A Practical On-line Pacing Scheme at Edges of Small Buffer Networks

Yan Cai†, Bo Jiang‡, Tilman Wolf†, Weibo Gong†
†Department of Electrical and Computer Engineering, ‡Department of Computer Science
University of Massachusetts, Amherst
Emails: {ycai,wolf,gong}@ecs.umass.edu, bjiang@cs.umass.edu

***Abstract*—For the optical packet-switching routers to be widely deployed in the Internet, the size of packet buffers on routers has to be significantly small. Such small-buffer networks rely on traffic with low levels of burstiness to avoid buffer overflows and packet losses. We present a pacing system that proactively shapes traffic in the edge network to reduce burstiness. Our queue length based pacing uses an adaptive pacing on a single queue and paces traffic indiscriminately where deployed. In this work, we show through analysis and simulation that this pacing approach introduces a bounded delay and that it effectively reduces traffic burstiness. We also show that it can achieve higher throughput than end-system based pacing.**

I. Introduction

Self-similarity of Internet traffic indicates that burstiness exhibits at a wide range of time scales [26], [33], [13]. It was pointed out in [16] that the long-term and short-term burstiness are mainly dominated by user/session attributes and TCP congestion control mechanisms, respectively. Router buffers play a critical role in absorbing bursts to maintain a certain level of performance of packet-switching networks. Nowadays, commercial routers are equipped with buffers of a rule-of-thumb size, that is, the bandwidth-delay product [11], [7]. A 10Gbps link requires 2.5Gbits of buffers to store packets of 250ms. However, in all-optical routers, traditional electronic buffers are replaced with optical buffers, which are usually implemented using the optical-delay-line technique [38], and whose buffer sizes are limited to a dozen of packets [32], [25], [27]. It is shown [15] that the bursty nature of TCP makes flows experience packet drop more frequently when buffer sizes are small, and as a result, the utilization of the shared link is limited to a fairly low level. Therefore, how to eliminate or reduce the short-term burstiness in small buffer networks becomes an important issue.

TCP pacing, as a natural way to reduce TCP burstiness, has been widely used in researches on small buffer networks performance [15], [18], [24]. However, as shown in these previous works, paced TCP only outperforms non-paced TCP when buffer sizes are small. Aggarwal et al. studied the performance of TCP pacing over a wide variety of large buffer networks and found that pacing improves throughput, reduces delays in some cases, but in general decreases performance [3]. As demonstrated in [3], paced TCP also has poor competitive capacity with non-paced TCP, which prevents TCP pacing from being widely adopted. Therefore, it is important to develop a novel pacing approach for small-buffer networks.

In this paper we present a practical packet pacing scheme, known as queue length based pacing (QLBP), to tackle the burstiness issue in small buffer networks. This pacing approach uses a single queue and paces traffic indiscriminately. The pacing rate is a function of the queue length of the buffer. When only a few packets are buffered, additional delay is introduced to achieve traffic pacing. As the queue length increases, the additional delay decreases to achieve the full link rate in the limit. This very simple approach to pacing can easily be implemented in commercial routers due to its low complexity. By deploying multiple QLBP pacers at access networks, traffic can be smoothed before entering a small-buffer core network. As a result, packet losses in the small buffer network are significantly reduced and high throughput can be achieved.

Our queue-length based pacing algorithm has been briefly described in [8]. The work presented in this paper significantly extends this initial idea, provides a thorough theoretical basis, and demonstrates its effectiveness through extensive simulations. Our specific contributions are:

1) We show theoretically that the pacing delay introduced by QLBP is adaptive to changes of the incoming traffic rate, and is upper bounded by a constant dependent only on the parameters of QLBP.
2) We quantitatively analyze the pacing effects of QLBP in reducing burstiness of network traffic in terms of the variance of the instantaneous rate of the input traffic in the context of a fluid model.
3) We evaluate the pacing performance of QLBP on a self-similar traffic generated by Tmix [40] in ns2 [37] and quantitatively compare its performance in improving link utilization of a non-bottleneck link with TCP pacing.

