

Capturing Router Congestion and Delay

Nicolas Hohn, Konstantina Papagiannaki, and Darryl Veitch, *Senior Member, IEEE*

Abstract—Using a unique monitoring experiment, we capture all packets crossing a (lightly utilized) operational access router from a Tier-1 provider, and use them to provide a detailed examination of router congestion and packet delays. The complete capture enables not just statistics as seen from outside the router, but also an accurate physical router model to be identified. This enables a comprehensive examination of congestion and delay from three points of view: the understanding of origins, measurement, and reporting. Our study defines new methodologies and metrics. In particular, the traffic reporting enables a rich description of the diversity of microcongestion behavior, without model assumptions, and at achievable computational cost.

Index Terms—Busy period, congestion, delay, modelling, router, utilization.

I. INTRODUCTION

END-TO-END packet delay is one of the canonical metrics in Internet Protocol (IP) networks, and is important both from the network operator and application performance points of view. For example the quality of Voice Over IP is directly dependent on delay, and network providers may have Service Level Agreements (SLAs) specifying allowable values of delay statistics across the domains they control. An important component of end-to-end delay is that due to forwarding elements, the fundamental building block of which is the delay incurred when a packet passes through a single IP router.

The motivation for the present work is a detailed knowledge and understanding of such “through-router” delays. A thorough examination of delay leads inevitably to deeper questions about congestion, and router queueing dynamics in general. We provide a comprehensive examination of these issues from three points of view: the understanding of origins, measurement, and reporting, all grounded in a unique data set taken from a router in the access network of a Tier-1 provider.

Although there have been many studies examining delay statistics and congestion measured at the edges of the network, very few have been able to report with any degree of authority on what actually occurs at switching elements. In [1] an analysis of single hop delay on an IP backbone network was presented. Only samples of the delays experienced by packets, on some links, were identified. In [2] single hop delays were also ob-

tained for a router. However since the router only had one input and one output link, which were of the same speed, the internal queueing was extremely limited. In this paper we work with a data set recording all IP packets traversing a Tier-1 access router over a 13 hour period. All input and output links were monitored, allowing a complete picture of congestion, and in particular router delays, to be obtained.

This paper is based on a synthesis of material first presented in two conference papers [3] and [4] based on the above data set. Section VI-A contains new results. In part because it is both costly and technically challenging to collect, this data set, although now a few years old, to the best of our knowledge remains unique. Consequently, the same is true of the kind of analysis it enables as presented here. What we emphasize in this presentation are the methodologies and approaches which have generic value. However, we believe that the detail and depth of the data analysis also makes this study worthwhile from the data archival point of view. Our specific conclusions are strictly speaking limited to a single data set with low loss and delay. However, since Tier-1 networks tend to be underprovisioned, we believe it remains representative of backbone access routers and their traffic today. This conjecture requires testing against more data sets.

The first aim of this paper is a simple one, to exploit this unique data set by reporting in detail on the magnitudes, and also the temporal structure, of delays on high capacity links with nontrivial congestion. The result is one of the most comprehensive pictures of router delay performance that we are aware of. As our analysis is based on empirical results, it is not reliant on assumptions on traffic statistics or router operations.

Our second aim is to use the completeness of the data as a tool to investigate how packet delays occur inside the router. In other words, we aim to provide a physical model capable of explaining the observed delay and congestion. Working in the context of the popular store and forward router architecture [5], we are able to justify the commonly held assumption that the bottleneck of such an architecture is in the output buffers, and thereby validate the fluid output queue model relied on routinely in the field of active probing. We go further to define a refined model with an accuracy close to the limits of timestamping precision, which is robust to many details of the architecture under reasonable loads.

Packet delays and congestion are fundamentally linked, as the former occur precisely because periods of temporary resource starvation, or *microcongestion episodes*, are dealt with via buffering. Our third contribution is an investigation of the origins of such episodes, driven by the question, “What is the dominant mechanism responsible for delays?”. We use a powerful methodology of virtual or “semi-” experiments, that exploits both the availability of the detailed packet data, and the fidelity of the router model. We identify, and evaluate the contributions of, three known canonical mechanisms: i) reduction in link bandwidth from core to access; ii) multiplexing of multiple input streams; iii) burstiness of the input traffic stream(s).

Manuscript received October 08, 2007; revised February 14, 2008; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor P. Barford. First published July 25, 2008; current version published June 17, 2009.

N. Hohn was with the ARC Special Research Centre for Ultra-Broadband Information Networks (CUBIN), The University of Melbourne, and also with Sprint ATL when this work was done (e-mail: nicolas.hohn@gmail.com).

K. Papagiannaki was with Sprint ATL and is now with Intel Research, Pittsburgh, PA 15213 USA (e-mail: dina.papagiannaki@intel.com).

D. Veitch is with the ARC Special Research Centre for Ultra-Broadband Information Networks (CUBIN), Department of Electrical and Electronic Engineering, The University of Melbourne, Melbourne, Australia 3010. He was on leave at Sprint ATL during this work (e-mail: dveitch@unimelb.edu.au).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TNET.2008.927258

To our knowledge, such a taxonomy of link congestion has not been used previously, and our findings for this specific data set are novel and sometimes surprising. Our broader contribution however is the methodology itself, including a set of metrics which can be used to examine the origins of congestion and hence delay.

The fourth part of the paper gives an innovative solution to a nontrivial problem: how the complexities of microcongestion and associated delay behavior can be measured and summarized in a meaningful way. We explain why our approach is superior to attempting to infer delay behavior simply from utilization, an approach which is in fact fatally flawed. In addition, we show how this can be done at low computational cost, enabling a compact description of congestion behavior to form part of standard router reporting. A key advantage is that a generically rich description is reported, without the need for any traffic assumptions.

The paper is organized as follows. The router measurements are presented in Section II, and analyzed in Section III, where the methodology and sources of error are described in detail. In Section IV we construct and justify the router model, and discuss its accuracy. Section V analyzes the origins of microcongestion episodes, whereas Section VI shows how they can be captured in a simple way, and efficiently reported.

II. FULL ROUTER MONITORING

In this section, we describe the hardware involved in the passive measurements, present the router monitoring setup, and detail how packets from different traces are matched.

A. Hardware Considerations

1) *Router Architecture*: Our router is of *store and forward* type, and implements *Virtual Output Queues* (VOQ) [5]. It is comprised of a switching fabric controlled by a centralized scheduler interconnecting *linecards* that process traffic from one or more *interfaces* which each control two *links*: one input and one output.

A packet *arrives* when it has exited the input link and is **stored** in linecard memory. VOQ means that each linecard has a dedicated first-in-first-out (FIFO) queue for each output interface. After consulting the forwarding table, the packet is stored in the appropriate VOQ, where it is decomposed into fixed length cells, to be transmitted through the switching fabric when the packet reaches the head of line (possibly interleaved with competing cells from VOQs at other linecards headed for the same output). At the output linecard the cells are re-assembled before being **forwarded** to the relevant output link scheduler. The packet might then experience queueing before being serialized onto the output link. In queueing terminology, it is “served” at a rate equal to the output link capacity. The packet might be queued both at the input and output linecards, however in practice the switch fabrics are overprovisioned and therefore very little queueing should be expected at the input queues.

2) *Layer Overheads*: Each interface on the router uses the High Level Data Link Control (HDLC) protocol as a transport layer to carry IP datagrams over a Synchronous Optical Network (SONET) physical layer, a popular combination known as Packet over SONET (PoS).

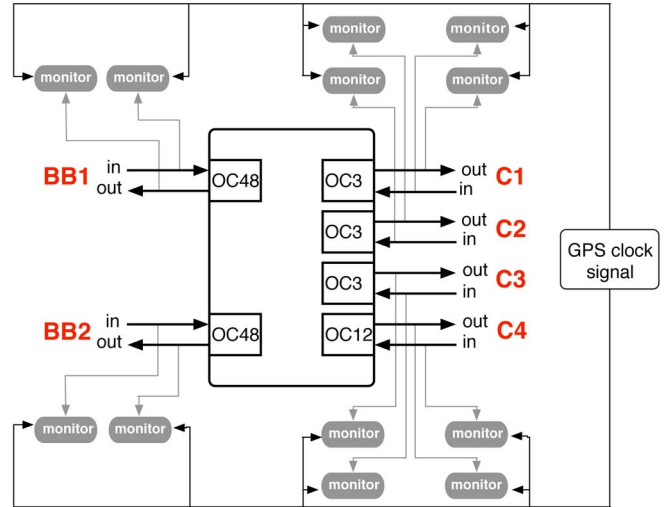


Fig. 1. Router Monitoring: 12 links over six interfaces (all active interfaces on four linecards) were instrumented with synchronized DAG cards.

A SONET OC-1 frame, including all overheads, offers an *IP bandwidth* of 49.92 Mb/s. OC- n bandwidth (with $n \in \{3, 12, 48, 192\}$) is achieved by merging n basic frames into a single larger one, yielding an IP bandwidth of $(49.92 \cdot n)$ Mb/s. The second level of encapsulation is the HDLC transport layer, which adds 5 bytes before and 4 bytes after each IP datagram, irrespective of SONET interface speed [6]. Hence, an IP datagram of size b bytes carried over OC-3 should be considered as a $b + 9$ byte packet transmitted at 149.76 Mb/s.

3) *Timestamping of PoS Packets*: All measurements are made using high performance passive monitoring DAG cards [7]. We use DAG 3.2 cards to monitor OC-3c and OC-12c links, and DAG 4.11 cards to monitor OC-48 links.

DAG 4.11 cards are dedicated to PoS measurement. They look past the PoS encapsulation (in this case HDLC) to consistently timestamp each IP datagram after the first (32 bit) word has arrived. DAG 3.2 cards have PoS functionality added on top of an ATM based architecture. As a result, timestamps on OC-3 links have a worst case precision of $2.2 \mu\text{s}$ [8]. Adding errors due to potential GPS synchronization problems between different DAG cards leads to a worst case error of $6 \mu\text{s}$ [9].

B. Experimental Setup

The data was collected in August 2003 at a gateway router of the Sprint IP backbone network. The router had four linecards supporting six active interfaces: 1: OC-48 (BB1); 2: OC-48 (BB2); 3: OC-12 (C4); and 4: OC-3 (C1, C2, C3). Traffic on 11 links over the 6 interfaces was monitored, accounting for more than 99.95% of all through-router traffic (link C3-in (< 5 pps) was not systematically monitored, and a 7th interface on a 5th GigE linecard with negligible traffic, was not monitored).

