Part III: Routers with very small buffers*

Mihaela Enachescu
Department of Computer Science
Stanford University
mihaela@cs.stanford.edu

Yashar Ganjali
Department of Electrical Engineering
Stanford University
yganjali@stanford.edu

Ashish Goel†
Dept of Management Sci. and Eng.
Stanford University
ashishg@stanford.edu

Nick McKeown
Department of Electrical Engineering
Stanford University
nickm@stanford.edu

Tim Roughgarden‡
Department of Computer Science
Stanford University
tim@cs.stanford.edu

ABSTRACT

Internet routers require buffers to hold packets during times of congestion. The buffers need to be fast, and so ideally they should be small enough to use fast memory technologies such as SRAM or all-optical buffering. Unfortunately, a widely used rule-of-thumb says we need a bandwidth-delay product of buffering at each router so as not to lose link utilization. This can be prohibitively large. In a recent paper, Appenzeller *et al.* challenged this rule-of-thumb and showed that for a backbone network, the buffer size can be divided by $\sqrt{N}$ without sacrificing throughput, where N is the number of flows sharing the bottleneck. In this paper, we explore how buffers in the backbone can be significantly reduced even more, to as little as a few dozen packets, if we are willing to sacrifice a small amount of link capacity. We argue that if the TCP sources are not overly bursty, then fewer than twenty packet buffers are sufficient for high throughput. Specifically, we argue that $O(\log W)$ buffers are sufficient, where W is the window size of each flow. We support our claim with analysis and a variety of simulations. The change we need to make to TCP is minimal—each sender just needs to pace packet injections from its window. Moreover, there is some evidence that such small buffers are sufficient even if we don't modify the TCP sources so long as the access network is much slower than the backbone, which is true today and likely to remain true in the future.

We conclude that buffers can be made small enough for all-optical routers with small integrated optical buffers.

*This work was supported under DARPA/MTO DOD-N award no. W911NF-04-0001/KK4118 (LASOR PROJECT) and the DARPA/MTO Buffer Sizing Grant no. W911NF-05-1-0224.

†Research also supported by an NSF career grant and an Alfred P. Sloan faculty fellowship.

‡Research also supported in part by ONR grant N00014-04-1-0725.

Categories and Subject Descriptors

C.2 [**Internetworking**]: Routers

General Terms

Design, Performance, Theory

Keywords

TCP, buffer size, congestion control, all-optical routers

1. MOTIVATION AND INTRODUCTION

Until quite recently, Internet routers were widely believed to need large buffers. Commercial routers today have huge packet buffers, often storing millions of packets, under the assumption that large buffers lead to good statistical multiplexing and hence efficient use of expensive long-haul links. A widely-used rule-of-thumb states that, because of the dynamics of TCP's congestion control mechanism, a router needs a bandwidth-delay product of buffering, $B = \overline{\text{RTT}} \times C$, in order to fully utilize bottleneck links [6, 15]. Here, C is the capacity of the bottleneck link, B is the size of the buffer in the bottleneck router, and $\overline{\text{RTT}}$ is the average round-trip propagation delay of a TCP flow through the bottleneck link. Recently, Appenzeller *et al.* proposed using the rule $B = \overline{\text{RTT}} \times C/\sqrt{N}$ instead, where N is the number of flows through the bottleneck link [3]. In a backbone network today, N is often in the thousands or the tens of thousands, and so the sizing rule $B = \overline{\text{RTT}} \times C/\sqrt{N}$ results in significantly fewer buffers.

In this paper, we explore if and how we could build a network with much smaller buffers still—perhaps with only a few dozen packet buffers in each router, and perhaps at the expense of 100% link utilization. While this is an interesting intellectual exercise in its own right, there would be practical consequences if it were possible.

First, it could facilitate the building of all-optical routers. With recent advances [8, 9, 12], it is now possible to perform all-optical switching, opening the door to routers with

huge capacity and lower power than electronic routers. Recent advances in technology make possible optical FCFS packet buffers that can hold a few dozen packets in an integrated opto-electronic chip [12]. Larger all-optical buffers remain infeasible, except with unwieldy spools of optical fiber (that can only implement delay lines, not true FCFS packet buffers). We are interested in exploring the feasibility of an operational all-optical network with just a few dozen optical packet buffers in each router.

