RZ 3849 (# ZUR1307-044) 08/06/2013
Computer Science 13 pages

# Research Report

## Occupancy Sampling for Terabit CEE Switches Resolving Output and Fabric-Internal Congestion

Fredy D. Neeser*, Nikolaos I. Chrysos*, Mitch Gusat*, Rolf Clauberg*, Cyriel Minkenberg*
Kenneth M. Valk‡, Claude Basso‡

*IBM Research – Zurich
8803 Rüschlikon
Switzerland

‡IBM Systems & Technology Group,
Rochester
USA

# Occupancy Sampling for Terabit CEE Switches Resolving Output and Fabric-Internal Congestion

Fredy D. Neeser, Nikolaos I. Chrysos, Mitch Gusat, Rolf Clauberg, Cyriel Minkenberg
*IBM Research, Zurich, Switzerland*
Kenneth M. Valk, Claude Basso
*IBM Systems & Technology Group, Rochester, USA*
{nfd,cry,mig,cla,sil}@zurich.ibm.com kmvalk@us.ibm.com, basso2@fr.ibm.com

***Abstract*—One consequential feature of Converged Enhanced Ethernet (CEE) is losslessness, achieved through L2 Priority Flow Control (PFC) and Quantized Congestion Notification (QCN). We focus on QCN and its effectiveness in identifying congestive flows in input-buffered CEE switches and switching fabrics. Our objective is to complement PFC's coarse per-port/priority granularity with QCN's per-flow control. By detecting buffer overload early, QCN can drastically reduce PFC's side effects. We install QCN congestion points at input buffers with virtual output queues and demonstrate that arrival-based marking cannot correctly discriminate between culprit and victim flows.**

**Our main contributions are the following. First, we propose occupancy sampling (QCN-OS), a novel, QCN-compatible marking scheme, and random occupancy sampling as a practical realization. For non-blocking switching fabrics, QCN-OS at input VOQ buffers is shown to correctly identify culprit flows, improving buffer utilization, switch efficiency, and fairness.**

**Next we consider blocking network topologies and show that (a) switch-internal blocking may prevent QCN-OS at the inputs from identifying flows bottlenecked inside the fabric and (b) QCN-OS combined with fabric-internal reliable delivery correctly identifies internally and/or externally bottlenecked flows. Finally, we propose *two-samples*, a refinement of QCN-OS that more easily identifies internally congested flows.**

## I. Introduction

Upcoming datacenter networks are "short and fat": Up to one million physical and virtual nodes are connected in a single Layer 2 (L2) domain with abundant multi-pathing across high-speed ("fat") 10–100 Gbps links of a few tens of meters. Unlike wide-area networks, the datacenter RTT is dominated by queuing delays rather than time-of-flight, which under bursty workloads leads to a difficult traffic engineering problem. A datacenter based on CEE must support lossless traffic to enable applications such as Fibre Channel over Ethernet (FCoE), business analytics, algorithmic trading, storage, and HPC workloads. In this paper, we focus on QCN and its implementation for large L2 switches.

### A. Datacenter Bridging / Converged Enhanced Ethernet

*1) PFC:* Traditional Ethernet is not lossless in the sense of guaranteed buffer space at the receiving end of a link; instead, packets are dropped whenever a receive buffer reaches its capacity. This lossy network behavior, however, does not meet the semantics of the above-mentioned applications. Therefore, IEEE 802.1Qbb has recently introduced PFC, based on the 802.1p Class of Service. Within each priority, PFC acts as 802.3x PAUSE, but a PAUSEd priority will not affect the others. However, PFC may cause saturation tree congestion and introduce deadlock in certain topologies and switch architectures.

*2) QCN:* As a prerequisite to the introduction of lossless operation and to counteract the potentially severe performance degradation due to saturation tree congestion, IEEE has also standardized a new L2 congestion control scheme, Quantized Congestion Notification (QCN, 802.1Qau) [2]. QCN performs congestion detection at Congestion Points (CPs) inside switches. Each CP samples the frames arriving at the queue according to a sampling interval in bytes. At the same time, it characterizes the queue congestion by two state variables, position (offset) $Q_{\text{off}}$ and velocity ($Q_\delta$), where $Q_{\text{off}}$ is defined with respect to an equilibrium setpoint $Q_{\text{eq}}$. When the CP detects congestion in the sense that the feedback value $F_b \triangleq Q_{\text{off}} + w \cdot Q_\delta$ is positive,[1] it sends a Congestion Notification Message (CNM) to the source of the most recent frame (or flow), which is considered culprit. Converged Network Adapters (CNAs)

The present manuscript is an extended version of an earlier conference paper [1].

[1] 802.1Qau [2] defines $F_b \triangleq -(Q_{\text{off}} + w \cdot Q_\delta)$ and then indicates congestion if $F_b$ is negative. However, for simplicity, we drop the double negation.

at the sources react to CNMs by instantiating rate limiters in their Reaction Points (RPs). In response to CNMs, a QCN RP multiplicatively decreases its rate limit as a function of the feedback value, whereas in the absence of CNMs, it autonomously increases its injection rate in a fashion somewhat similar to TCP CUBIC [3].

### B. Contents and Contributions

In Sec. II, we consider the relation between the switch architecture and QCN CP placement, motivated by emerging high-radix input-buffered switch designs. Such switches are likely to have dedicated input buffers for architectural reasons while supporting PFC and congestion detection.

Our first finding is that the arrival-based culprit flow identification (marking) of standard QCN, when applied at the input buffers of a Virtual Output Queue (VOQ) based scheduled fabric, is unable to discriminate between culprit and victim flows, often throttling innocent victims. In a VOQ-based switch, flows entering an input buffer may have different service time distributions. This is in contrast to the standard assumption whereby congestion detection is performed at a switch output queue[2] with single-server FIFO service, e.g., modeled as M/G/1. Hence, using a VOQ architecture exposes an undesirable property of standard QCN *arrival sampling*, because a high arrival rate of a flow in combination with $F_b > 0$ is not a reliable indicator of an arrival/departure rate mismatch.

In Sec. III, we introduce our main contribution, namely an alternate culprit-flow identification scheme called *occupancy sampling*. As a generalization of QCN arrival sampling, it removes the direct relationship between a flow's arrival rate and the rate at which it receives CNMs. We also contribute a succinct analysis demonstrating that occupancy sampling eliminates the unfairness of arrival sampling. We propose an efficient realization of the scheme, implementable at high link speeds. In Sec. IV, we provide a comparative evaluation of arrival vs. occupancy sampling, highlighting the benefits of the latter in CIOQ internally non-blocking switches and switching fabrics.

In Sec.V, we evaluate the performance of occupancy sampling at the input VOQs of *internally lossless but blocking* switching fabrics. In this setting, we find that switch-internal blocking may keep occupancy sampling at the inputs from identifying flows affected by fabric-internal bottleneck(s), because internal saturation trees may rate limit even flows that do not cross the fabric-internal bottleneck link(s), rendering the distinction of hot and cold flows based on VOQ backlogs more difficult.

We then describe the *two samples* method, which preserves the benefits of occupancy sampling while enhancing its capability to identify internally congested flows.

Next we find that occupancy sampling is able to correctly identify internally and/or externally bottlenecked flows if combined with fabric-internal end-to-end reliable delivery, a much desired feature of modern lossless switching fabrics. As it turns out in the reliable delivery setting, flows crossing internal bottlenecks are likely to have more unacknowledged packets and hence larger backlogs at the input buffers, a desirable form of congestion that helps QCN-OS at the inputs to quickly identify the true culprits.

We discuss related work in Sec. VI and then draw conclusions in Sec. VII, outlining some future work items.

## II. Motivation

### A. QCN for large switch fabrics

Crossbar chips with several tens of 10G and 40G Ethernet ports are now emerging in the market, and 100G Ethernet technology is also underway [4], [5]. Such high-radix crossbars enable *larger* switches and switching fabrics using *fewer* crossbar ASICs. Thus, high-radix switches lead to lower latency (fewer hops), lower cost (fewer chips), and lower power consumption (fewer chip boundary crossings). The lower latency also lowers cost by reducing buffer size requirements, in line with the trend towards shallow buffers.