In addition to TCP pacing, there have been other proposed schemes for resolving the packet drop problem in small buffer networks [29], [1], [2]. The work by Sivaraman et al., referred to as delay-budget based pacing, is similar to ours in terms of the deployment location of pacers. Our work differs in two aspects:

1) The computing complexity of QLBP is $O(1)$ whereas that of the delay budget-based pacing is $O(\log n)$, where n is the number of queued packets [34].
2) While every packet experiences the same delay in the delay-budget based pacing, QLBP adaptively adjusts

pacing delay for incoming traffics of various input rate: the lower the input rate, the shorter the pacing delay.

These differences have considerable practical impacts in terms of effectiveness of the pacing approach and the ease of deployment in practical networks.

The remainder of the paper is organized as follows. We introduce the QLBP scheme as well as its implementation in ns2 in Section III. We derive the upper bound on pacing delay and analyze quantitatively its effectiveness in smoothing packet traffic in Section IV. A set of experiments are conducted to show its pacing effect in Section V. The paper concludes in Section VI.

II. Related Work

The burstiness of Internet traffic is roughly categorized into two classes: "long-term" and "short-term". Feldmann et al. [16] pointed out that long-term (from hundreds of milliseconds to tens of minutes) and short-term (on the order of a few hundreds milliseconds and below) burstiness are mainly dominated by user/session attributes and TCP congestion control mechanisms.

The impacts of small buffers on network performance have been studied in the context of real-time traffic and TCP traffic [43], [35], [36], [15], [18], [24]. On one hand, small buffers significantly degrade network performance with ordinary TCP sessions by causing packet drop more frequently. Enachescu et al. [15] 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 al. [36] 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 conclusions show that small buffers are feasible for core routers through which tens of thousands of TCP sessions flow [15], [43], [35], [18], [24]. Enachescu et al. [15] 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. [18] demonstrated that a more than 90% link utilization is achievable in a X Gbps bottleneck link with a buffer of 20 packets, where X is in the range from 1 to 10. Lakshmikantha et al. [24] further showed that $O(1)$ buffer sizes, standing for 20 packets, are sufficient for good performance with no loss of link utilization when considering the impact of file arrivals and departures. We notice that all high performance results are achieved only when TCP sessions are paced by either some rate-control mechanism, known as TCP pacing, or access links with capacities much slower than the bottleneck link.

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 [12], [5]. Pacing was used in the TCP context to correct the compression of acknowledgements due to cross traffic [45], to avoid slow start [4], [31], after packet loss [20], or when an idle connection resumes [39]. Aggarwal et al. [3] 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 other proposals for resolving packet drops in small buffer networks [36], [29], [1], [2]. The work by Sivaraman et al. [36] stems from previous works on traffic conditioners for video transmission, called traffic conditioning *off-line*. They proposed an on-line version of traffic conditioner based on this traffic conditioning *off-line*. These approaches in [29], [1], [2] rely on the global network-wide coordinated scheduling.

An IPA (Infinitesimal Perturbation Analysis)-based analysis presented in [10] examines the impact of burstiness on network performance from a queueing system perspective and shows that there exists a linear relationship between average queue length and average input rate of a server under certain conditions.

III. Queue Length Based Pacing

We first briefly review previous works on burstiness in TCP. By reviewing this, we point out why TCP burstiness is still a big issue for the future Internet. We then introduce the ideas behind queue length based pacing (QLBP) and a specific algorithm that implements QLBP in ns2.

A. TCP Burstiness

TCP can pace itself due to ACK-clocking, that is, acknowledgments are spaced out by a bottleneck link and as a result, packets sent in the congestion avoidance phase are spaced by acknowledgement arrivals. However, as pointed out by Aggarwal et al. in [3], 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 [3]). Even though the impact of retransmissions of lost packets can somehow be mitigated by enabling TCP selective acknowledgement (SACK) options [28], [17], the negative impact of ACK-compression and multiplexing might become even worse in the future Internet with much larger bandwidth.

