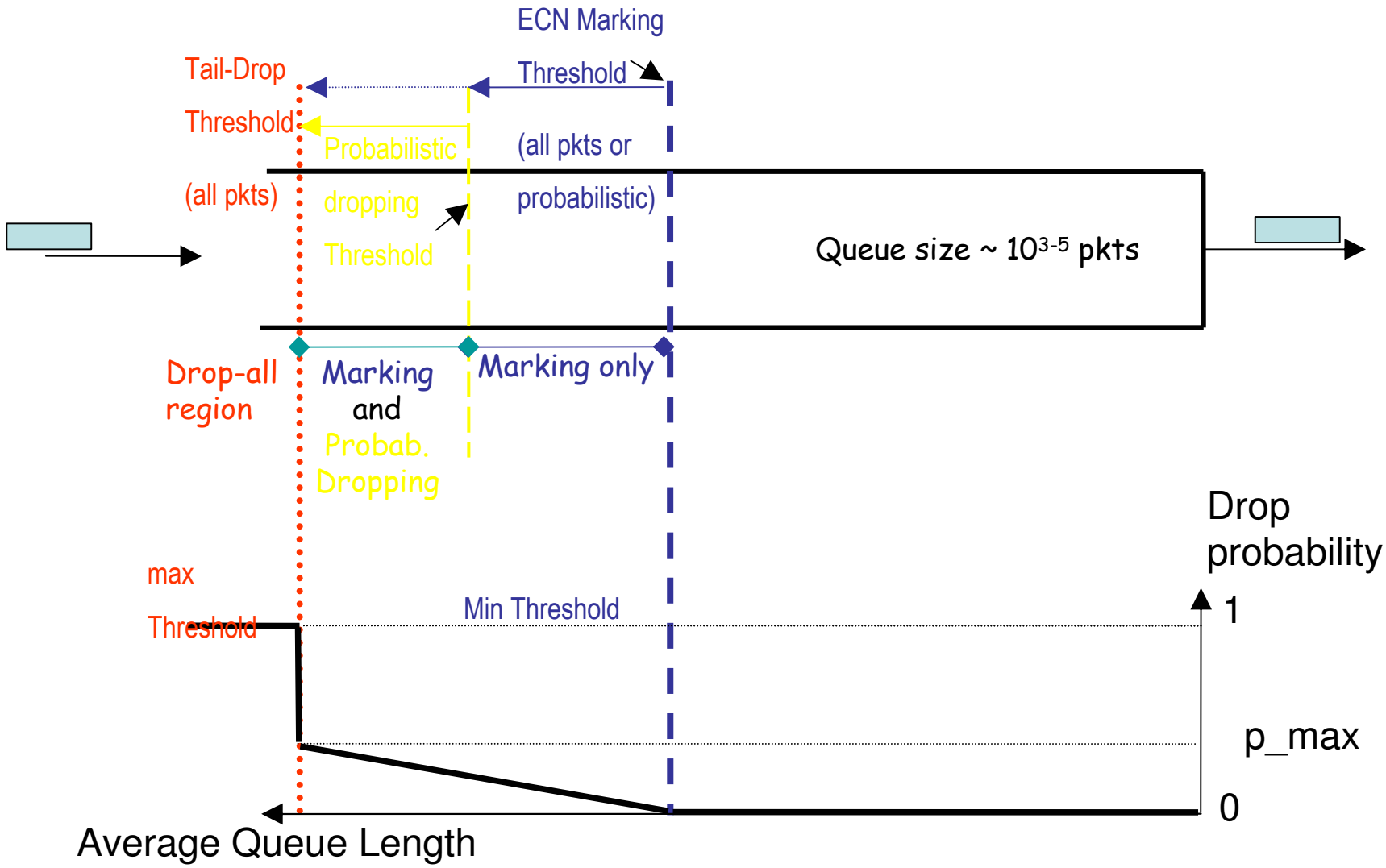
QCN Works, though Complex: Dozen+ Parms

Parameter	Value	Unit	Parameter	Value	Unit
TCP					
buffer size	128	KB	TX delay	9.5	μs
max buffer size	256	KB	RX delay	24	μs
default RTO	10	ms	timer quanta	1	μs
min RTO	2	ms	reassembly queue	200	seg.
RTO variance	20	ms			
ECN-RED					
min thresh.	25.6	KB	W_q	0.002	
max thresh.	76.8	KB	P_{max}	0.02	
QCN					
Q_{eq}	20 or 66	KB	fast recovery thresh.	5	
W_d	2		min. rate	100	Kb/s
G_d	0.5		active incr.	5	Mb/s
CM timer	15	ms	hyperactive incr.	50	Mb/s
sample interval	150	KB	min decr. factor	0.5	
byte count limit	150	KB	extra fast recovery	enabled	
PFC					
min thresh.	80	KB	max thresh.	97	KB
Network hardware					
link speed	10	Gb/s	adapter delay	500	ns
frame size	1500	B	switch buffer size/port	100	KB
adapter buffer size	512	KB	switch delay	100	ns

QCN Congestion Detection: L4 vs. L2

	L4 TCP (Reno)	L2 QCN
Detection Mechanism	1. @ destination (DupAck) 2. @ congestion point (AQM/ECN) 3. @ source (RTO)	@ congestion point (QCN sampler)
Feedback Type	1. Duplicate ACK (loss) 2. ECN/RED single-bit 3. Retransmission Timeout (latency)	Multi-bit: position, velocity
Burst Tolerance	Built-in	Depends on Q_{eq} threshold
Timescale	100s of **ms** (RTT dependent)	10s to 100s of **µs**

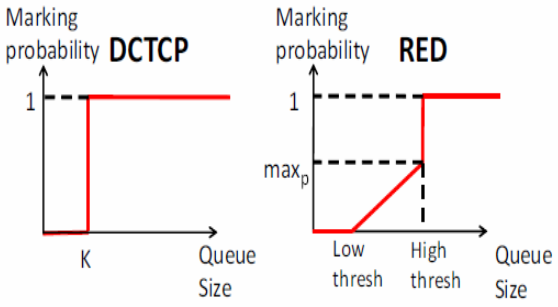
TCP/ECN Congestion Control in IP => deep buffers...
A View Inside a Congested Router's Queue: RED