Second, if big electronic routers required only a few dozen packet buffers, it could reduce their complexity, making them easier to build and easier to scale. A typical 10Gb/s router linecard today contains about one million packet buffers, using many external DRAM chips. The board space the DRAMs occupy, the pins they require, and the power they dissipate all limit the capacity of the router [3]. If a few dozen packet buffers suffice, then packet buffers could be incorporated inside the network processor (or ASIC) in a small on-chip SRAM; in fact, the buffers would only occupy a tiny portion of the chip. Not only would external memories be removed, but it would allow the use of fast on-chip SRAM, which scales in speed much faster than DRAM.

Our main result is that minor modifications to TCP would indeed allow us to reduce buffersizes to dozens of packets with the expense of slightly reduced link utilization. We obtain this result in a succession of steps. We will start by adopting two strong assumptions: (1) That we could modify the way packets are transmitted by TCP senders, and (2) That the network is over-provisioned. However, we will soon relax these assumptions.

We start by asking the following question: What if we kept the AIMD (Additive Increase Multiplicative Decrease) dynamics of TCP window control, but changed the TCP transmission scheme to "space out" packet transmissions from the TCP sender, thereby making packet arrivals less bursty? We assume that each TCP flow determines its window size using the standard TCP AIMD scheme. However, if the current window size at time t is W and the current round-trip estimate is RTT, then we assume the TCP sender sends according to a Poisson process of rate W/RTT at time t. This results in the same average rate as sending W packets per RTT. While this is a slightly unrealistic assumption (it can result in the window size being violated and so might alter TCP behavior in undesirable ways), this scenario yields important clues about the feasibility of very small buffers.

We are also going to assume that the network is over-provisioned—even if each flow is sending at its maximum window size, the network will not be congested.[1] Under these assumptions, we show that a buffer size of $O(\log W_{\max})$ packets is sufficient to obtain close to peak throughput, where $W_{\max}$ is the maximum window size in packets. Some elements of the proof are interesting in their own right.[2] The exact scenario is explained in Section 2 and the proof is presented in the extended version of this paper [5].

To get some feel for these numbers, consider the scenario where 1000 flows share a link of capacity 10Gbps. Assume that each flow has an RTT of 100ms, a maximum window size of 64KB, and a packet size of 1KB. The peak rate is roughly 5Gbps. The bandwidth-delay product rule-of-thumb suggests a buffer size of 125MB, or around 125,000 packets. The $\overline{\text{RTT}} \times C/\sqrt{N}$ rule suggests a buffer size of around 3950 packets. Our analysis suggests a buffer size of twelve packets plus some small additive constant, which brings the buffer size down to the realm where optical buffers can be built in the near future.

We then systematically remove the two assumptions we made above, using a combination of simulations and analysis. We first tackle the assumption that TCP sends packets in a locally Poisson fashion. Intuitively, sending packets at fixed (rather than random) intervals should give us the same benefit (or better) as sending packets at a Poisson rate. Accordingly, we study the more reasonable case where the TCP sending agent "paces" its packets deterministically over an entire RTT. Paced TCP has been studied before [2], and does not suffer from the problem of overshooting the window size. We perform an extensive simulation study of paced TCP with small buffers. When the network is over-provisioned, the performance of paced TCP closely mirrors our analytical bound of $O(\log W_{\max})$ for Poisson sources. This holds for a wide range of capacities and number of flows, and not just in the regime where one might expect the aggregate arrival process at the router to resemble a Poisson process [4]. These results are presented in Section 3. We provide additional intuition for this result in a the extended version of this paper [5]: if many paced flows are superimposed after a random jitter, then the packet drop probability is as small as with Poisson traffic.

The next assumption we attempt to remove is that of the network being over-provisioned. We consider a single bottleneck link, and assume that if each flow were to send at its maximum window size, then the link would be severely congested. In our simulations (presented in Section 4), Paced TCP results in high throughput (around 70-80%) with the relatively small buffers (10-20) predicted by the simple Poisson-transmissions analysis. While we have not been able to extend our formal analysis to the under-provisioned network case, some analytical intuition can also be obtained: if we assume that the TCP equation [11] holds and that the router queue follows the M/M/1/B dynamics, then buffers of size $O(\log W_{\max})$ suffice to utilize a constant fraction of the link capacity.

