

Practical Packet Pacing in Small-Buffer Networks

Yan Cai, Y. Sinan Hanay and Tilman Wolf
Department of Electrical and Computer Engineering
University of Massachusetts
Amherst, MA 01003
Email: {ycai,hanay,wolf}@ecs.umass.edu

Abstract—The demand for more bandwidth has led to proposals for an all-optical network core. Due to inherent constraints of optical technology, only routers with small packet buffers are feasible to be implemented. In order to ensure efficient operation of such small-buffer networks, it is necessary to ensure that traffic is less bursty than in conventional networks. We propose a novel packet pacing mechanism that can smooth traffic bursts. Our theoretical analysis shows that our pacing scheme can guarantee that queue length of routers is BIBO stable. Experimental results from our prototype implementation show the effectiveness of our pacer on in terms of reduced network congestion and improving network throughput.

Index Terms—Optical networks, traffic shaping, congestion control.

I. INTRODUCTION

Computer networks have become essential infrastructure for inter-personal communication, business, entertainment, and education. As the wide-spread use of the Internet grows, so does the demand for raw bandwidth available to every user. To provide orders of magnitude more network bandwidth, all-optical core networks have been considered as a promising technology. One major challenge in the design of optical routers is the lack of any efficient technique to buffer data that is transmitted as light. Therefore, optical systems are designed with very small packet buffers.

An important concern in the context of small-buffer networks is the performance of existing protocols. It has been shown that the TCP sliding window and congestion control mechanism can produce bursty traffic flows in case of high bandwidth multiplexed links [23]. Further, queuing theory shows that bursty traffic produces longer queues, and thus more packet loss in small buffers, resulting in lower throughput. Also, TCP performance can be significantly affected by small buffers when several flows share the network or the bandwidth-delay product is large [16].

To counter the bursty nature of TCP, there have been several proposals for smoothing TCP traffic by “pacing” or “spacing” packet transmissions more uniformly. Such pacing mechanisms have been proposed for deployment on transmitting end-systems and to ensure quality-of-service in traditional networks.

In this paper, we develop a novel pacing algorithm to reduce the burstiness of traffic in the network. The key aspect of our Queue Length Based Pacing (QLBP) algorithm is that it can be implemented easily in high-performance routers since it requires little processing and paces packets indiscriminately

(i.e., does not require per-flow queues). The main idea behind our QLBP algorithm is to determine the inter-packet delay based on the current queue length and the arrival rate of packets. Theoretical analysis shows that QLBP pacing makes the queue length of routers BIBO stable. It is possible to overlay a randomization of the packet delay to avoid synchronized packet drops when multiple streams of QLBP are aggregated. We demonstrate the effectiveness of QLBP by evaluating a prototype implementation in the Open Network Laboratory (ONL). Our results show that TCP connections traversing nodes with small buffers experience higher throughput when paced with QLBP compared to unpaced connections. The specific contributions of our work are:

- A novel pacing algorithm (QLBP) that can be easily implemented in high-performance routers and paces traffic indiscriminately.
- A stability analysis of the pacing algorithm that shows that the queue length of routers is BIBO stable.
- A study of a prototype implementation of QLBP on ONL that explore network performance in terms of traffic shaping and link utilization.

The remainder of the paper is organized as follows. Section II discusses related work. Section III discusses the design of QLBP pacing and analyzes the BIBO stability of router queues. Section IV presents experimental results to illustrate the effectiveness of QLBP. Section V concludes this paper.

II. RELATED WORK

Appenzeller et al. show that the buffer requirements on routers can be much lower than the conventionally assumed bandwidth-delay product [2]. Recent studies on the buffer sizing have shown that smaller buffers can be used with TCP [10] and Raina et al. [14], [15] show that small buffers provide greater network stability. However, Dhamdhare et al. argue that congested links and large numbers of TCP flows can lead to high loss rate and poor performance using the buffer sizing approach proposed by Appenzeller [7]. These conflicting results show that TCP performance in small-buffer networks is an important issue that needs to be studied.