Here we rethink the L2 congestion management of CEE networks with future large switches, namely that the switch architecture assumed by the IEEE 802.1 DCB task group in the QCN standardization is an ideal output-queued switch. Correspondingly, the standard associates one QCN CP with every switch output [6] [2, Sec. 30.2.1]. However, switch implementations using an output-queued or shared-memory architecture are not feasible with high port counts at 40+ Gb/s speeds, mainly because of their excessive memory bandwidth requirements.

Practical Ethernet switches must apply some form of per-input buffer allocation, which boils down to dedicating – either physically or logically – some buffer space to each input port. Therefore, most typical Ethernet switch implementations are either purely input-queued or adopt a hybrid combined-input-output-queued (CIOQ) architecture [7].

Figure 1 depicts a generic switch with input and output buffers. The incoming data frames are segregated at the

[2]The standard further assumes a separate output queue per priority.

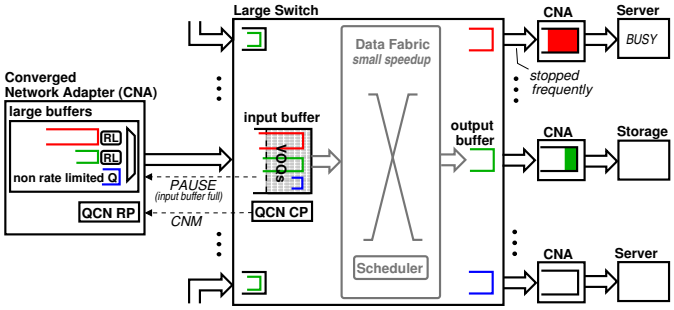



Figure 1. A generic Ethernet switch with input and output buffers. Shown are an input buffer that hosts virtual output queues (VOQs and the corresponding upstream CNA. When the input buffer is close to full, the input port sends PAUSE-ON to the upstream CNA, which stops forwarding data until it receives a PAUSE-OFF frame. This does not prevent an input buffer from being hogged with frames going to a congested destination, thus blocking other frames targeting available destinations.

inputs of the switch and stored in VOQs in front of the data fabric. A scheduler is responsible for transferring the frames from the input VOQs to the outputs, typically while taking care not to overflow the output buffer. Depending on the scale of the switch, a single crossbar or a multi-stage switching fabric may be used. The fabric may be bufferless or comprise internal staging buffers, and it may be overprovisioned to compensate for scheduling inefficiencies and internal overhead.

Such VOQ-based scheduled switches overcome the HOL blocking of input queuing. However, full flow isolation additionally requires (i) private per-VOQ buffers, which may not scale to large port counts, and (ii) per-VOQ (discriminate) flow control between the switch and the network adapter (CNA), which is not provisioned in Ethernet networks. As a result, the VOQs for different switch outputs have to share an input buffer, which can lead to *buffer hogging*. For instance, if a server is overloaded as shown in Fig. 1, then the traffic heading towards it will monopolize the input buffer. Eventually, the input port asserts PFC PAUSE, extending the saturation tree rooted at the overloaded server. Note that PFC, by performing flow control per input port and per priority, cannot selectively control on a per-output-port basis.

Saturation-tree congestion, also known as high-order HOL blocking [8], impedes the progress of data packets to uncongested destinations, deteriorating the throughput and delay performance. This is the main reason that legacy PAUSE is usually disabled in Ethernet networks. Priority flow control (PFC), which was a hard requirement to eliminate buffer overflows and to enable the convergence of storage and clustering traffic, cannot prevent HOL blocking within a priority. QCN, by effectively minimizing the formation of saturation trees, is complementary to PFC and a key prerequisite for the adoption of CEE.

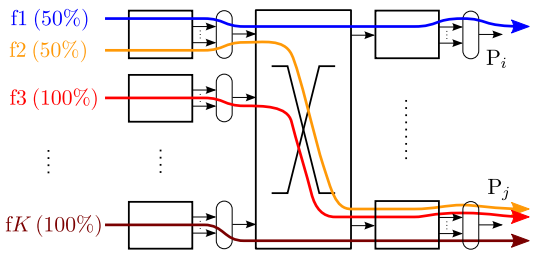



Figure 2. Input-generated (IG) hotspot scenario for a CIOQ switch.

### B. Input- vs. output-based congestion points

Standard QCN congestion detection is based on idealized output-queued switches having a CP monitoring each output queue. However, as illustrated above, in practical Ethernet switches, packets will queue up in input *and* output buffers. This suggests the possibility of QCN CPs at the input (VOQs) buffers instead of the outputs. Ideally, QCN at an input buffer should detect overload, mark and throttle the culprit flows and, by using an appropriate $Q_{\text{eq}}$, keep the input buffer backlog below the PFC-high threshold.In addition, CNMs generated by CPs at inputs do not traverse the fabric; hence they neither consume fabric-internal bandwidth nor incur any additional delays.

As we show next, naively applying standard QCN at inputs may severely reduce or even nullify the expected performance improvement over a PFC-only solution.

Our first experiment considers the input-generated (IG) hotspot scenario in Fig. 2, with a CIOQ switch as in Fig. 1. A QCN CP is installed at each input buffer. Flows f1 and f2 are injected by the same CNA, at 5 Gb/s each. Multiplexed on the same 10G input link of the switch, they share the input buffer of a switch port. Flow f1 targets the uncongested switch output port $\text{P}_i$, whereas f2 targets $\text{P}_j$, which is also the destination of flows f3 ... f$K$ from other switch inputs.

As shown in Fig. 3 for a PFC-only baseline test with QCN disabled, each flow receives a bandwidth share of $\frac{10}{4} = 2.5$ Gb/s during the congestive phase. This is fair for the $K - 1 = 4$ flows that share the 10G capacity of an output link. However, because of buffer hogging and ensuing PFC, victim f1 achieves only 2.5 Gb/s vs. the expected fair rate of 5 Gb/s.

Surprisingly, f1 also did not achieve the expected 5 Gb/s when we enabled QCN at the inputs: As shown in Fig. 4a for $K = 6$ flows, f1 is severely throttled between 1.5 and 3 Gb/s, converging towards 2 Gb/s. Interestingly,

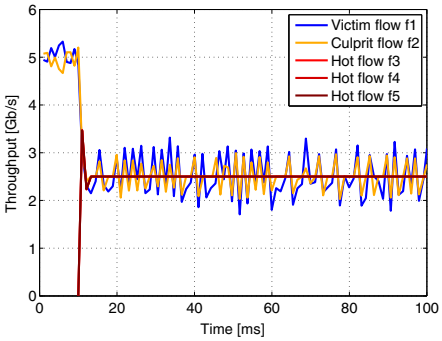



Figure 3. CNA TX raw throughput for all flows of the IG hotspot scenario in Fig. 2, with PFC only. We use $K = 5$ flows. Initially, flows f1 and f2 obtain their fair share of bandwidth. At 10 ms, flows f3 ... f5 are enabled as well. Because of indiscriminate PFC, buffer hogging at the common source port of f1 and f2 forces the victim flow f1 to the rate of the culprit-flow f2, which attains the fair share of the $K - 1 = 4$ flows targeting the same output.

as Fig. 4b shows, this is no longer caused by buffer hogging: After a transient PFC activity, QCN keeps the buffer occupancy close to $Q_{\text{eq}}$ and below PFC-high, so no PFC-PAUSE is asserted. Instead, the throughput limitation results from an inability of standard QCN sampling to discriminate, in the case of non-FIFO service, between culprits and victims.

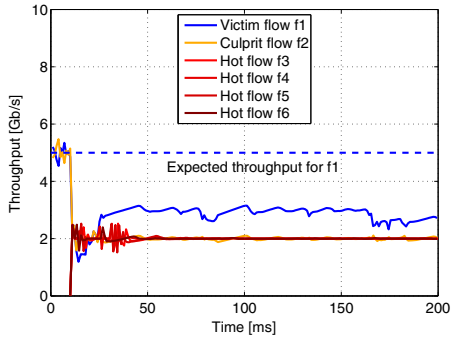
In the next (sub)section, we formally show that standard QCN will send CNMs to the source of a flow at an average rate proportional to its arrival rate at the CP: If flow f1 arrives at a higher rate than f2, then – whenever the input buffer occupancy in Fig. 4b exceeds $Q_{\text{eq}}$ –, f1 has a higher probability than f2 to receive a CNM. On the other hand, the input buffer occupancy does not stabilize while the culprit f2 arrives at more than $\frac{10}{K-1}$ Gb/s= 2 Gb/s. As a result, f1 is also forced towards a rate of 2 Gb/s. This is an unfair rate allocation that severely underutilizes the output link at $\text{P}_i$.