Our results are qualitatively different from the bandwidth-delay rule-of-thumb or from the results of Appenzeller *et al.* On the positive side, we have *completely removed* the dependence of the buffer size on the bandwidth-delay product. To understand the importance of this, consider the scaling where the RTT is held fixed at τ, but the maximum window size $W_{\max}$, the number of flows N, and the capacity C all go to ∞ such that $C = NW_{\max}/\tau$. This is a very reasonable scaling since τ is limited by the speed of light, whereas C, N, and $W_{\max}$ are all expected to keep growing as Internet traffic scales. Under this scaling, the sizing rule of Appenzeller *et al.* suggests that the buffer size should grow as $\sqrt{N}W_{\max}$, whereas our results suggest that the buffer size needs to grow only at the more benign rate of $\log W_{\max}$. On the negative side, unlike the result of Appenzeller *et al.* , our

[1]This assumption is less restrictive than it might appear. Current TCP implementations usually cap window sizes at 32 KB or 64 KB [10], and it is widely believed that there is no congestion in the core of the Internet. All optical networks, in particular, are likely to be significantly over-provisioned. Later we will relax this assumption, too.

[2]For example, we do not need to assume the TCP equation [11] or aggregate Poisson arrivals [13]—hence we do not rely on the simplifying assumptions about TCP dynamics and about a large number of flows that are required for these two results.

result is a tradeoff result—to obtain this large decrease in buffers, we have to sacrifice some fixed fraction (say around 25%) of link capacity. This might be a good tradeoff for an all-optical network routers where bandwidth is plentiful and buffers are scarce. But for electronic routers, this trade-off might not make sense.

We give evidence that our result is tight in the following sense.

1. Under the scaling described above, buffers must at least grow in proportion to $\log W_{\max}$ to obtain a constant factor link utilization. In Section 4.1, we present simulation evidence that constant sized buffers are not adequate as the maximum window size grows to infinity. We also perform a simple calculation which shows the necessity of the log-scaling assuming the TCP equation and M/M/1/B queueing.

2. When we run simulations without using Paced TCP, we can not obtain reasonable link utilizations with log-sized buffers, even in the over-provisioned case (Section 3).

While TCP pacing is arguably a small price to pay for drastic reduction in buffer sizes, it does require a change to end-hosts. Fortunately, we suspect this is not necessary, as two effects naturally provide some pacing in current networks. First, the access links are typically much slower than the core links, and so traffic entering the core from access links is automatically paced; we call this phenomenon "link-pacing". We present simulations showing that with link-pacing we only need very small buffers, because packets are spaced enough by the network. Second, the ACK-clocking scheme of TCP paces packets [2]. The full impact of these two phenomena deserves further study.

Other interesting directions for further study include the impact of packet sizes, the interaction of switch scheduling algorithms and small buffers, the impact of short flows, and the stability properties of TCP with our log-scaling rule. (Significant progress in analyzing stability was made recently by Raina and Wischik [13].)

Of course, significant additional work—including experimental verification, more detailed analysis, and larger simulation studies—is required before we undertake a drastic reduction in buffer sizes in the current Internet.

2. INTUITION: POISSON INJECTIONS AND AN OVER-PROVISIONED NETWORK

The intuition behind our approach is quite simple. Imagine for a moment that each flow is an independent Poisson process. This is clearly an unrealistic (and incorrect) assumption, but it serves to illustrate the intuition. Assume too that each router behaves like an M/M/1 queue. The drop-rate would be ρ^B, where ρ is the link utilization and B is the buffer size. At 75% load and with 20 packet buffers, the drop rate would be 0.3%, independent of the RTT, number of flows, and link-rate. This should be compared with a typical 10Gb/s router line-card today that maintains 1,000,000 packet buffers, and its buffer size is dictated by the RTT, number of flows and link-rate. In essence, the cost of not having Poisson arrivals is about five orders of magnitude more buffering! An interesting question is: How "Poisson-like" do the flows need to be in order to reap most of the benefit of very small buffers?

To answer our question, assume N long-lived TCP flows share a bottleneck link. Flow i has time-varying window size $W_i(t)$ and follows TCP's AIMD dynamics. In other words if the source receives an ACK at time t, it will increase the window size by $1/W_i(t)$, and if the flow detects a packet loss it will decrease the congestion window by a factor of two. In any time interval $(t, t']$ when the congestion window size is fixed, the source will send packets as a Poisson process at rate $W_i(t)/\text{RTT}$. Note that this is different from regular TCP, which typically sends packets as a burst at the start of the window.