Various traffic smoothing methods are proposed in [9], [17] based on smoothing at the edges using leaky buckets with regulators and bufferless multiplexing at the core. Traffic smoothing for VBR traffic is proposed in [18]. The smoothing algorithm achieves optimum smoothness in terms of minimum rate variability by employing piece-wise CBR transmitting

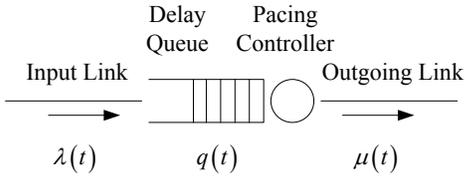


Fig. 1. Packet Pacing on a Router System.

behavior. However, these approaches depend on *a priori* knowledge of the data to be sent in order to schedule the transfer off-line. In our work, we propose an online method which does not require offline scheduling or beforehand knowledge of data rates.

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 [4], [5]. Pacing is implemented as the combination of these rate control protocols and TCP where the acknowledgments trigger the sending of data which are then sent at a fixed rate. Pacing was used in the TCP context to correct the compression of acknowledgements due to cross traffic [23], to avoid slow start [3], [12], after packet loss [11], or when an idle connection resumes [21]. Pacing at cell level in ATM networks has been shown to improve TCP throughput [8]. Aggarwal et al. studied the performance of TCP pacing over a wide variety of networks and found that pacing improves throughput, reducing delays in some cases but decreases performance generally [1].

Considering deployment of pacing across a network, it is more cost efficient to shape the traffic at the edge nodes of the network (as we propose to do with QLBP and has been proposed by Sivaraman et al. for optical edge nodes [19], [20]) rather than on end-systems. The effect of pacing with various versions of TCP and different routers has been studied, and it has been shown that the incentives of pacing are enough for applications to migrate to paced traffic model [22].

III. QUEUE LENGTH BASED PACING

We introduce the ideas behind Queue Length Based Pacing and a specific algorithm that implements QLBP efficiently on a network node. We also show the stability of QLBP in the context of router queues.

A. QLBP System and Notation

The general idea of packet pacing is illustrated in Figure 1. The figure shows a single input and output, but the concepts are applicable to routers with any number of ports. Packets arrive at a certain rate on the input link and are stored in the delay queue. If the queue is full, the arriving packet is dropped. The pacing controller determines when to send the next packet from the head of the queue.

To more formally describe the operation of QLBP, we introduce the following notation:

- $\lambda(t)$: the arrival rate of input traffic at time t .
- $\mu(t)$: the (paced) output rate of the pacing controller at time t .

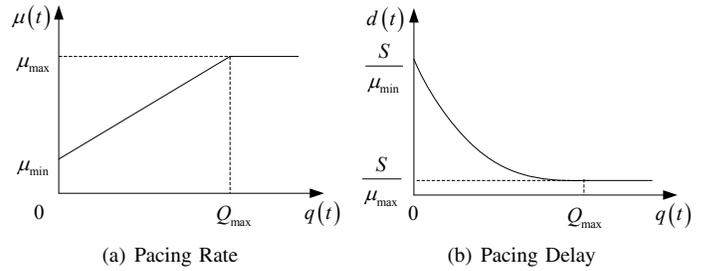


Fig. 2. Pacing Rate and Delay of QLBP

- $d(t)$: the pacing delay that the pacing controller introduces at time t .
- $q(t)$: the length of delay queue at time t .
- D_{max} (maximum pacing delay): the amount of time a packet is held by the pacing controller when the queue length is 0.
- Q_{max} (pacing cutoff queue length): the queue length at which no packet delays are introduced by the pacer.
- S : the size of a packet that is paced.
- S_{max} : the maximum packet size.
- μ_{min} (minimum pacing rate): the minimum rate of input traffic the pacing controller effects (i.e., when the input rate is smaller than this parameter, no pacing delay is introduced).
- μ_{max} (maximum pacing rate): the maximum rate at which the pacing controller transmits (ideally equal to the capacity of the outgoing link).