### C. Properties of QCN arrival sampling

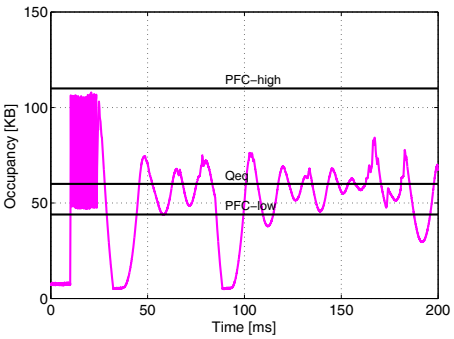
For a standard QCN CP, we first provide a clarification of two distinct aspects of "sampling", namely, (a) *sampling instant determination*, i.e., deciding *when* to mark a flow as congestive; and (b) *culprit flow identification*, i.e., deciding *which* flow to mark as congestive. In Sec. III, we will provide a corresponding characterization for an enhanced scheme.

If the standard CP[3] is associated with an output queue of an ideal output-queued bridge [2, Fig. 30-1], then the CP buffer is assumed to be a FIFO queue. However,

[3]A CP is associated with a single 802.1q priority; implementations typically provide a separate CP for each QCN-managed priority.




(a) CNA TX raw throughputs.




(b) Buffer occupancy of input port shared by f1 and f2.

Figure 4. Input generated (IG) hotspot test with $K = 6$ flows, using PFC and QCN CPs with standard sampling at the switch input ports. In (a), flows f1 and f2 initially obtain their fair share of bandwidth. At 10 ms, as the hot flows f3 ... f6 are enabled, victim f1 drops to $\approx 3$ Gb/s vs. the expected 5 Gb/s. In (b), the buffer occupancy shows a 20 ms phase of PFC activity due to the high overload starting at 10 ms. Thereafter, QCN keeps the occupancy around $Q_{\text{eq}} = 60$ KB.

the following also applies if the standard CP is (naively) associated with an arbitrarily scheduled buffer.

Consider a set of flows $f_n$, $n = 1 \ldots N$ with average frame sizes of $\overline{S}_n$ bytes, arriving at the CP with densities of $\lambda_n(t)$ [frames/s] and corresponding rates $R_n(t) = \overline{S}_n \lambda_n(t)$ [bytes/s]. The aggregate arrival rate is then $\lambda(t) = \sum_{i=1}^{N} \lambda_i(t)$ [frames/s] or $R(t) = \sum_{i=1}^{N} \overline{S}_i \lambda_i(t) = \overline{S}\lambda(t)$ [bytes/s], so the overall average frame size can be written as $\overline{S} = \left(\sum_{i=1}^{N} \overline{S}_i \lambda_i(t)\right) / \lambda(t)$. Furthermore, the departure rate will be denoted as $\mu(t)$ [frames/s] or $X(t)$ [bytes/s].

Standard QCN uses *arrival sampling* (QCN-AS), i.e., both (a) and (b) are based on arrivals: The aggregate arrivals at the CP, measured from the previous sampling instant and counted in bytes, are compared with a current sampling interval $I_s$ (also in bytes) to trigger the computation of a congestion estimate $F_b(t)$; if $F_b(t) > 0$, then a CNM is generated, targeting the culprit flow implied by the most recent arrival.

QCN-AS has a time-varying overall sampling rate[4] [5]

$$\rho(t) = R(t)/I_s \tag{1}$$

denoting the rate at which congestion estimates are computed.

The following discussion is restricted to slowly time-varying arrival rates, i.e., we assume $\lambda_n(t) \approx \lambda_n$, $R_n(t) \approx R_n$, $R(t) \approx R = \overline{S}\lambda = \sum_{i=1}^{N} \overline{S}_i\lambda_i$ and $\rho(t) \approx \rho = R/I_s$ over the duration of a sampling interval. The expected duration for receiving $I_s$ bytes is then $\overline{T}_s \triangleq E[T_s] = \left(I_s/\overline{S}\right)/\lambda = I_s/R$.

Let $A_n$ denote the number of arrivals (frames) of flow $f_n$ during an interval of duration $T_s$. As we can in fact define $\lambda_n \triangleq E[A_n]/T_s$, we have $E[A_n] = \lambda_n T_s$. Conditioned on the event that the CP has counted $I_s$ bytes of payload at the end of an interval of duration $T_s$, the probability that the most recent frame belongs to $f_n$ is given by the expected fraction of $I_s$ that is received for $f_n$, viz.

$$P_n^{(s)} = \frac{E[A_n]\,\overline{S}_n}{\sum_{i=1}^{N} E[A_i]\,\overline{S}_i} = \frac{\lambda_n\overline{S}_n}{\sum_{i=1}^{N}\lambda_i\overline{S_i}} = R_n/R. \tag{2}$$

The proportionality of the *flow sampling probability* $P_n^{(s)}$ to its *arrival rate* $R_n(t) = \overline{S}_n\lambda_n(t)$ is the main characteristic of the QCN-AS culprit flow identification.

If $P_n^{(s)}(t)$ denotes $P_n^{(s)}$ for a sampling interval ending at time $t$ then, conditioned on the event that the CP takes a sample at $t$, the probability that it sends a CNM to the source of $f_n$ becomes

$$\begin{aligned} P_n^{(r)}(t) &= P_n^{(s)}(t) \cdot \Pr\{F_b(t) > 0\} \\ &= (R_n(t)/R(t)) \cdot \Pr\{F_b(t) > 0\}, \end{aligned} \tag{3}$$

referred to as the QCN-AS *flow-reflection probability*. The proportionality of (3) to $R_n(t)$ tends to equalize the flow injection rates (in bytes/s) whenever there is congestion: A flow with a higher arrival rate at the bottleneck queue is likely to be marked more often than one with a lower arrival rate. This is desirable in the sense of also minimizing buffer congestion *if* the contribution of a flow to the overall buffer occupancy is proportional to its arrival rate $R_n(t) = \overline{S}_n\lambda_n(t)$.

[4]In reality, $I_s$ depends on the quantized feedback value at the previous sampling instant [2]. However, as this is unimportant for understanding the flow selectivity of QCN-AS, $I_s$ is assumed to be constant.

[5]Eq. 1 is sometimes written as $\rho(t) = \lambda(t)\,P_s$, where $P_s \triangleq \overline{S}/I_s$ is loosely referred to as a sampling probability, with values such as $\overline{S} = 1500$B, $I_s = 150,000$B and $P_s = 1\%$. However, $P_s$ should not be confused with the conditional probability $P_n^{(s)}$ in (2).

## III. QCN OCCUPANCY SAMPLING

As shown in the previous section, QCN-AS characterizes a flow as congestive (culprit) based on its contribution $r_n(t)$ (in bytes/s) to the overall arrival rate $r(t) = \sum_{i=1}^{N} r_i(t)$, ignoring the departure rate $x_n(t)$ of the flow[6].

We remove this restriction by using the rate *mismatch* $r_n(t) - x_n(t)$ as a discriminator, exploiting the fact that a buffer acts as a rate mismatch integrator: In a lossless system, the contribution of flow $f_n$ to the CP buffer occupancy with given initial condition is $q_n(t) = q_n(0) + \int_0^t (r_n(\tau) - x_n(\tau))\,d\tau$. We maintain that for any non-FIFO discipline, e.g., the multiserver case typical for a CIOQ switch input queue (see Sec. II-A), $q_n(t)$ is generally not proportional to $r_n(t)$.

Referring to Fig. 5, we propose a QCN-compatible sampling method called *occupancy sampling* (QCN-OS, in blue), which (i) implicitly or explicitly uses the buffer occupancy of a flow as the cost function; (ii) eliminates the proportionality of the flow sampling probability to the flow arrival rate; (iii) is suitable for monitoring a switch input buffer that hosts many scheduled VOQs by not mandating FIFO service; and (iv) is compatible with PFC.

