SCISPACE
formerly Typeset

Open access • Journal Article • DOI:10.1109/JSAC.2011.110502

Delaying Transmissions in Data Communication Networks to Improve Transport-Layer Performance — Source link

Yan Cai, Tilman Wolf, Weibo Gong

Institutions: University of Massachusetts Amherst

Published on: 21 Apr 2011 - IEEE Journal on Selected Areas in Communications (IEEE)

Topics: End-to-end delay, Processing delay, Transmission delay, Network traffic control and Network delay

Related papers:

- A Practical On-line Pacing Scheme at Edges of Small Buffer Networks
- Practical Packet Pacing in Small-Buffer Networks
- Delay-based Back-Pressure scheduling in multi-hop wireless networks
- Opportunistic Scheduling and Performance Analysis on Wireless Network Coding
- Hybrid Resource Allocation in Wireless Ad Hoc Networks

Share this paper:

View more about this paper here: https://typeset.io/papers/delaying-transmissions-in-data-communication-networks-to-3haygs4sqb

Delaying Transmissions in Data Communication Networks to Improve Transport-Layer Performance

Yan Cai *Student Member, IEEE*, Tilman Wolf, *Senior Member, IEEE,* and Weibo Gong, *Fellow, IEEE*

***Abstract*—Packet losses in the network have a considerable performance impact on transport-layer throughput. For reliable data transfer, lost packets require retransmissions and thus cause very long delays. This tail of the packet delay distribution causes performance problems. There are several approaches to trading off networking resources up-front to reduce long delays for some packets (e.g., forward error correction, network coding). We propose packet pacing as an alternative that changes traffic characteristics favorably by adding intentional delay in packet transmissions. This intentional delay counters the principle of best effort but can reduce the burstiness of traffic and improve overall network operation – in particular in network with small packet buffers. As a result, pacing improves transport-layer performance, providing a tradeoff example where small amounts of additional delay can significantly increase connection bandwidth. We present a Queue Length Based Pacing (QLBP) algorithm that paces network traffic using a single queue and that can be implemented with small computational and memory overhead. We present a detailed analysis on delay bounds and the quantitative impact of QLBP pacing on network traffic. Through simulation, we show how the proposed pacing technique can improve connection throughput in small-buffer networks.**

***Index Terms*—transport layer, traffic pacing, small-buffer network.**

I. Introduction

MANY data communication networks use a layered network architecture, where each layer implements different networking protocols [1]. The separation of networking functionality into layers simplifies the design of network protocols, but also implies that the performance that can be achieved within a protocol layer is highly dependent on the performance achieved by underlying layers. Specifically, the performance of transport layer protocols relies on the performance achieved by packet delivery in the network layer.

In our work, we discuss how to improve the throughput performance of transport layer protocols by adjusting the operation of the network at the network layer. The main idea is to adjust the characteristics of network traffic at the edge of the network to ensure better performance in the core of the network. Specifically, we propose to introduce intentional delay in network layer transmissions to reduce the occurrence of traffic bursts, which have detrimental effects on transport layer performance as they can lead to packet loss due to buffer overflow. Our focus is on networks with small packet buffers (e.g., all-optical packet-switched networks, wireless networks with low-performance nodes) [2].

Manuscript received 1 July 2010; revised 10 December 2010. This material is based upon work supported by the National Science Foundation under Grant No. CNS-0721790.

Y. Cai, T. Wolf and W. Gong are with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA, 01003 USA (e-mail: {ycai,wolf,gong}@ecs.umass.edu).

Digital Object Identifier 10.1109/JSAC.2011.110502.

A. Packet Loss in Networks

One of the most problematic events for data transmissions in the network layer is a packet loss. The two main causes for packet loss in networks are:

- Bit errors in physical layer: Bit errors in the physical layer most commonly occur in wireless transmissions due to interference, but can also occur in wired links. These bit errors cause checksums in the data link layer to fail, triggering a packet drop.
- Congestion in network layer: Statistical multiplexing of network traffic implies that there are no guarantees about the available bandwidth on any given link. Thus, network traffic can congest the outgoing port of a router and cause transmission buffers to fill up. If a packet arrives at such a transmission queue when no more buffer space is available, then it is dropped.

While these causes of packet loss are fundamentally different, their effects result in the same performance degradation in the transport layer.

In practice, many application require reliable (i.e., lossless) data transfer. While some applications can compensate for lost data in the application layer, lossy transmission are only useful in very specific application domains (e.g., video playback). To recover from a loss event, the transport layer initiates a retransmission of the lost packet. This is a problematic solution for applications where data needs to be delivered with low delay (e.g., cyber-physical control, online gaming, etc.), since retransmission of a packet can incur considerable delay (time to discover loss plus one round-trip time). Therefore, there is considerable need to develop mechanisms that allow for reliable data communication while ensuring low delay.

B. Delay and Bandwidth Tradeoffs

There are several possible approaches to addressing the problem of reducing the impact of packet loss on the delay in transport layer communication. Figure 1 illustrates how some of these techniques relate. The figure shows the amount of delay incurred at the transport layer versus the amount of bandwidth used at the transport layer. The main techniques noted in this figure are:

- Lossy transmission: Using lossy transmission protocols (e.g., User Datagram Protocol (UDP) [3]) places the bandwidth needs and delay close to the ideal lower bounds. Marginal amounts of additional bandwidth are

0733-8716/11/$25.00 © 2011 IEEE

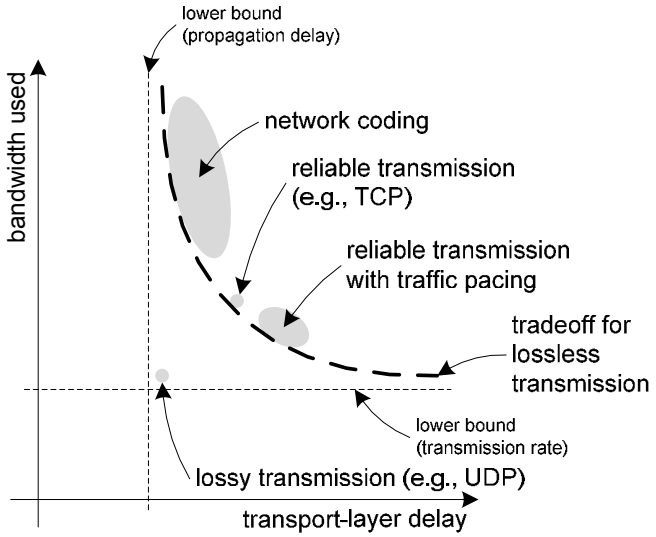


Fig. 1. Tradeoff of delay and bandwidth consumption for different lossless transmission techniques.

necessary for packet headers and additional delay is incurred due to the packetized transmission of data.

- Reliable transmission: The baseline protocol for reliable transmission is the Transmission Control Protocol (TCP) [4]. Compared to UDP, TCP requires more bandwidth since some packets need to be retransmitted. It also incurs additional delay due to these retransmissions.
- Network coding: There are several coding techniques to reduce packet loss in networks. To reduce bit errors, error correction coding can be used [5]. To avoid packet losses, transmission information can be spread across multiple paths in the network using network coding [6]. These techniques require additional bandwidth since they rely on redundant transmission of information. They also exhibit increased delay over a lossy transmission due to the need for data reconstruction at the receiver. However, these techniques incur less delay than TCP.
- Traffic pacing: Traffic pacing is based on TCP, but uses traffic conditioning techniques in the network to reduce traffic bursts. By delaying some packet transmissions, less packet losses occur and thus less retransmissions are needed. Traffic pacing incurs a small additional delay, but uses less bandwidth than TCP since fewer retransmissions are necessary.

Overall, Figure 1 shows that there is a general tradeoff between bandwidth use and delay for lossless transmission in the transport layer.

While network coding and traffic pacing trade bandwidth versus delay in different manner, it is interesting to note that they both target the same problem of packet loss. When considering a distribution of end-to-end packet delays in networks, it can be expected that most packets are transmitted successfully in the first attempt. However, packets that get lost and are retransmitted exhibit much longer delays. This "tail" of the packet delay distribution is the main problem for transport layer performance. When requiring lossless data transfers, long delays of a few packets limit overall throughput performance. Thus, it is critical to eliminate (or at least reduce) this tail in the delay distribution.

C. Traffic Pacing in Networks

A key operational principle in the Internet is "best effort." Network resources are used when there is traffic to be sent and link schedulers on routers use "work-conserving" scheduling disciplines. This approach of not wasting opportunities to transmit packets intuitively seems to lead to the best possible network performance. However, a significant drawback is that best-effort forwarding propagates traffic bursts through the network and leads to potential buffer overflows (and thus packet loss). In contrast to best effort, several traffic pacing approaches have been proposed. In traffic pacing, transmission of some packets are intentionally delayed (despite link availability) to improve the characteristics of network traffic as a whole and thus reduce the probability of packet loss due to buffer overflows.

In our work, we present a traffic pacing technique that can reduce the burstiness of traffic and improve the throughput of transport layer TCP connections. The design of our traffic pacing system is particularly suitable for emerging network architectures for two reasons:

- Indiscriminate pacing does not require per-flow state: Many existing pacing techniques determine packet delays on a per-flow basis. This process requires computationally expensive packet classification and the maintenance of per-flow state on the router. For high-bandwidth links, this technique does not scale well. In our work, we pace packets indiscriminately of what flow they belong to. Thus, we only need to maintain a single packet queue and pacing parameters.
- Pacing algorithm improves operation of small-buffer networks: As we show in this work, the proposed pacing technique improves throughput in networks with small packet buffers on routers. Since these small-buffer networks are expected to be deployed in the next-generation Internet [7], our solution presents an important contribution to the efficient operation of these networks.