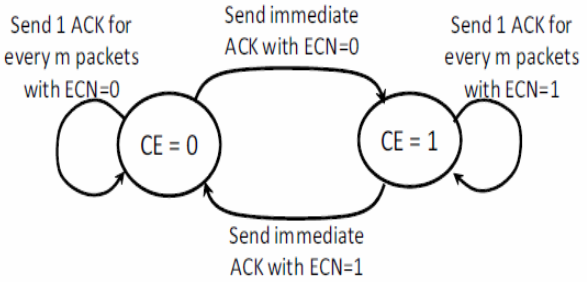


RED/ECN and DCTCP Marking

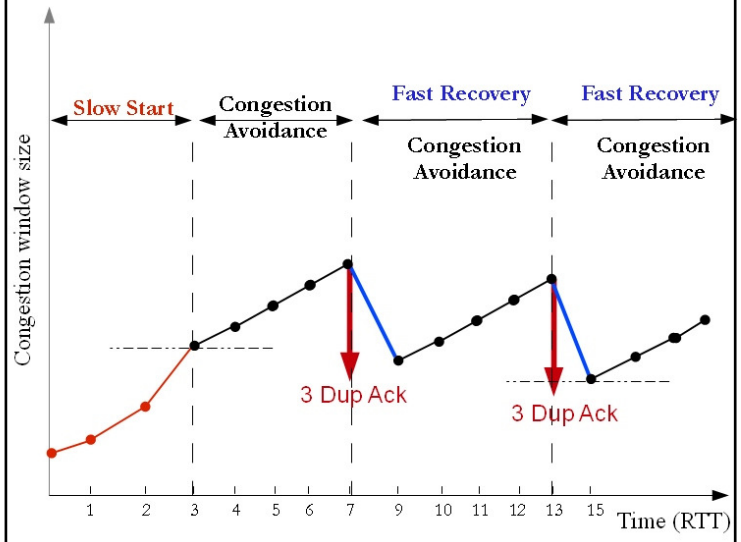
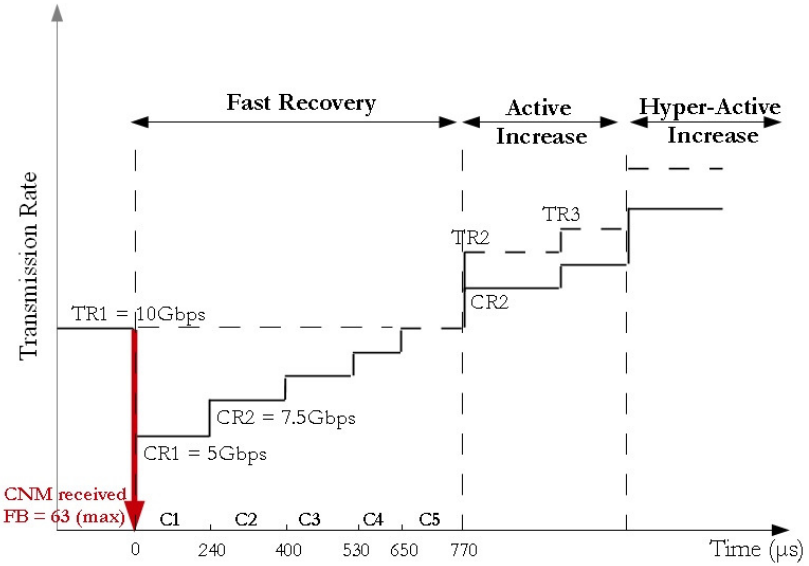
DCTCP: Packets are sampled with 100% probability. When the queue occupancy is above K, the packet is marked as RED/ECN does it today, i.e. by setting the CE code point. This marking continues as long as the queue occupancy is above Qeq [, and is RFC 3168 and 6040 compliant, to survive SDN tunnels].

This Feedback loop is fully ECN compatible - as it uses the ECE, CE and CRW code points between SRC and DST. It also incurs the full end-to-end RTT delays of RED/ECN, as any DCTCP-compliant scheme.

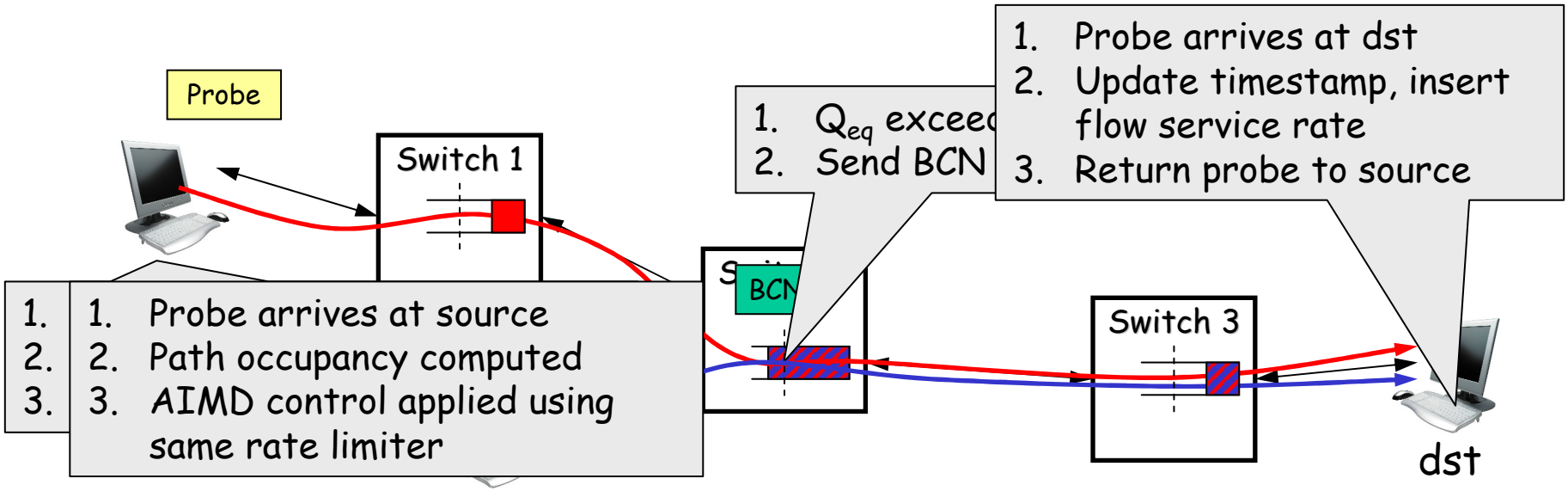




Congestion Control: L4 vs. L2

	L4 TCP (Reno)	L2 QCN
Principle of Operation	Window Controller @ SRC	Rate-based Controller @ SRC Finite State Machine
Increase & Decrease Control Law	Additive Increase Multiplicative Decrease (AIMD) 	Fb-proportional Decrease / Fast Recovery + Active Increase + Hyper Active Increase 

E2CM: Per-Path Occupancy Alternative



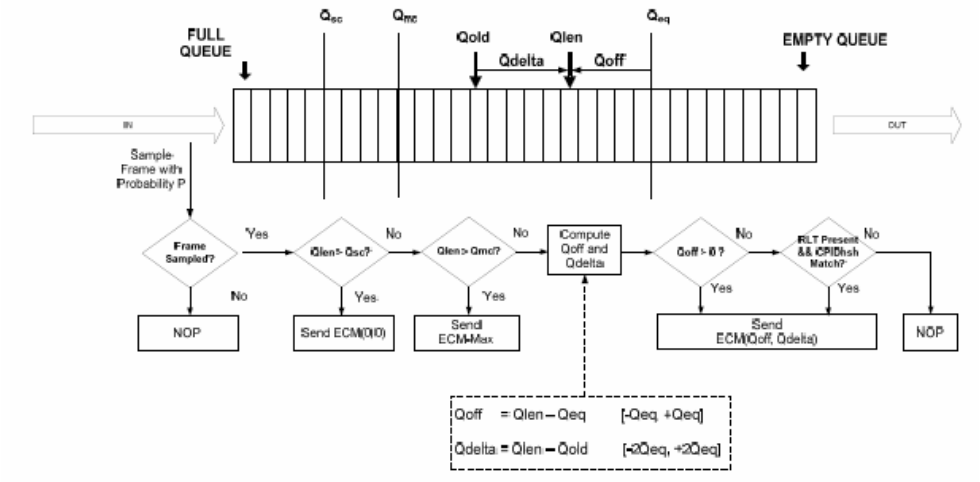
- Probing is triggered by BCN frames; only rate-limited flows are probed
  - Insert one probe every X KB of data sent per flow, e.g. X = 75 KB
  - Probes traverse network inband: Objective is to observe real current queuing delay