B. Adaptive Pacing Delay

One of the key aspects of any pacing algorithm is how the inter-packet pacing delay is determined. 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 an empty queue, the maximum delay is limited. This rule ensures that packets do not get delayed indefinitely.
- The first packet arriving at the pacer does not get delayed. This rule ensures that pacing is only activated when multiple packets (i.e., a burst) are observed.

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

$$\mu(t) = \begin{cases} \frac{\mu_{max} - \mu_{min}}{Q_{max}} q(t) + \mu_{min}, & q(t) < Q_{max} \\ \mu_{max}, & q(t) \geq Q_{max} \end{cases} \quad (1)$$

where

$$\mu_{min} = \frac{S_{max}}{D_{max}}. \quad (2)$$

The pacing delay is calculated by the equation below,

$$d(t) = \frac{S}{\mu(t)}. \quad (3)$$

The graphs of $\mu(t)$ and $d(t)$ with respect to $q(t)$ are shown in Figure 2.(a) and (b).

C. QLBP Algorithm

Based on the above equations to determine rate and delay, we need to design an algorithm that can efficiently implement this pacing mechanism in the data path of a router. This algorithm is shown as Algorithm 1.

The algorithm consists of two functions: `handle_packet` and `send_packets`. The `handle_packet` function is called by the operating system every time a packet arrives. The `send_packets` function uses `transmit_packet` to pass packets back to the operating system for transmission. To manage the pacing delay, the algorithm maintains a packet queue, q , and two global variables, t_{last} and t_{next} , which record the last time a packet was transmitted and the next time one could be transmitted, respectively.

Whenever a packet arrives, it is enqueued (line 5). Then the algorithm determines when the packet may be transmitted (line 6) while maintaining the delay define in Equation (3). If the next transmission time is in the future, a callback is scheduled with the operating system (line 8). Otherwise, `send_packets` is called directly (line 10).

When `send_packets` is called (either directly from `handle_packet` or through a callback), the function enters a while loop (lines 15–20). While there are packets in the queue and the next transmission time has passed, packets are dequeued (line 16), transmitted (line 17), and their transmission time recorded (line 18). Then the next transmission time is determined based on the previous transmission time and the queue length (line 19). Once the while loop terminates, the next call back is scheduled if there are more packets to be transmitted.

There are several important observations 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.
- The algorithm does not explicitly cancel scheduled callbacks that become unnecessary (e.g., when t_{next} is reduced due to arrival of another packet and the packet transmission is triggered by the `handle_packet` calling `send_packets`). The check for `(system_time() ≥ tnext)` in line 15 ensures that “old” callbacks do trigger premature transmissions.
- S_p refers to the size of the packet at the head of queue q , both in line 6 and line 19.
- The initialization of t_{last} to 0 (i.e., $t_{last} \ll \text{system_time}()$) in line 2 ensures that the first packet traversing the node does not get delayed when calculating t_{next} in line 6.

D. Stability Discussion

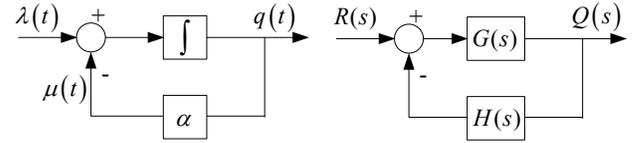
When introducing a pacing function into the data path of a router, an immediate concern is its effect on the network as a whole. In the following, we discuss the stability of QLBP in terms of router queue lengths. From a control system perspective, we prove that this pacing system is BIBO stable,