Let us take a closer look at TCP dynamics. For simplicity, we only examine the TCP congestion control 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 window size. 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 rate of transmitting a burst of packets is determined by the physical speed of the

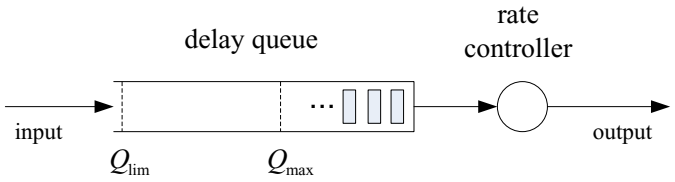

Fig. 1. QLBP System

slowest link along the path of that TCP session. However, this rate may be much higher than the TCP connection's available bandwidth. This difference is the source of burstiness in the TCP session. As physical link speeds increase in the future Internet [18], this burstiness will be more severe.

B. Overview of Queue Length Based Pacing System

The general ideal of QLBP is illustrated in Figure 1, and the major notation used in this paper is summarized in Table I.

The figure shows a single input and output, but the concepts are applicable to routers with any number of ports. A QLBP system includes a delay queue and a rate controller, and has three parameters: μ_{max}, μ_{min} and Q_{max}. The delay queue in Figure 1 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.

A simple and straightforward example is to apply QBLP 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.

C. Pacing Delay

One of the key aspects of any pacing algorithm is how the inter-packet pacing delay is determined. In TCP pacing [3], the inter-packet pacing delay is roughly set to the ratio of the current RTT over the congestion window size. In the pacing scheme proposed by Sivaranman [36], 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:

- For longer queue lengths, the pacing delay is lower. This rule ensures that the link can be fully utilized.
- For packets that arrive at a rate lower than μ_{min}, they do not get delayed. This rule ensures that pacing is only activated when multiple packets arrive at a certain high rate.

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

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

TABLE I
MAJOR NOTATION USED IN THIS PAPER

Defined in Section III-B	
$q(t)$	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
μ_{max}	maximum rate at which the rate controller transmits packets when pacing is enabled
μ_{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-D	
q	a delay queue object
p	a packet object
t_{last}	time at which the last packet was sent from q
t_{next}	time at which the next packet is allowed to send from q
S_p	size of the packet at the head of queue q
Defined in Section IV-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 IV-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

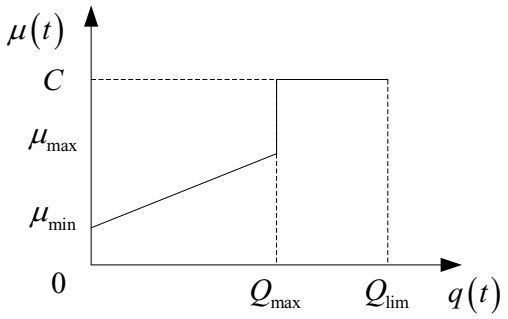

Fig. 2. Pacing rate $\mu(t)$ v.s. queue length $q(t)$

Figure 2 depicts the output rate $\mu(t)$ versus the instantaneous queue length $q(t)$.

In the following we use a simple example 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, μ_{min} and μ_{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 $a > b$.

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 μ_{min} (i.e., $\mu(t)$), a queue begins to be built up, i.e., $q(t) > 0$, which causes $\mu(t)$ to

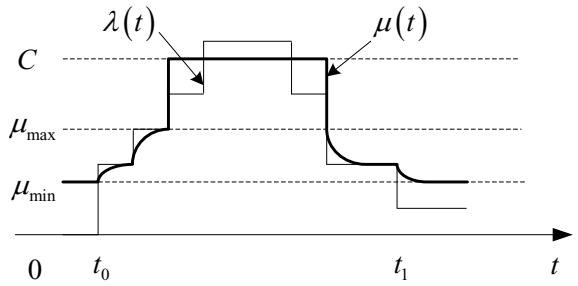

Fig. 3. Illustration about how $\mu(t)$ traces $\lambda(t)$

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 growing up to μ_{max}, $q(t)$ increases towards Q_{max} and $\mu(t)$ increases to follow $\lambda(t)$. When $\mu_{max} < \lambda(t) \leq C$, $q(t)$ is close to but slightly larger than Q_{max} and $\mu(t)$ is set to C.