We will assume that the window size is bounded by $W_{\max}$. Implementations today typically have a bound imposed by the operating system (Linux defaults to $W_{\max} = 64\text{KB}$), or the window size is limited by the speed of the access link. We'll make the simplifying assumption that the two-way propagation delay of each flow is RTT. Having a different propagation delay for each flow leads to the same results, but the analysis is more complicated. The capacity C of the shared link is assumed to be at least $(1/\rho) \cdot NW_{\max}/\text{RTT}$ where ρ is some constant less than 1. Hence, the network is over-provisioned by a factor of $1/\rho$, i.e. the peak throughput is ρC. The effective utilization, θ, is defined as the achieved throughput divided by ρC.

In this scenario, the following theorem holds:

Theorem 1. *To achieve an effective utilization of θ, a buffer of size*

$$B \geq \log_{1/\rho}\left(\frac{W_{\max}^2}{2(1-\theta)}\right) \tag{1}$$

suffices.

The proof of Theorem 1 is presented in the extended version of this paper [5]. As an example of the consequences of this simple model, if $W_{\max} = 64$ packets, $\rho = 0.5$, and we want an effective utilization of 90%, we need a buffer size of 15 packets *regardless of the link capacity.* In other words, the AIMD dynamics of TCP don't necessarily force us to use larger buffers, if the arrivals are well-behaved and non-bursty. So what happens if we make the model more realistic? In the next section we consider what happens if instead of injecting packets according to a Poisson process, each source uses Paced TCP in which packets are spread uniformly throughout the window.

3. PACED TCP, OVER-PROVISIONED NETWORK

It should come as no surprise that we can use very small buffers when arrivals are Poisson: arrivals to the router are benign and non-bursty. Queues tend to build up—and hence we need larger buffers—when large bursts arrive, such as when a TCP source sends all of its outstanding packets at the start of the congestion window. But we can prevent this from happening if we make the source spread the packets over the whole window. Intuitively, this modification should prevent bursts and hence remove the need for large buffers. We now show that this is indeed the case. Throughout this section, we assume that the bottleneck link is over-provisioned in the same sense as in the previous section. In the next section we remove this assumption.

First, suppose N flows, each with maximum window size $W_{\max}$, share a bottleneck link. Then the following is true,