Specifically, as further explained below, we randomly sample the CP buffer in response to a stimulus, if $F_b(t) > 0$. The stimulus is the arrival of $I_s$ bytes of payload [2]. Additionally, since occupancy sampling does not rely on frame arrivals, an external clock source can be used during PFC-PAUSE to keep generating CNMs while the CP does not receive new frames.Such a *QCN keep-alive* mechanism can reduce the duration of PFC activity by issuing CNMs during PFC-PAUSE.

Thus, instead of sending the CNM to the source of the frame that just entered the CP buffer, with (random) QCN-OS we identify a culprit by *randomly picking* one occupied buffer unit and locating the corresponding frame header. Herein, a buffer unit refers to a fixed-size unit of memory or storage. A frame may be stored in multiple buffer units. The random selection from a pool of fixed-size buffer units will pick a particular frame and flow with a probability given by the fraction of the overall CP buffer occupancy $q(t) = \sum_{i=1}^{N} q_i(t)$ taken, respectively, by this frame and flow. Hence, QCN-OS has an (instantaneous) flow sampling probability

$$P_n^{(s)}(t) = q_n(t)/q(t) \tag{4}$$

[6]Here we use $r_n(t)$, $r(t)$, $x_n(t)$ and $x(t) = \sum_{i=1}^{N} x_i(t)$ to denote instantaneous (realized) rates.

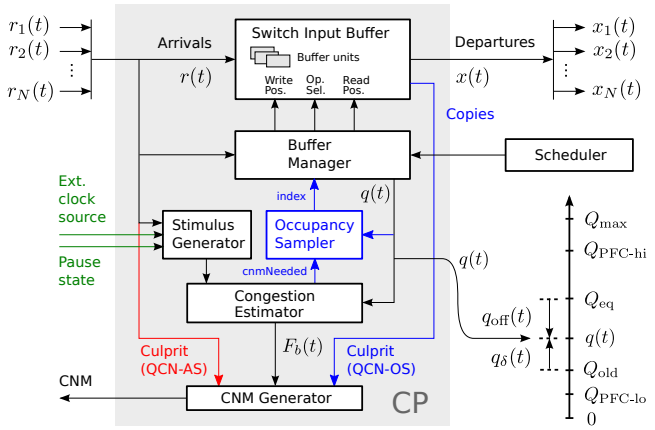



Figure 5. QCN congestion point (CP) at a switch input port. Arrivals and departures are shown for $N$ flows. Governed by the Scheduler, the switch input buffer allows flows to depart at different speeds. The Buffer Manager provides the aggregate queue size $q(t)$ to the Congestion Estimator (CE). Triggered by a stimulus, the CE computes a feedback value $F_b(t) \triangleq q_{\text{off}}(t) + w \cdot q_\delta(t)$.

If $F_b(t) > 0$, the CNM Generator forms a CNM targeting the source of a culprit. For arrival sampling (QCN-AS, red), the culprit flow is implied by the most recent frame. For occupancy sampling (QCN-OS, blue), the Occupancy Sampler randomly selects an occupied buffer unit containing a frame header, which identifies the culprit. The CNM includes $F_b(t)$ quantized to six bits.

Also shown are $Q_{\text{eq}}$ (setpoint), $q_{\text{off}}(t) \triangleq q(t) - Q_{\text{eq}}$ (position/offset), $q_\delta(t) \triangleq q(t) - Q_{\text{old}}$ (velocity), $Q_{\text{old}}$ (previous queue size), $Q_{\text{PFC-hi}}$ (stop threshold) and $Q_{\text{PFC-lo}}$ (go threshold).

The stimulus generator is based on comparing the received payload with a current sampling interval $I_s$ in bytes [2]. During PFC-PAUSE, it may also use an external clock source (green).

and a flow reflection probability

$$P_n^{(r)}(t) \quad = \quad (q_n(t)/q(t)) \cdot \Pr\{F_b(t) > 0\}. \qquad (5)$$

It also follows that the average probability of selecting flow $f_n$ over multiple sampling periods is given by the average fraction of the CP buffer occupancy of the flow.

We emphasize that an implementation need not hold any state to keep track of the flow buffer occupancies $q_n(t)$: The proportionality of (5) to $q_n(t)/q(t)$ simply follows from random sampling. Moreover, the feedback value itself need not depend on the culprit flow identified; here it is computed according to the QCN standard.

As an example, suppose that flows $f_1$, $f_2$ and $f_3$ are passing through a CP, with initial buffer occupancies $q_i(t_0) \approx Q_{\text{eq}}/3$. The flows have the same initial arrival rate $r_i(t_0) = r(t_0)$. The departure rates of $f_1$ and $f_2$ are limited by the outgoing link capacity $C$ [bits/s] , and the departure rate of $f_3$ is limited by some maximum service rate $x_{3,\max} \ll r_3(t_0)$. The CP will detect $F_b(t) > 0$ as $q_3(t)$ grows rapidly; on the other hand, $q_{1,2}(t)$ stays at or below $Q_{\text{eq}}/3$. While $F_b(t) > 0$, occupancy sampling sends CNMs to the source of $f_i$ with probability $p_i(t) = \frac{q_i(t)}{q_1(t)+q_2(t)+q_3(t)}$, where $p_3(t) \gg p_{1,2}(t)$ until the flow buffer occupancies are balanced.

For the same example, arrival sampling sends CNMs to the source of $f_i$ with approximately equal probability $p_i(t) = \frac{r_i(t)}{r_1(t)+r_2(t)+r_3(t)} \approx 1/3$ until *all* arrival rates fall below the minimum of the service rates given by $x_{3,\max}$ and $F_b(t) \leq 0$.

Table I
SIMULATION PARAMETERS

| Module | Parameter | Value |
|---|---|---|
| Switch | Input buffer per port | 150 KB |
| Switch | Output buffer per port | 150 KB |
| PFC | $Q_{\text{PFC-hi}}$(STOP threshold) | 110 KB |
| PFC | $Q_{\text{PFC-lo}}$(GO threshold) | 44 KB |
| QCN CP | $Q_{\text{eq}}$(equilibrium) | 60 KB |
| QCN CP | w (weight for velocity) | 2 |
| QCN CP | $I_s$ (base sampling interval) | 150 KB |

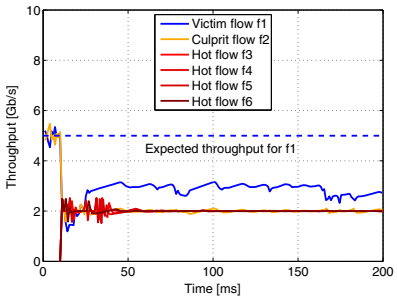
## IV. SIMULATIONS OF OCCUPANCY SAMPLING IN OUTPUT-PORT CONGESTION OF CIOQ SWITCHES

Our simulation environment is based on Venus [9], a detailed L2 network simulator based on OMNeT++ [10]. Our CEE switch model is a CIOQ as shown in Fig. 1, with sufficient internal speedup for the Ethernet line rate. Operating at flit level, it accurately represents the CIOQ queuing, buffering, scheduling, and link-level flow control. We use dedicated input (output) buffers per port with associated PFC thresholds. The input buffers of a port are shared by logical VOQs. The key parameters are listed in Table I.

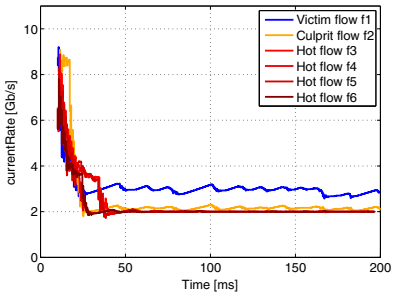
Our CNA and CIOQ switch models support both PFC and QCN. We can configure QCN CPs at the switch input and/or output buffers. The CNA model is capable of instantiating or removing QCN rate limiters on demand.

We simulate a single CEE priority, using 1522B MTUs for data traffic and 64B frames for PFC-PAUSE and QCN CNMs. We performed extensive simulations for 10G and 100G links; however, here we show only the 10G results to simplify comparisons with the recent literature and the 802 DCB archives.