The experimental setup is illustrated in Fig. 1. The interfaces BB1 and BB2 connect to two backbone routers, while the other four connect customer links: two trans-Pacific OC-3 links to Asia (C2 and C3), and one OC-3 (C1) and one OC-12 (C4) link to domestic customers.

The DAG cards were each synchronized to a common GPS signal, and output a fixed length 64 byte record for each packet. The details of the record depend on the link type (ATM, SONET

TABLE I
TRACES COLLECTED BETWEEN 03:30–16:30 UTC, AUG. 14, 2003

Set	Link	# packets	Rate (Mbps)	Matched (% total)	Copies (% total)	Router (% total)
BB1	in	817883374	83	99.87	0.045	0.004
	out	808319378	53	99.79	0.066	0.014
BB2	in	1143729157	80	99.84	0.038	0.009
	out	882107803	69	99.81	0.084	0.008
C1	out	103211197	3	99.60	0.155	0.023
	in	133293630	15	99.61	0.249	0.006
C2	out	735717147	77	99.93	0.011	0.001
	in	1479788404	70	99.84	0.050	0.001
C3	out	382732458	64	99.98	0.005	0.001
	in	16263	0.003	N/A	N/A	N/A
C4	out	480635952	20	99.74	0.109	0.008
	in	342414216	36	99.76	0.129	0.008

or Ethernet). Here all IP packets are PoS packets, and each record consists of 8 bytes for the timestamp, 12 for control and PoS headers, 20 for the IP header, and the first 24 bytes of the IP payload. We captured a trace for each interface over 13 hours, representing more than 7.3 billion IP packets or 3 Terabytes.

C. Packet Matching

Packet matching consists in identifying the records corresponding to the same packet appearing at different interfaces at different times. Matching is computationally intensive and demanding in terms of storage. Our methodology (see [1] for details) uses CRC based hashing and takes into account DAG packet padding. All unmatched packets were carefully analyzed. These either involve link C3-in (see above), or are sourced or sunk at the router interfaces themselves (0.01%). The router did not drop a single packet, but packet copies are occasionally produced by the physical layer (the reasons for this are not clear). We discard these, but monitor their frequency. Table I summarizes the matching statistics. The percentage of matched packets is extremely high.

We define a *matching function* \mathcal{M} and write

$$\mathcal{M}(\Lambda_j, m) = (\lambda_i, n) \quad (1)$$

if the m th packet of output link Λ_j corresponds to the n th packet of input link λ_i . The matching procedure effectively defines \mathcal{M} for all packets over all output links.

In the remainder of the paper we focus on C2-out, an OC-3 link, because it is the most highly utilized. Its traffic is overwhelming due to the two OC-48 backbone links BB1-in and BB2-in carrying 47.00% and 52.89% of its traffic respectively (almost equal due to the Equal Cost Multi Path policy deployed in the network when packets may follow more than one path to the same destination). This is clearly seen in the packet time series of Fig. 2 across the full 13 hours. The bytes time series has an almost identical shape.

III. EXTERNAL DELAY ANALYSIS

In this section, we view the router as a “black box” system, and concentrate on the statistics of delays crossing it. In the next section, we begin to look inside the router, and examine delays and congestion in greater detail.

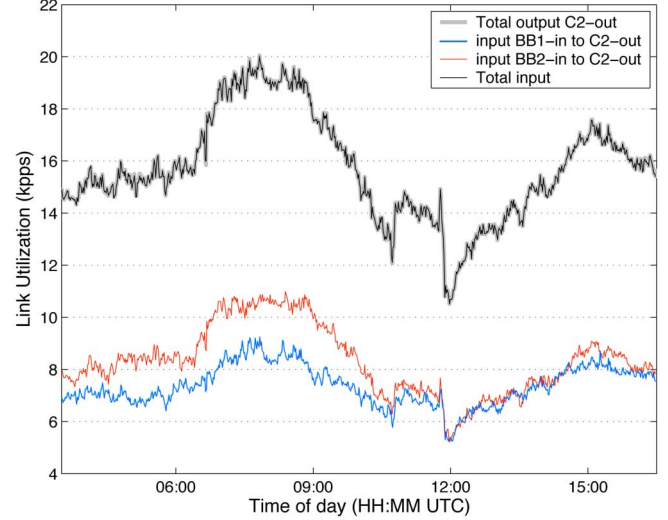


Fig. 2. Utilization for link C2-out in kilo packet per second (kpps).

A. System Definition

The DAG timestamps an IP packet on the incoming interface at time $t(\lambda_i, n)$, and on the outgoing interface at time $t(\Lambda_j, m)$. As the DAG cards are physically close to the router, one might think to define the through-router delay as $t(\Lambda_j, m) - t(\lambda_i, n)$. However, this ignores the bit position at which DAG timestamps are made. Furthermore, for a store and forward router no action is performed until the packet has fully entered the router. It is therefore convenient for the input buffer to be considered as part of the input link. Hence, we consider that the packet’s *arrival to the system* occurs just after the arrival of the last bit, and the *departure from the system* corresponds to the instant when the last bit leaves the output buffer. Thus, the *system*, the part of the router which we study, is not exactly the same as the physical router, as the input buffer (a component which is understood and does not require study) is excised.

We now establish the precise relationships between the DAG timestamps and the time instants $\tau(\lambda_i, n)$ of arrival and $\tau(\Lambda_j, m)$ of departure of a given packet to the system as just defined. Denote by $l_n = L_m$ the size of the packet in bytes when indexed on links λ_i and Λ_j respectively, and let θ_i and Θ_j be the corresponding link bandwidths in bits per second. We denote by H the function giving the depth of bytes into the IP packet where the DAG timestamps it. H is a function of the link speed, but not the link direction. For a given link λ_i , H is defined as

$$H(\lambda_i) = 4 \quad \text{if } \lambda_i \text{ is an OC - 48 link} \\ = b \quad \text{if } \lambda_i \text{ is an OC - 3 or OC - 12 link}$$

where we take b to be a uniformly distributed integer between 0 and $\min(l_n, 53)$ to account for the ATM-based discretization. We can now derive the desired system arrival and departure event times as

$$\tau(\lambda_i, n) = t(\lambda_i, n) + 8(l_n - H(\lambda_i)) / \theta_i \\ \tau(\Lambda_j, m) = t(\Lambda_j, m) + 8(L_m - H(\Lambda_j)) / \Theta_j. \quad (2)$$

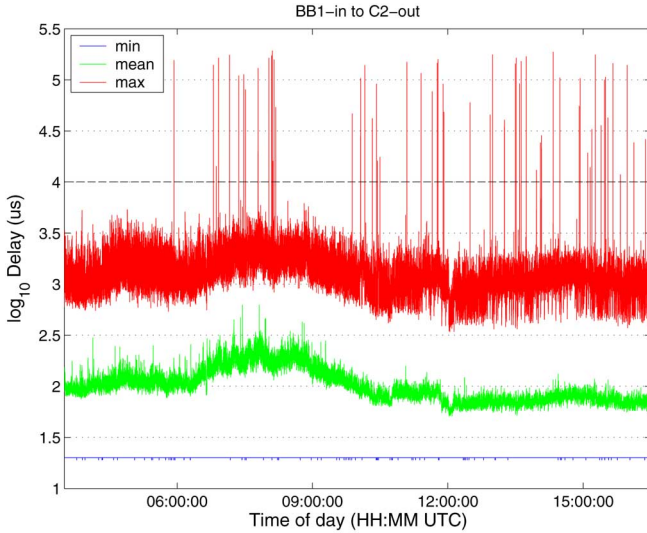


Fig. 3. Delays: BB1-in to C2-out. Those >10 ms are due to option packets.

With the above notations, the through-system delay experienced by packet m on link Λ_j is defined as

$$d_{\lambda_i, \Lambda_j}(m) = \tau(\Lambda_j, m) - \tau(\lambda_i, n). \quad (3)$$

To simplify notations we shorten this to $d(m)$ in what follows.

B. Delay Statistics

A thorough analysis of single hop delays was presented in [1]. Here we follow a similar methodology and obtain comparable results, but with the added certainty gained from not needing to address the sampling issues caused by unobservable packets on the input side.

Fig. 3 shows the minimum, mean and maximum delay experienced by packets going from input link BB1-in to output link C2-out over consecutive 1 minute intervals. As observed in [1], there is a constant minimum delay across time, up to timestamping precision. The fluctuations in the mean delay follow roughly the changes in the link utilization presented in Fig. 2. The maximum delay value has a noisy component with similar variations to the mean, as well as a spiky component. All the spikes above 10 ms have been individually studied. The analysis revealed that they are caused by IP packets carrying options, representing less than 0.0001% of all packets. Option packets take different paths through the router since they are processed through software, while all other packets are processed with dedicated hardware on the so-called “fast path.”

In any router architecture it is likely that many components of delay will be proportional to packet size. This is certainly the case for store and forward routers, as discussed in [10]. To investigate this here we compute the “excess” minimum delay experienced by packets of different sizes, that is not including their transmission time on the output link, a packet size dependent component which is already understood. Formally, for every packet size L we compute

$$\Delta_{\lambda_i, \Lambda_j}(L) = \min_m \{d_{\lambda_i, \Lambda_j}(m) - 8l_m/\Theta_j | l_m = L\}. \quad (4)$$

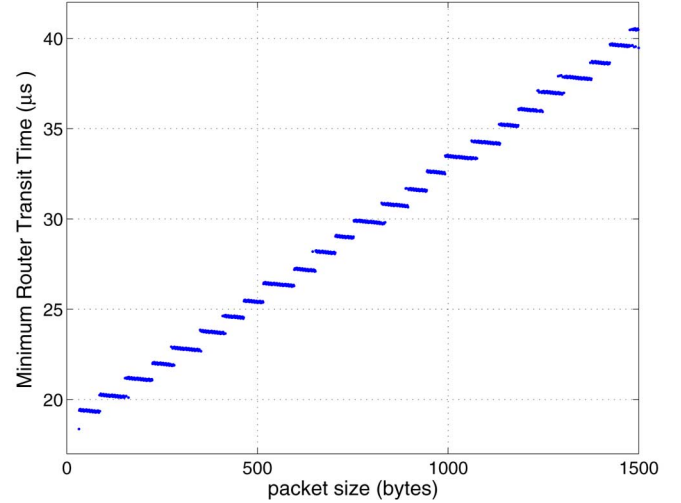


Fig. 4. Measured minimum excess system transit times: BB1-in to C2-out.