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

When $\lambda(t)$ decreases, a similar but reversed process follows. $\mu(t)$ will jump back from C to μ_{max} as soon as $q(t) = Q_{max}$. If $\lambda(t)$ stays around some rate in between μ_{min} and μ_{max}, $\mu(t)$ will chase $\lambda(t)$ to that rate after a certain period of time. Whenever $\lambda(t)$ is further smaller than μ_{min}, $\mu(t)$ will stay at μ_{min} but the actual output rate is still $\lambda(t)$.

Figure 3 shows how $\mu(t)$ reacts to $\lambda(t)$'s changes. The vertical lines of $\lambda(t)$ in the figure ideally represent the rapid changes in $\lambda(t)$ and facilitate our understanding to the QLBP mechanism.

It can be seen that $\mu(t)$ changes automatically when $\lambda(t)$ changes. A QLBP system uses $q(t)$ as a congestion signal. If $\lambda(t)$ is very small (i.e., $< \mu_{min}$), which represents a case in which the router operates at a very low load level, no pacing is necessary. At the opposite extreme end, a $\lambda(t)$ larger than μ_{max} results in a $q(t)$ higher than Q_{max}, disables the pacing feature and makes the delay queue work at the full line speed.

D. QLBP Implementation in NS2

For our simulation study, we developed an implementation of QLBP that realizes Equation (1). A QLBP implementation in practical routers in the Open Network Laboratory (ONL) test-bed [14] has been presented in [8]. However, to test the QLBP mechanism in a larger-scale network, we require an implementation for ns2.

The algorithm used in our work is described in detail as Algorithm 1. There are four functions: `handle_packet()`, `send_packet()`, `resume()` and `target()`. The `handle_packet()` function is triggered by a packet arrival event. The `send_packet()` function uses the `target()` function to deliver a packet to the link. After it delivers the packet to its associated link, the queue is blocked for a certain period of time which equals to the transfer time, that is, S_p/C, where S_p is the size of the delivered packet. The `resume()` function is invoked when a queue is awaken up by a timer expiration. The timer could be set by either the queue itself or its downstream link that receives the packet delivered by the queue.

In our ns2 implementation, we use t_{next} to control when a packet at the head of the delay queue is allowed to send. Variable t_{last} is used to keep track of the last packet's sending time. The difference of $t_{next} - t_{last}$ is the delay we intend to control to implement the pacing effect. A longer difference means a lower output rate of the rate controller.

There are several important observations we make about the QLBP algorithm:

- The delay (i.e., $t_{next} - t_{last}$) is updated every time the queue length changes. Thus, the pacing delay always considers the most recent state of the delay queue. Also the complexity of updating the delay is $O(1)$. The calculation of delay can be executed based on specific hardware.
- Whenever a packet arrives at the delay queue, it will be forwarded immediately if the queue is not blocked and $now() \geq t_{next}$. This behavior ensures that an *the first packet arriving after a timer expires (at t_{next}) does not get delayed, which is critical to implement the adaptive pacing delay*. By "first", we mean such a packet that the queue is empty and non-blocked when it arrives.

IV. Delay Bound and Effectiveness Analysis

In this section we show that the pacing delay depends on the incoming traffic rate and is upper bounded by a constant that is determined by the parameters of the QLBP system. Then we give an analytical model to show the effectiveness of QLBP on reducing the variance of the instantaneous traffic rate in the context of a fluid model.

A. Delay Guarantee

First we 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 queueing delay. As the delay queue itself is the packet-storing queue, a packet might experience either queueing 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 μ_{max}, μ_{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 $\frac{Q_{max}}{\mu_{max}}$.