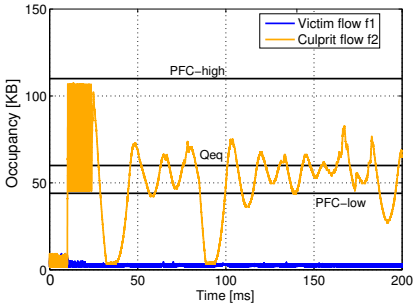
To evaluate the sampling methods, we use two traffic scenarios [11], namely, an input-generated hotspot in Sec. IV-A. Here we present the results for input-generated hotspots; our results for output-generated hotspots can be found in [1]. In both scenarios, a switch inputport is shared by two flows f1 and f2. Flow f1 (a.k.a. the victim) targets an uncongested output port, whereas f2 (a.k.a. the culprit) targets a congested one.



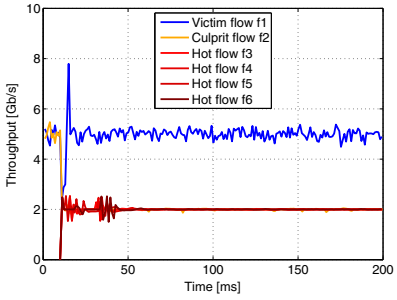

(a) QCN-AS, CNA TX gross throughputs.



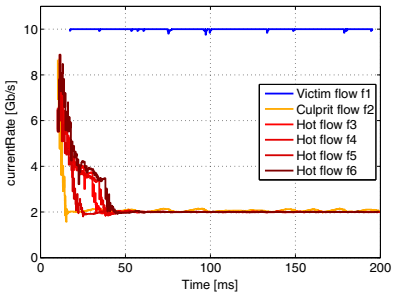

(b) QCN-AS, CNA rate limits.



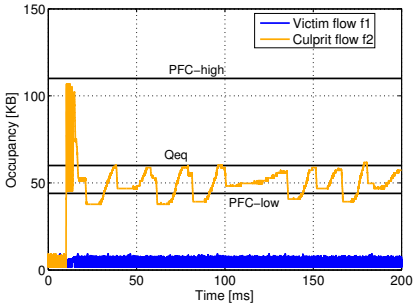

(c) QCN-AS, input buffer occupancy per flow. The PFC-PAUSE activity at 10 ms lasts for $\approx 14$ ms.




(d) QCN-OS, CNA TX gross throughputs.




(e) QCN-OS, CNA rate limits. Flow f1 is virtually unaffected by flow f2 targeting the congested destination.




(f) QCN-OS, input buffer occupancy per flow. The PFC-PAUSE activity at 10 ms lasts for $\approx 6$ ms.

Figure 6. Input-generated (IG) hotspot test with PFC and QCN at the switch input ports, for QCN arrival sampling (QCN-AS, top row) and occupancy sampling (QCN-OS, bottom row). (c) and (f) show the input buffer occupancy per flow for the input shared by f1 and f2; the congestion at 10 ms results in a transient PFC-PAUSE activity, where the sum of the flow occupancies frequently attains PFC-high.

A good QCN implementation should ensure that the culprit flow does not impede the throughput of the victim flow and that both output ports are maximally utilized. We measure the raw throughputs[7] of the flows in bits/s at the exit points of the upstream CNAs. These CNA TX throughputs correspond to the rates at which flows are injected into the switch. As we consider lossless operation, the flow arrival rates at the downstream CNAs closely match their injection rates.

### A. *Input-generated hotspot*

We consider the scenario depicted in Fig. 2 for $K = 6$ flows, so the congestion at output port $P_j$ results from the $K - 1 = 5$ flows f2 ... f6. During the first 10 ms, the system warms up with only f1 and f2 active. Then the remaining flows are also activated for the remaining simulation.

For PFC and QCN at the switch input ports, Fig. 6 provides a detailed performance comparison with arrival sampling (QCN-AS) vs. occupancy sampling (QCN-OS).

[7]Raw throughput includes the 20B per-frame overhead for the interframe gap, preamblem and start frame delimiter.

With QCN-AS, the congestion at 10 ms results in rapid rate-limit and throughput decreases for *all* flows. At 25 ms, f1 and f2 are rate limited to $\approx 4.5$ and $\approx 3$ Gb/s, respectively. Hence, whenever the CNA is able to send (not PAUSEd), it injects f1 at a higher rate than f2. During the transient PFC activity, the effective injection rates are lower than the rate limits.

As shown in Fig. 6b, the rate recovery phases of f2 result in rapid increases of its input buffer occupancy and, for f1 *and* f2, in correspondingly higher flow reflection probabilities (3).

The rate limits in Fig. 6 show that for QCN-AS, both f1 and f2 are throttled, whereas for QCN-OS, f1 is virtually unaffected by the congestion.

In Fig. 6c and Fig. 6f, the input buffer has a backlog of f2 around $Q_{\text{eq}}$ with both sampling schemes, but QCN-OS enjoys a shorter PFC activity and improved stability in the QCN-only regime.

Fig. 7 depicts the performance of the standard QCN configuration using QCN-AS at the switch *outputs*. In this configuration, the victim flow recovers as fast as with QCN-OS at the switch inputs –see Fig. 6d. However, unlike occupancy sampling, which achieves a strictly

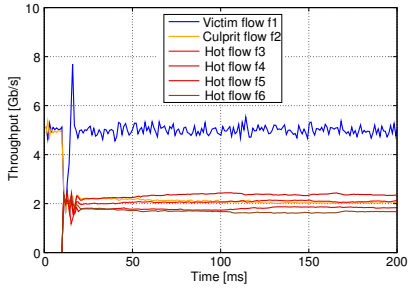



Figure 7. Input-generated (IG) hotspot test, QCN-AS at the switch outputs, CNA TX raw throughputs.

fair rate allocation of the output capacity, QCN-AS at the outputs results in a $\pm 0.4$ G unfairness among the hot flows. The unfairness of QCN-AS at the outputs is to be attributed to the statistical errors of a single QCN congestion point handling multiple flows: *the flows that are sampled first may result with smaller rates*. By contrast, the system deploying QCN-OS at inputs has a separate contention point for each flow, thus yielding more fair rates.

## V. Occupancy Sampling At Fabric VOQ Buffers Resolving In-Fabric Congestion

The key characteristic of QCN occupancy sampling at input buffers is that it identifies and throttles the flows that tend to monopolize the buffer memory available for VOQs. In an ideal CIOQ switch, the backlog of an input VOQ depends on the rates of the flows targeting the same output; its backlog does not depend on flows heading to different outputs.

However, many interesting switch architectures are prone to *internal blocking*, where divergent flows may bottleneck on a shared internal hotspot link. Such internal blocking is a symptom of switches and switching fabrics that employ a (not non-) blocking network topology, and is more common in scalable multi-stage and hierarchical network topologies. Single-stage switches based on shared memory can also exhibit internal blocking due to contention for memory buffers. Internal blocking can also be present in switching fabric architectures and datacenter networks that cannot exploit the available bisection bandwidth because of routing constraints.

Two (side) effects hold true for switching fabrics that use hop-by-hop (link-level) flow-control to prevent buffer overflows. First, because of the conservation law for bits, the input (VOQ) backlog of a flow bottlenecked at an internal link $l$ depends on the mismatch between the flow's arrival rate and its fair share at link $l$. This effect enables QCN at the input VOQs to detect and throttle internally bottlenecked flows. However, if the

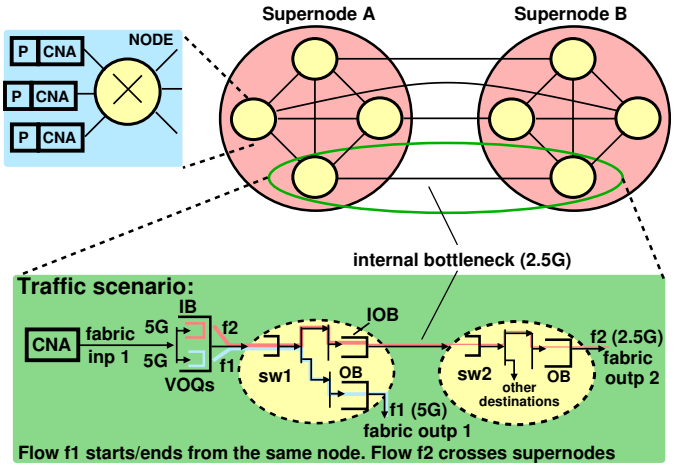



Figure 8. A multi-stage switch using an topology with internal blocking: the bisection bandwidth per CNA exceeds the Ethernet link capacity $C_E$ (10G) for nodes on the same supernode, but is significantly below $C_E$ for nodes on different supernodes (2.5 G). The traffic scenario that we simulate comprises two flows initiated from the same processor (and CNA) in supernode A: f1 goes to a local destination at the sourcing node, whereas f2 heads to a destination at supernode B, and is bottlenecked at a link connecting the two supernodes, filling up the internal output buffer (IOB) in front of that it. The figure depicts the arrival rates of the two flows (5G for both f1 and f2), as well as their *ideal, fair* output rates: 5G for f1 which is not bottlenecked at any link, and 2.5G for f2 which is bottlenecked at an inter-supernode link.