- Per flow, BCN and probes employ the same rate limiter
  - Control per-flow (probe) as well as per-queue (BCN) occupancy
  - CPID of probes = destination MAC
  - Rate limiter is never associated with probe CPID
  - Parameters re. probes may be set differently (in particular $Q_{eq,flow}$, $Q_{max,flow}$, $G_{d,flow}$, $G_{i,flow}$)

Ethernet's BCN Load Sensor Analysis

- Observation: Baseline BCN has robust performance in the linear region!
  - ➢ However, its dynamic range (DR) is limited by the queue capacity
  - ➢ Furthermore, possibly fed by n simultaneous arrivals…

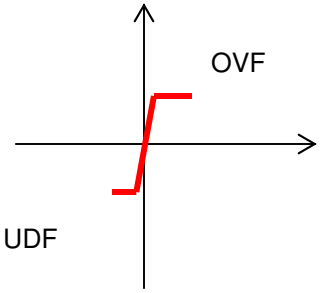
from ECM Spec



Saturated Integrator behavior…

$\Rightarrow$ Feedback: $F_b(t) = -(q(t) - Q_{eq}) + w*(dq/dt) / (\mu_j * p_s)$ =>

$0 \le q(t) \le q_{max}$

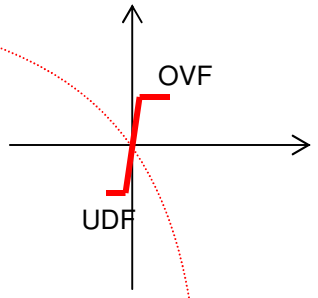


- ➢ Fast transition between lower/upper saturation (n+1 stochastic procs)
  - requires frequent use of saturation signals: BCN_Max, BCN(0,0)
  - non-linear saturation patches reduce the efficiency of the baseline control alg.

Extending the Linear Region of a Saturated Integrator

- How to scale BCN's stability properties w/ network size?
  1. Increase the dynamic range by chaining the j queues along the path i...
  2. Control the chain of queues instead of the individual queue

  ➢ **Introduce per path probing ...**



- Concatenate multiple queues along a path into a Path Queue -> Σq_{ij}

  ➢ State equations

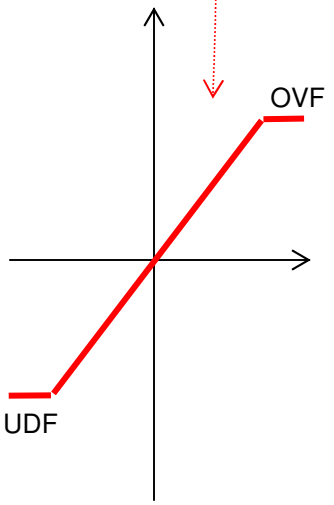
  From local queue stability to per path stability:

LQueue) $dq/dt = HSD*\lambda(t) - \mu_j$, where

$\max(HSD) = N$, and $\max(\mu_j) = C_j$

PQueue) $Q'_{ij} = \Sigma_i dq_{ij}/dt = \Sigma_i \lambda_{ij}(t) - \mu_j$, $1 < i \leq HSD$

Obs.: Slope steepness decreases: from n+1 to 2 stochastic procs



How about using QCN beyond just congestion control...?

I) QCN-based Switch Adaptive Routing

Cyriel Minkenberg[1], Mitchell Gusat[1], German Rodriguez[2]

[1] *IBM Research - Zurich*

[2] *Barcelona Supercomputing Center*

Congestion management vs. adaptive routing

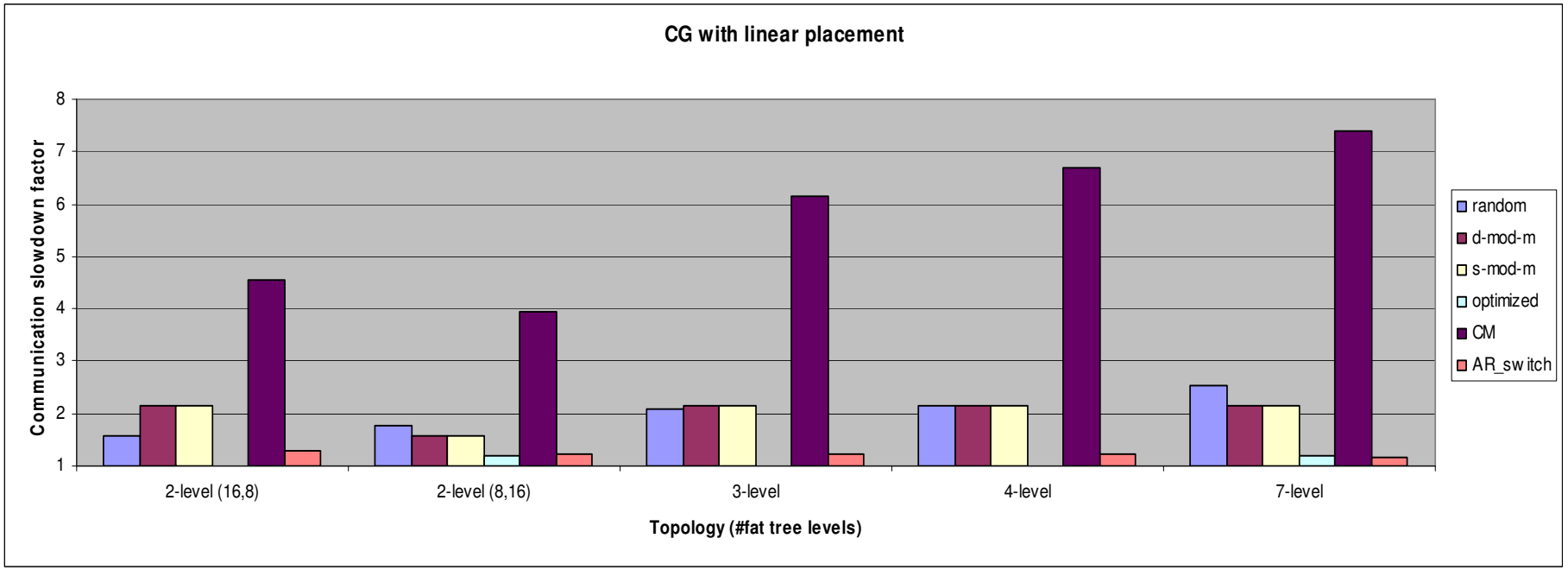
- CM solves congestion by reducing injection rate
  - Needs culprits and victims
  - Doesn't exploit path diversity
    - Typical DCN topologies offer high path diversity
      Fat tree, mesh, torus
- Adaptive routing (AR) approach
  - Enables multipath routing
  - Default route: Shortest path (latency and determinism)
  - Detect downstream congestion by means of BCN/CNM
  - If congestion
    1. Try to reroute hot flows on alternative paths
    2. Iff no uncongested alternative exists ➔ Reduce send rate

Dynamic adaptive routing based on QCN

- Concept
  - Upstream switches **snoop** congestion notifications,
  - **Annotate** routing tables with congestion information, and
  - **Modify route selection** to fwd on the **least congested** port among those enabled for a given destination
- Routing table
  - Maps a destination MAC to one or more switch port numbers, listed in order of preference, e.g., shortest path first
- Congestion table
  - Maps a key <destination MAC, switch port number> to a congestion entry comprising the following information:
- Receiver checks frame order and performs resequencing if needed

field	type	meaning
congested	boolean	Flag indicating whether port is congested
local	boolean	Flag indicating whether congestion is local or remote
fbCount	integer	Number of notifications received
feedback	integer	Feedback severity value