Algorithm 1 QLBP Algorithm

```
1: q ← empty_queue()
2: t_last ← 0
3:
4: function handle_packet(p)
5:    enqueue(q,p)
6:    if isblocked(q) then
7:       t_next ← t_last + S_p/((μ_max − μ_min)/Q_max · length(q) + μ_min)
8:       if q.timer.status() == PENDING then
9:          if now() ≥ t_next then
10:            q.timer.reschedule(now(), resume())
11:         else
12:            q.timer.reschedule(t_next, resume())
13:         end if
14:      end if
15:   else
16:      if now() ≥ t_next then
17:         if q.timer.status() == PENDING then
18:            q.timer.cancel()
19:         end if
20:         send_packet()
21:         block q
22:      else
23:         q.timer.schedule(t_next, resume())
24:      end if
25:   end if
26: end function
27:
28: function send_packet()
29:   p ← dequeue(q)
30:   t_last ← now()
31:   t_next ← t_last + S_p/((μ_max − μ_min)/Q_max · length(q) + μ_min)
32:   target(q,p)
33: end function
34:
35: function resume()
36:   if now() ≥ t_next then
37:      if q.timer.status() == PENDING then
38:         q.timer.cancel()
39:      end if
40:      if length(q) > 0 then
41:         send_packet()
42:      else
43:         unblock q
44:      end if
45:   else
46:      q.timer.reschedule(t_next, resume())
47:   end if
48: end function
49:
50: function target(q,p)
51:   target processes packet p
52:   target.timer.schedule(now(), q.resume())
53: end function
```

The proof is provided on-line in [9].

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 μ_{max} is set to 600Mbps. In Theorem 1 we focus only on pacing delay in the steady state. In practise, the incoming traffic rate changes over time. In this case a more complicated analysis is required.

B. Smoothness Effect Analysis

In this sub-section 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 smooths 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., $\mu(0) = \mu_{max}$). Then we solve for $\mu(t)$ as follows,

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

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 4. 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 smooths traffic in term of reducing auto-covariance of network traffic rate.

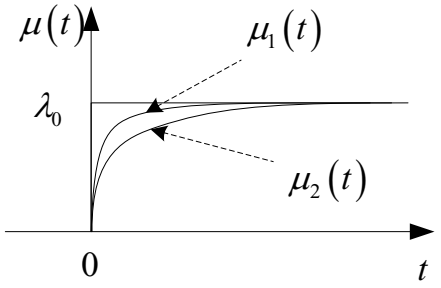

Fig. 4. How $\mu(t)$ responds to changes in $\lambda(t)$

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 [19], [30] and to show the impact of the auto-covariance of network traffic on buffer size [6], [22], [44], [21]. Also Willinger et al. [41], [42] 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 with peak rate h, ON and OFF Poisson counters N_1 and N_2 with arrival rate r_1 and r_2. Thus $\lambda(t)$ is given by a Poisson Counter Driven Stochastic Differential Equation (PCSDE) [6]

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

where $dx(t) = (1 - x(t))dN_1(t) - x(t)dN_2(t)$.

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

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

where α is given by (3).

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

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

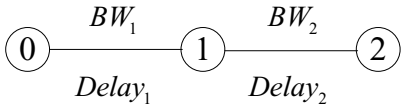

Fig. 5. A three node topology

TABLE II
PACING DELAY V.S. INPUT RATE

CBR Rate (Mbps)	Pacing Delay (ms)
1	0
2	0
4	4
6	5.33
8	6
10	6.4

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 on-line in [9].

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 [6], [22]. 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.

V. PERFORMANCE EVALUATION

In this section we conduct several sets of experiments in ns2 to: (1) validate the adaptive pacing delay introduced by QLBP, (2) quantitatively evaluate its effectiveness on reducing burstiness of traffic in terms of the variance of the instantaneous traffic rate and (3) compare its performance with TCP pacing in improving link utilization.

A. Adaptive Pacing Delay

In this experiment, we send a CBR traffic through a QLBP pacer and examine the pacing delay. Figure 5 shows the topology. A CBR traffic 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. $BW_1 = BW_2 = 10$Mbps, and $Delay_1 = Delay_2 = 10$ms. $\mu_{max} = 10$Mbps, $\mu_{min} = 2$Mbps and $Q_{max} = 10$pkts. UPD packet size is 1000 Bytes.

Table II shows different pacing delays under different CBR rates. When the CBR rate is smaller than or equal to μ_{min}, no pacing delay is introduced. As the rate increases, the pacing delay grows. The delay bound in this case is 8ms (10 pkts * 8000 bits per packet / 10Mbps).