Fig. 4 shows the values of $\Delta_{\lambda_i, \Lambda_j}(L)$ for packets going from BB1-in to C2-out. The IP packet sizes observed varied between 28 and 1500 bytes. We assume that these are true minima, in other words that the system was empty from the point of view of this input–output pair. This means that the excess minimum delay corresponds to the time taken to make a forwarding decision (not packet size dependent), to divide the packet into cells, transmit it across the switch fabric and reassemble it (each being packet size dependent operations), and finally to deliver it to the appropriate output queue. The step-like curve means that there exist ranges of packet sizes with the same minimum transit time, consistent with the fact that each packet is divided into fixed length cells, transmitted through the backplane cell by cell, and reassembled. A given number of cells can therefore correspond to a contiguous range of packet sizes with the same minimum transit time.

IV. MODELLING

We are now in a position to exploit the completeness of the data set to look inside the system. This enables us to find a physically meaningful model which can be used both to understand and predict the end-to-end system delay.

A. The Fluid Queue

We first recall some basic properties of FIFO queues. Consider a FIFO queue with a single server of deterministic service rate μ , and let t_i be the arrival time to the system of packet i of size l_i bytes. We assume that the entire packet arrives instantaneously (which models a fast transfer across the switch), but it leaves progressively as it is served (modelling the output serialization). Thus it is a fluid queue at the output but not at the input. Nonetheless we will for convenience refer to it as the “fluid queue.”

Let W_i be the time packet i waits before being served. The service time of packet i is simply l_i/μ , so the system time, that is, the total amount of time spent in the system, is

$$S_i = W_i + \frac{l_i}{\mu}. \quad (5)$$

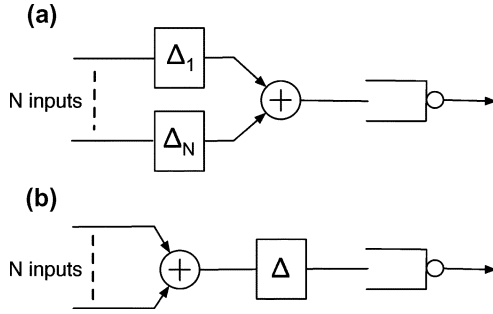


Fig. 5. Router mechanisms: (a) simple conceptual picture including VOQs; (b) actual model with a single common minimum delay.

The waiting time of the next packet ($i + 1$) to enter the system can be expressed by the following recursion:

$$W_{i+1} = \left[W_i + \frac{l_i}{\mu} - (t_{i+1} - t_i) \right]^+ \quad (6)$$

where $[x]^+ = \max(x, 0)$. The service time of packet $i + 1$ is

$$S_{i+1} = [S_i - (t_{i+1} - t_i)]^+ + \frac{l_{i+1}}{\mu}. \quad (7)$$

We denote by $U(t)$ the amount of *unfinished work* at time t , that is the time it would take, with no further inputs, for the system to completely drain. The unfinished work at the instant following the arrival of packet i is nothing other than the end-to-end delay that packet will experience across the queueing system. Note that it is defined at all real times t .

B. A Simple Router Model

The delay analysis of Section III revealed two main features of the system delay which should be taken into account: the minimum delay experienced by a packet, which is size, interface, and architecture dependent, and the delay corresponding to the time spent in the output buffer, which is a function of the rate of the output interface and the occupancy of the queue. The delay across the output buffer could be modelled by the fluid queue as described above, however it is not immediately obvious how to incorporate the minimum delay property.

Assume for instance that the router has N input links $\lambda_1, \dots, \lambda_N$ contributing to a given output link Λ_j and that a packet of size l arriving on link λ_i experiences at least the minimum possible delay $\Delta_{\lambda_i, \Lambda_j}(l)$ before being transferred to the output buffer [Fig. 5(a)]. Our first problem is that given different technologies on different interfaces, the functions $\Delta_{\lambda_1, \Lambda_j}, \dots, \Delta_{\lambda_N, \Lambda_j}$ are not necessarily identical. The second is that we do not know how to measure, nor to take into account, the potentially complex interactions between packets which do *not* experience the minimum excess delay but some larger value due to contention in the router arising from cross traffic.

We address this by simplifying the picture still further, in two ways. First we assume that the minimum delays are identical across all input interfaces: a packet of size l arriving on link λ_i and leaving the router on link Λ_j now experiences an excess minimum delay

$$\Delta_{\Lambda_j}(l) = \min_i \{ \Delta_{\lambda_i, \Lambda_j}(l) \}. \quad (8)$$

In the following we drop the subscript Λ_j to ease the notation. Second, we assume that the multiplexing of the different input streams takes place before the packets experience their minimum delay. By this we mean that we preserve the order of their arrival times and consider them to enter a single FIFO input buffer. In doing so, we effectively ignore all complex interactions between the input streams. Our highly simplified picture, which is in fact the model we propose, is shown in Fig. 5(b). We will justify these simplifications *a posteriori* in Section IV-C where the comparison with measurement shows that the model is very accurate. We now explain why we can expect this accuracy to be robust.

Suppose that a packet of size l enters the system at time t^+ and that the amount of unfinished work in the system at time t^- was $U(t^-) > \Delta(l)$. The following alternative scenarios produce the same total delay:

- (i) the packet experiences a delay $\Delta(l)$, then reaches the output queue and waits $U(t) - \Delta(l) > 0$ before service;
- (ii) the packet reaches the output queue straight away and has to wait $U(t)$ before being served.

In other words, as long as there is more than an amount $\Delta(l)$ of work in the queue when a packet of size l enters the system, the fact that the packet should wait $\Delta(l)$ before reaching the output queue can be neglected. Once the system is busy, it behaves exactly like a simple fluid queue. This implies that no matter how complicated the front end of the router is, one can simply neglect it when the output queue is sufficiently busy. The errors made through this approximation will be strongly concentrated on packets with very small delays, whereas the more important medium to large delays will be faithfully reproduced.

A system equation for our two stage model can be derived as follows. Assume that the system is empty at time t_0^- and that packet k_0 of size l_0 enters the system at time t_0^+ . It waits $\Delta(l_0)$ before reaching the empty output queue where it immediately starts being served. Its service time is l_0/μ and therefore its total system time is

$$S_0 = \Delta(l_0) + \frac{l_0}{\mu}. \quad (9)$$

Suppose a second packet enters the system at time t_1 and reaches the output queue before the first packet has finished being served, i.e., $t_1 + \Delta(l_1) < t_0 + S_0$. It will start being served when packet k_0 leaves the system, i.e., at $t_0 + S_0$. Its system time will therefore be

$$S_1 = S_0 - (t_1 - t_0) + \frac{l_1}{\mu}.$$

The same recursion holds for successive packets k and $k + 1$ as long as the amount of unfinished work in the queue remains above $\Delta(l_{k+1})$ when packet $k + 1$ enters the system:

$$t_{k+1} + \Delta(l_{k+1}) < t_k + S_k. \quad (10)$$

Therefore, as long as (10) is verified, the system times of successive packets are obtained by the same recursion as for the case of a busy fluid queue:

$$S_{k+1} = S_k - (t_{k+1} - t_k) + \frac{l_{k+1}}{\mu}. \quad (11)$$

Suppose now that packet $k+1$ of size l_{k+1} enters the system at time t_{k+1}^+ and that the amount of unfinished work in the system

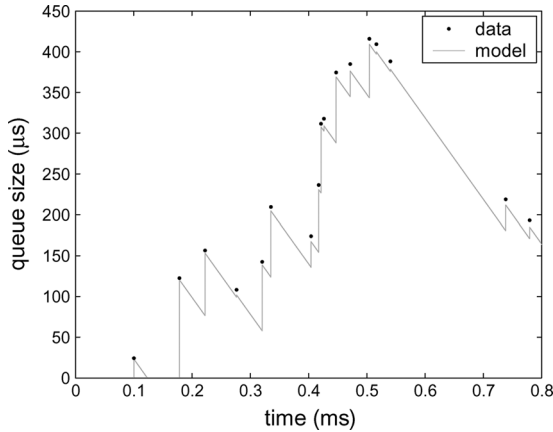


Fig. 6. Comparison of measured and predicted delays on link C2-out: grey line: unfinished work $U(t)$ in the system according to the model; black dots: measured delay value for each packet.

at time t_{k+1}^- is such that $0 < U(t_{k+1}^-) < \Delta(l_{k+1})$. In this case, the output buffer will be empty by the time packet $k+1$ reaches it after having waited $\Delta(l_{k+1})$ in the first stage of the model. The service time of packet $k+1$ therefore reads

$$S_{k+1} = \Delta(l_{k+1}) + \frac{l_{k+1}}{\mu}. \quad (12)$$

A crucial point to note here is that in this situation, *the output queue can be empty but the system still busy with a packet waiting in the front end*. This is also true of the actual router.

Once the queue has drained, the system is idle until the arrival of the next packet. The time between the arrival of a packet to the empty system and the time when the system becomes empty again defines a *system busy period*. In this brief analysis we have assumed an infinite buffer size, a reasonable assumption in a low loss context (it is quite common for a line card to be able to accommodate up to 500 ms worth of traffic).

C. Evaluation

We compare model results with empirical delay measurements. The model delays are obtained by using the system arrival timestamps of all packets destined for C2-out (almost all originated from BB1-in and BB2-in) and feeding the resulting multiplexed packet stream into the model in an exact trace driven simulation. Fig. 6 shows a sample path of the unfinished work $U(t)$ corresponding to a fragment of real traffic destined to C2-out. The process $U(t)$ is a right continuous jump process where each jump marks the arrival time of a new packet. The resultant new local maximum is the time taken by the newly arrived packet to cross the system, that is its delay. The black dots represent the actual measured delays for the corresponding input packets. In practice the queue state can only be measured when a packet enters the system. Thus the black dots can be thought of samples of $U(t)$ obtained from measurements, and agreement between the two seems very good.