Algorithm 1 QLBP Algorithm

```

1:  $q \leftarrow \text{empty\_queue}()$ 
2:  $t_{last} \leftarrow 0$ 
3:
4: function handle_packet(p)
5:   enqueue(q,p)
6:    $t_{next} \leftarrow t_{last} + S_p / (\frac{\mu_{max} - \mu_{min}}{\text{max\_length}(q)} \cdot \text{length}(q) + \mu_{min})$ 
7:   if  $t_{next} > \text{system\_time}()$  then
8:     callback(tnext,send_packets())
9:   else
10:    send_packets()
11:   end if
12: end function
13:
14: function send_packets()
15:   while  $(\text{system\_time}() \geq t_{next}) \wedge (\text{length}(q) > 0)$ 
16:      $p \leftarrow \text{dequeue}(q)$ 
17:     transmit_packet(p)
18:      $t_{last} \leftarrow \text{system\_time}()$ 
19:      $t_{next} \leftarrow t_{last} + S_p / (\frac{\mu_{max} - \mu_{min}}{\text{max\_length}(q)} \cdot \text{length}(q) + \mu_{min})$ 
20:   end while
21:   if  $\text{length}(q) > 0$  then
22:     callback(tnext,send_packets())
23:   end if
24: end function

```



(a) Control System Corresponding (b) Transfer Function of Control System to Pacing shown in Figure 1

Fig. 3. Control System and Transfer Function

i.e., the queue length is bounded at all times. BIBO stability is defined as follows (refer to [13]):

Definition A system is bounded-input, bounded-output stable, if, for every bounded input, the output remains bounded for all time.

For a pacing system shown in Figure 1, the relationship among $q(t)$, $\lambda(t)$ and $\mu(t)$ can be expressed by the equation below,

$$q(t) = \int_0^t (\lambda(t) - \mu(t)) dt. \quad (4)$$

A control-system diagram is shown in Figure 3.(a), and the transfer functions of the components of the control system are shown in Figure 3.(b). From control theory, it is known that

$$G(s) = \frac{1}{s} \quad (5)$$

and

$$H(s) = \beta, \quad (6)$$

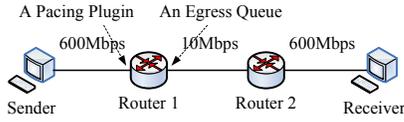


Fig. 4. Network Topology for Single TCP Flow.

where

$$\beta \approx \frac{\mu_{max} - \mu_{min}}{Q_{max}}, \quad (7)$$

given the assumption that $\mu_{max} \gg \mu_{min}$. In this case, the last term in Equation (1) is dropped.

Thus, we may calculate the transfer function of the pacing system as:

$$T(s) = \frac{G(s)}{1 + G(s)H(s)} = \frac{1}{s + \beta}. \quad (8)$$

From Equation (7), it is concluded that the root of the characteristic equation lies in the left half of the s -plane. The natural-response term of the system is $ke^{-\beta t}$. This proves that the pacing system is BIBO stable.

Given the formal description of QLBP, a practical algorithm for its implementation, and an assurance that it does not cause instability, we now turn to experimental evaluation.

IV. EXPERIMENTAL RESULTS

To obtain an experimental validation of the effectiveness of QLBP, we have implemented this pacing mechanism on the Open Network Laboratory (ONL) [6]. We evaluate not only how well bursts of traffic are smoothed by QLBP, but also how much throughput improvement on TCP flows can be observed.

A. Open Network Laboratory Setup

The Open Network Laboratory is a testbed facility maintained by researchers at Washington University in St. Louis. ONL users can set up different network topologies and extend the data path functionality of routers by adding so-called “plugins” to the packet forwarding process. We have developed a QLBP plugin that performs the QLBP algorithm as described in Section III.