The specific contributions of our work are:

- Queue Length Based Pacing (QLBP): We present a novel pacing algorithm that decreases the burstiness of network traffic by delaying packets based on the length of the local packet buffer.
- Analysis of QLBP: We present a formal analysis of QLBP that provides delay bounds and a quantitative understanding of the effect of traffic smoothing.
- Simulation Results: We present simulation results that show the effectiveness of QLBP and its improvements of transport layer performance in small-buffer networks.

We believe that these contributions present an important step towards more effective operation of networks.

The remainder of this paper is organized as follows. Section II introduces the network architecture for pacing and details on the Queue Length Based Pacing algorithm. Analytical results are presented in Section III. Simulation results on the effectiveness of QLBP are presented in Section IV. Section V discusses related work, and Section VI summarizes and concludes this paper.

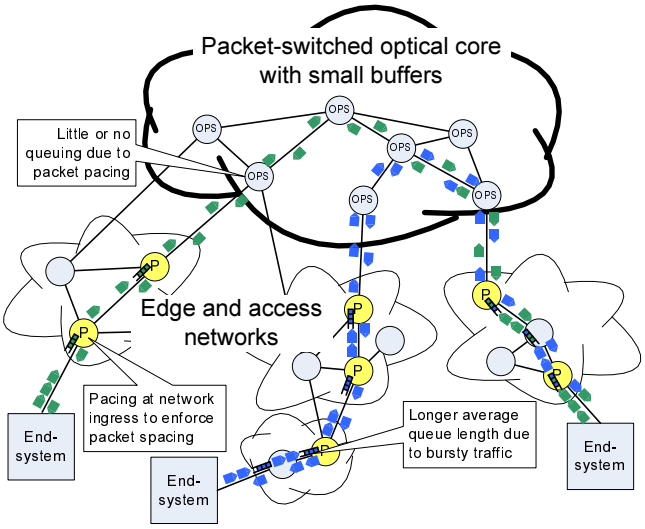


Fig. 2. Network architecture with opportunistic pacing.

II. Queue-Length Based Pacing

The pacing technique that we propose in this work aims to reduce the burstiness of network traffic. Before detailing the pacing algorithm, we briefly discuss background on TCP burstiness and an overview of a network architecture that uses our pacing technique.

A. TCP Burstiness

TCP is the most widely used transport layer protocol in the Internet. Its traffic characteristics have considerable impact on the operation of the network. As we discuss here, TCP traffic is inherently bursty due to the design of the protocol and can cause problems in networks with small buffers.

The TCP protocol can pace itself due to ACK-clocking, where acknowledgments are spaced out by the bottleneck link. As a result, packets sent in the congestion avoidance phase are spaced by acknowledgement arrivals. However, as pointed out by Aggarwal et al. in [8], a number of factors inherent to TCP can cause burstiness in the behavior of a TCP flow, such as slow start, lost packet retransmission, ACK-compression and multiplexing (for details, see [8]). Even though the impact of retransmissions of lost packets can somehow be mitigated by enabling TCP selective acknowledgement (SACK) options [9], [10], the negative impact of ACK-compression and multiplexing might become even worse in the future Internet with much larger bandwidth.

To illustrate this point, consider the detailed dynamics of TCP. (For simplicity, we only examine the TCP congestion avoidance phase.) For a long-lived TCP session, its available bandwidth is determined by the capacity of the bottleneck link. In particular, the available bandwidth is equal to the bottleneck link capacity divided by the number of long-lived TCP sessions that compete for the bottleneck link. (Here, we assume only long-lived TCP sessions exist.) If there are UDP sessions, then the bandwidth of bottleneck link is equal to the total bandwidth minus the UDP sessions' bandwidth. We ignore the impact of short-lived TCP sessions because of their small congestion windows. Due to ACK-compression and multiplexing, all packets belonging to one congestion window can go through the bottleneck link in a back-to-back manner. Thus, the transmission rate within a burst of packets is likely to be close to the link speed of the bottleneck link, which might be much higher than the long-term throughput of the underlying TCP session. This difference is the source of burstiness in the TCP session. As physical link speeds increase in the future Internet [11], this burstiness will be more severe.

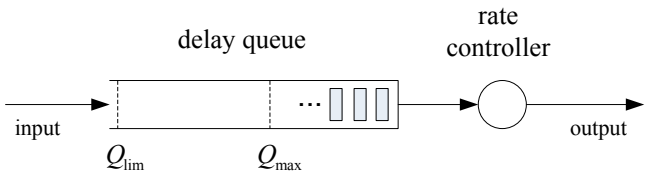


Fig. 3. QLBP system for router buffer.

B. Pacing Network Architecture

To reduce the burstiness of TCP traffic (and any other traffic), we propose a pacing technique that delays some packet transmissions. This pacing process can be implemented on the outgoing interfaces of routers. We envision an overall network architecture as shown in Figure 2. Pacing is deployed on several (but not necessarily all) nodes in the network. Since pacing cannot be practically implemented on optical packet switches, it is constrained to non-optical routers. These routers have sufficiently large buffers that allow moderate traffic bursts to be absorbed and paced without packet loss. At the network edge, routers with pacing capabilities reduce the burstiness of traffic before it enters the small-buffer network core. Within the network core, packet drops are reduced since non-bursty traffic is less likely to fill up router queues, even when they are small.

It is important to note that all traffic on an outgoing link uses only one queue and pacer. Thus, pacing is done *indiscriminately* and can be implemented efficiently for high-performance routers. Also, pacing can be performed *opportunistically*: the more pacing nodes are traversed by traffic, the less bursty it becomes.

C. Queue Length Based Pacing System

The general ideal of Queue Length Based Pacing (QLBP) is to dynamically adjust the sending rate of a queue according to the queue length, rather than to send packets at a constant rate. The structure of a QLBP system is shown in Figure 3, and the major notation used in this paper is summarized in Table I.

The figure shows a single input and output, but the concept can be applied to routers with any number of ports. A QLBP system includes a delay queue and a rate controller, and has three parameters: $\mu_{\max}, \mu_{\min}$ and $Q_{\max}$. The delay queue in Figure 3 is an ordinary FIFO queue. Packets arrive at a certain rate on the input link and are stored in the delay queue. If the queue is full (i.e. $q(t) = Q_{\lim}$), the arriving packet is dropped. The output rate $\mu(t)$ is controlled by a rate controller according to the queue length $q(t)$: if $0 \leq q(t) \leq Q_{\max}$, $\mu(t)$ is calculated in a deterministic way (will be specifically introduced in the next sub-section); if $Q_{\max} < q(t) \leq Q_{\lim}$, $\mu(t)$ is set to the capacity C of the outgoing link.

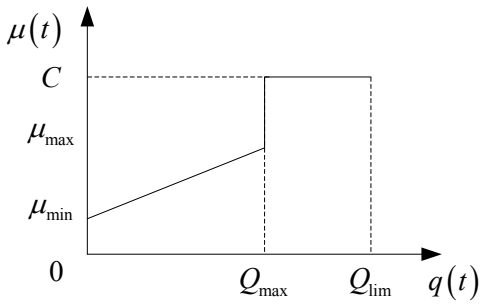


Fig. 4. Pacing rate $\mu(t)$ vs. queue length $q(t)$.

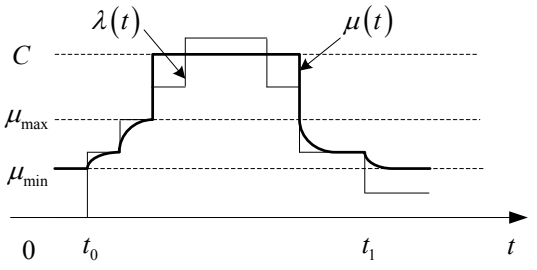


Fig. 5. Relationship between $\mu(t)$ and $\lambda(t)$.

Typically, QLBP would be used on an egress port of a router. In this case, the delay queue is the output queue of the egress port, and C is the link capacity of the egress port.

D. Pacing Delay

One of the key aspects of any pacing algorithm is how the inter-packet pacing delay is determined. In TCP pacing [8], the inter-packet pacing delay is roughly set to the ratio of the current RTT to the congestion window size. In the pacing scheme proposed by Sivaranman [12], the inter-packet pacing delay is calculated based on the packet arrival curve and the packet deadline curve within the same pacing interval. In QLBP, we determine this delay based on some very simple rules:

- If the pacing queue increases due to a higher input traffic rate, QLBP intentionally lowers the introduced pacing delay. This rule ensures that the link can be fully utilized under heavy load.
- For packets that arrive at a rate lower than $\mu_{\min}$, they do not get delayed. This rule ensures that pacing is only activated when packets arrive at a certain high rate.

Based on these rules, we have designed the queue length dependent output rate $\mu(t)$ as follows:

$$\mu(t) = \begin{cases} \frac{\mu_{\max}-\mu_{\min}}{Q_{\max}} q(t) + \mu_{\min}, & 0 \le q(t) \le Q_{\max}, \\ C, & \text{otherwise.} \end{cases} \quad (1)$$

Figure 4 depicts the output rate $\mu(t)$ versus the instantaneous queue length $q(t)$.