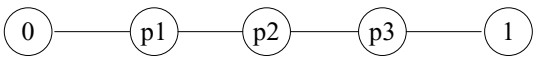

Fig. 6. A tandem queue topology

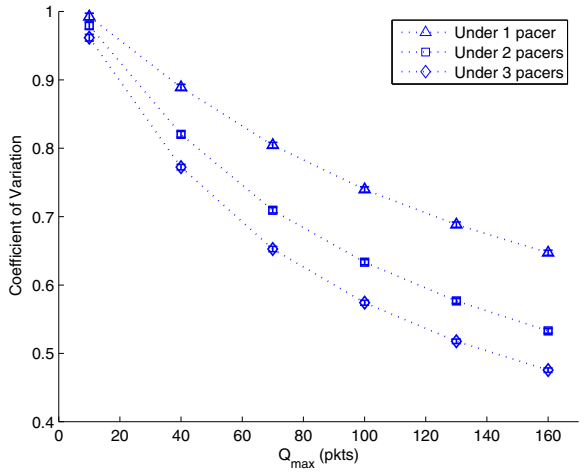

Fig. 7. Pacing effect of QLBP on Markov ON-OFF modeled process

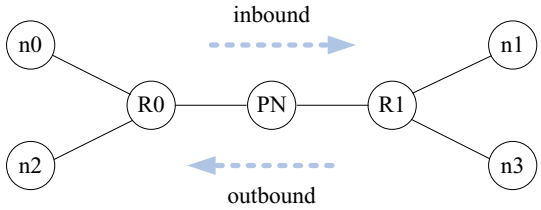

Fig. 8. A Tmix topology

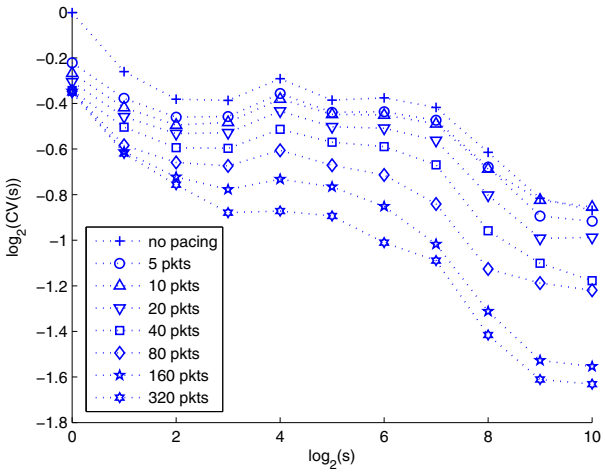

Fig. 9. Pacing effect of QLBP on self-similar Internet traffic

B. Pacing Effectiveness

We are interested in how QLBP affects traffic burstiness. The metric in this sub-section is the coefficient of variation of the traffic rate, which is used in [36] 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 a ns2-integrated traffic generator, Tmix [40] 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 [40].

1) QLBP on Markov ON-OFF Modeled Process: Figure 6 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 = 2Mbps. The average busy and idle periods are 100ms and 200ms, respectively. μ_{max} = 10Mbps and μ_{min} = 10Kbps. UPD 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 X_i represents the amount of bytes arriving at node 1 during the i-th interval.

Figure 7 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 IV-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 ns2 to replicate a piece of Internet trace file that has been show to be self-similar with Hurst parameter $H = 0.95$ [40].

Figure 8 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 ns2 manual (for details, see Chapter 43 in the ns2 manual [37]). The inbound and outbound connection vectors files (inbound.cvec and outbound.cvec) are provided by Professor M.C. Weigle [23]. We slightly modify the script to insert a pacer (i.e., 'PN' node as shown in Figure 8) in 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 [40] 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 320pkts.

Figure 9 plots the scale versus the coefficient of variation under different settings of Q_{max} on a log-log scale. X-axis is the time scale at which the amount of bytes arriving at node R1 is counted. The basic time resolution is 5ms. A scale s represents an interval of $2^s \times 5$ms.

It is observed that QLBP with a small Q_{max} (10 or 20 packets) can reduce the coefficient of variation of $\{X_i\}$ at $s = 0$ by 50% and its effect diminishes as s increases. Its impact disappears at $s = 10$. The larger Q_{max}, the wider the range of burstiness QLBP can affect. QLBP with a large Q_{max} (160 or 320 packets) results in a significant reduction at large time scales ($s = 9, 10$). This is because a large Q_{max} forces the rate-controller in QLBP less sensitive to burstiness in a wider range, and as a result, smoothing effect is more significant at high time scales.