In the ONL implementation of QLBP, we need to consider the practical restrictions imposed by limited timer granularity. The kernel of the router’s operating system only supports integer operation and a timer granularity of 1 millisecond. Thus, we introduce a scale factor α to adjust the pacing delay curve. In particular, the pacing delay is given by the following equation:

$$d(t) = \alpha \frac{S}{\mu(t)}, \quad (9)$$

where $\alpha \geq 1$. In contrast to Equations (3) and (9), Equation (9) yields longer pacing delays. Theoretically it affects μ_{min} and μ_{max} as well. However, in the context of more granular timers, the adjusted μ_{max} is still equivalent to its intended setting for a large range of α . If the queue length exceeds Q_{max} , then the pacing delay is always 0.

We use the following settings for the pacing plugin in our experiments: $\mu_{max}=200\text{Mbps}$, $S_{max}=1518\text{KB}$,

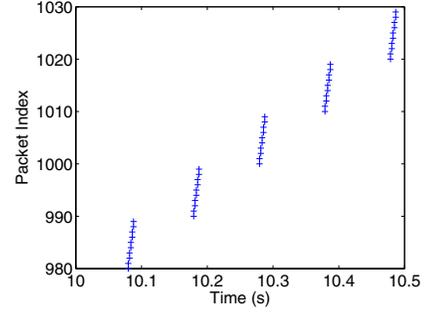


Fig. 5. Arrival Process of TCP Packets without Pacing.

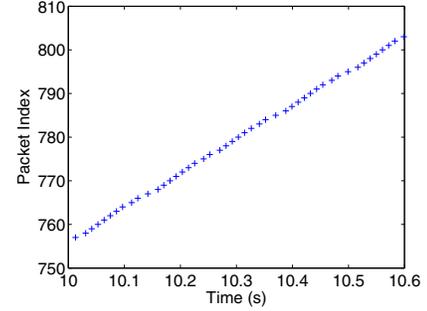


Fig. 6. Arrival Process of TCP Packets with QLBP Pacing.

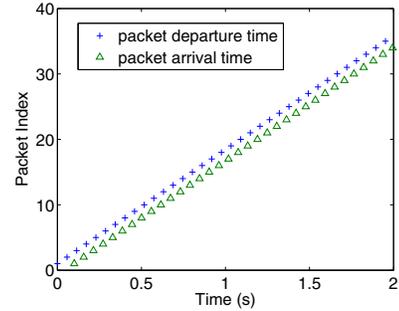


Fig. 7. Arrival and Departure Time of 200Kbps CBR Traffic.

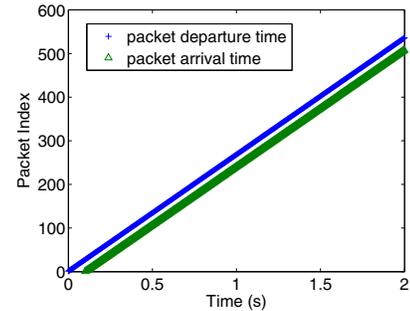


Fig. 8. Arrival and Departure Time of 3Mbps CBR Traffic.

$Q_{max}=100\text{pkts}$ and $S \approx 1500\text{KB}$. The round-trip time (RTT) from sender and receiver is always 100ms. To create a RTT of 100ms, two 50ms `pDelay` plugins are installed at two egress ports of the router attached to the receiver. D_{max} and α vary in different groups as necessary.

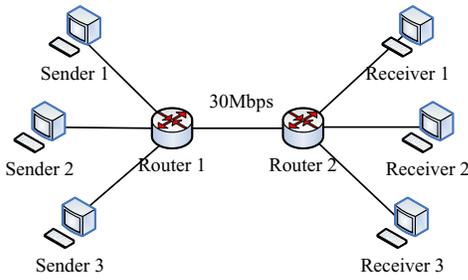


Fig. 9. Network Topology for Multiple TCP Flows.

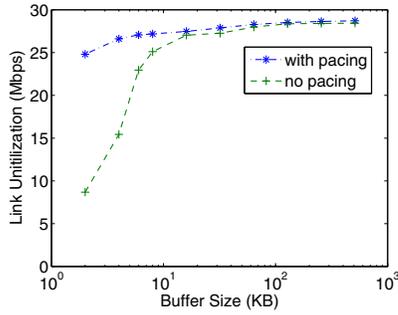


Fig. 10. Bottleneck Link Utilization for 20 TCP Flows and Varying Buffer Sizes.