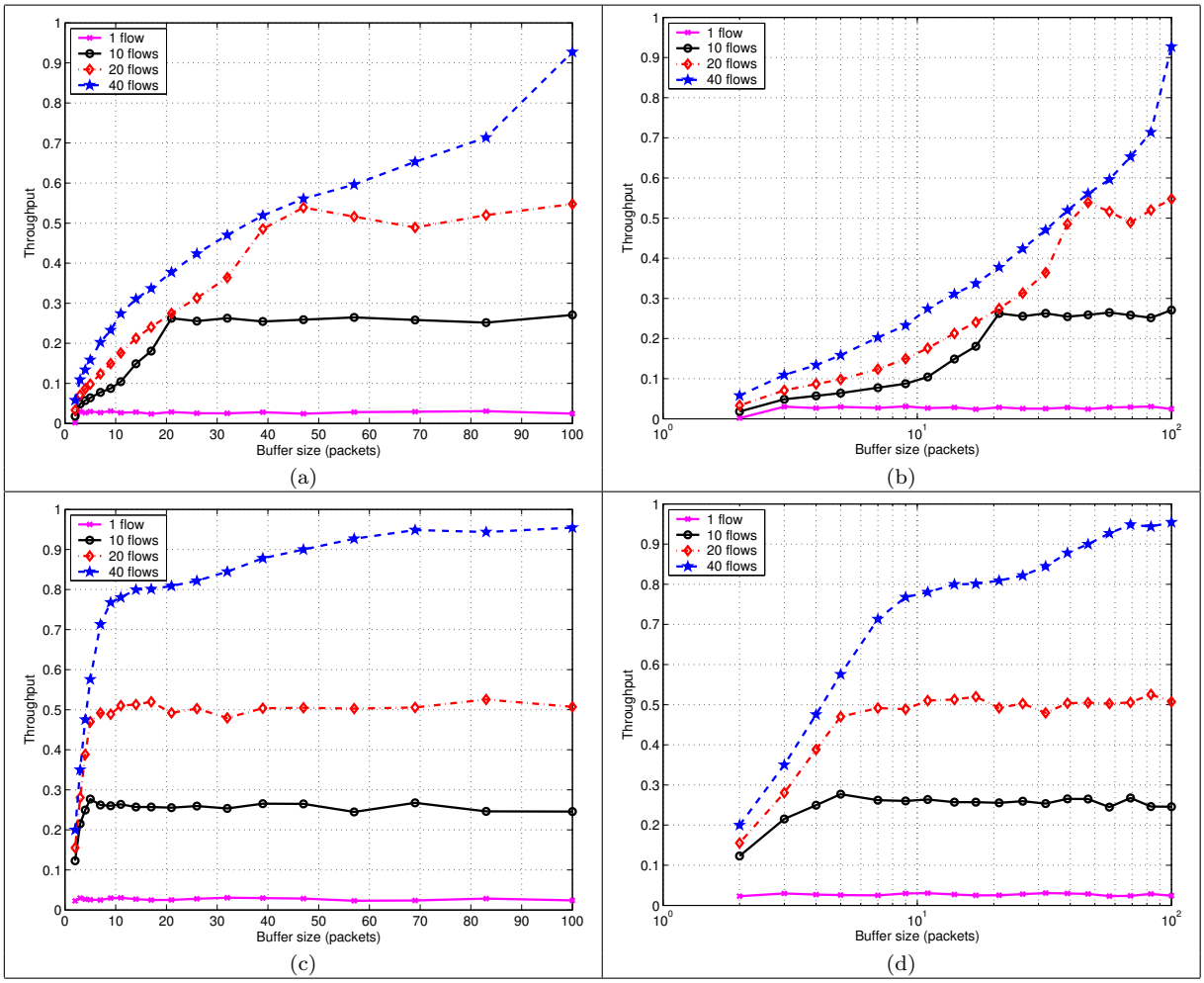

Figure 1: Bottleneck link utilization for different buffer sizes and number of flows. (a) unmodified TCP (b) unmodified TCP with logarithmic x-axis (c) paced TCP (d) paced TCP with logarithmic x-axis. The maximum possible offered load is 0.026 with one flow, 0.26 with 10 flows, 0.52 with 20 flows, and 1 with 40 flows.

under some mild assumptions (laid out in the extended version [5] along with the proof):

Theorem 2. *The packet loss probability during a single RTT is $O(1/W_{\max}^2)$, if (1) The buffer size is at least $c_B \log W_{\max}$ packets, where $c_B > 0$ is a sufficiently large constant; and (2) Each flow sends packets at a rate at most a $1/c_S \log W_{\max}$ fraction times that of the bottleneck link, where c_S is a sufficiently large constant.*

The buffer size requirement for Theorem 2 (Assumption (1)) is comparable to that in Theorem 1—a few dozen packets for present-day window sizes, independent of the link capacity, number of flows, and RTT. This requirement appears to be necessary to achieve constant throughput, even with Paced TCP (see Section 4.1). The packet loss probability in Theorem 2 is comparable to that for Poisson traffic with the same buffer size. To understand the second assumption of Theorem 2, note that if flows can send at the same rate as the bottleneck link, then there is no pacing of traffic whatsoever. In this case, our simulations indicate that constant throughput is not achievable with log-sized buffers. The natural goal is thus to obtain good throughput with small buffers provided flows are "sufficiently non-bursty". Theorem 2 quantifies this: as long as all flows send at a rate that is roughly a $\log W_{\max}$ factor slower than that of the bottleneck link, a Poisson-like throughput-buffer size tradeoff is achievable. This slowdown factor is only a few dozen for present-day window sizes, while access links are often orders of magnitude slower than backbone links. This huge difference in access link and backbone link speeds also seems likely to persist in the near future (especially with an all-optical backbone).

To explore the validity of Theorem 2, we performed simulations using the popular *ns*2 simulation tool [1]. We implemented Paced TCP and used various values of RTT, different number of flows, and buffer sizes. In Figure 1 we compare the number of buffers needed by TCP Reno with Paced TCP. We plot the throughput of the system as function of the buffer size used in the router, for various number of flows. The capacity of the bottleneck link is 100Mb/s, and the average RTT is 100ms. In this experiment, the maximum congestion window size is set to 32 packets, and the size of packets is 1,000 bytes. The simulation is run for 1,000 seconds, and we start recording the data after 200 seconds.

As we can see, with 40 unmodified TCP (Reno) flows, we need to buffer about 100 packets to achieve a throughput above 80%. However, in the same setting, Paced TCP achieves 80% throughput with just 10 packet buffers.

In Figure 1 we increased the system load as we increased the number of flows. It's also interesting to see what happens if we keep the system load constant (at 80% in this case) while increasing the number of flows. This is illustrated in Figure 2 (a), for flows with a maximum congestion window of 32 packets. As we increase the number of flows from one to more than a thousand, we also increase the bottleneck link capacity from 3.2Mb/s to 3.4Gb/s to keep the peak load at 80%. The buffer size is still set to 10 packets. The graph shows that regardless of the number of flows, throughput is improved by Paced TCP. The throughput of Paced TCP is around 70% (i.e., the effective utilization is more than 85%) while the throughput of the TCP Reno is around 20% (with an effective utilization of around 25%) regardless of the number of flows in the system.