Results (1 of 3) – CG with linear task placement



- CG with linear task placement
  - Random, d-mod-, s-mod-m all perform about equally well
  - Colored is almost ideal
  - CM by itself is terrible: rate limiting online delays without solving congestion
  - AR performs significantly better than oblivious algorithm (25 to 45% better than d-mod-m; within 25% of ideal)

Conclusions: QCN-based SwitchAR

- IEEE 802.1Qau framework
  - QCN defines **congestion management** for CEE / DCNs
  - provides **temporal** reaction: selectively rate-limiting flow sources
  - Can also harm performance for HPC and DC workloads
  - Provides sufficient **hooks** to implement fully **adaptive routing**

- Snoop-based adaptive routing (QCN-based sAR)
  - snoops QCN notifications to annotate routing tables with congestion information
  - takes advantage of **multi-path** capabilities
  - **reduces latency** and **increases saturation throughput** under uniform traffic
  - efficiently reroutes specific flows in case of congestion
  - interoperates seamlessly with underlying CM scheme
  - good results for HPC workloads

Ditto, Source-driven...?

II) R^3C^2: Reactive Route & Rate Control for CEE

QCN-based Source-Adaptive Routing

Mitch Gusat, Daniel Crisan,
Cyriel Minkenberg, and Casimer DeCusatis

IBM Research GmbH, Zürich Research Laboratory

Introduction & Motivation

- Existing congestion mgmt schemes for CEE
  - Rate-only: Quantized Congestion Notification
    - Rate/window reduction don't benefit trading, HPC, BA apps
  - Route-only: Switch Adaptive Routing
    - Based on QCN's load sensor
    - Exploits path diversity

- Proposal: $R^{3/2}C^2$
  - Dual Route & Rate control
  - Source-driven

$R^{3/2}C^2$ Concept

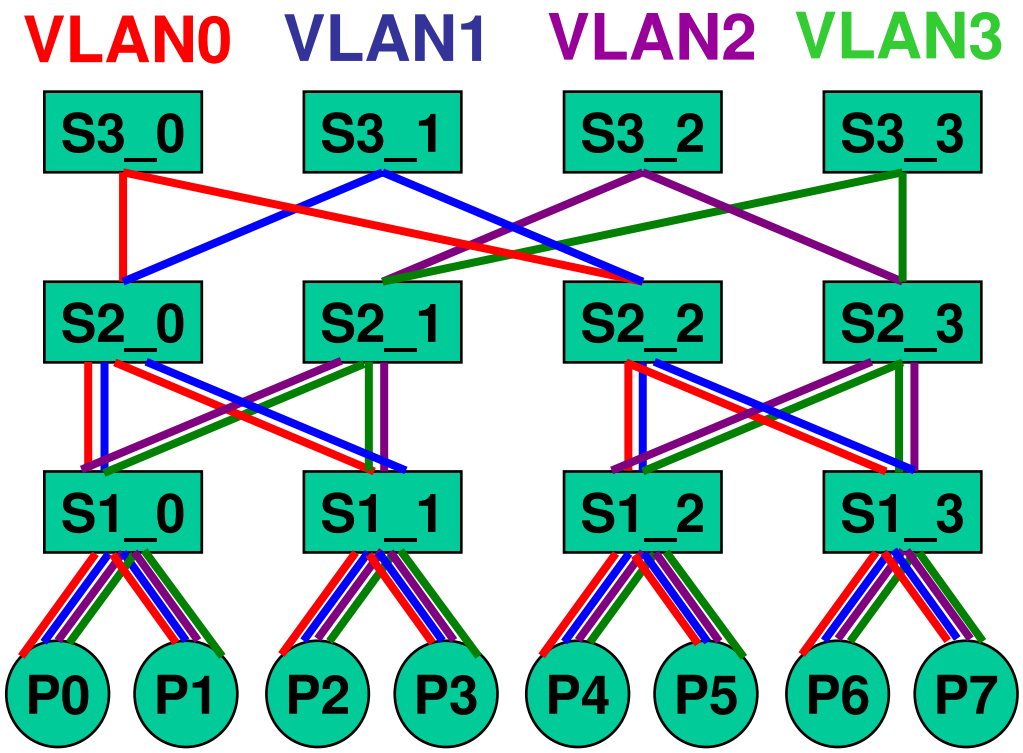
Take advantage of CNMs at the source for adaptive load-balancing

- **Congestion Point** issues CNMs
  - Where is the hotspot?
  - How severe is the hotspot?

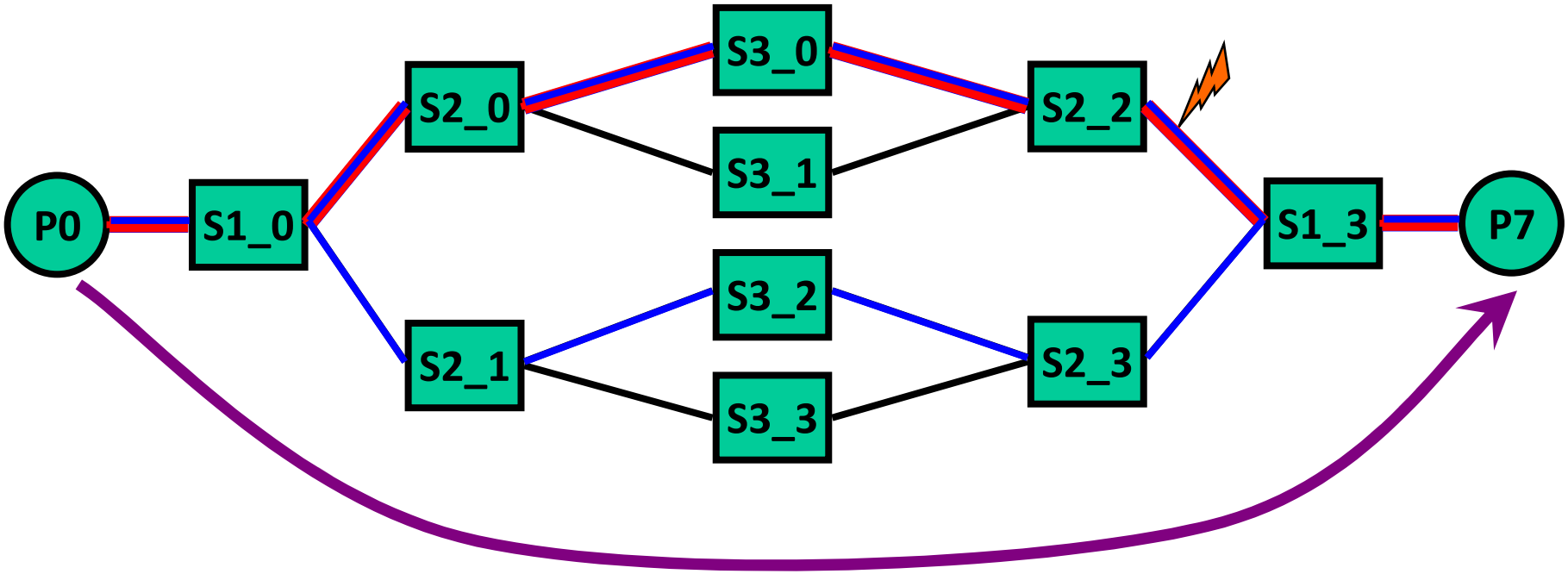
- **Source** receives the CNMs
  - Identifies the most severe hotspots
  - Reroutes traffic around the hotspots
  - Splits flows and rate-limits subflows