We now focus on a set of busy periods on link C2-out involving 510 packets. The top plot of Fig. 7 shows the system times experienced by incoming packets, both from the model and measurements. The largest busy period on the figure has a duration of roughly 16 ms and an amplitude of more than 5 ms. Once again, the model reproduces the measured delays very well. The lower plot in Fig. 7 shows the error, that is the

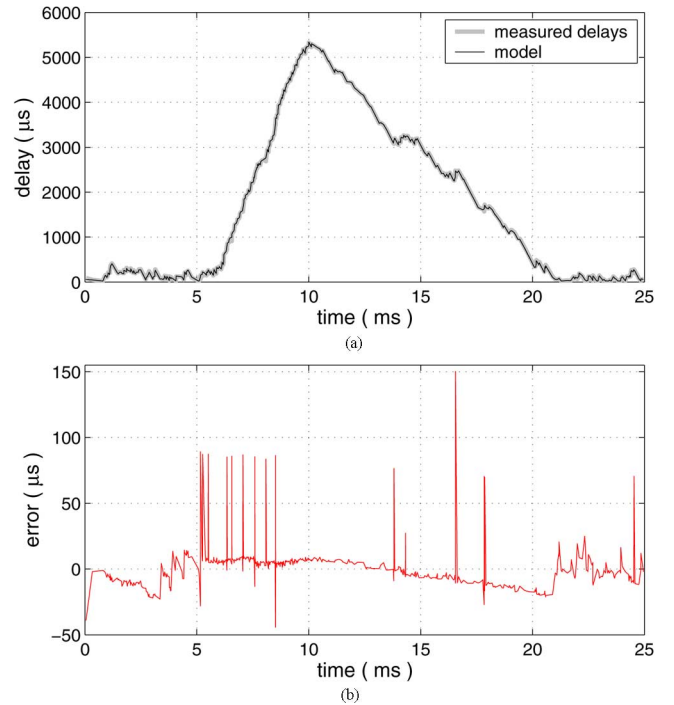


Fig. 7. (top) Measured delays and model predictions; (bottom) difference.

difference between measured and modeled delays at each packet arrival time.

There are three main points one can make about the model accuracy. First, the absolute error is within $30 \mu\text{s}$ of the measured delays for almost all packets. Second, the error is much larger for a few packets, as shown by the spiky behavior of the error plot. These spikes are due to a local reordering of packets inside the router that is not captured by our model. Local reordering typically occurs when a large packet arriving at the system is overtaken by a small one which arrives just after on another interface. Such errors are local only. They do not accumulate since once both packets have reached the output buffer, the amount of work in the system is the same irrespective of arrival order. Local reordering requires that two packets arrive almost at the same time on different interfaces. This is much more likely to happen when the links are busy, in agreement with Fig. 7 which shows that spikes always occur when the queueing delays are increasing.

The last point is the systematic linear drift of the error across a busy period duration, indicating that our queueing model drains slightly faster than the real queue. We could not confirm any physical reason why the IP bandwidth of the link C2-out is smaller than what was predicted in Section II-A2. However, the important observation is that this phenomenon is only noticeable for very large busy periods, and is lost in measurement noise for most busy periods.

Despite its simplicity, our model is considerably more accurate than other single-hop delay models. Fig. 8(a) compares the errors made on the packet delays from the OC-3 link C2-out presented in Fig. 7 with three different models: our two stage model, a fluid queue with OC-3 nominal bandwidth, and a fluid queue with OC-3 IP bandwidth. As expected, with a simple fluid model, i.e., when one does not take into account the minimum transit time, all the delays are systematically underesti-

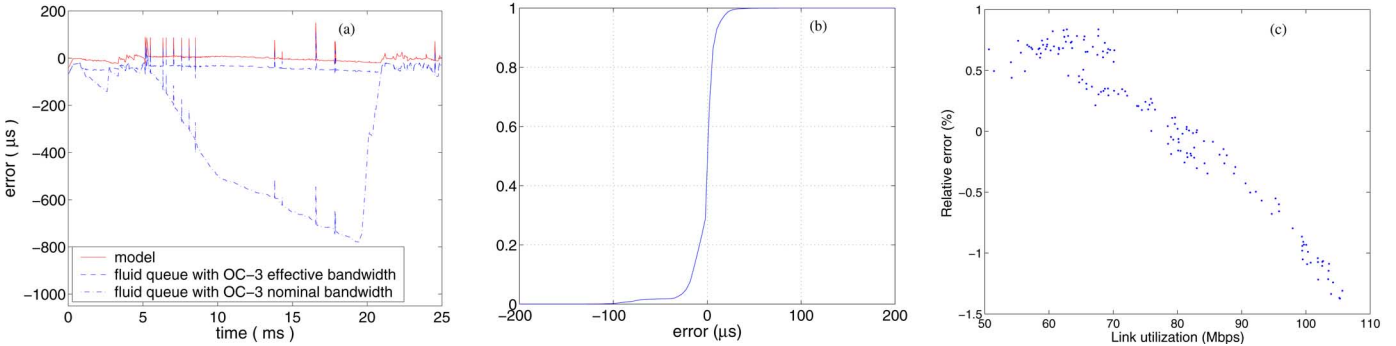


Fig. 8. (a) Comparison of error in delay predictions from different models of the sample path from Fig. 7. (b) Cumulative distribution function of model error over a 5 minute window on link C2-out. (c) Relative mean error between delay measurements and model on link C2-out versus link utilization.

ated. If moreover one chooses the nominal link bandwidth (155.52 Mb/s) for the queue instead of a carefully justified IP bandwidth (149.76 Mb/s), the errors inside a busy period build up very quickly because the queue drains too fast. There is in fact only a 4% difference between the nominal and effective bandwidths, but this is enough to create errors up to 800 μs inside a moderately large busy period!

Fig. 8(b) shows the cumulative distribution function of the delay error for a 5 minute window of C2-out traffic. Of the delays inferred by our model, 90% are within 20 μs of the measured ones. Given the timestamping precision issues described in Section II-A3, these results are very satisfactory.

Finally, we evaluate the performance of the model on C2-out over the entire 13 hours as follows. We divide the period into 156 intervals of 5 min. For each interval, we plot the average relative delay error against the average link utilization. The results, presented in Fig. 8(c), show an absolute relative error less than 1.5% over the whole trace.

D. Router Model Summary

We propose the following simple approach for modeling store and forward routers. For each output link Λ_j :

- (i) Measure the minimum excess packet transit time $\Delta_{\lambda_i, \Lambda_j}$ between each input λ_i and the given output Λ_j (4). Define the overall minimum packet transit time Δ_{Λ_j} as the minimum over all input links λ_i , as per (8).
- (ii) Calculate the IP bandwidth of the output link, taking into account packet encapsulation (Section II-A2).
- (iii) Obtain packet delays by aggregating the input traffic corresponding to the given output link, and feeding it to a simple two stage model [Fig. 5(b)], where packets are first delayed by an amount Δ_{Λ_j} before entering a FIFO queue. System equations are given in Section IV-B.

A model of a full router can be obtained by putting together the models obtained for each output link Λ_j .

Although simple, this model performed very well for our data set, where the router was lightly loaded and the output buffer was clearly the bottleneck. We expect the model to continue to perform well even under heavier load where interactions in the front end become more pronounced, but not dominant. The accuracy would drop off under loads heavy enough to shift the bottleneck to the switching fabric.

V. UNDERSTANDING MICROCONGESTION

In this section, we exploit the model and methodology developed above to investigate the microcongestion in our data in

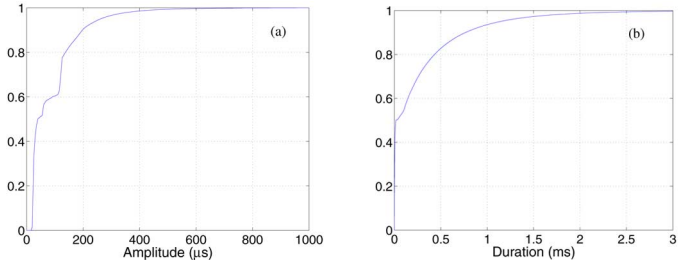


Fig. 9. Busy period statistics: (a) CDF of amplitudes; (b) CDF of durations.

more detail, and then proceed to inquire into its origins. The key concept is that of a system *busy period*.

A *busy period* is the time between the arrival of a packet to the empty system and the time when it returns to its empty state. Equivalently: a busy period starts when a packet of size l bytes crosses the system with a delay $\Delta(l) + l/\mu$, and ends with the last packet before the start of another busy period. This definition is far more robust than one based solely on packet inter-arrival times at the output link. If one were to detect busy periods by using timestamps and packet sizes to group together back-to-back packets, timestamping errors could lead to wrong busy period separations. More importantly, according to (12) from Section IV-B, packets belonging to the same busy period are not necessarily back-to-back at the output.

To understand the nature of the busy periods in this data, we begin by collecting per busy period statistics, such as duration, number of packets and bytes, and *amplitude* (maximum delay experienced by a packet inside the busy period). The cumulative distribution functions (CDF) of busy period amplitudes and durations are plotted in Fig. 9(a) and (b) for a 5 minute traffic window, in which 90% of busy periods have an amplitude smaller than 200 μs , and 80% last less than 500 μs . We expand upon the study of busy period durations and amplitudes in Section VI.

A. Busy Period Origins: a First Look

The completeness of the data set, combined with the router model, allows us to empirically answer essentially any question regarding the formation and composition of busy periods, or on the utilization and delay aspects of congestion, that we wish. In queueing terminology we have full access to the entire sample path of both the input and output processes, and the queueing system itself. Remarkable as this is, we can however go much further.