C. 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 are used in [15] 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 10. Ten sender nodes, denoted by Si ($1 \leq i \leq 10$) are connected to each access router, denoted by Aj ($1 \leq j \leq 4$). Four access routers are connected to core router C0. Core router C0 is connected to core router C1 which connects ten receiver nodes, denoted by 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. Delays between sender nodes and access routers and between core router C1 to receiver nodes are uniformly distributed in $[1\ 10ms]$ to reduce the impact of TCP synchronization. The average RTT is about 100ms. We apply four QLBP pacers on four access routers, one pacer on each link Aj-C0 ($1 \leq j \leq 4$) with $\mu_{max} = 100$Mbps and $\mu_{min} = 1$Mbps. Q_{max}'s at four QLBP pacers are the same and vary from 10 to 160packets. 40 long-lived TCP flows are sent from 40 senders to 10 receivers. For each TCP flow, the maximum congestion window is set 32packets 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 time of a TCP session is uniformly distributed in $[0\ 100s]$. Each simulation run lasts 1000s and the steady state starts at 200s. The metric is the link utilization in the steady state.

Figure 11 shows the improvement of the link utilization. For a small buffer of 5 packets, QLBP with Q_{max} of 10 packets can improve link utilization by around 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.

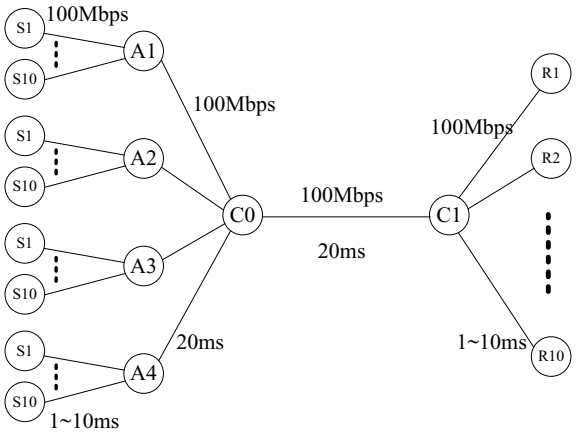

Fig. 10. A dumbbell topology

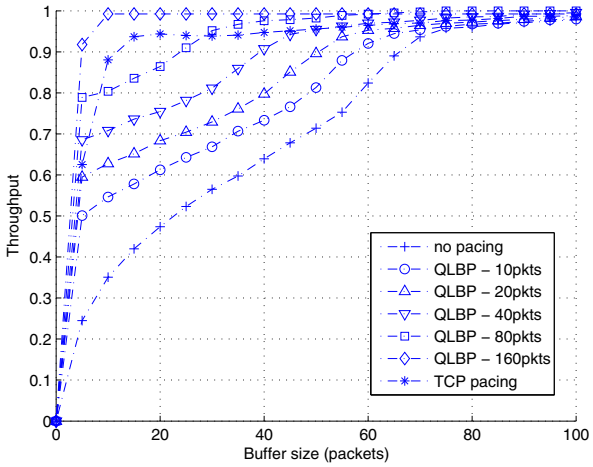

Fig. 11. Link utilization v.s. various buffer size

VI. Conclusions

The use of small buffers in the core of future networks raises the question of how to ensure that traffic bursts do not lead to degradation in network performance due to packet losses. We propose the use of an adaptive pacing system at the edge of these small-buffer networks. Our pacer is simple to implement due to its $O(1)$ complexity. Our analysis shows that the delay introduced by the proposed pacing is bounded. We also show that the throughput that can be achieved for TCP using our pacing algorithm exceeds that of end-system based TCP pacing. Even for a small buffer size, our system can achieve near 100% link utilization in these networks. We believe this pacing system provides an important solution to the burstiness problem and makes it practical to deploy small-buffer networks in the future Internet.