Source Routing in CEE: VLAN

- Plain Ethernet is not source-routed
- Solution: VLAN
  - One tree per VLAN
- Source
  - Set VLAN# at injection → path selection

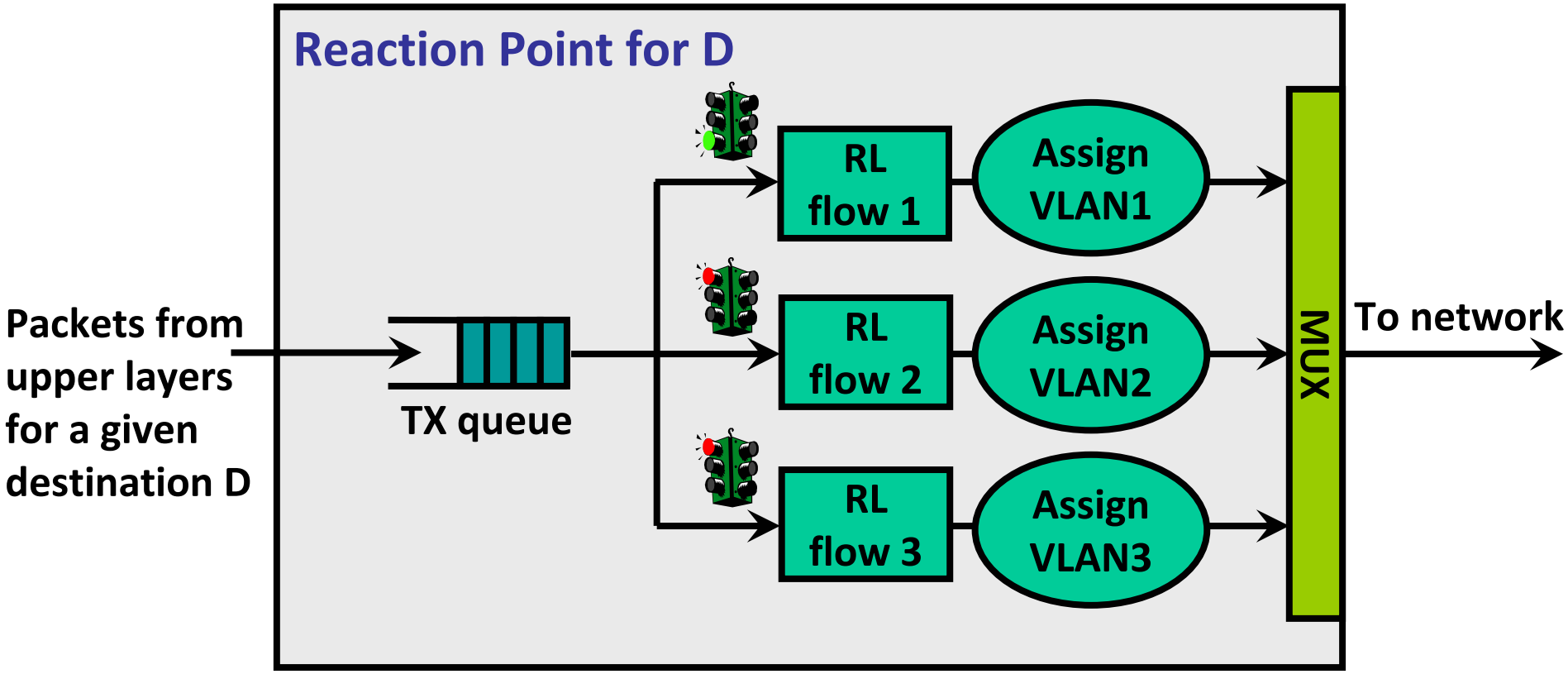


$R^{3/2}C^2$ Algorithm



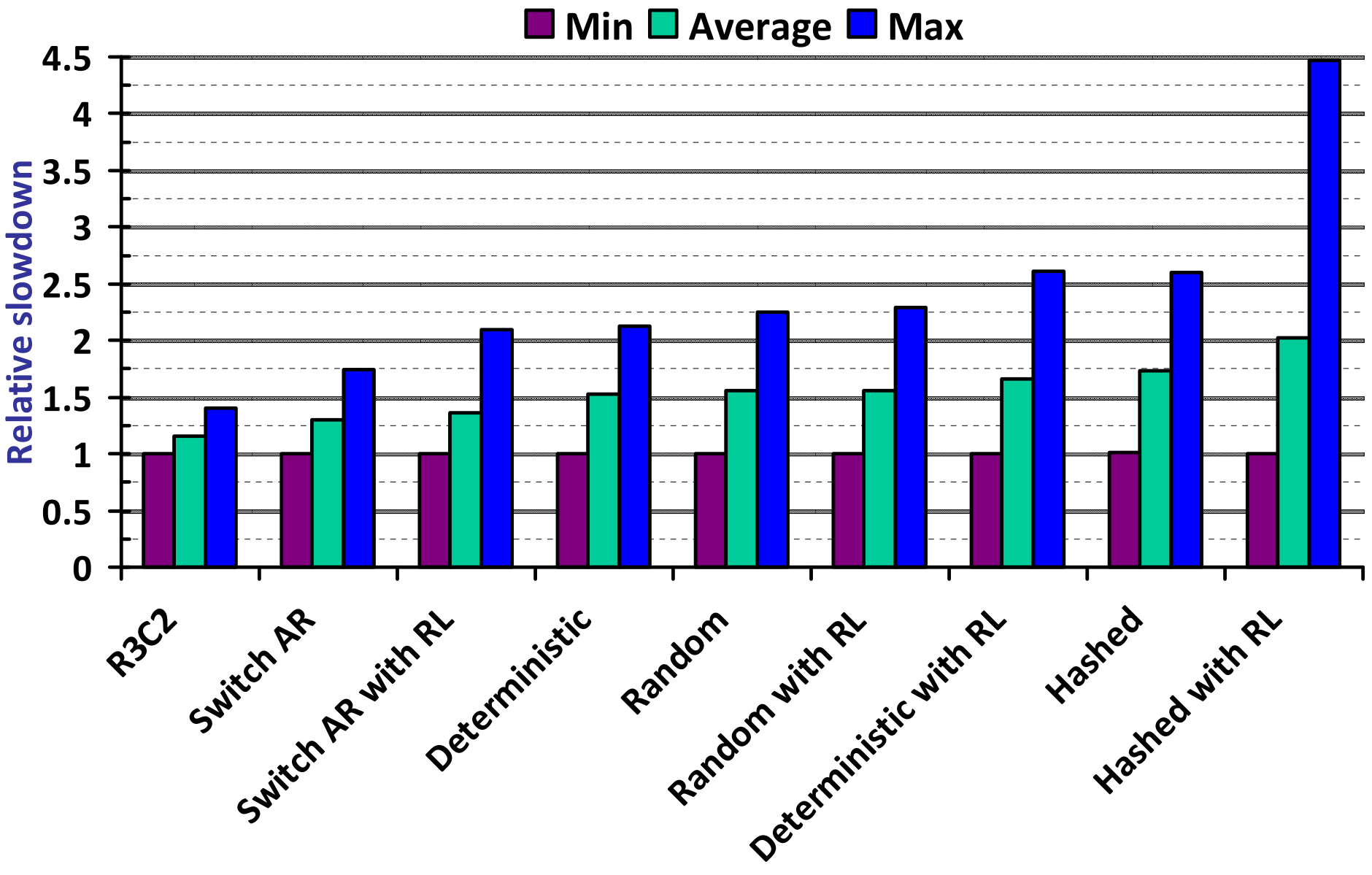
- No overload: Deterministic single path (shortest)
- Congestion: Activate additional paths
- Path activation: avoid hotspots
- Use RL along each path