VOQ backlog does not form fast enough to let QCN-OS throttle the bottlenecked flow to its fair share at link $l$, a saturation tree routed at the bottleneck link $l$ may form, which limits the rate of all flows that share a buffer with the flow crossing $l$. In particular, the saturation tree may fill a switch-internal input buffer upstream of link l and downstream of the VOQs, which will limit all flows passing through it to the rate of the bottlenecked flow. This in turn leads to similar VOQ backlogs for hot (culprit) and cold (victim) flows, rendering QCN-OS unable to identify the true culprit(s).

Consider, for instance, the traffic scenario depicted in Fig. 8. Flows f1 and f2 enter a blocking, multi-stage switching fabric from the 10G fabric input 1, at 5G each, and target the 10G fabric outputs 1 and 2, respectively. No flow competes with them at the fabric outputs. The switching fabric uses link-level flow control to avoid packet loss. Along its route, flow f2 needs to cross a 2.5G link that connects switch 1 to switch 2, which becomes its bottleneck and limits its throughput to 2.5 Gb/s. Not shown in Fig. 8 is the filling level of buffers and the exertion of backpressure. Instead, the figure depicts the ideal, fair allocation of rates: f1 crosses the fabric at its arrival rate (5 Gb/s), and flow f2 is limited at its internal bottleneck at 2.5 Gb/s.

On the other hand, the simulations results in Fig. 9

show that the two flows receive equal throughputs. In the simulation model, the flows pass through a CNA upstream to their input VOQs, which can react to PFC signals, and enforce rate limits in response to CNMs. Flows f1 and f2 are active throughout the experiment and between 50–150 ms, respectively. The model employs link-level flow control to avoid buffer overflows. The parameters for QCN and the fabric-input/fabric-output buffer sizes are as described in Table I. In addition, the buffer in front of the bottleneck link connecting switch 1 to switch 2 has space for 32KB.

Our results are the following. In Fig. 9a, we use *only PFC* (there is no QCN). Note that ideally, f1 should retain a throughput of 5G throughout the experiment. As shown in the figure, with only PFC, f1's throughput drops to f2's bottleneck fair share. In Fig. 9c, we apply standard **QCN-AS at fabric outputs**. Here the behavior is identical to that in Fig. 9a, as fabric outputs are uncongested and therefore QCN does not react to input and internal backlogs, letting PFC define flow rates. Finally, in Fig. 9c, we apply **QCN-AS at fabric inputs**, which eliminates PFC after a short phase of simultaneous PFC and QCN activity[8]. Now, however, both flows are equally limited by the QCN rate limiters, as they arrive at the CP of the VOQs at equal rates.

Figure 9d shows that also QCN-OS at fabric-input VOQs cannot protect the victim flow f1 in this scenario. As mentioned, this happens because the VOQs are backpressured indiscriminately inside the saturation tree that is rooted at the internal bottleneck; therefore the VOQs of the two flows drain at the same rate, despite their different fair shares. It thus follows that VOQ occupancies are dictated by flow arrival rates. As the arrival rates of the two flows are equal, so are the VOQ backlogs (Fig. 9f). Effectively QCN-OS behaves similarly as QCN-AS. As shown in Fig. 9e, the CNA rate limits of the two flows follow each other on their way down, and stabilize once f2 drops to the rate of the bottleneck link, approximately 30 ms after the onset of the congestive event. It takes another 20 ms for the VOQ backlogs to stabilize at values that sum around the QCN set point (60 KB).

Although QCN-OS at inputs fails in this setting, its strength in resolving output contention remains valuable[9]. Next, we describe two mechanisms that maintain the spirit of QCN-OS, but in addition attack internal bottlenecks.

[8]QCN keep-alive can be used to reduce the duration of such transient periods.

[9]Note that fabric-output contention can also induce the creation of saturation trees, which in lossless multi-stage fabrics can indiscriminately limit the rate of any VOQ flow. However, it is possible to avoid saturation trees routed at fabric output ports by regulating flow injection rates in an end-to-end fashion —see for instance [12].

### *A. Two (random) samples: multi-criterion culprit resolution*

In our first method, we first do twice what random-based QCN-OS does once: we randomly sample two (occupied) buffer units from the CP buffer. Effectively, we now have *two (randomly-sampled) culprit flows*, indicated by two frame headers, and we can select one or the other using some appropriate criterion. The criterion that we use here ranks flows based on a (dynamic) path-congestion index that we maintain for each flow.

In principle, any path-congestion index can be used, but in this paper we select the maximum filling level of the buffers along the flow's path. In many interesting topologies, this index can be defined as the filling level of the queues in front of outgoing links of the first-level switch, which links are close enough to allow feeding the index back to the CP of the VOQs. In other network topologies, the path-congestion index may need to be routed or piggybacked to the CPs of the VOQs.

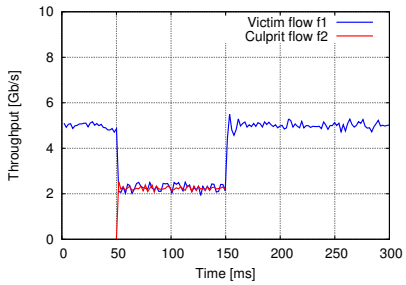
**Algorithm 1** Two samples: multi-criterion culprit resolution

**Select culprit flow:**
*A := Flow of randomly selected occupied buffer unit.*
*B := Flow of randomly selected occupied buffer unit.*
**if** *path-congestion index (A)* $\leq$ *path-congestion index (B).* **then**
  *return B.  // the CP will issue a CNM to flow B*
**else**
  *return A.  // the CP will issue a CNM to flow A*
**end if**
**Path-congestion index (Y):**
return maximum normalized filling level of all buffers in the route of flow Y.

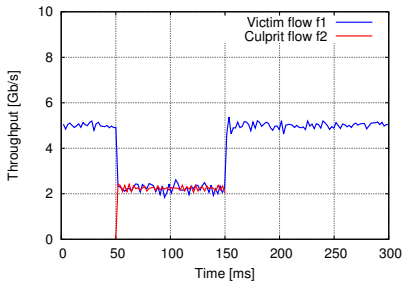
Observe that our two samples algorithm maintains the buffer occupancy of a flow as the main cost function. If a flow monopolizes the VOQ (CP) buffers, then it is very likely that both samples will be coming from it, in which case two samples is indifferent to QCN-OS. On the other hand, if the backlog of a flow is too small, then it is very unlikely that the flow will be sampled.

We further adapted the two samples method in order to perform congestion detection at the VOQ buffers of the topology in Fig. 8. In our simulations, we use the filling level of the buffer in front of the first hop of a flow to determine its path-congestion index. An alternative way to see this is that the path-congestion index is evaluated as the filling level of the buffer in front of the first hop link(s) connecting the source to the targeted cluster (destination/node/supernode).

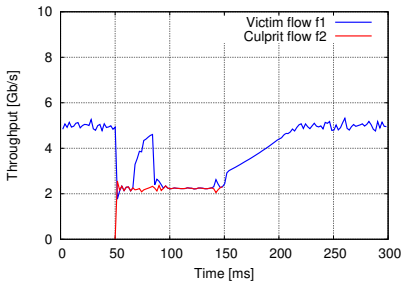
Figure 10a verifies that two-samples correctly identifies the internally bottlenecked flow and protects the



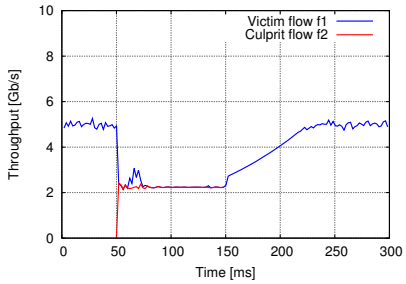

(a) PFC-only: CNA TX throughputs.