In the following, we use a simple example shown in Figure 5 to illustrate how a QLBP system paces packets. Suppose that at time t_0, $\lambda(t)$ is zero. From that moment on, $\lambda(t)$ begins to increase. Without loss of generality, $\mu_{\min}$ and $\mu_{\max}$ are set to $\frac{C}{a}$ and $\frac{C}{b}$, and $Q_{\max}$ is set to $\frac{Q_{\lim}}{c}$, where $a, b, c > 1$ and $a > b$.

TABLE I
NOTATION USED IN THIS PAPER

Defined in Section II-C	
$q(t)$	instantaneous length of the delay queue at time t
$\lambda(t)$	arrival rate of input traffic at time t
$\mu(t)$	output rate of the rate controller at time t
$\mu_{\max}$	maximum rate at which the rate controller transmits packets when pacing is enabled
$\mu_{\min}$	minimum rate at which the rate controller transmits packets when pacing is enabled
$Q_{\max}$	(pacing cutoff queue length) queue length beyond which no pacing delays are introduced by the pacer
$Q_{\lim}$	buffer size of the delay queue
C	capacity of the outgoing link
Defined in Section III-A	
d	pacing delay
d_{pacer}	delay a packet experiences when passing through a QLBP pacer
d_{FIFO}	delay a packet experiences when passing through a FIFO queue
Defined in Section III-B	
N_1	ON Poisson counter of the Markov ON-OFF modeled process
N_2	OFF Poisson counter of the Markov ON-OFF modeled process
r_1	rate of ON Poisson counter N_1
r_2	rate of OFF Poisson counter N_2
h	peak rate during ON periods

When $\lambda(t) < \mu_{\min}$, $q(t) = 0$ and $\mu(t) = \mu_{\min}$ according to (1). As a result, no packets are paced and the actual output rate is still $\lambda(t)$. When $\lambda(t)$ exceeds $\mu_{\min}$ (i.e., $\mu(t)$), a queue begins to be built up, i.e., $q(t) > 0$, which causes $\mu(t)$ to increase to follow $\lambda(t)$. When the equilibrium is reached, $\mu(t) = \lambda(t)$, and the corresponding $q(t)$ is given by

$$q(t) = \frac{\lambda(t) - \mu_{\min}}{\mu_{\max} - \mu_{\min}} Q_{\max}.$$

As $\lambda(t)$ continues to grow up to $\mu_{\max}$, $q(t)$ increases towards $Q_{\max}$, causing $\mu(t)$ to further increase. *When* $\mu_{\max} < \lambda(t) \le C$, $q(t)$ *is equal to* $Q_{\max}$ *and* $\mu(t)$ *is* C.

It is possible for $\lambda(t)$ to be even larger than C (considering an egress port as an example). In this case, $q(t)$ grows up to $Q_{\lim}$ and eventually causes overflow.

When $\lambda(t)$ decreases, a similar but reversed process follows.

Given the detailed description of QLBP, we turn towards the analysis of its properties.

III. Analysis

In this analysis, we show two important results: (1) the pacing delay depends on the incoming traffic rate and is upper-bounded by a constant (depending on QLBP parameters), thus limiting delay introduced by QLBP, and (2) the effectiveness of QLBP on reducing burstiness in network traffic can be quantified by evaluating the variance of the instantaneous traffic rate in the context of a fluid model.

A. Delay Guarantee

To show the bounds on delay, we first give a precise definition of pacing delay.

Definition 1: For a packet, the pacing delay, denoted by d, is defined as the time difference of $d_{pacer} - d_{FIFO}$, where

d_{pacer} and d_{FIFO} represent the delay the packet experiences when passing through a QLBP queue and an ordinary FIFO (drop-tail) queue, respectively.

Remark: This definition differentiates pacing delay from queuing delay. As the delay queue itself is the packet-storing queue, a packet might experience either queuing delay or pacing delay, or both when it passes through the delay queue. This extra amount of delay is counted as the pacing delay in the sense that packets are not sent at a full line speed but, instead, a pacing rate, which is smaller than or equal to the full line speed.

Given the definition of pacing delay, we now have the following theorem.

Theorem 1: Given parameters $\mu_{\max}, \mu_{\min}$ and $Q_{\max}$, for an input traffic with rate λ, the pacing delay d in steady state depends on λ and is upper bounded by a constant $\frac{Q_{\max}}{\mu_{\max}}$.

Proof is provided in Appendix A.

Remark: For a 600Mbps OC-12 link equipped with a QLBP pacer of $Q_{\max} = 150$KB (i.e., 100 of 1500 Byte packets) and $\mu_{\max} = 300Mbps$, the delay bound is 4ms. The delay bound is reduced to 2ms when $\mu_{\max}$ is set to 600Mbps. In Theorem 1, we focus only on pacing delay in the steady state. In practice, the incoming traffic rate changes over time. In this case, a more complicated analysis is required.

B. Reduction of Traffic Burstiness

We quantitatively analyze the pacing effect of a QLBP system in two aspects: (1) how quickly a QLBP system responds to the change in the input rate, (2) how a QLBP system smoothes the input traffic by reducing the auto-covariance. Even though the modeling and analysis are established based on some simple toy traffic models, they still unveil the fundamental natures of QLBP. To this end, our work can be viewed as the first step towards a more realistic and complicated modeling and analysis.

In the following analytical analyses, we have the assumption on the parameters of QLBP and the input rate $\lambda(t)$.

Assumption 1: The parameters of the QLBP system are set as follows: $\mu_{\min} = 0$, $\mu_{\max} = C$, $Q_{\max} = \frac{Q_{\lim}}{a}$, where a ($a > 1$) is an arbitrary real number, and for any $t > 0$, $0 \leq \lambda(t) < C$.

This corresponds to a scenario where the QLBP system is applied to a campus edge router in which the input traffic rarely overflows the outbound link of capacity C.

1) Response Speed of QLBP: Under Assumption 1 the QLBP system can be described by the following equations,

$$\begin{cases} dq(t) &= (\lambda(t) - \mu(t))1_{(q>0)}dt, \\ \mu(t) &= \frac{\mu_{\max}-\mu_{\min}}{Q_{\max}}q(t) + \mu_{\min}, \end{cases} \tag{2}$$

where $1_{(X)}$ is an indicator function, which is 1 if X is true, and 0 otherwise.

Now we examine how $\mu(t)$ responds when $\lambda(t)$ changes. Assume $\lambda(t)$ changes from 0 to λ_0 at time 0. $\lambda(t)$ can be expressed by $\lambda(t) = \lambda_0 U(t)$, where $U(t)$ is a step function. Also assume the initial condition $q(0) = 0$ (i.e., $\mu(0) = \mu_{\min}$). Then, we solve for $\mu(t)$ as follows,

$$\mu(t) = -(\lambda_0 - \mu_{\min})e^{-\frac{\mu_{\max}-\mu_{\min}}{Q_{\max}}t} + \lambda_0, \text{ for } t > 0.$$

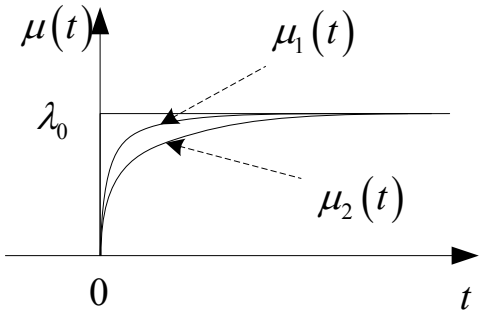


Fig. 6. Relationship between $\mu(t)$ and changes to $\lambda(t)$.

Define the response constant α by

$$\alpha = \frac{\mu_{\max} - \mu_{\min}}{Q_{\max}}. \tag{3}$$

The larger α, the faster $\mu(t)$ converges to $\lambda(t)$, as shown in Figure 6. Under the same initial condition, $\mu_1(t)$ with a larger α converges to λ_0 faster than $\mu_2(t)$ does.

2) Reduction of Auto-covariance: Next we propose a fluid model that describes the dynamics of the QLBP system. Our goal is to provide insights into how the QLBP system smoothes traffic in term of reducing auto-covariance of network traffic rate.

In this case, once the queue becomes nonempty, it remains so, though it may be very arbitrarily close to zero. Then, Equation (2) gives

$$\frac{d\mu(t)}{dt} = -\alpha\mu(t) + \alpha\lambda(t).$$

To investigate the impact of QLBP on auto-covariance of the network traffic, we consider a special case where incoming traffic is modeled as a Markov ON-OFF modeled process. The Markov ON-OFF model has been used to model voice data [13], [14] and to show the impact of the auto-covariance of network traffic on buffer size [15]–[18]. Also Willinger et al. [19], [20] characterized Ethernet LAN traffic as ON-OFF processes and interpreted the measurements in terms of exponential and heavy-tailed distributed ON/OFF durations.

Now the input traffic is modeled as a Markov ON-OFF modeled process, $\lambda(t)$, with peak rate h, ON and OFF Poisson counters N_1 and N_2 with arrival rates r_1 and r_2. Thus, $\lambda(t)$ is given by a Poisson Counter Driven Stochastic Differential Equation (PCSDE) [15]

$$\lambda(t) = hx(t),$$

where $dx(t) = (1 - x(t))dN_1(t) - x(t)dN_2(t)$. Note that the average ON and OFF period durations are $1/r_2$ and $1/r_1$, respectively, and, as a result, $E[\lambda] = hE[x] = hr_1/(r_1 + r_2)$ (for details, see [15]).