It is important to note that this significant discrepancy between paced and regular TCP is observed only with small buffers. If we use the bandwidth-delay rule for sizing buffers, this discrepancy vanishes.

4. UNDER-PROVISIONED NETWORK, LIMITED ACCESS LINK CAPACITY

So far we have assumed that the network is over-provisioned and we do not have congestion on the link under study. Even though this is true for most links in the core of the Internet, it is also interesting to relax this assumption. We next study, via simulations, how congestion affects link utilization.

We repeat an experiment similar to that depicted in Figure 1. However, we increase the number of flows to up to 100. The average RTT is 100ms, and the maximum window size is 32 packets. Each packet is 1000 bytes, which means each flow can contribute a load of $32*1000*8/0.1 \simeq 2.5$Mb/s. The capacity of the core link is 100Mb/s, which mean if we have more than 40 flows, the core link will become congested.

Figure 2 (b) shows the throughput of the bottleneck link as a function of the buffer size for various number of flows. We can see that as we increase the number of flows from 20 to 40 (at which point the link starts to be saturated) the throughput goes from around 50% to about 80-90%. As we keep increasing the number of flows, to 100, and 200 flows, for some buffer sizes we see a degradation in throughput, but the throughput never goes below 80% even though the buffer size is 10 packets.

We have shown that Paced TCP can gain a very high throughput even with very small buffers. A very interesting observation is this: if the capacity of the access links is much smaller than the core link, packets entering the core will automatically have spacing between them even without modifying TCP. We did some experiments to verify if this spacing can result in the same throughput as Paced TCP. The core link bandwidth is set to 1Gb/s, and we vary the capacity of the access links. The maximum window size is very large (set to 10,000), and the buffer size is set to 10 packets, and the average RTT is set to 100ms. Figure 3 (a) shows that we still gain a high utilization even though we are not using Paced TCP. Here, the x-axis represents the capacity of the access links, and the y-axis represents the throughput. We can see that at the beginning the throughput increases (almost) linearly with access link capacity. For example, with 100 flows, this happens when the access link capacity is below 8-9Mb/s. Note that the normalized throughput is close to 100% in this case since the core link is not the bottleneck. As we increase the access link capacity, the throughput gradually decreases. This is because we lose the natural spacing between packets as the capacity of access links is increased.

4.1 The necessity of logarithmic scaling of buffer-sizes

We have not been able to extend our proof of theorem 1 to the case when the network is under-provisioned. However, the TCP equation [11] gives interesting insights if we assume that the router queue can be modeled as an M/M/1/B system [14]. Consider the scaling (described in the introduction) where the RTT is held fixed at τ, but the maximum window size $W_{\max}$, the number of flows N, and the capacity C all go to ∞. To capture the fact that the network is under-provisioned, we will assume that $C = \frac{NW_{\max}}{2\tau}$ i.e. the link