B. Impact of QLBP on Single TCP and UDP Flows

This group of experiments focuses on the impact of QLBP pacing on a single TCP or UDP flow. The topology is shown in Figure 4. A pacing plugin is applied at the ingress port of router 1. The buffer size of the queue associated with the 10Mbps link is 25KB. The pacing plugin is with D_{max} of 10ms and α of 5.

When using a TCP connection, Figure 5 shows that without pacing the packets within one RTT window are sent as a burst, e.g., several packets depart at the very beginning of an RTT period. Whereas, Figure 6 indicates that the QLBP pacing plugin creates a packet departure sequence that is much smoother.

When sending UDP traffic with constant bit rate (CBR), we observe the packet arrival and departure times shown in Figures 7 and 8. Figure 7 uses CBR traffic with a lower data rate of 200Kbps and Figure 8 uses a higher data rate of 3Mbps. The topology is the same as for the TCP case (shown Figure 4) and the pacing plugin is configured with $D_{max}=10$ ms and $\alpha=5$. These figures illustrate that QLBP pacing has no impact on CBR traffic of any data rate in steady state.

C. Impact of QLBP on Multiple TCP Flows

It has been shown that the QLBP pacing scheme helps reduce the burstiness of a single TCP flows. As a result, it improves network performance in the case of multiple TCP flows competing for a bottleneck link with a small buffer. Figure 9 shows the topology used in this group of experiments. The bottleneck link between Router 1 and Router 2 has a bandwidth of 30Mbps. The rest of the links in this topology are 600Mbps. There are 7 TCP sessions between each pair

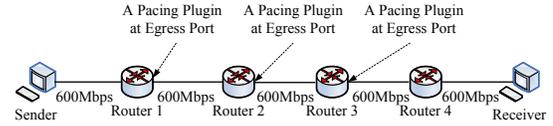


Fig. 11. Network Topology with Multiple QLBP Pacers.

of sender i and receiver i , where $i \in \{1, 2, 3\}$ (for a total of 21 TCP sessions). Each sender is connected to a router in which a QLBP pacing plugin is installed. For simplicity these intermediate routers between senders and router 1 are not displayed in Figure 9.

Figure 10 shows the relationship between link utilization and buffer size. The link utilization is measured at the receiver end in terms of goodput of all 21 TCP flows. The TCP connection duration is 100s. For every buffer size, results are average over multiple repetitions of the same experiments. From Figure 10, it can be seen that a 180% gain of link utilization is obtained at buffer size of 2KB.

D. Sequence of Multiple QLBP Pacers

When deploying QLBP in a practical network, pacing may occur at any node within the network. Such indiscriminate pacing (independent of location or packet flow) simplifies the deployment as pacers can be installed opportunistically and without central coordination. A key question is how traffic is affected by a sequence of multiple pacers. From the earlier results in Figures 6 and 7, we see that bursty traffic gets smoothed and in the limit (i.e., CBR traffic) remains unchanged.

To show the impact of multiple pacers in more detail, we use the topology shown in Figure 11. There are three pacing plugins in a row between the sender and the receiver. Every pacing plugin is configured with $D_{max}=2$ ms and $\alpha=1$. Starting with a single pacer, we enable an additional pacer in each experiment.

Figure 12 shows the arrival processes of a single TCP flow passing through 0, 1, 2 and 3 pacers respectively. To allow for a comparison of changes in the packet arrival process, we show all four trace files in one figure. The packet indexes and arrival time have been shifted accordingly. From Figure 12, it can be observed that more pacers lead to smaller gaps between two consecutive RTT periods. And thus traffic approaches the properties of CBR after only 3 pacing steps.