Combining them together, we have the following description of the QLBP system with a Markov ON-OFF input process,

$$\begin{cases} \lambda(t) &= hx(t), \\ dx(t) &= (1 - x(t))dN_1 - x(t)dN_2, \\ d\mu(t) &= -\alpha\mu(t)dt + \alpha\lambda(t)dt, \end{cases} \tag{4}$$

where α is given by (3).

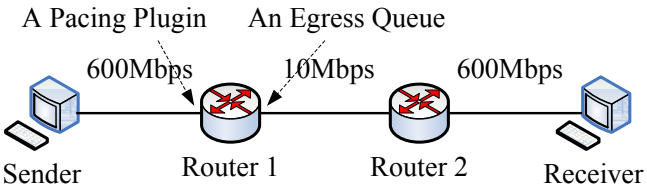


Fig. 7. Network topology for single TCP flow.

Theorem 2: Under Assumption 1, for a QLBP system described by Equation (4), the steady-state auto-covariances of the input and output processes are given by

$$C_{\lambda\lambda}(\tau) \triangleq \lim_{t\to\infty} \mathrm{Cov}(\lambda_{t+\tau}, \lambda_t) = \frac{h^2 r_1 r_2}{(r_1+r_2)^2} e^{-(r_1+r_2)\tau}, \quad (5)$$

and

$$C_{\mu\mu}(\tau) \triangleq \lim_{t\to\infty} \mathrm{Cov}(\mu_{t+\tau}, \mu_t) = \begin{cases} Ae^{-(r_1+r_2)\tau} + Be^{-\alpha\tau}, & \text{if } \alpha \neq r_1+r_2, \\ \frac{h^2 r_1 r_2}{2(r_1+r_2)^2}(1+\alpha\tau)e^{-\alpha\tau}, & \text{if } \alpha = r_1+r_2, \end{cases} \quad (6)$$

where

$$A = \frac{\alpha^2 h^2 r_1 r_2}{(r_1+r_2)^2(\alpha+r_1+r_2)(\alpha-r_1-r_2)},$$

and

$$B = -\frac{\alpha h^2 r_1 r_2}{(r_1+r_2)(\alpha+r_1+r_2)(\alpha-r_1-r_2)}.$$

Proof is provided in Appendix B.

Remark: Note that

$$C_{\mu\mu}(\tau) \approx \frac{\alpha}{\alpha+r_1+r_2}[1+(r_1+r_2)\tau]C_{\lambda\lambda}(\tau) < C_{\lambda\lambda}(\tau)$$

for small τ, which means the short-term burstiness is reduced [15], [16]. The compromise is a slower decay of the auto-covariance for large τ. However, since the decay is still exponential, this is not a great concern. Especially when the buffer is small, a significant reduction in the short-term burstiness is more desirable.

These analytical results show that QLBP has limited effect on the delay of packet transmissions, but can effectively reduce burstiness of traffic.

C. Parameter Selection

Given a QLBP system of $(Q_{\lim}, C)$, an important question that remains to be answered is how the parameters in QLBP are chosen. We formulate it as an optimization problem,

$$\min B = F(\lambda(t), \mu_{\max}, \mu_{\min}, Q_{\max}), \quad (7)$$

subject to

$$d \leq D_{\max},$$

where B is a measure of the burstiness of the underlying traffic and $D_{\max}$ is the maximum delay tolerance.

The challenge for solving the above optimization problem lies in the lack of the suitable definition of the burstiness, i.e., B, and the relationship between the burstiness and the parameters, i.e., F in (7).

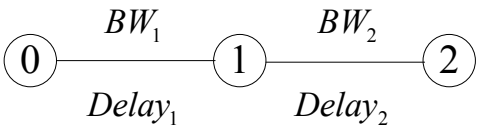


Fig. 8. A three node topology.

Our rule-of-thumb parameter settings of a QLBP system applied at a link with bandwidth C and link utilization ρ are:

$$*\mu_{\max} = \frac{1+\rho}{2}C,$$
$$\mu_{\min} = \frac{\rho}{2}C,$$
$$Q_{\max} = D_{max}\mu_{\max}.$$

IV. Simulation Results

The reduction of burstiness in network traffic translates into increased throughput performance for TCP traffic. In this section, we present results from a QLBP prototype implementation on the Open Network Laboratory (ONL) [21]. We also show results from simulation using larger-scale network configurations in ns-2 [22]. These results (1) show the pacing effect of QLBP on TCP and UDP flows, (2) validate the adaptive pacing delay introduced by QLBP, (3) quantitatively evaluate QLBP effectiveness on reducing burstiness of traffic in terms of the variance of the instantaneous traffic rate, (4) compare QLBP performance with TCP pacing in improving link utilization, and (5) show that the end-to-end delay distribution of paced traffic has a smaller tail.

A. Impact of QLBP on Single TCP and UDP Flows

This set of experiments is conducted using prototype implementation of QLBP in the Open Network Laboratory. More details on this implementation of QLBP can be found in [23]. The topology for these experiments is shown in Figure 7. A QLBP pacer is implemented as an ONL plugin and applied at the ingress port of router 1. A TCP or UDP flow is transmitted between the sender and the receiver.

The experimental setup is as follows: $\mu_{\max} = 200$Mbps, $\mu_{\min} = 1.2$Mbps, $Q_{\max}$=100pkts. The round-trip time (RTT) from the sender and the receiver is always 100ms. To create a RTT of 100ms, two 50ms `pdelay` plugins are installed at two egress ports of router 2. The buffer size of the egress queue at the 10Mbps link is 16pkts.

When using a TCP connection, Figure 9 shows that without pacing the packets within one RTT window are sent as a burst, e.g., a bunch of packets depart at the very beginning of each RTT period. Whereas, Figure 10 indicates that the QLBP pacing plugin creates a packet departure sequence that is much smoother.

When sending UDP traffic with a constant bit rate (CBR), we observe the packet arrival and departure processes shown in Figures 11 and 12. Figure 11 uses CBR traffic with a lower data rate of 200Kbps ($< \mu_{\min}$) and Figure 12 uses a higher data rate of 3Mbps ($> \mu_{\min}$). These figures show that QLBP pacing does not affect the data rate of CBR traffic in steady state.

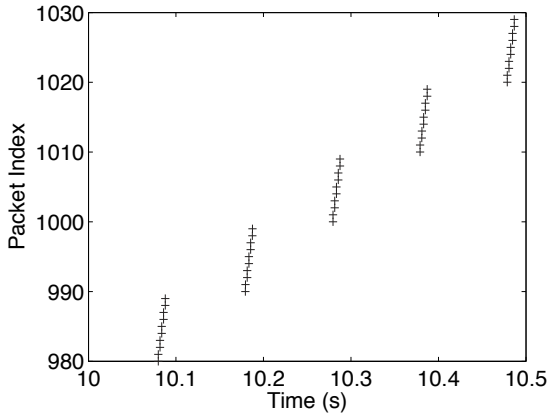


Fig. 9. Arrival process of TCP packets without pacing.

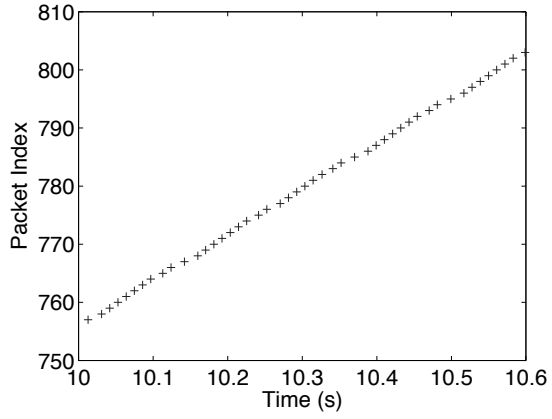


Fig. 10. Arrival process of TCP packets with QLBP pacing.

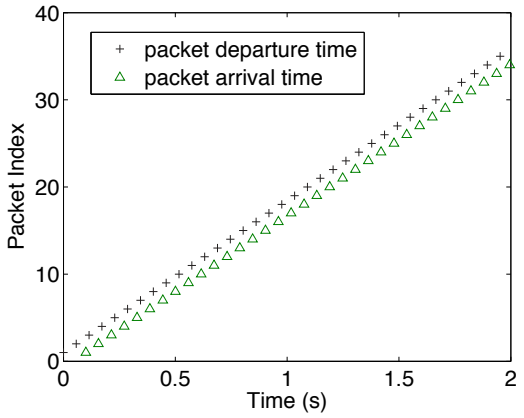


Fig. 11. Arrival and departure time of 200Kbps CBR traffic.

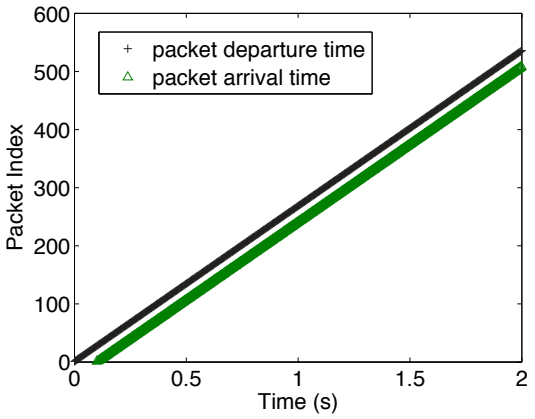


Fig. 12. Arrival and departure time of 3Mbps CBR traffic.