$R^{3/2}C^2$ Reaction Point



- Packet assigned the VLAN# of the 1st eligible Rate Limiter

HPC Traces (9 Benchmarks) with Hotspots



Conclusions (QCN-based SRC-AR)

- Introduced $R^{3/2}C^2$
  - Source-driven adaptive routing scheme
  - QCN and VLAN are key
- Dual Route & Rate control
  - Improved stability and performance
    - even better than our previous Switch AR scheme
- Performance benefits
  - 80% over Deterministic
  - 40% over Random

QCN Parting Thoughts

- 2nd generation of L2 CM
- Implemented in most high/med. end current CEE fabrics 10/40/100G
- Significantly more principled and stable than CCA
  - for a QCN-CCA comparison, see Gilles Cressier's EPFL Master Thesis 2013 (attached)
- It has its own limitations
  - complexity
  - stability with # of flows
  - fairness (addressed my multiple papers, attached)

- Surprisingly versatile for OTHER purposes
  - load balancing, adaptive routing
  - DCN monitoring

BKUP
Appendix B

Queues will swing with large RTTs. Must they?

- Yes - with uncompensated CM loops...
- Q swings can be treated as symptoms or cause, hence
- Compensation methods
  - A) Implicit: extend the stability region thru state space => q, q', q",....
    - ✓ issue is how to add new 'describing' equations to the ODE system
      - measurement
      - noise: d(d(q)dt)/dt => high pass filter is bad... low pass adds more lag to q
  - B) Explicit: dynamic compensation
    - ✓ live demo

Why High Slope SRF? Better Tracking...

"we believe that by the time that much higher bandwidth (in the order of 100Gbps) becomes available, the network must have more advanced AQM schemes deployed so that we can use a higher slope response function.

A less aggressive (i.e., lower slope) protocol, however, is less responsive to available bandwidth; so short file transfers will suffer.

We believe that fast convergence to efficiency requires a separate mechanism that detects the availability of unused bandwidth which has been an active research area lately. We foresee that advance in this field greatly benefits the congestion control research." Injong Rhee et al. "Binary Increase Congestion Control for Fast Long-Distance Networks", INFOCOM 2004

Fluid "Modeling" of a Queue

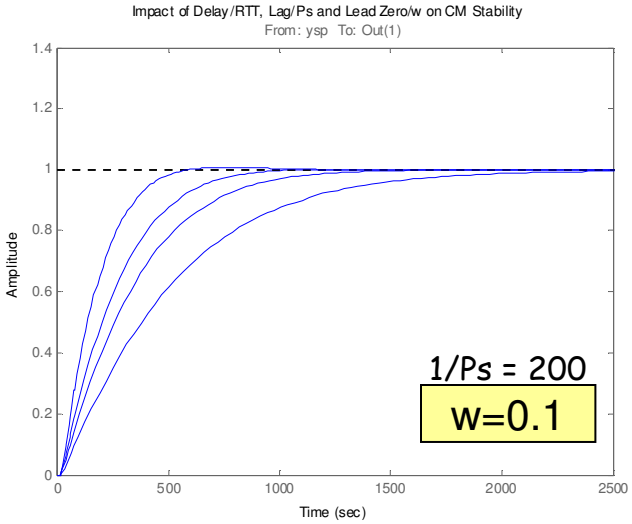
- What is a queue?
- A mismatch integrator between IN and OUT rates $Q = \int_0^\tau (r(t) - c(t))dt$
  - ➢ hence a low-pass filter is already in, and...
    - ✓ a phase lag to be compensated
      => intuitively adding lead zero(s) could help

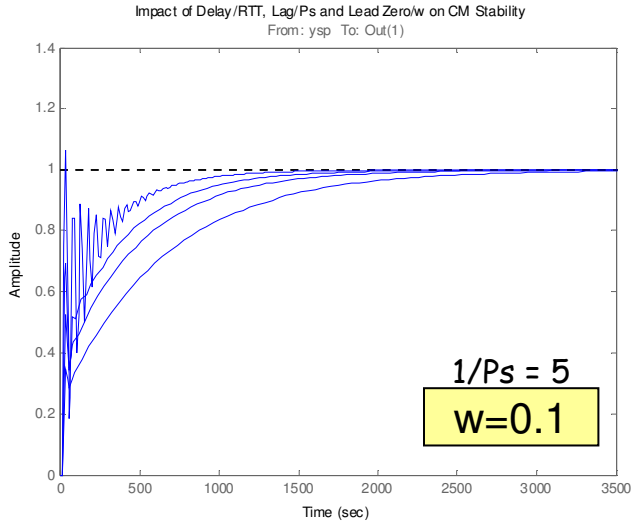
- AQM's conservation equation: dq(t)/dt = N*r(t) – c(t)

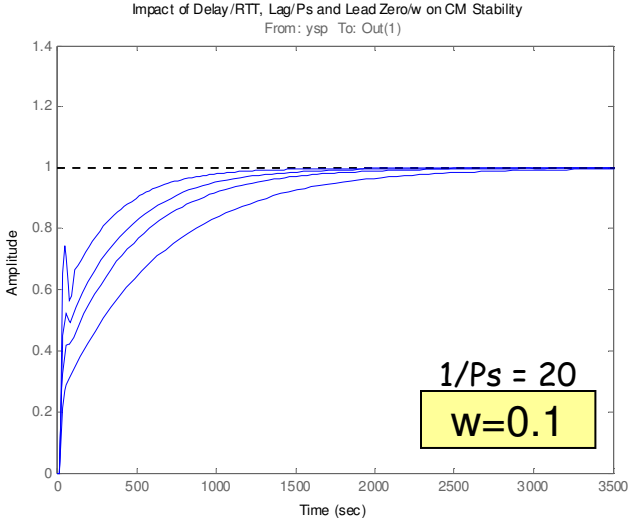
- Water glass contest: Fastest 'perfect' fill !