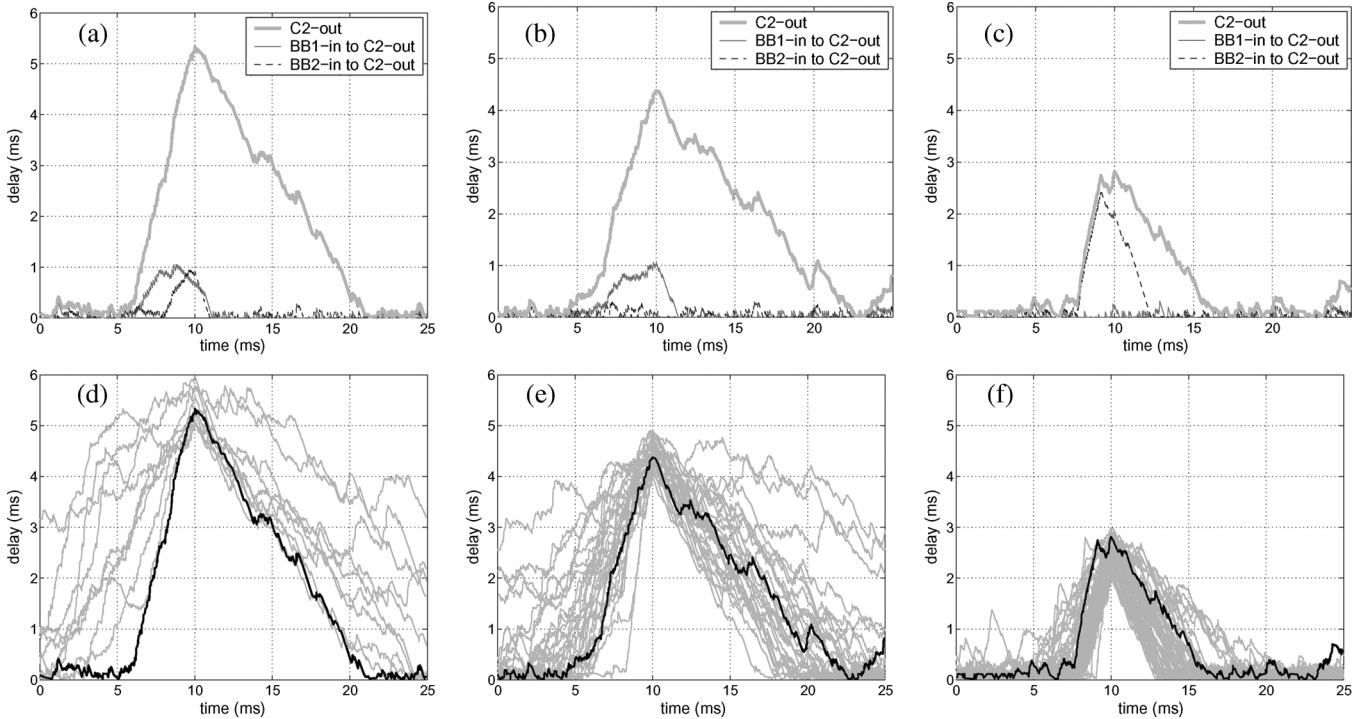


Fig. 10. (a), (b), (c) Illustration of the multiplexing effect leading to a busy period on the output link C2-out. (d), (e), (f) Collection of largest busy periods in each 5 minute interval on the output link C2-out.

We use our knowledge of the input packet streams on each interface, combined with the exact trace simulation idea used in evaluation methodology of Section IV-C, as a means of learning what the busy periods *would have* looked like under different circumstances. In this way we can investigate the mechanisms that create microcongestion observed at the output links. To illustrate the power of this, consider the three examples in the top row of Fig. 10. Each is centered on a different large busy period of C2-out, accompanied by the *virtual* busy periods obtained by individually feeding the packet streams BB1-in to C2-out, and BB2-in to C2-out into the model (recall that these together account for almost all of the traffic appearing at C2-out).

In Fig. 10(a) (the same time window as in Fig. 7) it appears as if the largest delays from C2-out are generated by busy periods from the individual input streams which peak at around the same time. However, there is a strong nonlinear effect, as the component amplitudes are each around 1 ms, whereas the amplitude for C2-out is around 5 ms. Fig. 10(b) is a more extreme example of this nonlinearity, showing one input stream creating at most a 1 ms packet delay by itself and the other only a succession of 200 μ s delays. The resulting multiplexed amplitude is nonetheless much larger than the individual ones. A different situation is shown in Fig. 10(c), where one link contributes almost all the traffic of the output link for a short time period.

From these examples it is apparent that it is not obvious how to adequately explain the formation of busy periods, in particular their amplitudes, based on the behavior of component substreams in isolation. This motivates the following deeper look.

B. Congestion Mechanisms

Fundamentally, all congestion is due to one thing—too much traffic. However, traffic can be built up or concentrated in different ways, leading to microcongestion of very different character. We examine the following three mechanisms.

Bandwidth Reduction Clearly, in terms of average rate, the input link of rate μ_i could potentially overwhelm the output link of rate $\mu_o < \mu_i$. This does not happen for our data over long timescales, but *locally* it can and does occur. The fundamental effect is that a packet of size p bytes, which has a width of p/μ_i seconds on the input wire, is stretched to p/μ_o seconds at the output. Thus, packets which are too close together at the input may be “pushed” together and forced to queue at the output. In this way busy periods at the input can only worsen: individually they are all necessarily stretched, and they may also then meet and merge with others. Furthermore new busy periods (larger than just a single packet) can be created which did not exist before. This “stretching” effect also corresponds, clearly, to an increase in link utilization, however it is the small scale effects, and the effect on delay, that we emphasize here. Depending on other factors, stretching can result in very little additional delay, or significant buildups.

Link Multiplexing For a given output link, input traffic will typically arrive from different traffic streams over different input links. Whether these streams are correlated or not, the superposition of multiple streams increases the packet density at all scales and thereby encourages both the creation of busy periods at the output, and the inflation of existing ones. To first order this is simply an additive increase in utilization level. Again however, the effect on delays could either be very small or significant, depending on other factors.

Flow Burstiness It is well known that traffic is highly variable or bursty on many timescales. The duration and amplitude of busy periods will depend upon the details of the packet spacings at the input, which is another way of saying that it depends on the input burstiness. For example packets which are already highly clustered can more easily form busy periods via the bandwidth-induced stretching above. To put it in a different way, beyond the first order effect of utilization level, effects at second order and above can have a huge impact on the delay process.

C. Metrics and Results

Just as for Fig. 10, we use the router model and the measurements of separate input traffic streams to virtually explore different scenarios (this approach can be generalized and has been called the semi-experimental method [11]).

The experiments take the following form. First, a “total” traffic stream S_T is selected. It is fed through the model with output rate μ , and the locations and characteristics of all the resulting busy periods are recorded. Note that even in the case when S_T is the full set of measured traffic, we still must use the model to determine when busy periods begin and end, as we can only measure the system when packets arrive or depart, whereas the model operates in continuous time.

For a given busy period we denote its starting time by t_s , its *duration* by D , its *amplitude*, that is the maximum of the workload function (the largest of the delays $\{d_j\}$ suffered by any packet), by A , and let t_A be the time when it occurred. When we need to emphasize the dependence on the stream or the link bandwidth, we write $A(S_T, \mu)$ and so on.

We next select a substream S_S of traffic according to some criteria. We wish to know the extent to which the substream contributes to the busy periods of the total stream. We evaluate this by feeding the substream into the model, since the detailed timing of packet arrivals is crucial to their impact on busy period shape and amplitude. The focus remains on the busy periods of the total stream even though the substream has its own busy period structure. Specifically, for each busy period of S_T we will look at the contribution from S_S appearing in the interval $[t_s, t_A]$ during which it was building up to its maximum A . Exactly how to measure the contribution will vary depending upon the context.

It is in fact not possible in general to fully separate the congestion mechanisms, as the busy period behavior is a result of a detailed interaction between all three. The extent to which separation is feasible will become apparent as the results unfold.

D. Reduction in Bandwidth

To fix ideas, we illustrate the first mechanism in Fig. 11. The two bar plots visualize the locations of busy periods on the OC-48 input link (lower bar), and the resulting busy periods following transmission to the smaller OC-3 output link (upper bar). For the latter, we also graph the corresponding system workload induced by the packets arriving to the busy period, obtained using the model. We clearly see that the input busy periods, which consist typically of just one or two packets, have been stretched and merged into a much smaller number of much longer busy periods at the output.

In order to quantify the stretching effect, we perform a virtual experiment where the total traffic stream S_T is just one of the main input OC-48 streams. By restricting to just a single

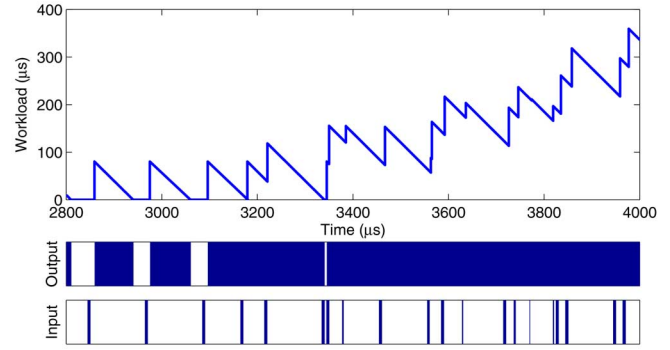


Fig. 11. The bandwidth reduction effect. Bottom: input/output bar plots: busy periods on the OC-48 input and OC-3 output links. Top: resulting OC-3 queuing workload process. The output busy periods are longer and far fewer than at the input.

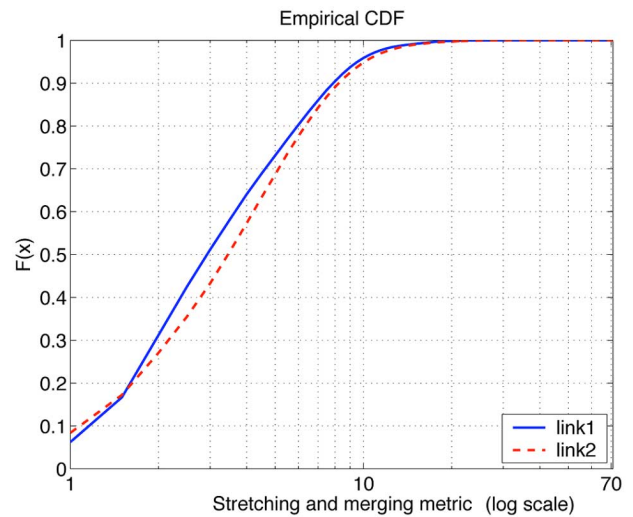


Fig. 12. Empirical distribution functions of AF for the OC-48 input streams.

input stream, we can study the effect of link bandwidth reduction without interference from multiplexing across links.