Another important observation from this experiment is that multiple pacers with low values of D_{max} are as effective as a single pacer with high values of D_{max} . The QLBP pacer in Figure 6 uses $D_{max}=10$ ms to obtain a departure process that is as smooth as three independent pacers in Figure 6 using $D_{max}=2$ ms. That indicates that a small amount of pacing on many nodes in the network is just as effective as high levels of pacing on selected nodes.

E. Discussion

As shown in our experimental results, the proposed QLBP mechanism has the following characteristics:

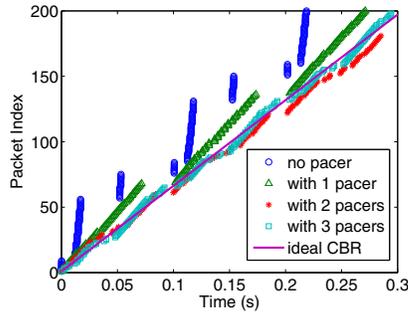


Fig. 12. Arrival Process with Multiple Pacers

- 1) The QLBP mechanism does not harm traffic that is smooth already (in the sense of making it burstier). This feature guarantees that a QLBP module can be deployed in carrier-level routers as a traffic-engineering system without creating negative effects on traffic shaping.
- 2) QLBP improves the link utilization of bottleneck link for small buffer sizes by reducing burstiness of network traffic. For links with larger delay-bandwidth products, this benefit becomes even more significant.
- 3) The effect of QLBP pacing can be aggregated over multiple hops. Thus, benefits of pacing are increased as QLBP becomes more widely deployed. This provides an incentive for network service providers to perform pacing in their systems.

These benefits together with the simplicity of the pacing algorithm make our pacing approach a simple network extension that can improve the operational performance of small-buffer networks.

V. CONCLUSIONS

In this paper we have presented a novel packet pacing system that can smooth traffic bursts. Reducing bursts in the network helps to operate small-buffer networks more efficiently and increases throughput for TCP connections. We have discussed our QLBP algorithm and illustrated its operation using an example. We have also shown a stability analysis and experimental evidence indicating the effectiveness of pacing. The use of multiple pacers that smooth traffic indiscriminately across the network can help reduce burstiness in TCP traffic and improve the utilization of bottleneck links.

We believe that packet pacing is an essential component at the network edge of future all-optical core networks. Our QLBP algorithm provides an important step towards making the operation of such network more efficient and practical.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. CNS-0721790. The Open Network Laboratory (ONL) that was used to obtain experimental results is supported by the National Science Foundation under Grant Nos. CNS-0551651, DRL-0632580, and DRL-0632586.