ACKNOWLEDGMENTS

This work is supported in part by National Science Foundation under grants CNS-0519922, CNS-0524323, CNS-0721790, EFRI-0735974, and DARPA under Contract W911NF-08-1-0233. The authors would like to thank Michele C. Weigle of the Old Dominion University for her great support and Pak-Ching Lee of the Chinese University of Hong Kong for his valuable suggestions.

REFERENCES

[1] M. Adler, S. Khanna, R. Rajaraman, and A. Rosen. Time-constrained scheduling of weighted packets on trees and meshes. *Algorithmica*, 36(2):123–152, 2003.

[2] M. Adler, A. L. Rosenberg, R. K. Sitaram, and W. Unger. Scheduling time-constrained communication in linear networks. *Theoretical Comp. Sc*, 35(6):559–623, 2002.

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

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

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

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

[7] R. Bush and D. Meyer. RFC 3439: Some internet architectural guidelines and philosophy. Dec. 2003.

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

[9] Y. Cai, B. Jiang, T. Wolf, and W. Gong. A practical on-line pacing scheme at edges of small buffer networks. Technical report.

[10] Y. Cai, Y. Liu, W. Gong, and T. Wolf. Impact of arrival burstiness on queue length: An infinitesimal perturbation analysis. In *Proc. of CDC '09*, Shanghai, China, Dec. 2009.

[11] Cisco line cards. http://www.cisco.com/en/US/products/hw/modules /ps2710/products data sheets list.html.

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

[13] M. Crovella and A. Bestavros. Self-similarity in World Wide Web traffic: Evidence and possible causes. *IEEE/ACM Transactions on Networking*, 5:835–846, Dec. 1997.

[14] 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*, 35(5):75–78, Oct. 2005.

[15] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden. Routers with very small buffers. In *Proc. of IEEE INFOCOM 06*, pages 1–11, Spain, Apr. 2006.

[16] A. Feldmann, A. Gilbert, P. Huang, and W. Willinger. Dynamics of ip traffic: A study of the role of variability and the impact of control. In *Proc. of ACM SIGCOMM '99*, pages 301–313, Aug. 1999.

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

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

[19] H. Heffes and D.Lucantoni. A markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance. In *IEEE Journal on selected areas in communications*, pages 856–868, Sept. 1986.

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

[21] C. V. Hollot, Y. Liu, V. Misra, and D. Towsley. Unresponsive Flows and AQM Performance. In *Proc. of IEEE INFOCOM*, Apr 2003.

[22] 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.

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

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

[25] V. Lal, J. A. Summers, M. L. Masanovic, L. A. Coldren, and D. J. Blumenthal. Novel compact inpbased monolithic widelytunable differential mach-zehnder interferometer wavelength converter for 40gbps operation. In *Indium Phosphide and Related Materials*, Scotland, 2005.

[26] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Transactions on Networking*, 2(1):1–15, Feb. 1994.

[27] M. L. Masanovic, V. Lal, J. S. Barton, E. J. Skogen, J. A. Summers, L. Rau, L. A. Coldren, and D. J. Blumenthal. Widely-tunable monolithically-integrated all-optical wavelength converters in inp. 23(3), 2005.

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

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

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

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

[32] H. Park, E. F. Burmeister, S. Bjorlin, and J. E. Bowers. 40-Gb/s optical buffer design and simulations. In *Numerical Simulation of Optoelectronic Devices (NUSOD)*, 2004.

[33] V. Paxson and S. Floyd. Wide-area traffic: the failure of poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, June 1995.

[34] F. Preparata. An optimal real-time algorithm for planar convex hull. *Communication of the ACM*, 22:402–405, 1979.

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

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

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

[38] R. S. Tucker. The role of optics and electronics in high-capacity routers. *Journal of Lightwave Technology*, 24(12):4655–4673, 2006.

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

[40] 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*, 36(3):67–76, 2006.

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

[42] 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 Transactions on Networking*, 5:71–86, 1997.

[43] D. Wischik and N. McKeown. Part I: Buffer sizes for core routers. *ACM SIGCOMM Comput Commun Rev*, pages 75–78, July 2005.

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

[45] 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. of ACM SIGCOMM 91*, pages 133–147, Zurich, Switzerland, Sept. 1991.