B. Adaptive Pacing Delay

In this ns-2 experiment, we send CBR traffic through a QLBP pacer and examine the pacing queue length Q_p and the pacing delay D_p. Figure 8 shows the topology. A CBR traffic with rate λ flows from node 0 to node 2. A QLBP pacer is placed at node 1 to pace the traffic towards node 2. The parameters are set as follows. $BW_1 = BW_2 = 15$Mbps, and $Delay_1 = Delay_2 = 10$ms. $\mu_{\max} = 10$Mbps, $\mu_{\min} = 2$Mbps, $Q_{\max} = 10$pkts and $Q_{\lim} = 1000$pkts. UDP packet size is 1000 Bytes.

Table II shows different pacing queue lengths and pacing delays under different CBR rates. When λ is smaller than or equal to $\mu_{\min}$, the pacing queue length is zero and no pacing delay is introduced. As λ increases while being still below $\mu_{\max}$, the pacing delay grows. When λ exceeds $\mu_{\max}$, the pacing delay stays at $Q_{\max}$. Since $\mu = \lambda$ in steady state, the pacing delay goes down as μ and λ increases. The relationship between λ, Q_p and D_p satisfies $D_p = Q_p/\lambda$. The delay bound in this case is 8ms (10pkts * 8000 bits per packet / 10Mbps).

C. Pacing Effectiveness

We are interested in how QLBP affects traffic burstiness. The metric of concern in this ns-2 experiment is the coefficient of variation of the traffic rate, which is used in [12] to measure the extent to which traffic is bursty. There are two sets of experiments. In the first set, we apply QLBP on a Markov ON-OFF modeled process. Using this toy model, we show how the pacing effect of QLBP can be enhanced by increasing $Q_{\max}$ or deploying multiple pacers. In the second set, we use an ns-2-integrated traffic generator, Tmix [24] to replicate a 3600 second Internet trace that was captured on a campus edge router of North Carolina State University. This traffic trace has been shown to be self-similar [24].

1) QLBP on Markov ON-OFF Modeled Process: Figure 13 shows a tandem queue topology. A Markov ON-OFF modeled process models a traffic flow from node 0 to node 1. The flow rate in the ON state is h, and 0 otherwise. We run experiments with 1, 2 and 3 pacer nodes, respectively. Even though we draw all three pacer nodes in the figure, in an experiment with i pacer nodes ($1 \leq i \leq 3$), only P1 to Pi exist to pace traffic. Parameter settings are set as follows. All links have the same delay of 2ms and bandwidth of 10Mbps. $h = 2$Mbps. The average busy and idle periods are 100ms and 200ms, respectively. $\mu_{\max} = 10$Mbps and $\mu_{\min} = 10$Kbps. UDP packet size is 1000 Bytes. $Q_{\max}$ varies from 10 to 160 and the number of pacer nodes is 1, 2 or 3, respectively. We run a 1900 second long simulation with the same $Q_{\max}$ and the number of pacer nodes 10 times to obtain the average. We analyze the trace file from 100 second to 1900 second. We set 50ms as the interval and count the amount of bytes arriving at node 1 per interval. We obtain a time series $X = \{X_i\}$ where

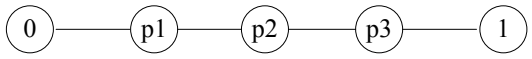


Fig. 13. A tandem queue topology.

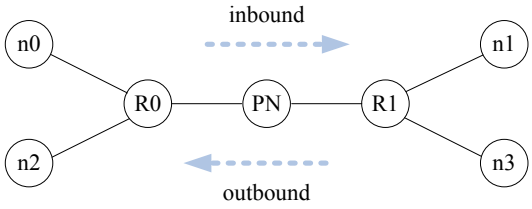


Fig. 14. A Tmix topology

X_i represents the amount of bytes arriving at node 1 during the i-th interval.

Figure 15 shows the coefficient of variation of X as well as the 95% confidence interval. X-axis is $Q_{\max}$ and Y-axis is the coefficient of variation divided by the coefficient of variation of the time series X that is generated without QLBP. Even though we do not plot it out, the average arrival rate of paced traffic (i.e., $E[X]$) is the same for all cases no matter whether and how many pacers are used, which implies that QLBP does not hurt the long-term throughput.

It is observed that a larger $Q_{\max}$ results in a smaller coefficient of variation, which is consistent with the analysis in Section III-B. Also, deploying multiple pacers can further reduce the coefficient of variation.

2) QLBP on Self-similar Internet Traffic: It is interesting how QLBP affects burstiness of real Internet traffic. We make use of Tmix in ns-2 to replicate a piece of Internet trace file that has been show to be self-similar with Hurst parameter $H = 0.95$ [24].

Figure 14 shows the topology used in this experiment. We use the exactly same topology and parameters described in a TCL script that can be found in the ns-2 manual (for details, see Chapter 43 in the ns-2 manual [22]). The inbound and outbound connection vectors files (inbound.cvec and outbound.cvec) are provided by Weigle [25]. We slightly modify the script to insert a pacer (i.e., 'PN' node as shown in Figure 14) between two Tmix-Delaybox nodes (R0 and R1) to pace inbound traffic. All the links in this topology are 1Gbps. An inbound traffic is sent from n0 to n1 while an outbound traffic is sent from n2 to n3. Figure 7 in [24] shows that inbound traffic rate varies from 10Mbps to 35Mbps with an average of 16Mbps. To better investigate the QLBP's effect on the inbound traffic, the parameters of the pacer node 'PN' are set as follows. $\mu_{\min} = 1$Mbps and $\mu_{\max} = 35$Mbps. $Q_{\max}$ varies from 5 to 320 packets.

Figure 16 shows the coefficient of variation, $CV(s)$, versus the time scale s on a $\log_2$-$\log_2$ scale. The x-axis is the base-2 logarithm of s and the y-axis the base-2 logarithm of $CV(s)$. The basic time resolution is 5ms. A point x of coordinate $(\log_2(s_0), \log_2(CV(s_0)))$ represents the base-2 logarithm of the coefficient of variation CV at time scale $5 * 2^{s_0}$ ms.

From Figure 16 we make the following observations. First, QLBP with a small $Q_{\max}$ (e.g., 5 or 10 packets) affects the coefficient of variation at small time scales. Comparing the plots of $\log_2(CV(s))$ with no pacing, $Q_{\max}$ of 5pkts and 10pkts, we see that QLBP with $Q_{\max}$ of 5pkts or 10pkts

TABLE II
PACING DELAY VS. INPUT RATE

λ (Mbps)	Q_p (pkts)	D_p (ms)
1	0	0
2	0	0
4	2	4
6	4	5.33
8	7	7
10	9	7.2
12	10	6.67
15	10	5.3

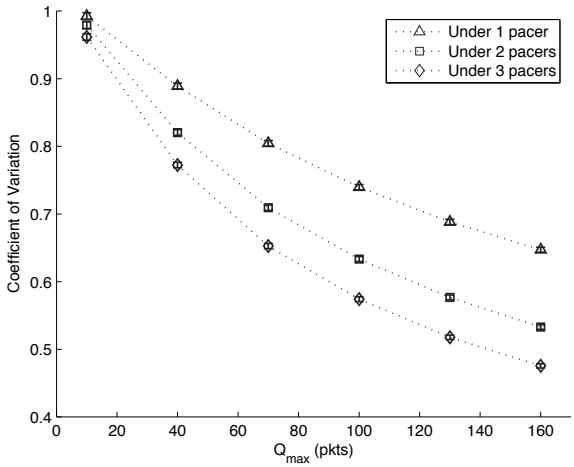


Fig. 15. Pacing effect of QLBP on Markov ON-OFF modeled process.

reduces the coefficient of variation by nearly 50% at time scale 5ms ($s = 0$). As s goes up, $\log_2(CV(s))$ with $Q_{\max}$ of 5 or 10 packets converts to that with no pacing, indicating the fading impact of pacing. Second, the larger $Q_{\max}$, the wider the range of time scale in which QLBP has a significant impact on burstiness. A larger $Q_{\max}$ (e.g., 160pkts or 320pkts) results in a significant reduction at large time scales (e.g., 2.5s ($s = 512$) or 5s ($s = 1024$)). This is because a large $Q_{\max}$ makes the rate-controller of QLBP respond less sensitively to the change in the instant input rate.

D. Improvement on Link Utilization

In this sub-section we investigate the impact of short-term burstiness on a non-bottleneck link in terms of link utilization. This set of experiments is used in [7] to show the performance improvement of TCP pacing in small buffer networks. The topology used in this set of experiments is a dumbbell one, as shown in Figure 17.