In this case, our substream is the same as the total stream ($S_S = S_T$), but evaluated at a different link rate. We quantify “stretching and merging” relative to a given busy period of S_T using the normalized *amplification factor*

$$AF = \frac{A(S_S, \mu_o)}{\max_k A_k(S_T, \mu_i)} \frac{\mu_o}{\mu_i}$$

where $AF \geq 1$. The amplitude for the substream is evaluated across all k busy periods (or partial busy periods) of S_T that fall in $[t_s, t_A]$ of S_S .

In simple cases where packets are well separated, so that all busy periods at both the input and output consist of just a single packet, then stretching is purely linear and $AF = 1$. If queuing occurs so that nontrivial busy periods form at the output, then $AF > 1$. The size of AF is an indication of the extent of the delay increase due to stretching. If the utilization at the output exceeds 1 then it will grow without bound over time.

We present the cumulative distribution function for AF in Fig. 12 for each of the main input streams separately. Less than 5% of the busy periods are in the linear regime with minimal delay detected via $AF = 1$. The majority are significantly amplified by the nonlinear merging of input busy periods into larger

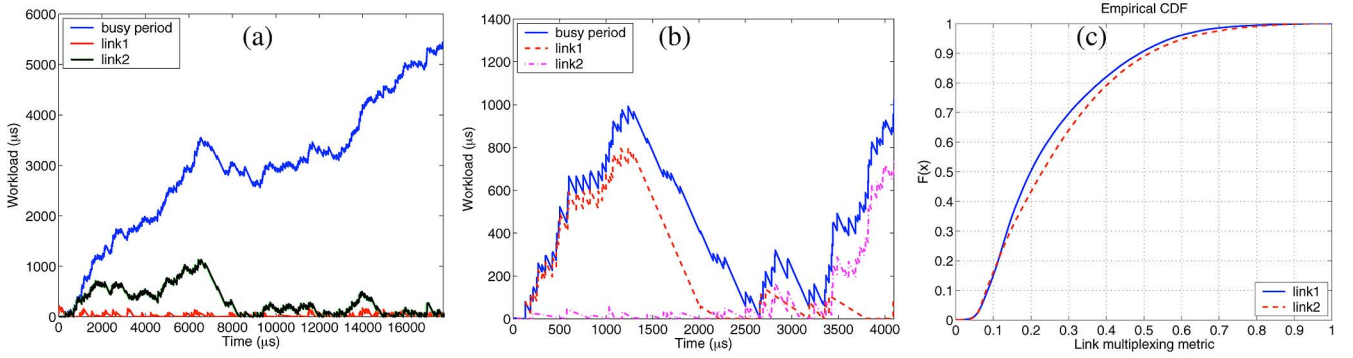


Fig. 13. Link multiplexing. Effect of multiplexing on the formation of busy periods (from t_s to t_A): (a) strong nonlinear magnification; (b) a “bimodal” busy period, assessing the contribution to A is ambiguous; (c) empirical CDF of LM for the OC-48 input streams.

output busy periods. If instead we had found that in most cases that AF was close to 1, it would have been an indication that most of the input traffic on that link was shaped at OC-3 rate upstream.

To get a feeling for the size of the values reported in Fig. 12, note that a realistic upper bound is given by $AF = 240\,000$, corresponding roughly to a 500 ms buffer being filled (in a single busy period) by 40 byte packets well separated at the input, that would induce a maximum workload of $129\ \mu\text{s}$ when served at OC-48 rate. A meaningful value worthy of concern is $AF = 1030$, corresponding to delays of 20 ms built up from 375 byte packets, the average packet size in our data. Since the worst value encountered in the data was $AF \approx 20$, stretching is not a dominant factor here.

E. Link Multiplexing

To examine the impact of multiplexing across different input links, we let the total stream S_T be the full set of measured traffic. The ramp-up period, $[t_s, t_A]$, for two busy periods of S_T are shown as the topmost curves in Fig. 13(a) and (b). We select our substreams to be the traffic from the two OC-48 backbone links, S_1 and S_2 . By looking at them separately, we again succeed in isolating multiplexing from the other two mechanisms in some sense. However, the actual impact of multiplexing is still intimately dependent on the “stretch transformed” burstiness structure on the separate links. What will occur cannot be predicted without the aid of detailed traffic modelling. Instead, we will consider how to measure what *does* occur, and see what we find for our data.

Fig. 13(a) and (b) show the delay behavior (consisting of multiple busy periods) due to the separate substreams over the ramp-up period of S_T . The nonlinearity is clear: the workload function is much larger than the simple sum of the workload functions of the two input substreams, although they comprise virtually all of the total traffic. For example in Fig. 13(a) the individual links each contribute less than 1 ms of workload at worst. Nonetheless, the multiplexed traffic leads to a significantly longer ramp-up period that reaches more than 5 ms of maximum workload at t_A on the right of the plot.

To quantify this effect we define the “link multiplexing” ratio

$$LM = \frac{\max_k A_k(S_i, \mu_o)}{A(S_T, \mu_o)}$$

which obeys $LM \in [0, 1]$. Values close to zero indicate that the link has a negligible individual contribution. Therefore if all substreams have small values, the nonlinear effect of multiplexing is very strong. In contrast, if $LM \approx 1$ for some substream, then it is largely generating the observed delays itself, and multiplexing *may* not be playing a major role. Large values of LM are in fact subject to ambiguity, as illustrated in Fig. 13(b), where the ratio is large for both links. The busy period has a bimodal structure. The first mode is dominated by link 1, however its influence has died off at time t_A , and so is not significantly responsible for the size of A .

The results for the data are presented in Fig. 13(c). In more than 95% of the busy periods traffic from each individual link contributes to less than 60% of the actual busy period amplitude. Therefore, it appears that multiplexing is an important factor overall for the delays experienced over the access link.

F. Flow Burstiness

There are many definitions of burstiness. It is not possible to fully address this issue without entering into details which would require traffic models, which is beyond the scope of this paper. We therefore focus on burstiness related to 5-tuple flows, to investigate the impact that individual flows, or groups of flows, have on overall delays. We begin by letting the total traffic S_T be that from a single link, to avoid the complications induced by multiplexing.

In order to obtain insight into the impact of flow burstiness we first select as a substream the “heaviest” individual flow in S_T in the simple sense of having the largest number of packets in the ramp-up period $[t_s, t_A]$ of S_T . Two extreme examples of what can occur in the ramp-up period are given in Fig. 14(a) and (b). In each case, the busy period amplitude is large, however the flow contribution varies from minimal in Fig. 14(a) to clearly dominant in Fig. 14(b).

To refine the definition of heaviest flow and to quantify its impact we proceed as follows. For each busy period in S_T we classify traffic into 5-tuple flows. We then use each individual flow S_j as a substream and measure the respective $A(S_j, \mu_o)$. The *top* flow is the one with the largest individual contribution. We define *flow burstiness* (FB) as

$$FB = \max_j \frac{\max_k A_k(S_j, \mu_o)}{A(S_T, \mu_o)}$$

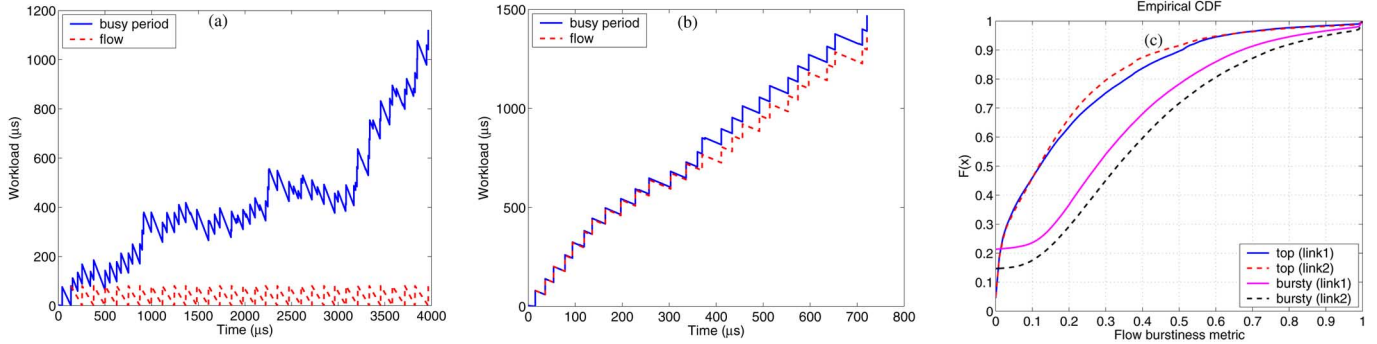


Fig. 14. Flow burstiness. Examples of the effect on queue buildup of flows with multiple packets: (a) no significant impact; (b) dominant flow; (c) empirical CDF of FB for the OC-48 input streams.

where as before the inner maximum is over all busy periods (or partial busy periods) of the relevant substream falling in $[t_s, t_A]$. The top flow may or may not be the one with the greatest number of packets.

Our top flow metric takes values $FB \in (0, 1]$. If FB is close to zero then we know that **all** flows have individually small contributions. Alternatively, if FB is large then, similarly to LM, there is some ambiguity. We certainly know that the top flow contributes significant delay but in case of bimodality this flow may not actually be responsible for the peak defining the amplitude. In addition, knowledge about the top flow can say nothing about the other flows.

We present the cumulative distribution function for FB in Fig. 14(c) for each of the OC-48 links. For more than 90% of the busy periods the contribution of the top flow was less than 50%. In addition for 20% of the busy periods the contribution of the top flow was *minimal* [for example as in Fig. 14(a)], that is it was the smallest possible, corresponding to the system time of a single packet with size equal to the largest appearing in the flow.

If the top flow has little impact, it is natural to ask if perhaps a small number of top flows together could dominate. One approach would be to form a stream of the n largest flows in the sense of FB. However, as the choice of n is arbitrary, and it is computationally intensive to look over many values of n , we first change our definition to select a more appropriate substream. We define a flow to be *bursty* if its substream generates a packet delay which exceeds the minimal delay (as defined above) during the ramp-up period. Note that only very tame flows are not bursty by this definition! We denote by S_b the substream of S_T that corresponds to **all** bursty flows, and compute the new flow burstiness metric:

$$FB' = \frac{\max_k A_k(S_b, \mu_o)}{A(S_T, \mu_o)}.$$

Now $FB' \in [0, 1]$, and its value can be interpreted in an analogous way to before. The difference is that, as the substream is much larger in general, a small value is now extremely strong evidence that individual flows do not dominate. Note that it is possible that no flow is bursty, in which case $FB' = 0$, and therefore that the top flow is not necessarily bursty. This has the advantage of avoiding the classification of a flow as dominant, thereby giving the impression that it is bursty in some sense, simply because a trivial flow dominates a trivial busy period.