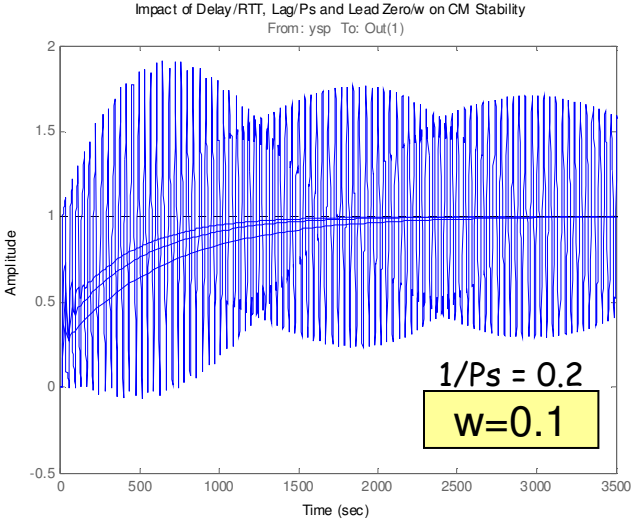
1. open loop
2. closed loop
   1. low delay
   2. high delay

Impact of variable sampling frequency Ps @ constant RTT (=19)







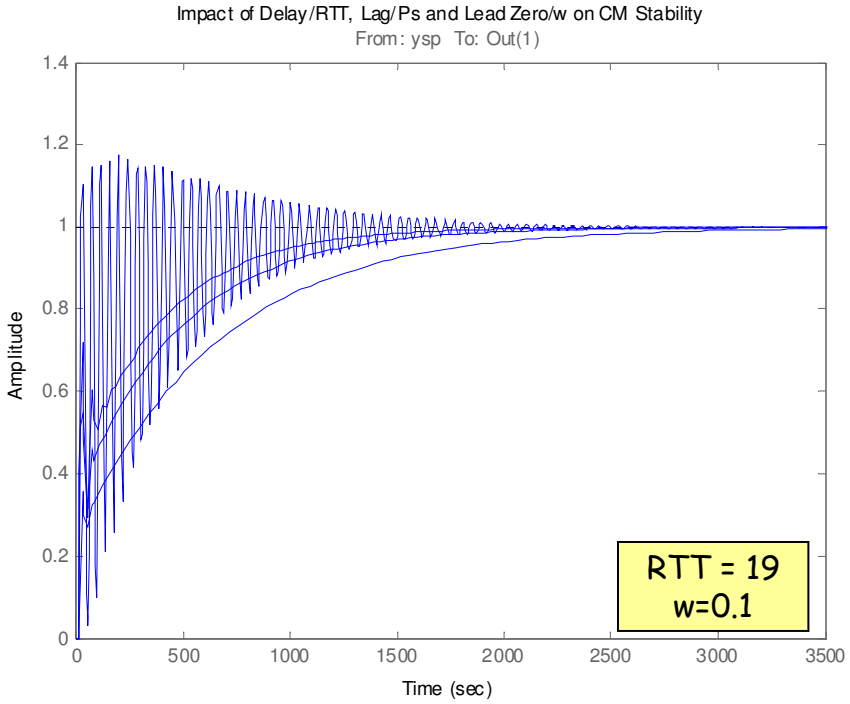


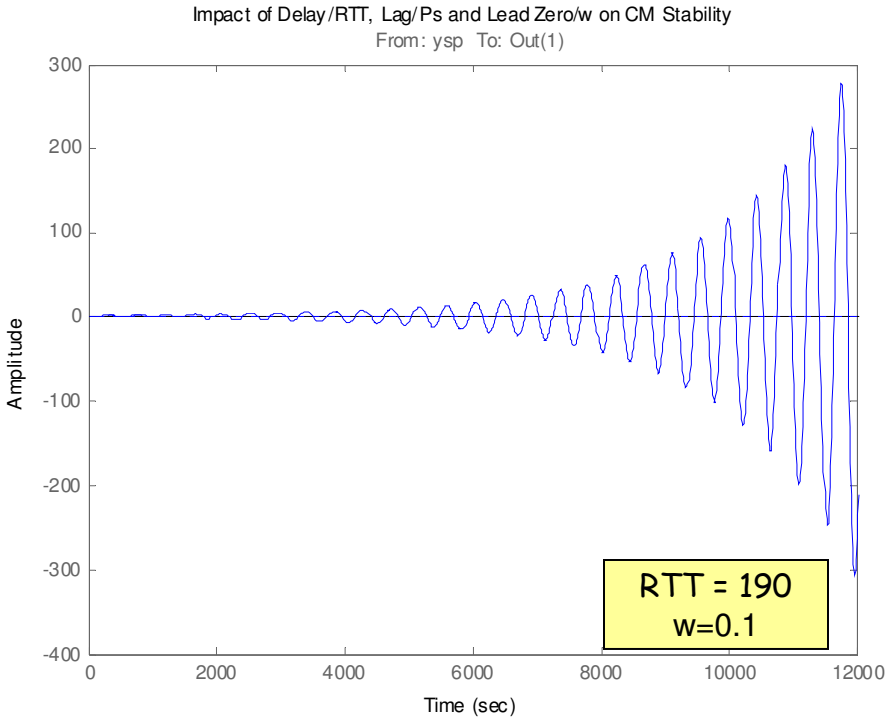
Why QCN's Adaptive Sampling Depends on RTT probing?

- Observations
  1. Whenever delay exceeds sampling lag the loop becomes unstable
     1. Hence the intrinsic conflict between increasing P_s and delay stability
     2. No clear trade-off is possible w/ RTT knowledge
  2. Sampling is aggregate @ CP, while F_b is per flow @ RP
  3. CP does not know RTT, nor "n" (# flows)
  4. Flooding RPs w/ bursts of outdated feedback requires adaptivity
     1. near RP's benefit directly from an increased P_s
     2. remote RP's don't... (must filter - decimation, Kalman)

see "Effects of long RTT [and P_s] on QCN"

Impact of variable RTT @ constant sampling $1/P_s$ (= 2)