Core router C0 is connected to four access routers Aj ($1 \leq j \leq 4$), each connecting ten sender nodes Si ($1 \leq i \leq 10$). Core router C1 is connected to ten receiver nodes Ri ($1 \leq i \leq 10$). The bandwidths of all links are 100Mbps. Delays between Aj's and C0 and between C0 and C1 are set to 20ms and delays between sender nodes and access routers and between C1 to receiver nodes are uniformly distributed in $[1 : 10ms]$ to reduce the impact of TCP synchronization. The average RTT is about 100ms. 40 long-lived TCP flows are sent from 40 senders to 10 receivers. For each TCP flow, the maximum congestion window is set 32packets

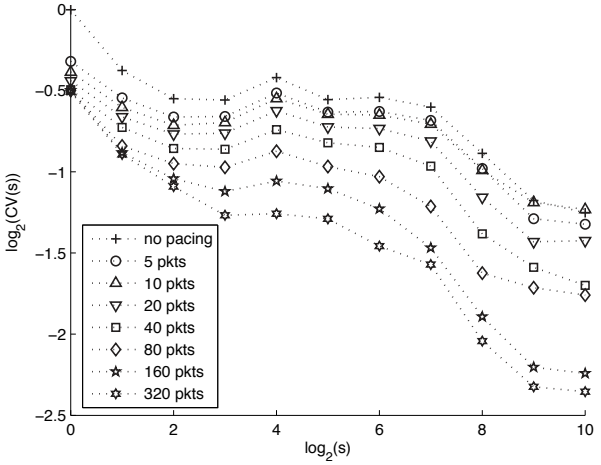


Fig. 16. Pacing effect of QLBP on self-similar Internet traffic

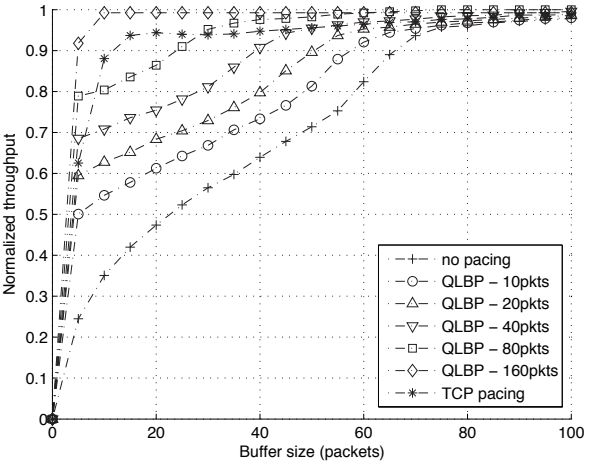


Fig. 18. Link utilization vs. various buffer sizes.

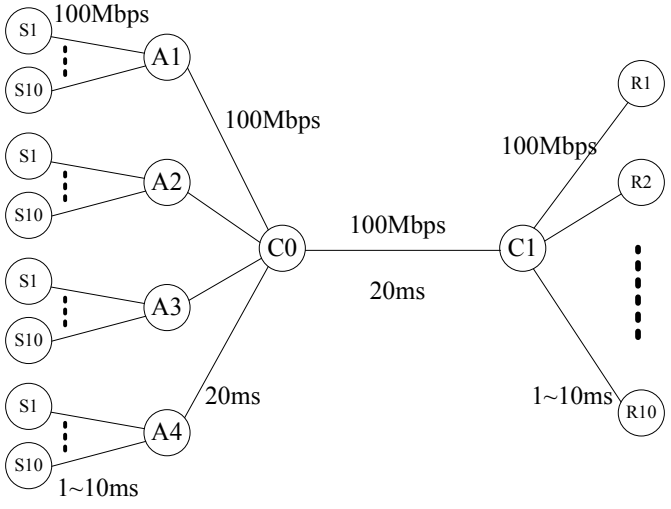


Fig. 17. A dumbbell topology

and packet size is set 1000Bytes. The maximum throughput of one TCP session on average is bounded by 2.5Mbps ($\approx 1000Bytes/packet * 8bits/byte * 32packets/100ms$). To reduce the impact of synchronization, the start times of 40 TCP sessions are uniformly distributed in $[0 : 100s]$. We apply four QLBP pacers on four access routers, each on the link Aj-C0 ($1 \leq j \leq 4$) with $\mu_{\max} = 100$Mbps and $\mu_{\min} = 1$Mbps. Buffer sizes of Aj ($1 \leq j \leq 4$) are set to be 2000 packets. The $Q_{\max}$ values at the four QLBP pacers are the same, varying from 10 to 160 packets. The buffer size at C0 varies from 1 to 100 packets. Each simulation run lasts one thousand seconds and the steady state starts at 200s. The metric is the normalized throughput (defined as the ratio of the total throughput to the link bandwidth) of link C0-C1 in steady state.

Figure 18 shows the normalized throughput (i.e., the link utilization) versus the buffer size at router C0. For a small buffer of 5 packets, QLBP with $Q_{\max}$ of 10 packets can improve link utilization by nearly 100%. QLBP with $Q_{\max}$ of 80 packets outperforms TCP pacing when the buffer size grows beyond 30 packets. QLBP with $Q_{\max}$ of 160 packets outperforms TCP pacing over the whole range of buffer size.

TABLE III
LINK UTILIZATION AND DELAY FOR NON-PACING AND QLBP PACING.

Pacing technique	link utilization	delay		
		average	minimum	maximum
no pacing	47.57%	56.9ms	50.8ms	63.4ms
QLBP (40pkts)	76.33%	51.9ms	46.3ms	57.5ms
QLBP (160pkts)	98.48%	60.8ms	55.2ms	66.5ms

E. Delay Distribution

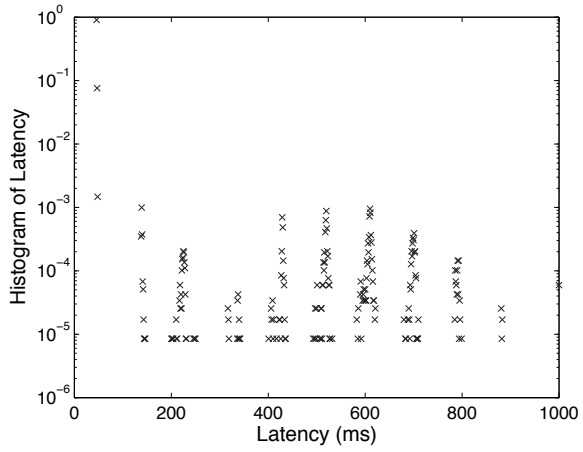
In the introduction to this paper, we argued that a long tail in the delay distribution for packet transmission in the transport layer leads to poor performance. In Figure 19, we show the delay distributions for successful packet transmissions in TCP connections in ns-2 simulations. The different figures show the distribution for a network without pacing, for QLBP pacing with a small amounts of pacing ($Q_{\max}$=40 packets), and for QLBP pacing with a large amounts of pacing ($Q_{\max}$=160 packets). As expected, the tail of the distribution decreases with more pacing.

Table III shows the corresponding link utilization and average, minimum, and maximum packet delays. These results confirm that QLBP pacing meets the goals that we set in our work: we achieve better throughput performance (as indicated by higher link utilization) at the cost of a slightly larger delay (when comparing QLBP ($Q_{\max}$=160) with no pacing). Interestingly, QLBP ($Q_{\max}$=40) achieves both higher bandwidth and lower delay. This is accomplished by avoiding packet loss with only small amounts of additional delay.

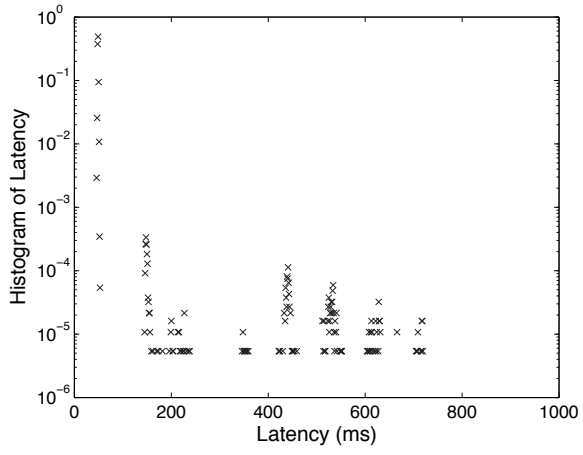
V. RELATED WORK

The impacts of small buffers on transport-layer network performance have been studied in the context of real-time traffic and TCP traffic [7], [11], [12], [26]–[28]. Interestingly, the results of these studies are not conclusive.

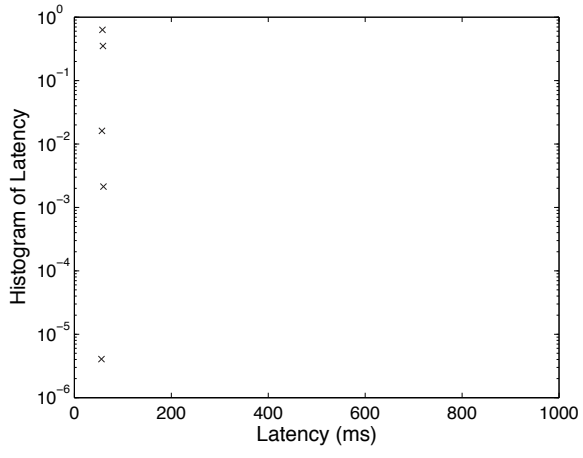
On one hand, it has been shown that small buffers significantly degrade network performance with ordinary TCP sessions by causing packet drop more frequently. Enachescu et al. [7] showed that a 80% workload consisting of long-lived TCP sessions only achieves a 20% link utilization when the buffer size of the shared link is 10 packets. Sivaramman et



(a) No pacing.



(b) QLBP pacing (40 packets).



(c) QLBP pacing (160 packets).

Fig. 19. End-to-end delay distribution for reliable packet transmissions. Long delays are caused by retransmissions in transport layer.

al. [12] demonstrated that "a 10Gbps optical packet switching (OPS) node with 10 to 20 packets can experience significant losses even at low (40%) to moderate (60% for long-range dependent or 80% for short-range dependent) traffic loads."