Our results are presented in Fig. 14(c). As expected, the contribution of S_b to the busy period is more significant: in 15% of cases it exceeds 60%. On the other hand, 20% of the busy periods had FB' equal or close to zero, indicating that they had no bursty flows. Indeed, we found that only 7.7% of all flows in our dataset were classified as “bursty” according to our definition. Only in a very small number of cases does the top or the subset of bursty flows account for the majority of the workload [for example as in Fig. 14(b)]. Consequently, it seems from this data that flow dynamics have little impact on the delays experienced by packets in core networks.

VI. REPORTING MICROCONGESTION

In this section we discuss how microcongestion can be efficiently summarized and reported. We focus on measurement, not modelling, and in this context point out the dangers of oversimplifying a complex reality.

A. The Inutility of Utilization

From Section IV we know that our router model can accurately predict congestion and delays when the input traffic is fully characterized. However in practice the traffic is unknown, which is why network operators rely on available simple statistics, such as curves giving upper bounds on delay as a function of link utilization, in order to infer packet delays through their networks. The problem is that these curves are not unique since delays depend not only on the mean traffic rate, but also on more detailed statistics, for example second order properties such as long-range dependence. To illustrate this important point, we briefly give two examples.

Consider the following “semi-” experiment [11]: within each 5 minute window of aggregated traffic destined to C2-out, we replace the original packet arrival times by a Poisson process with the same rate but keep the packet sizes, and their arrival order, unchanged. The average utilization over each window is (very close to) unaffected by this packet repositioning. For each window we separately feed the real and surrogate traffic through the model, record the mean packet delays, and plot them against the utilization. From Fig. 15, we find that packets from the Poisson traffic systematically experience smaller delays. Although not surprising, this shows very clearly that a curve of mean delay versus link utilization is far from universal. On the contrary, it depends strongly on the statistics of the input traffic.

Next, suppose that there is a group of back-to-back packets on link C2-out, which means that the local link utilization is 100%. However this does not imply that they experienced large

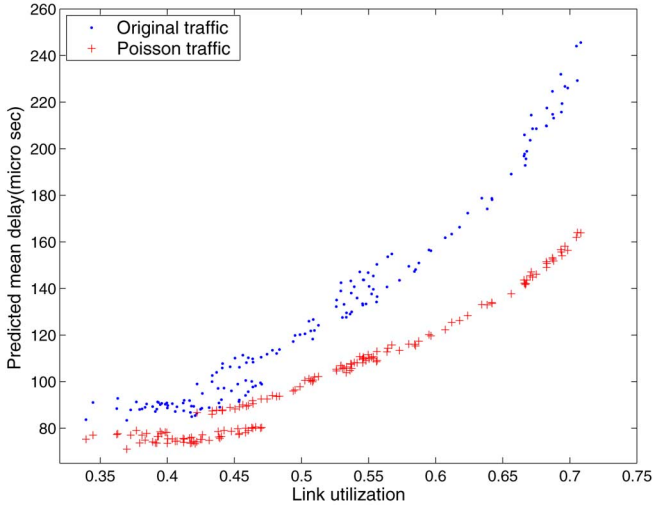


Fig. 15. Mean packet delays depends on traffic statistics beyond utilization.

delays. They could very well be coming back-to-back from the input link C1-in with the same capacity as C2-out. In this case they would actually cross the router with *minimum* delay in the absence of cross traffic!

The above arguments show how misleading utilization (even timescale dependent utilization [12]) can be for the understanding of congestion. In terms of measurement, inferring packet delays from link utilization is fundamentally flawed.

B. How to Summarize Congestion?

A complete and objective view of congestion behavior would provide a detailed account of busy period structure. However, busy periods are complex and diverse, and it is not clear how to compactly represent them in any meaningful way.

A hint of a solution can be seen in Fig. 10. Each of the large busy periods in the top row, discussed in Section V-A, have a roughly triangular shape. The bottom row shows many more busy periods, obtained using the following ad hoc heuristic. For each 5 minute interval, we detect the largest packet delay, store the corresponding packet arrival time t_0 , and plot the delays experienced by packets in a window 10 ms before and 15 ms after t_0 . The resulting sets of busy periods are grouped according to the largest packet delay observed: Fig. 10(d) when the largest amplitude is between 5 ms and 6 ms, (e) between 4 ms and 5 ms, and (f) between 2 ms and 3 ms (in each plot the black line highlights the busy period in the plot directly above it). The striking feature is that we systematically find a roughly triangular shape. As a result, the amplitude and duration can be used to approximate the form of the entire busy period.

We speculate that an explanation for this universality can be found in the theory of large deviations, which states that rare events happen in the most likely way. Some hints on the shape of large busy periods in (Gaussian) queues can be found in [13] where it is shown that, in the limit of large amplitude, busy periods tend to be antisymmetric about their midway point, in agreement with what we see here.

C. Modelling Busy Period Shape

We begin by illustrating a fundamental bound on the shape of any busy period of duration D seconds. As shown in Fig. 16, it is bounded above by the busy period obtained when D seconds

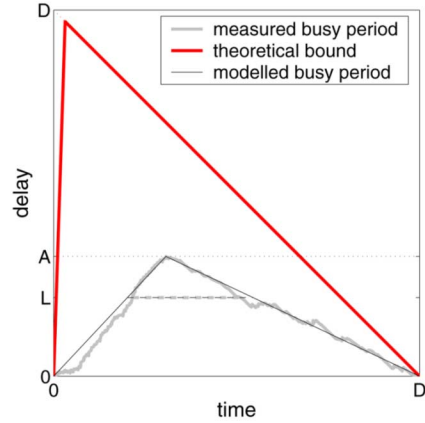


Fig. 16. A triangular bound and triangular model of busy period shape.

worth of work arrives in the system at maximum input link rate. The queue occupancy then decreases with slope -1 since no more packets enter the system. In the case of the OC-3 link C2-out fed by the two OC-48 links BB1 and BB2, it takes at least $D/32$ seconds for the D seconds of load to enter the system. The busy period shown in Figs. 7 and 10(a) is replotted in Fig. 16 for comparison. Its amplitude A is much lower than the theoretical maximum.

In the rest of the paper we model the shape of a busy period of duration D and amplitude A by a triangle with base D , height A and the same apex position as the busy period. This is illustrated in Fig. 16 by the triangle superimposed over the measured busy period. This very rough approximation can give surprisingly valuable insight into packet delays. For example, let L be the delay experienced by a packet crossing the router. A network operator might be interested in knowing how long a congestion level larger than L will last, because this gives a direct indication of the performance of the router.

Let $d_{L,A,D}$ be the length of time the workload of the system remains above L during a busy period of duration D and amplitude A . Let $d_{L,A,D}^{(T)}$ be the duration obtained from the shape model. Both $d_{L,A,D}$ and $d_{L,A,D}^{(T)}$ are plotted with a dashed line in Fig. 16. From basic geometry one can show that

$$d_{L,A,D}^{(T)} = \begin{cases} D \left(1 - \frac{L}{A}\right), & \text{if } A \geq L \\ 0, & \text{otherwise.} \end{cases} \quad (13)$$

In other words, $d_{L,A,D}^{(T)}$ is a function of L , A and D only. For the metric considered, the two parameters (A, D) are therefore enough to describe busy periods, the knowledge of the apex position does not improve our estimate of $d_{L,A,D}$.

Denote by $\Pi_{A,D}$ the random process governing $\{A, D\}$ pairs for successive busy periods over time. The mean length of time during which packet delays are larger than L reads

$$T_L = \int d_{L,A,D} d\Pi_{A,D}. \quad (14)$$

T_L can be approximated by our busy period model with

$$T_L^{(T)} = \int d_{L,A,D}^{(T)} d\Pi_{A,D}. \quad (15)$$

We use (15) to approximate T_L on C2-out. The results are plotted on Fig. 17 for two 5 minute windows of traffic with different average utilizations. In each case, the measured durations

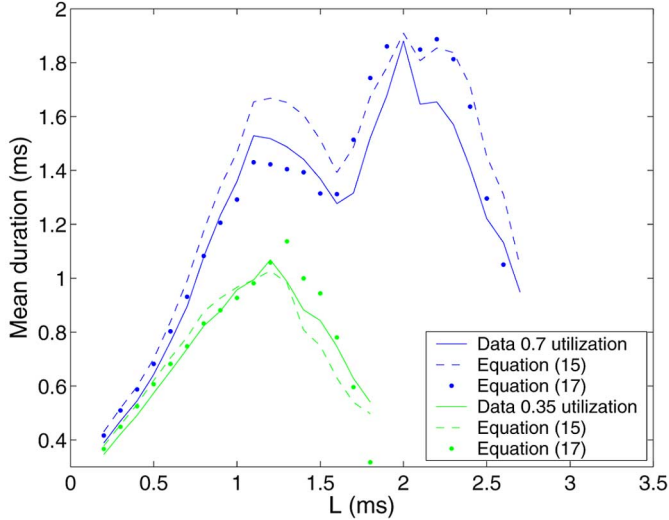


Fig. 17. Average duration of a congestion episode above L ms defined by equation (15), for two different utilization levels (0.3 and 0.7) on link C2-out. Solid lines: data, dashed lines: Equation (15), dots: Equation (17).

(solid line) and the results from the triangular approximation (dashed line) are remarkably similar.

We now qualitatively describe the behaviors observed in Fig. 17. For a small congestion level L , the mean duration of the congestion episode is also small. This is due to the fact that, although a large number of busy periods have an amplitude larger than L [see Fig. 9(a)], most do not exceed L by a large amount. It is also worth noticing that the results are very similar for the two different utilizations. This means that busy periods with small amplitude are roughly similar at this timescale, independent of large scale statistics. As the threshold L increases, the (conditional on L) mean duration first increases as there are still a large number of busy periods with amplitude greater than L on the link, and of these, most are considerably larger than L . With an even larger value of L however, fewer and fewer busy periods qualify. The ones that do cross the threshold L do so for a smaller and smaller amount of time, up to the point where there are no busy periods larger than L in the trace.