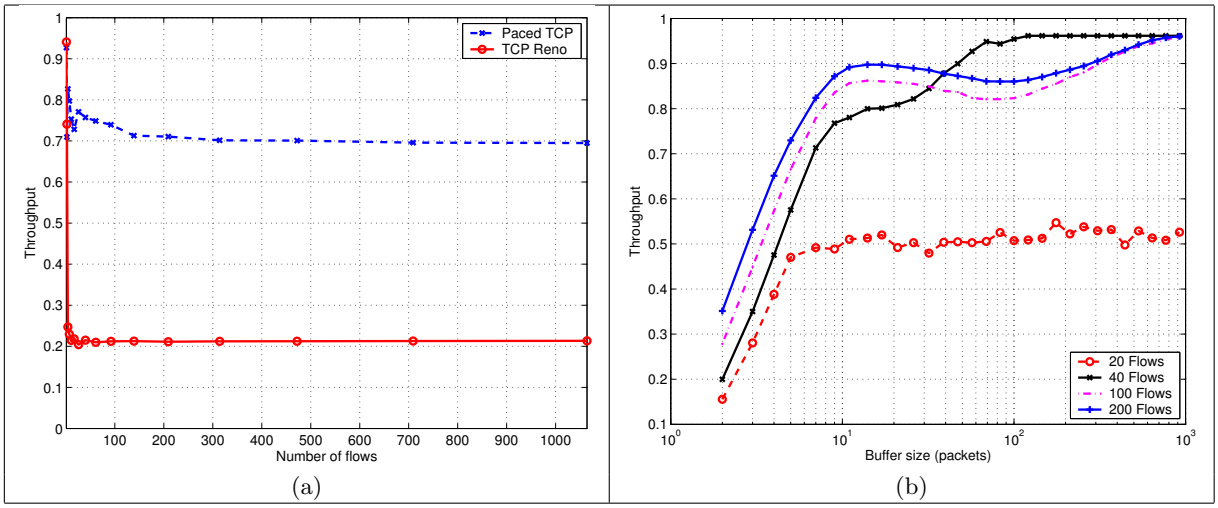

Figure 2: (a) Paced TCP vs. TCP Reno. (b) Bottleneck link utilization vs. the buffer size. With only 40 flows the core link becomes saturated, but even if we increase the number up to 200 flows, the throughput does not go below 80%.

can only support half the peak rate of each flow. Similarly, $C = \frac{2NW_{\max}}{\tau}$ represents the under-provisioned case.

Let p be the drop probability, and ρ the link utilization. Clearly, $\rho = RN/C$, where R is the average throughput of each flow. Then, the TCP equation states:

$$R = \frac{1}{\tau}\sqrt{\frac{3}{2p}} + o(1/\sqrt{p}) \simeq \frac{1}{\tau}\sqrt{\frac{3}{2p}}. \quad (2)$$

The M/M/1/B assumption yields [7]:

$$p = \rho^B \frac{1-\rho}{1-\rho^B}\frac{\rho}{1+\rho} \simeq \rho^{B+1} \quad (3)$$

Equations 2, 3 immediately yield the following:

1. Assume $C = \frac{NW_{\max}}{2\tau}$. For any constant $\alpha < 1$, there exists another constant β such that setting $B = \beta \log W_{\max}$ yields $\rho > \alpha$. In other words, logarithmic buffer-sizes suffice for obtaining constant link utilization even when the network is under-provisioned.

2. Assume $C = \frac{2NW_{\max}}{\tau}$. If $B = o(\log W_{\max})$ then $\rho = o(1)$. In other words, if the buffer size grows slower than $\log W_{\max}$ then the link utilization drops to 0 even in the over-provisioned case.

Obtaining formal proofs of the above statements remains an interesting open problem. Simulation evidence supports these claims, as can be seen in Figure 3 (b) which describes the throughput for a constant vs. a logarithmic sized buffer. For this simulation we are using Paced TCP, N is held fixed at 10, $W_{\max}$ varies from 10 to 1529, and C varies as follows: C is chosen initially so that the peak load is constant and a little over 50% and this choice determines the initial value for the ratio $\frac{C\overline{\text{RTT}}}{NW_{max}}$; then, since we fix N and $\overline{\text{RTT}}$, C varies proportionally to W_{max} keeping the above ratio constant as in our theoretical modeling. The buffer size is set to 5 packets when $W_{\max} = 10$. Thereafter, it increases in proportion with $\log W_{\max}$ for the log-sized-buffer case, and remains fixed at 5 for the constant buffer case. Here, initially the throughput is around 50% for both buffer sizing schemes. However, the throughput for the constant sized buffer drops significantly as $C, W_{\max}$ increase, while for the logarithmic sized buffer the throughput remains approximately the same, just as predicted by our theoretical model.

5. CONCLUSIONS

The main conclusion, of course, is that our results suggest packet buffers can be made much smaller; perhaps as small as 10-20 packets, if we are prepared to sacrifice some of the link capacity. It appears from simulation - though we have not been able to prove it - that the buffer size dictates directly how much link capacity is lost, however congested the network is. For example, a 40Gb/s link with 15 packet buffers could be considered to operate like a 30Gb/s link. This could, of course, be compensated by making the router run faster than the link-rate, and so not lose the link capacity at all. In a future network with abundant link capacity, this could be a very good tradeoff: Use tiny buffers so that we can process packets optically. In the past, it was reasonable to assume that packet buffers were cheap, and long-haul links were expensive and needed to be fully utilized. Today, fast, large packet buffers are relatively painful to design and deploy; whereas link capacity is plentiful and it is common for links to operate well below capacity. This is even more so in an all-optical network where packet buffers are extremely costly and capacity is abundant.