On the other hand, theoretical analyses and empirical results show that small buffers are feasible for core routers through which tens of thousands of TCP sessions flow [7], [11], [26]–[28]. Enachescu et al. [7] argued that $O(\log W)$ buffers are sufficient for high throughput, where W is congestion window size of each flow, and router buffer can even be reduced to a few dozen packets if a small amount of link utilization is sacrificed. Gu et al. [11] demonstrated that more than 90% link utilization is achievable in a 1–10 Gbps bottleneck link with a buffer of 20 packets. Lakshmikantha et al. [28] further showed that $O(1)$ buffer sizes (20 packets) are sufficient for good performance with no loss of link utilization when considering the impact of file arrivals and departures. We note that all high performance results are achieved only when TCP sessions are paced by either some rate-control mechanism (i.e., TCP pacing) or access links with capacities much slower than the bottleneck link.

The main concern with the small buffer core networks is the high packet loss probability due to the small buffer size and the bursty behavior of TCP. Several techniques are proposed to lower the drop probability in small buffer networks by smoothing network traffic. Packet pacing finds its roots in the explicit rate control non-TCP protocols, which send data at a fixed rate irrespective of the receipt of acknowledgments [29], [30]. Pacing was used in the TCP context to correct the compression of acknowledgements due to cross traffic [31], to avoid slow start [32], [33], after packet loss [34], or when an idle connection resumes [35]. Aggarwal et al. [8] concluded that pacing improves throughput in some cases but in general decreases performance. The poor performance of pacing is attributed mostly to "synchronized drops" and packet delays being misinterpreted as congestion.

In addition to TCP pacing, there have been several proposals for resolving packet drops in small buffer networks [12], [36]–[39]. The work by Alparslan et al. [36] shares a very similar idea with our, i.e., turning the pacing rate based on the buffer occupancy, and the effect of the pacing is evaluated in a large-scale hypothetic network. The work by Sivaraman et al. [12] stems from previous works on traffic conditioners for video transmission, called traffic conditioning *off-line* [40]. They proposed an on-line version of traffic conditioner based on this traffic conditioning *off-line*. The approaches in [37]–[39] rely on the global network-wide coordinated scheduling.

Unlike the above pacing-based approaches, Vishwanath et al. proposed to recover lost packets by using the packet-level forward error correction (FEC) scheme [41]. Their coding-based approach works based on an observation that "loss at core links is due to contention, not congestion." Through simulation they show the efficiency of the FEC-based approach.

VI. Summary

Our work presents a novel view on the tradeoff between link bandwidth and packet delay. Instead of using an error correction or network coding approach where more bandwidth is used to avoid packet losses, we proposed to delay packet transmissions to reduce the burstiness of traffic and thus reduce packet losses in small-buffer networks. We present Queue Length Based Pacing, which is a pacing technique that uses a single pacing queue on router ports and adapts its sending rate based on the amount of traffic that is buffered at that port. Our analysis shows that pacing delay due to QLBP is bounded and

that the variance of the instantaneous traffic rate is reduced. We show the effectiveness of QLBP through a prototype implementation and simulation. Specifically, we show that TCP connections in a small-buffer network with QLBP pacing achieve higher link utilization than in non-paced networks. Therefore, we believe that QLBP is an effective approach to improving the operation of networks and improving the effective bandwidth of connections at the cost of only small amounts of additional delay.

APPENDIX

In this section we provide the proofs of Theorem 1 and Theorem 2 made in Section III.

A. Proof of Theorem 1

Proof: According to the amplitude of λ (i.e., the input rate in steady state), we prove Theorem 1 in four cases. Note that $d_{FIFO} = 0$ for $\lambda \leq C$.

Case 1: $\lambda \leq \mu_{\min}$

$d = d_{pacer} - d_{FIFO} = 0 < \frac{Q_{\max}}{\mu_{\max}}$.

Case 2: $\mu_{\min} < \lambda \leq \mu_{\max}$

Without loss of generality, let $\lambda = \beta\mu_{\max} + (1-\beta)\mu_{\min}$, where $0 < \beta \leq 1$. Thus, we have

$$\begin{aligned} d = d_{pacer} - d_{FIFO} &= \frac{q_\lambda}{\mu} - 0 \\ &= \frac{Q_{\max}\frac{\lambda-\mu_{\min}}{\mu_{\max}-\mu_{\min}}}{\lambda} = \frac{Q_{\max}}{\mu_{\max} + \frac{1-\beta}{\beta}\mu_{\min}} \\ &\leq \frac{Q_{\max}}{\mu_{\max}}. \end{aligned}$$

Case 3: $\mu_{\max} \leq \lambda \leq C$

In this case, the pacing queue length stays at $Q_{\max}$, as demonstrated in Section IV-B. $d = d_{pacer} - d_{FIFO} = \frac{Q_{\max}}{\lambda} - 0 < \frac{Q_{\max}}{\mu_{\max}}$.

Case 4: $\lambda > C$

In this case the input traffic saturates the bottleneck link and overflows the router buffer. For the packets who successfully pass the delay/FIFO queue, we have $d = d_{pacer} - d_{FIFO} = \frac{Q_{\lim}}{C} - \frac{Q_{\lim}}{C} = 0 < \frac{Q_{\max}}{\mu_{\max}}$.

Thus, we always have $d \leq \frac{Q_{\max}}{\mu_{\max}}$ no matter how big λ is. Hence, Theorem 1 is proved. ∎

B. Proof of Theorem 2

This subsection will guide through the detailed process of calculating the auto-covariance of a paced On-Off process.

Proof:

For the sake of clarity, we will use the subscript notations, i.e. write x_t for $x(t)$, etc. In steady state, the expectation of x_t is $E[x] \triangleq \lim_{t\to\infty} E[x_t] = \frac{r_1}{r_1+r_2}$ and its auto-covariance is $C_{xx}(\tau) \triangleq \lim_{t\to\infty} \mathrm{Cov}(x_t, x_{t+\tau}) = \frac{r_1 r_2}{(r_1+r_2)2} e^{-(r_1+r_2)\tau}$. Therefore,

$$E[\lambda] \triangleq \lim_{t\to\infty} E[\lambda_t] = \lim_{t\to\infty} hE[x_t] = \frac{hr_1}{r_1+r_2},$$

and

$$C_{\lambda\lambda}(\tau) \triangleq \lim_{t\to\infty} \mathrm{Cov}(\lambda_t, \lambda_{t+\tau}) = \frac{h2r_1r_2}{(r_1+r_2)2} e^{-(r_1+r_2)\tau}.$$

Moreover,

$$E[\mu] \triangleq \lim_{t\to\infty} E[\mu_t] = E[\lambda] = \frac{hr_1}{r_1+r_2}.$$

Next we compute the steady-state cross-covariance $C_{x\mu}(\tau)$. Note that $d(x_t\mu_t) = \mu_t(1-x_t)dN_1 - \mu_t x_t dN_2 - \alpha x_t\mu_t dt + \alpha h x_t dt$. Taking expectations gives

$$E[x\mu] \triangleq \lim_{t\to\infty} E[\mu_t x_t] = \frac{hr_1(r_1+\alpha)}{(r_1+r_2)(r_1+r_2+\alpha)}.$$

Note also that $d(x_t\mu_s) = \mu_s(1-x_t)dN_1 - \mu_s x_t dN_2$. where s is held constant. Taking expectations gives

$$\frac{d}{dt}E[x_t\mu_s] = r_1 E[\mu_s] - (r_1+r_2)E[x_t\mu_s],$$

which yields

$$\begin{aligned} E[x_t\mu_s] =& \frac{r_1}{r_1+r_2}E[\mu_s] \\ &+ \left(E[x_s\mu_s] - \frac{r_1}{r_1+r_2}E[\mu_s]\right) e^{-(r_1+r_2)(t-s)}. \end{aligned}$$

Letting $t, s \to \infty$ such that $t - s = \tau$ is constant, we have

$$\begin{aligned} C_{x\mu}(\tau) &= \lim_{s\to\infty} E[x_{s+\tau}\mu_s] - E[x]E[\mu] \\ &= \frac{\alpha h r_1 r_2}{(r_1+r_2)2(r_1+r_2+\alpha)} e^{-(r_1+r_2)\tau}. \end{aligned}$$

Finally, we compute the auto-covariance $C_{\mu\mu}(\tau)$. Note that $d\mu_t 2 = -2\alpha\mu_t 2dt + 2\alpha h x_t\mu_t dt$. Taking expectations, we have

$$E[\mu 2] \triangleq \lim_{t\to\infty} E[\mu_t 2] = hE[x\mu] = \frac{h2r_1(r_1+\alpha)}{(r_1+r_2)(r_1+r_2+\alpha)}.$$

Note also that $d(\mu_t\mu_s) = -\alpha\mu_t\mu_s dt + \alpha h x_t\mu_s dt$, which, upon taking expectations, gives

$$\frac{d}{dt}E[\mu_t\mu_s] = -\alpha E[\mu_t\mu_s] + \alpha h E[x_t\mu_s].$$

Plugging in the formula for $E[x_t\mu_s]$ and solving for $E[\mu_t\mu_s]$,

$$E[\mu_t\mu_s] = \frac{hr_1}{r_1+r_2}E[\mu_s] + A(s)e^{-(r_1+r_2)(t-s)} + B(s)e^{-\alpha(t-s)}.$$

where $A(s) = \frac{\alpha h}{\alpha - r_1 - r_2}\left(E[x_s\mu_s] - \frac{r_1}{r_1+r_2}E[\mu_s]\right)$ and $B(s) = E[\mu_s 2] - \frac{hr_1}{r_1+r_2}E[\mu_s] - A(s)$, assuming $\alpha \neq r_1 + r_2$. Letting $t, s \to \infty$ such that $t - s = \tau$ is constant, we have

$$\begin{aligned} C_{\mu\mu}(\tau) &= \lim_{s\to\infty} E[\mu_{s+\tau}\mu_s] - (E[\mu])2 \\ &= Ae^{-(r_1+r_2)\tau} + Be^{-\alpha\tau}. \end{aligned}$$

where A and B are as in the theorem. When $\alpha = r_1 + r_2$, l'Hôpital's rule gives

$$C_{\mu\mu}(\tau) = \frac{h2r_1r_2}{2(r_1+r_2)2}[1 + (r_1+r_2)\tau]e^{-(r_1+r_2)\tau}.$$

Thus, $C_{\mu\mu}(\tau)$ in Theorem 2 is derived. ∎

REFERENCES

[1] *International Standard ISO/IEC 7498-1*, 2nd ed., International Organization for Standardization / International Electrotechnical Commission, Geneve, Switzerland, Nov. 1994.