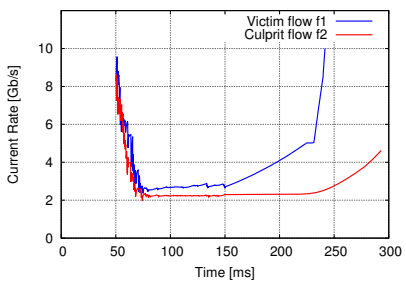

(b) QCN-AS at fabric outputs: CNA TX throughputs.



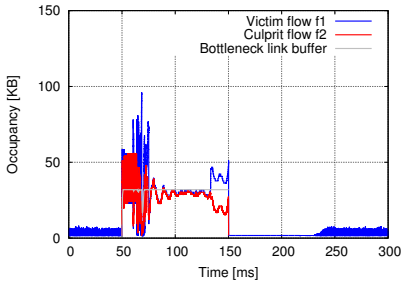

(c) QCN-AS at fabric inputs: CNA TX throughputs.




(d) QCN-OS at fabric inputs: CNA TX throughputs.




(e) QCN-OS at fabric inputs: rate limiters.




(f) QCN-OS at fabric inputs: the occupancies of the VOQs and of the internal buffer in front of the bottleneck.

Figure 9. Performance of PFC-only, QCN-AS, and QCN-OS in the scenario of Fig. 8. Flow f2 is active between 50 ms and 150 ms.

victim one. As shown in Fig. 10c, this is achieved despite the fact that the two flows initially build up similar VOQ backlogs. Figure 10b reveals however that the victim flow still receives a considerable number of CNMs, especially after the culprit flow has been rate limited close to its fair share.

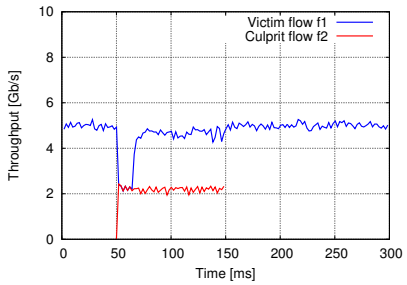
### *B. Improvements due to reliable delivery schemes*

Our next observation is that QCN-OS at input VOQs will perform better when the switching fabric implements an end-to-end reliable delivery scheme. In a reliable delivery scheme, the buffer holding the input VOQs maintains a copy of each injected packet until it receives an acknowledgement from the targeted end-point that the packet has been properly received by the corresponding fabric output buffer. It is very likely that a culprit flow, which crosses the bottleneck link, has a higher number of outstanding packets than a cold (victim) flow, which does not cross the bottleneck link. Hence the input backlogs of the two flows will also differ.

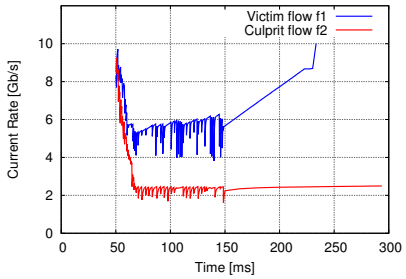
As shown in Figs. 10d-10f, this effect allows QCN-OS to mainly sample the culprit flow, leaving the victim one virtually unaffected. Effectively, the addition of fabric-internal reliable (end-to-end) delivery causes a culprit flow to quickly form VOQ backlog for the outstanding packets at the fabric input buffer, which is detected by QCN-OS[10]. Through the QCN feedback loop, this ensures that the injection rate of the culprit is reduced to the bandwidth available on the corresponding path(s) to the fabric output buffer. In other words, QCN-OS at input (VOQs) buffers, combined with fabric-internal reliable delivery, provides the full benefits of occupancy sampling in switch architectures that can be subject to internal blocking.

Finally, Fig. 11 shows the throughput performance when we apply two samples together with end-to-end reliable delivery. Additional simulation results, not presented here due to space limitations, show that two samples helps to reduce the transient period in traffic scenarios where f1 and f2 each comprise multiple smaller Ethernet subflows, which is not the case for the experiment in Fig. 11. In such scenarios, the VOQ backlogs of the culprit subflows (belonging to f2) and of victim subflows (belonging to f1), as well as the differences between these backlogs, will be relatively low due to the small subflow arrival rates. Hence QCN-OS (with or without end-to-end ACKs) will take longer to (statistically) identify the true culprits.

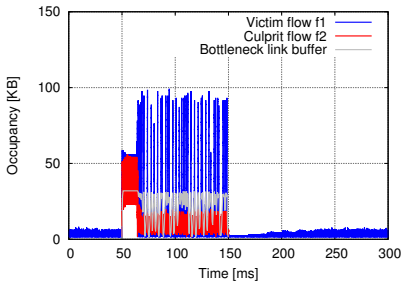
[10]Note that the switching fabric that we simulate remains internally lossless thanks to link-level flow control.



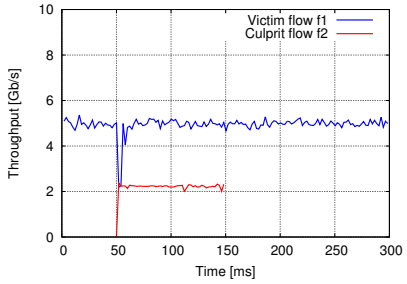

(a) Two-samples at inputs: CNA TX throughputs.



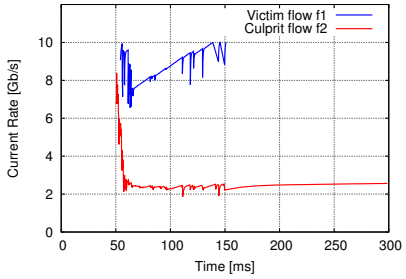

(b) Two-samples at fabric inputs: rate limiters.



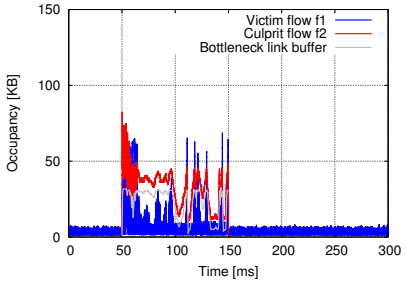

(c) Two-samples at fabric inputs: the occupancies of the VOQs and of the internal buffer in front of the bottleneck.




(d) QCN-OS at fabric inputs with end-to-end ACKs: CNA TX throughputs




(e) QCN-OS at fabric inputs with end-to-end ACKs: rate limiters.




(f) QCN-OS at fabric inputs with end-to-end ACKs: the occupancies of the VOQs and of the internal buffer in front of the bottleneck.

Figure 10. Performance of *two samples* and of *QCN-OS with end-to-end ACKs* in the scenario of Fig. 8. Flow f2 is active between 50-150ms.

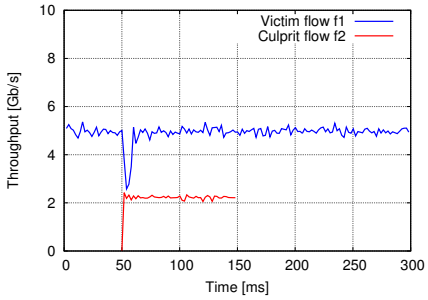



Figure 11. Performance in terms of CNA TX throughputs of two samples with end-to-end ACKs in the scenario of Fig. 8. Flow f2 is active between 50-150 ms. Flows f1 and f2 are here single flows, i.e. they are not comprised of multiple smaller Ethernet subflows.

## VI. Selected Related Work

The present paper is an extended version of an earlier conference paper [1], where we first proposed and described QCN-OS. In addition to that previous work, the present paper evaluates QCN-OS performance in scalable, multi-stage switching fabrics, whereas the simulations in [1] considered only CIOQ switches. Furthermore, the present paper describes why QCN-OS may fail when flows bottleneck at internal links, and how fabric-internal, edge-to-edge reliable delivery schemes, inducing *additional backlogs* at inputs, may in fact *improve* the performance of QCN-OS. Finally, it also presents and evaluates *two samples*, a practical scheme that can be used to identify internally bottlenecked flows, while maintaining QCN-OS's benefits. More generally, two samples provides a general framework to build a multi-criterion culprit-flow identification method on top of QCN.