REFERENCES

- [1] AGGARWAL, A., SAVAGE, S., AND ANDERSON, T. Understanding the performance of TCP pacing. In *Proc. of IEEE INFOCOM 2000* (Tel Aviv, Israel, Mar. 2000), pp. 1157–1165.
- [2] APPENZELLER, G., KESLASSY, I., AND MCKEOWN, N. Sizing router buffers. In *Proc. of ACM SIGCOMM 04* (Portland, Oregon, Oct. 2004), pp. 281–292.
- [3] ARON, M., AND DRUSCHEL, P. TCP: Improving startup dynamics by adaptive timers and congestion control. Technical Report TR98-318, Rice University, 1998.
- [4] BONOMI, F., AND FENDICK, K. The rate based flow control framework for the available bit rate ATM service. *IEEE Network Magazine* (1998), 25–39.
- [5] CLARK, D. D., LAMBERT, M. M., AND ZHANG, L. NETBLT: A high throughput transport protocol. *ACM SIGCOMM Comp. Comm. Rev.* 17 (Aug. 1987), 353–359.
- [6] DEHART, J., KUHN, F., PARWATIKAR, J., TURNER, J., WISEMAN, C., AND WONG, K. The open network laboratory: a resource for networking research and education. *ACM SIGCOMM Computer Communication Review* 35, 5 (Oct. 2005), 75–78.
- [7] DHAMDHARE, A., JIANG, H., AND DOVROLIS, C. Buffer sizing for congested internet links. In *Proc. of IEEE INFOCOM 05* (Miami, FL, Mar. 2005), pp. 1072–1083.
- [8] EWY, B. J., EVANS, J. B., FROST, V. S., AND MINDEN, G. J. TCP/ATM experiences in the MAGIC testbed. In *HPDC '95: Proceedings of the 4th IEEE International Symposium on High Performance Distributed Computing* (Washington, DC, Aug. 1995), pp. 87–93.
- [9] GROSSGLAUSER, M., KESHAV, S., AND TSE, D. N. C. RCBP: a simple and efficient service for multiple time-scale traffic. *IEEE/ACM Transactions on Networking* 5, 6 (Dec. 1997), 741–755.
- [10] GU, Y., TOWSELY, D., HOLLLOT, C. V., AND ZHANG, H. Congestion control for small buffer high speed networks. In *Proc. of IEEE INFOCOM 07* (Anchorage, Alaska, May 2007), pp. 1037–1045.
- [11] HOE, J. Start-up dynamics of TCP's congestion control and avoidance schemes. Masterthesis, MIT, June 1995.
- [12] PADMANABHAN, V. N., AND KATZ, R. H. TCP Fast Start: A technique for speeding up web transfers. In *Proc. of IEEE GLOBECOMM* (Sydney, Australia, Nov. 1998).
- [13] PHILLIPS, C. L., AND HARBOR, R. D. *Basic FeedBack Control Systems*, second ed. Prentice Hall, Englewood Cliffs, New Jersey, 1991.
- [14] RAINA, G., TOWSELY, D., AND WISCHIK, D. Part II: Control theory for buffer sizing. *ACM SIGCOMM Comput Commun Rev* (July 2005), 79–82.
- [15] RAINA, G., AND WISCHIK, D. Buffer sizes for large multiplexers: TCP queuing theory and instability analysis. In *Proc. of EuroNGI* (Rome, Italy, Apr. 2005), pp. 173–180.
- [16] RAZDAN, A., NANDAN, A., WANG, R., SANADIDI, M. Y., AND GERLA, M. Enhancing TCP performance in networks with small buffers. In *Proc. of ICCCN 02* (Miami, FL, Oct. 2002), pp. 39–44.
- [17] REISSLEIN, M., ROSS, K. W., AND RAJAGOPAL, S. A framework for guaranteeing statistical QoS. *IEEE/ACM Transactions on Networking* 10, 1 (Feb. 2002), 27–42.
- [18] SALEHI, J. D., ZHANG, Z.-L., KUROSE, J. F., AND TOWSELY, D. F. Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing. *IEEE/ACM Transactions on Networking* 6, 4 (Aug. 1998), 397–410.
- [19] SIVARAMAN, V., MORELAND, D., AND OSTRY, D. Ingress traffic conditioning in slotted optical packet switched networks. In *ATNAC* (Sydney, Australia, Dec. 2004).
- [20] SIVARAMAN, V., MORELAND, D., AND OSTRY, D. A novel delay-bounded traffic conditioner for optical edge switches. In *Workshop on High performance Routers and switches (HPSR)* (Hong Kong, May 2005), pp. 182–186.
- [21] VISWESWARAIAH, V., AND HEIDERMAN, J. Improving restart of idle TCP connections. Technical Report TR97-661, University of Southern California, 1997.
- [22] WEI, D. X., CAO, P., AND LOW, S. H. TCP pacing revisited. unpublished, 2006.
- [23] ZHANG, L., SHENKER, S., AND CLARK, D. D. Observations on the dynamics of a congestion control algorithm: the effects of two way traffic. In *Proc. of ACM SIGCOMM 91* (Zurich, Switzerland, Sept. 1991), pp. 133–147.