[2] M. Shifrin and I. Keslassy, "Small-buffer networks," *Computer Networks*, vol. 53, no. 14, pp. 2552–2565, Sep. 2009.

[3] J. Postel, "User Datagram Protocol," Information Sciences Institute, RFC 768, Aug. 1980.

[4] ——, "Transmission Control Protocol," Information Sciences Institute, RFC 793, Sep. 1981.

[5] T. K. Moon, *Error Correction Coding: Mathematical Methods and Algorithms*. Wiley-Interscience, Jun. 2005.

[6] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[7] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden, "Routers with very small buffers," in *Proc. Twentyfifth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2006)*, Barcelona, Spain, Apr. 2006.

[8] A. Aggarwal, S. Savage, and T. Anderson, "Understanding the performance of TCP pacing," in *Proc. IEEE INFOCOM 2000*, Tel Aviv, Israel, Mar. 2000, pp. 1157–1165.

[9] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "RFC 2018: Tcp selective acknowledgement options," Apr. 1996.

[10] S. Floyd, J. Mahdavi, M. Mathis, and A. Romanow, "RFC 2883: An extension to the selective acknowledgement (SACK) option for tcp," Jul. 2000.

[11] Y. Gu, D. Towsely, C. V. Hollot, and H. Zhang, "Congestion control for small buffer high speed networks," in *Proc. IEEE INFOCOM 07*, Anchorage, Alaska, May 2007, pp. 1037–1045.

[12] V. Sivaraman, H. Elgindy, D. Moreland, and D. Ostry, "Packet pacing in short buffer optical packet switched networks," in *Proc. IEEE INFOCOM 06*, Spain, Apr. 2006.

[13] H. Heffes and D.Lucantoni, "A markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance," in *IEEE J. Sel. Areas Commun.*, Sep. 1986, pp. 856–868.

[14] I. Nikolaidis and I. Akyildiz, "Source characterization and statistical multiplexing in ATM networks," Georgia Tech., Technical Report GIT-CC 92-24, 1992.

[15] R. W. Brockett, W. Gong, and Y. Guo, "Stochastic analysis for fluid queueing systems," in *IEEE CDC*, Dec. 1999.

[16] Y. Huang, Y. Liu, W. Gong, and D. Towsley, "Two-level Stochastic Fluid Tandem Queuing Model for Burst Impact Analysis," in *IEEE CDC*, Dec. 2007.

[17] Y. Wu, W. Gong, and D. Towsley, "Analysis of abstract simulation via stochastic differential equation models," in *IEEE CDC '03*, Dec 2003.

[18] C. V. Hollot, Y. Liu, V. Misra, and D. Towsley, "Unresponsive Flows and AQM Performance," in *Proc. IEEE INFOCOM*, Apr 2003.

[19] W. Willinger, M. Taqqu, R. Sherman, and D. Wilson, "Self-similarity through highvariability: statistical analysis of ethernet lan traffic at the source level," pp. 100–113, Aug. 1995.

[20] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson, "Self-similarity through high-variability: Statistical analysis of ethernet lan traffic at the source level," *IEEE/ACM Trans. Netw.*, vol. 5, pp. 71–86, 1997.

[21] J. DeHart, F. Kuhns, J. Parwatikar, J. Turner, C. Wiseman, and K. Wong, "The open network laboratory: a resource for networking research and education," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 5, pp. 75–78, Oct. 2005.

[22] The network simulator - ns-2, http://www.isi.edu/nsnam/ns/.

[23] Y. Cai, S. Hanay, and T. Wolf, "Practical packet pacing in small-buffer networks," in *ICC '09*, Dresden, Germany, Jun. 2009.

[24] M. C. Weigle, P. Adurthi, F. H.-C. K. Jeffay, and F. D. Smith, "Tmix: A tool for generating realistic tcp application workloads in ns-2," *SIGCOMM Computer Communication Review*, vol. 36, no. 3, pp. 67–76, 2006.

[25] inbound.cvec and outbound.cvec, http://www.cs.odu.edu/ netsim/ TrafGen/Traces-tmix-ccr06.

[26] D. Wischik and N. McKeown, "Part I: Buffer sizes for core routers," *ACM SIGCOMM Comput Commun Rev*, pp. 75–78, Jul. 2005.

[27] G. Raina, D. Towsely, and D. Wischik, "Part II: Control theory for buffer sizing," *ACM SIGCOMM Comput Commun Rev*, pp. 79–82, Jul. 2005.

[28] A. Lakshmikantha, R. Srikant, and C. Beck, "Impact of File Arrivals and Departures on Buffer Sizing in Core Routers," in *Proc. IEEE INFOCOM*, 2008.

[29] D. D. Clark, M. M. Lambert, and L. Zhang, "NETBLT: A high throughput transport protocol," *ACM SIGCOMM Comp. Comm. Rev.*, vol. 17, pp. 353–359, Aug. 1987.

[30] F. Bonomi and K. Fendick, "The rate based flow control framework for the available bit rate ATM service," *IEEE Network*, pp. 25–39, 1998.

[31] L. Zhang, S. Shenker, and D. D. Clark, "Observations on the dynamics of a congestion control algorithm: the effects of two way traffic," in *Proc. ACM SIGCOMM 91*, Zurich, Switzerland, Sep. 1991, pp. 133–147.

[32] M. Aron and P. Druschel, "TCP: Improving startup dynamics by adaptive timers and congestion control," Rice University, Technical Report TR98-318, 1998.

[33] V. N. Padmanabhan and R. H. Katz, "TCP Fast Start: A technique for speeding up web transfers," in *Proc. IEEE GLOBECOMM*, Sydney, Australia, Nov. 1998.

[34] J. Hoe, "Start-up dynamics of TCP's congestion control and avoidance schemes," Masterthesis, MIT, Jun. 1995.

[35] V. Visweswaraiah and J. Heidermann, "Improving restart of idle TCP connections," University of Southern California, Technical Report TR97-661, 1997.

[36] O. Alparslan, S. Arakawa, and M. Murata, "Node pacing for optical packet switching," in *Proc. Photonics in Switching, 2008*, Sapporo, Aug. 2008.

[37] J. Naor, A. Rosen, and G. Scalosub, "Online time-constrained scheduling in linear networks," in *Proc. IEEE INFOCOM 05*, Miami, FL, Mar. 2005.

[38] M. Adler, S. Khanna, R. Rajaraman, and A. Rosen, "Time-constrained scheduling of weighted packets on trees and meshes," *Algorithmica*, vol. 36, no. 2, pp. 123–152, 2003.

[39] M. Adler, A. L. Rosenberg, R. K. Sitaram, and W. Unger, "Scheduling time-constrained communication in linear networks," *Theoretical Comp. Sc*, vol. 35, no. 6, pp. 559–623, 2002.

[40] J. D. Salehi, Z. Zhang, J. Kurose, and D. Towsley, "Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing," *IEEE/ACM Trans. Netw.*, vol. 6, no. 4, pp. 397–410, 1998.

[41] A. Vishwanath, V. Sivaraman, M. Thottan, and C. Dovrolis, "Enabling a bufferless core network using edge-to-edge packet-level fec," in *Proc. IEEE INFOCOM 10*, San Diego, CA, Mar. 2010.

Yan Cai (M'10) received his B.E. and M.S. from the Department of Automation at Tsinghua University, Beijing, China in 2000 and 2003, respectively. He is a doctoral candidate in the Department of Electrical and Computer Engineering at the University of Massachusetts Amherst. His research interests include analytic modeling of networks and switch design.

Tilman Wolf (M'02-SM'07) is an associate professor in the Department of Electrical and Computer Engineering at the University of Massachusetts Amherst. He received a D.Sc. in computer science in 2002, all from Washington University in St. Louis. His research interests include network processors, their application in next-generation Internet architectures, and embedded system security.

Weibo Gong (M'87-SM'97-F'99) received the Ph.D. degree from Harvard University, Cambridge, MA, in 1987. He is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, since 1987. His major research interests include control and systems methods in communication networks, network security, and network modeling and analysis.