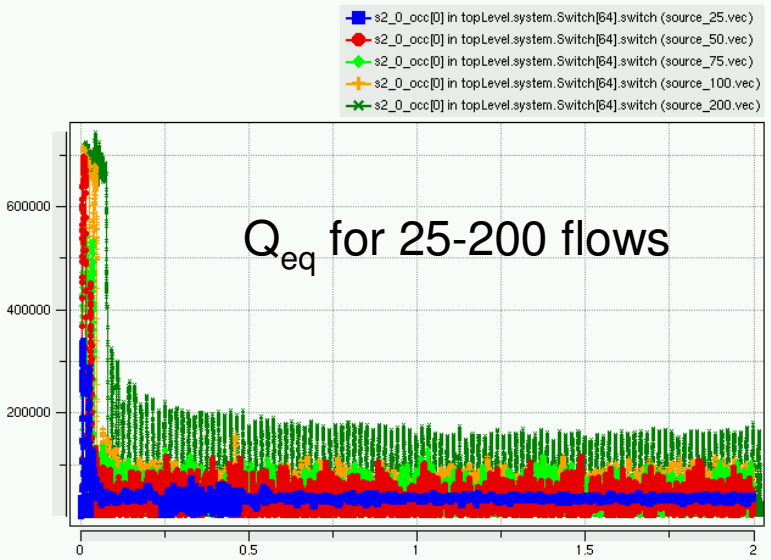


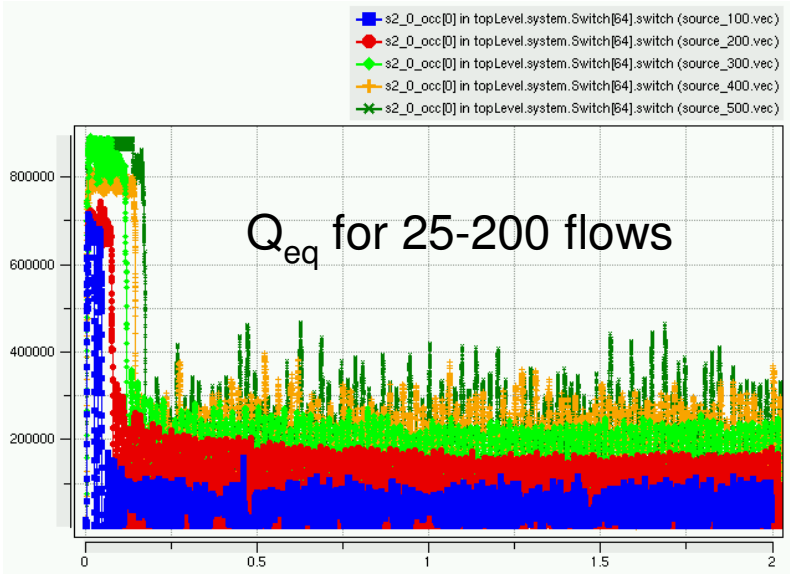


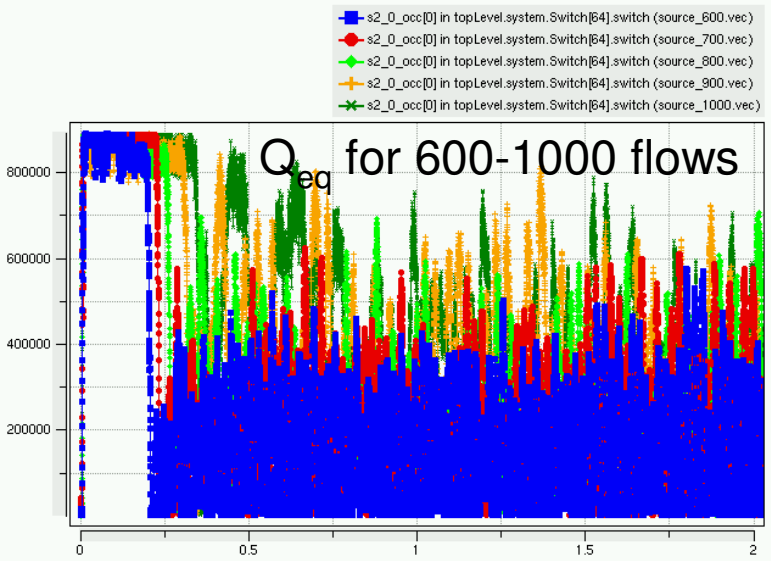
QCN Challenges

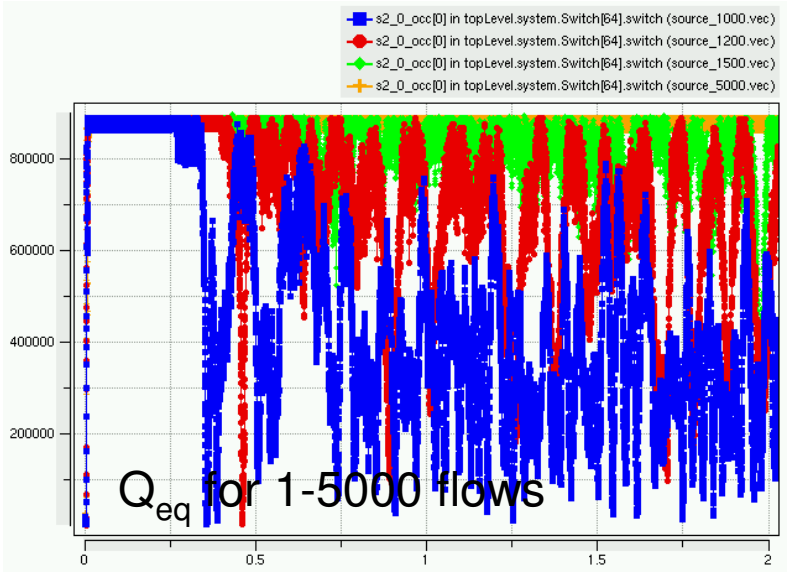
Many Flows; Unfairness.

QCN Buffer Control f(# flows): Q_{eq}=33KB

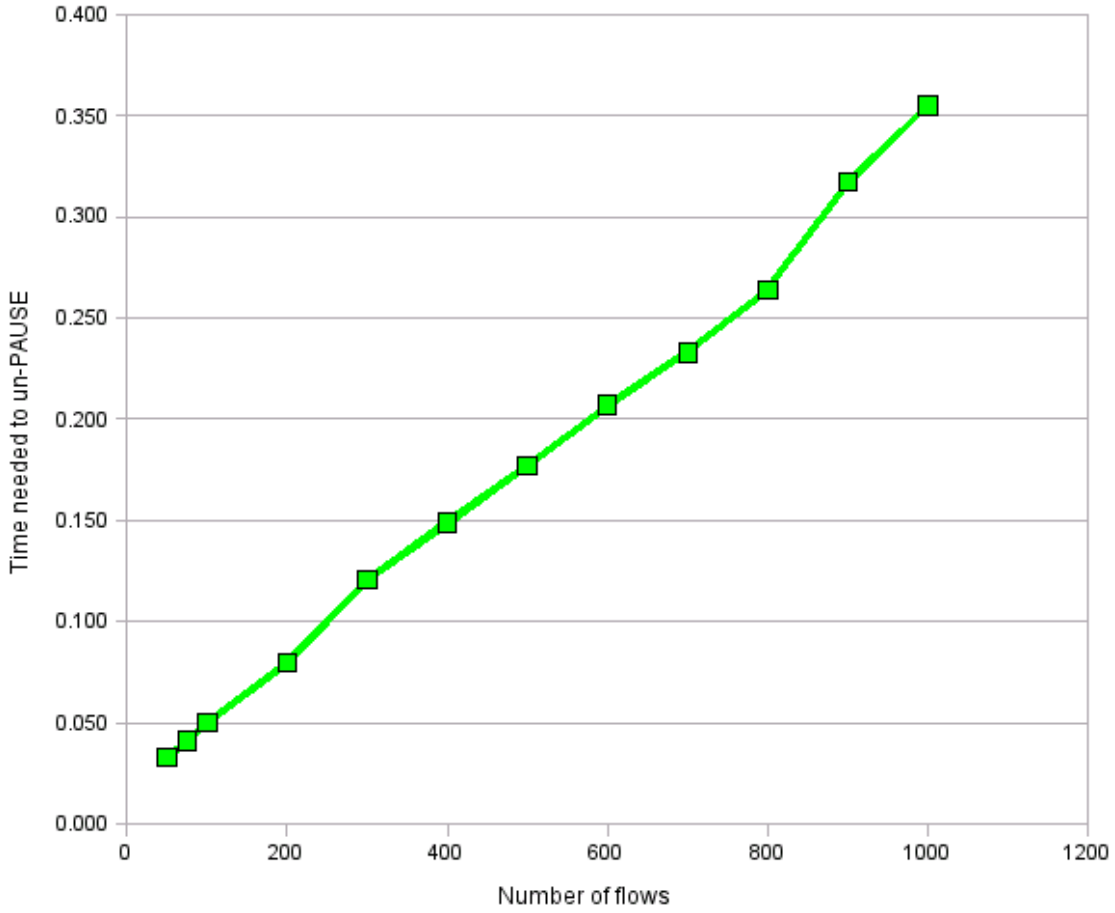








QCN control lag: Until PFC=0 and within Q_{eq} limits



Unfairness: An Extreme Case of 1 Winner + (N-1) Loosers