D. Reporting Busy Period Statistics

The study presented above shows that one can get useful information about delays by jointly using the amplitude and duration of busy periods. Now we look into ways in which such statistics could be concisely reported.

We start by forming busy periods from buffer occupancy measurements and collecting (A, D) pairs during 5 minute intervals (a feasible number for SNMP reporting). This is feasible in practice since the buffer size is already accessed, for example by active queue management schemes. Measuring A and D is easily performed on-line. In principle we need to report the pair (A, D) for each busy period in order to recreate the process $\Pi_{A,D}$ and evaluate (15). Since this represents a very large amount of data in practice, we instead assume that busy periods are independent and therefore that the full process $\Pi_{A,D}$ can be described by the joint marginal distribution $F_{A,D}$ of A and D . Thus, for each busy period we need simply update a sparse 2-D histogram.

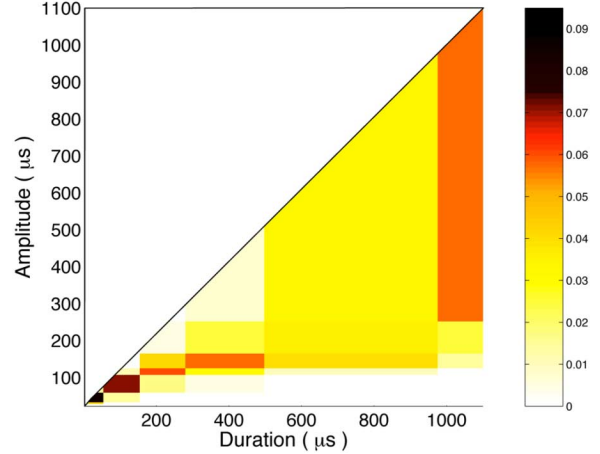


Fig. 18. Histogram of the quantized joint probability distribution of busy period amplitudes and durations with $N = 10$ equally spaced quantiles along each dimension for a 5 minute window on link C2-out.

The bin sizes should be as fine as possible consistent with available computing power and memory. We do not consider these details here. They are not critical since at the end of the 5 minute interval a much coarser discretization is performed in order to limit the volume of data finally exported. We control this directly by choosing N bins for each of the amplitude and the duration dimensions.

As we do not know *a priori* what delay values are common, the discretization scheme must adapt to the traffic to be useful. A simple and natural way to do this is to select bin boundaries for D and A separately based on quantiles, i.e., on bin populations. For example a simple equal population scheme for D would define bins such that each contained $(100/N)\%$ of the measured values. Denote by M the $N \times N$ matrix representing the quantized version of $F_{A,D}$. The element $p(i, j)$ of M is defined as the probability of observing a busy period with duration between the $(i - 1)$ th and i th duration quantile, and amplitude between the $(j - 1)$ th and j th amplitude quantile. Given that for every busy period $A < D$, the matrix is triangular, as shown in Fig. 18. Every 5 min, $2N$ bin boundary values for amplitude and duration, and $N^2/2$ joint probability values, are exported.

The 2-D histogram stored in M contains the 1-D marginals for amplitude and duration, characterizing respectively packet delays and link utilization. In addition however, from the 2-D histogram we can see at a glance the relative frequencies of different busy period *shapes*. Using this richer information, together with a shape model, M can be used to answer performance related questions. Applying this to the measurement of T_L introduced in Section VI-C, and assuming independent busy periods, (15) becomes

$$T_L^{(T)} = \int d_{L,A,D}^{(T)} dF_{A,D} = \int_{A>L} D \left(1 - \frac{L}{A}\right) dF_{A,D}. \quad (16)$$

To evaluate this, we need to determine a single representative amplitude A_i and average duration D_j for each quantized probability density value $p(i, j)$, $(i, j) \in \{1, \dots, N\}^2$, from M . One can for instance choose the center of gravity of each of the tiles

plotted in Fig. 18. For a given level L , the average duration T_L can then be estimated by

$$\widetilde{T}_L^{(T)} = \frac{1}{n_L} \sum_{j=1}^N \sum_{\substack{i=1 \\ A_i > L}}^j d_{L, A_i, D_j}^{(T)} p(i, j) \quad (17)$$

where n_L is the number of pairs (A_i, D_j) such that $A_i > L$. Estimates obtained from (17) are plotted in Fig. 17. They are fairly close to the measured durations despite the strong assumption of independence.

The above gives just one example, for the metric T_L , of how the reported busy period information can be used, however other performance related questions could be answered in a similar way. In any case, our reporting scheme provides far more insight into packet delays than currently available statistics based on average link utilization, despite its simplicity.

VII. CONCLUSION

We explored in detail router congestion and packet delay, based on a unique data set where all IP packets crossing a Tier-1 access router were measured. We presented authoritative empirical results about packet delays, but also looked inside the router to provide a physical model of router operation which can very accurately infer, not just packet delay, but the entire structure of microcongestion (busy periods). We then used semi-experiments to inquire into the causes of microcongestion based on three canonical mechanisms: bandwidth reduction, link multiplexing, and burstiness. We defined new corresponding metrics, and used them to show that, for this data, the classical nonlinear effect of the multiplexing of flows was primarily responsible for the observed delay behavior. Finally, we examined the difficult problem of compactly summarizing the richness of microcongestion behavior, beginning by explaining why an approach based on utilization is fundamentally flawed. By exploiting an observed commonality in shape of large busy periods, we were able to capture much of the busy period diversity using a simple joint distribution of busy period amplitude and duration. We showed how this metric captures important information about the delay process without any model based preconceptions, and gave a sketch for on-line algorithms for its collection and export.

ACKNOWLEDGMENT

The authors would like to thank C. Diot, G. Iannaccone, E. Kress, and T. Ye for facilitating the collection of the measurement data and making it possible, and for the development of the packet matching code.

REFERENCES

- [1] K. Papagiannaki, S. Moon, C. Fraleigh, P. Thiran, F. Tobagi, and C. Diot, "Analysis of measured single-hop delay from an operational backbone network," in *Proc. IEEE INFOCOM*, New York, Jun. 2002, pp. 535–544.
- [2] Waikato Applied Network Dynamics. [Online]. Available: <http://www.wand.cs.waikato.ac.nz/wand/wits/>
- [3] N. Hohn, D. Veitch, K. Papagiannaki, and C. Diot, "Bridging router performance and queueing theory," in *Proc. ACM Sigmetrics*, New York, Jun. 12–16, 2004, pp. 355–366.

- [4] K. Papagiannaki, D. Veitch, and N. Hohn, "Origins of microcongestion in an access router," in *Proc. Passive and Active Measurement Workshop (PAM2004)*, Juan-Les-Pins, France, Apr. 19–20, 2004, Lecture Notes in Computer Science (LNCS), pp. 126–136, Springer.
- [5] N. McKeown, "SLIP: A scheduling algorithm for input-queued switches," *IEEE/ACM Trans. Networking*, vol. 7, no. 2, pp. 188–201, Apr. 1999.
- [6] W. Simpson, "PPP in HDLC-like framing," RFC 1662 [Online]. Available: <http://www.ietf.org/rfc/rfc1662>
- [7] Endace Measurement Systems. [Online]. Available: <http://www.endace.com/>
- [8] S. Donnelly, "High precision timing in passive measurements of data networks," Ph.D. dissertation, Univ. of Waikato, Hamilton, New Zealand, 2002.
- [9] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-level traffic measurements from the Sprint IP backbone," *IEEE Network*, vol. 17, no. 6, pp. 6–16, Nov.–Dec. 2003.
- [10] S. Keshav and S. Rosen, "Issues and trends in router design," *IEEE Commun. Mag.*, vol. 36, no. 5, pp. 144–151, May 1998.
- [11] N. Hohn, D. Veitch, and P. Abry, "Cluster processes, a natural language for network traffic," *IEEE Trans. Signal Processing*, vol. 51, Special Issue on Signal Processing in Networking, no. 8, pp. 2229–2244, Aug. 2003.
- [12] K. Papagiannaki, R. Cruz, and C. Diot, "Network performance monitoring at small timescales," in *Proc. ACM Internet Measurement Conf.*, Miami, FL, 2003, pp. 295–300.
- [13] R. Addie, P. Mannersalo, and I. Norros, "Performance formulae for queues with Gaussian input," in *Proc. 16th Int. Teletraffic Congr.*, Edinburgh, Scotland, U.K., Jun. 1999, pp. 1169–1178.



Nicolas Hohn received the Ingénieur degree in electrical engineering in 1999 from Ecole Nationale Supérieure d'Electronique et de Radio-électricité de Grenoble, Institut National Polytechnique de Grenoble, France. In 2000 he received the M.Sc. degree in bio-physics from the University of Melbourne, Australia, while working for the Bionic Ear Institute on signal processing techniques in auditory neurons. In 2004, he received the Ph.D. degree from the E&EE Department at The University of Melbourne, Australia, in Internet traffic statistics, sampling, and modelling. He is currently a researcher in a hedge fund in Melbourne, Australia. Dr. Hohn received the best student paper award at the 2003 ACM Internet Measurement Conference.



Konstantina Papagiannaki received the Ph.D. degree from the Computer Science Department of University College London, London, U.K., in 2003. Her Ph.D. dissertation was awarded the Distinguished Dissertations Award 2003. She has been with Intel Research working in wired and wireless networking since January 2004. From 2000 to 2004, she was a member of the IP research group at the Sprint Advanced Technology Labs in Burlingame, CA. Her main interests lie in network design and planning for wired and wireless networks, traffic modeling and enterprise security.



Darryl Veitch (SM'02) received the B.Sc. (Hons.) degree from Monash University, Australia, in 1985, and the Ph.D. degree in mathematics from Cambridge University, Cambridge, U.K., in 1990. He worked at TRL (Telstra, Melbourne), CNET (France Telecom, Paris), KTH (Stockholm), INRIA (Sophia Antipolis, France), Bellcore (New Jersey), RMIT (Melbourne), and EMUlab and CUBIN at the University of Melbourne, where he is a Principal Research Fellow. His research interests include traffic modelling, parameter estimation, active measurement, traffic sampling, and clock synchronization. Dr. Veitch is a member of the Association for Computing Machinery (ACM).