Our overall work builds on the newly standardized QCN [2], which lies at the intersection of two established classes of congestion management. On one hand are the TCP/IP-based Layer 3/4 congestion detection, signaling and control schemes in lossy networks. Most widely implemented in routers are schemes based on RED [13] and REM [14], using ECN for single-bit congestion signaling feedback as in RFC 3168. As for L4 control, we mention only CUBIC [3].

On the other hand are the L2 flow and congestion control schemes for lossless computer and storage interconnection networks. Besides a faster control loop – with lags a few orders of magnitude lower than TCP's typical few 100s of ms – two key aspects differentiate these schemes from their TCP counterparts in lossy networks. First is the avalanche effect of high-order HOL blocking [8], identified in the mid 80s as saturation trees [15]. A second key characteristic of datacenter and HPC networks is the prevalence of queuing delays over the transport lag [16].

Inspired by CUBIC and ECN, QCN introduces an *instantaneous* queue sensor sensitive to temporary bursts. Moreover, increasing the difference to ECN, is that QCN conveys explicit multibit feedback in the form of 64B CNM packets sent backwards directly to their culprit sources. QCN is further analyzed in [17], [18].

QCN-OS was designed and implemented as a practical solution to the high-order HOL blocking and saturation tree challenges specific to lossless CEE fabrics. Related to it, albeit in the lossy context, are push-out/back methods [19], [20]. Most recently, [21] applies a phantom queue latency reduction method for lossy datacenter fabrics. By associating congestion (price) with the link capacity, rather than with the buffer occupancy, it is an antithetical proposal.

## VII. Conclusions and Future Work

We have proposed occupancy sampling QCN for congestion detection at the input buffers of high-radix switches. QCN-OS detects overload, throttles the *actual* culprit flows, and maintains – via an appropriate $Q_{\text{eq}}$ setting – the input buffer backlog below the PFC-high threshold. The CNMs thus generated by the input congestion points neither consume bandwidth nor incur additional delays by traversing the switch fabric.

When comparing QCN-OS with QCN-AS, both at inputs, our results show significant performance improvements under IG hotspot scenarios. We attribute this to the correct culprit identification of the QCN-OS scheme. When comparing QCN-OS at inputs with the typical QCN-AS at outputs, our results in [1] show that the former has remarkably faster reaction under OG hotspot scenario. Furthermore, unlike QCN-AS at outputs, QCN-OS at inputs achieves strictly fair rate allocation of the output capacity under the IG scenario.

These results substantiate our first contributions: *(i)* A new QCN-compatible L2 congestion marking scheme, particularly amenable to input buffered modern switches; *(ii)* a simple analytical formulation of its sampling function as compared with the standard QCN; and *(iii)* a practical implementation that randomly picks an occupied unit within the buffer to identify a flow as a congestive culprit.

We also evaluated QCN-OS in scalable, multi-stage switching fabrics. We described why QCN-OS may fail when flows bottleneck at internal links, and how fabric-internal end-to-end reliable delivery schemes, by causing *additional backlogs* at inputs, may in fact *improve* the performance of QCN-OS at input VOQs to identify and throttle internally bottlenecked flows. Finally, we also proposed *two samples*, a practical scheme that can be used to identify internally bottlenecked flows, while maintain QCN-OS's benefits. Two samples also provides a general framework on how to build a *multi-criterion* culprit selection method on top of QCN.

Currently we are performing the next steps, such as testing against more complex traffic scenarios with more flows in faster (40/100G) and larger fabrics.

## References


[1] F. Neeser, N. Chrysos, R. Clauberg, D. Crisan, M. Gusat, C. Minkenberg, K. Valk, and C. Basso, "Occupancy Sampling for Terabit CEE Switches," in *Proc. IEEE Hot Interconnects 2012*.

[2] *802.1Qau - Virtual Bridged Local Area Networks - Amendment: Congestion Notification*, IEEE Std., 2010. [Online]. Available: http://www.ieee802.org/1/pages/802.1au.html

[3] I. Rhee and L. Xu, "CUBIC: A new TCP-friendly high-speed TCP variant," in *Proc. PFLDnet*, Lyon, France, February 2005.

[4] S. Scott, D. Abts, J. Kim, and W. J. Dally, "The BlackWidow high-radix Clos network," in *ISCA*. IEEE Computer Society, 2006, pp. 16–28.

[5] G. Passas, M. Katevenis, and D. N. Pnevmatikatos, "A 128 x 128 x 24 Gb/s crossbar interconnecting 128 tiles in a single hop and occupying 6% of their area," in *NOCS*, Grenoble, France, May 2010.

[6] M. Alizadeh, B. Atikoglu, A. Kabbani, A. Lakshmikantha, R. Pan, B. Prabhakar, and M. Seaman, "Data center transport mechanisms: Congestion control theory and IEEE standardization," in *Proc. Allerton Conference on Communication, Control, and Computing*, Sept. 2008.

[7] S.-T. Chuang, A. Goel, N. McKeown, and B. Prabhakar, "Matching output queueing with a combined input/output-queued switch," *IEEE JSAC*, vol. 17, pp. 1030–1039, June 1999.

[8] T. Jurczyk, M. Schwederski, "Phenomenon of higher order head-of-line blocking in multistage interconnection networks under nonuniform traffic patterns," *IEICE Trans. on Information and Systems*, vol. 79, no. 8, pp. 1124–1129, Aug. 1996.

[9] C. Minkenberg and G. Rodriguez, "Trace-driven co-simulation of high-performance computing systems using OMNeT++," in *Proc. 2nd SIMUTools Internat. Workshop on OMNeT++*, Rome, Italy, March 2009.

[10] A. Varga, "The OMNeT++ discrete event simulation system," in *Proc. European Simulation Multiconference (ESM 2001)*, Prague, Czech Republic, June 2001.

[11] M. Wadekar, "CN-SIM: Topologies and workloads," Feb. 8 2007. [Online]. Available: http://www.ieee802.org/1/files/public/docs2007/au-sim-wadekar-reqd-extended-sim-list020807.pdf

[12] N. Chrysos and M. Katevenis, "Scheduling in non-blocking buffered three-stage switching fabrics," in *Proceedings of IEEE INFOCOM*, Barcelona, Spain, April 2006.

[13] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. on Networking*, vol. 1, no. 4, pp. 397–413, August 1993.

[14] S. Athuraliya, S. Low, V. Li, and Q. Yin, "REM: Active queue management," *Network, IEEE*, vol. 15, no. 3, pp. 48–53, May 2001.

[15] G. Pfister and V. Kumar, "The onset of hotspot contention," in *Proc. ICPP*, University Park, PA, August 1986.

[16] M. Gusat, R. Birke, and C. Minkenberg, "Delay-based cloud congestion control," in *Proc. IEEE GLOBECOM*, Honolulu, HI, December 2009.

[17] Y. Lu *et al.*, "Congestion control in networks with no congestion drops," in *Proc. 44th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Sept. 2006.

[18] J. Jiang and R. Jain, "Analysis of backward congestion notification (BCN) for Ethernet in datacenter applications," in *Proc. IEEE INFOCOM*, Anchorage, AK, May 2007, pp. 2456–2460.

[19] S. Fong, S. Singh, and M. Atiquzzaman, "An analytical model and performance analysis of shared buffer ATM switches under non-uniform traffic," *J. of CSSE, Special Issue on ATM Networks*, pp. 125–137, 1997.

[20] R. Pan, B. Prabhakar, and K. Psounis, "CHOKe - a stateless active queue management scheme for approximating fair bandwidth allocation," in *IEEE INFOCOM*, Tel Aviv, Israel, March 2000.

[21] M. Alizadeh, A. Kabbani, T. Edsall, B. Prabhakar, A. Vahdat, and M. Yasuda, "Less is more: Trading a little bandwidth for ultra-low latency in the data center,," in *Proc. of the 9th ACM/USENIX Symposium on NSDI, San Jose, CA*, 2012.