The buffer-size we propose depends on the maximum window size. Today, default settings in operating systems limit window size, but this limitation will probably go away over time. However, even if the maximum window size were to increase exponentially with time according to some form of "Moore's law", the buffer size would only need to increase linearly with time, which is a very benign scaling given recent technology trends.

Our results also assume that packets are sufficiently spaced out to avoid heavy bursts from one flow. Again, slow access links help make this happen. But if this is not true - for ex-

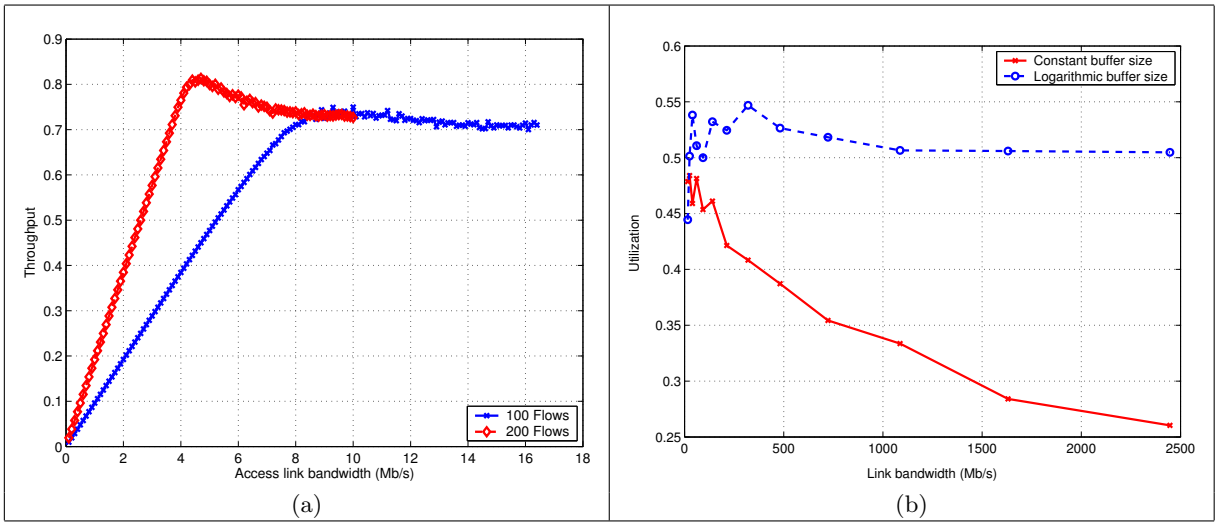

Figure 3: (a) Throughput as a function of access link capacity. (b) Constant vs. logarithmic buffers.

ample, when two supercomputers communicate - the TCP senders can be modified to use Paced TCP instead.

Our results lead to some other interesting observations. First, it seems that TCP dynamics have very little effect on buffer-sizing, and hence these results should apply to a very broad class of traffic. This is surprising, and counters the prevailing wisdom (and our own prior assumption) that buffers should be made large because of TCP's sawtooth behavior.

Acknowledgement

The authors would like to thank Guido Appenzeller, Neda Beheshti, Peter Glynn, and Damon Mosk-Aoyama for helpful discussions.

6. REFERENCES

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

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

[3] G. Appenzeller, I. Keslassy, and N. McKeown. Sizing router buffers. In *SIGCOMM '04*, pages 281–292, New York, NY, USA, 2004. ACM Press.

[4] J. Cao, W. Cleveland, D. Lin, and D. Sun. Internet traffic tends to poisson and independent as the load increases. Technical report, Bell Labs, 2001.

[5] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden. Routers with very small buffers. Technical Report TR05-HPNG-060606, Stanford University, High Performance Networking Group, June 2005.

[6] V. Jacobson. [e2e] re: Latest TCP measurements thoughts. Posting to the end-to-end mailing list, March 7, 1988.

[7] F. P. Kelly. *Chapter 3: Queueing Networks*, pages 57–94. Wiley, Chichester, 1979.

[8] V. Lal, J. A. Summers, M. L. Masanovic, L. A. Coldren, and D. J. Blumenthal. Novel compact inPbased monolithic widely-tunable differential Mach-Zehnder interferometer wavelength converter for 40Gbps operation. In *Indium Phosphide and Related Materials*, Scotland, 2005.

[9] 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. *Journal of Lightwave Tehcnology*, 23(3), 2005.

[10] Microsoft. TCP/IP and nbt configuration parameters for windows xp. Microsoft Knowledge Base Article - 314053, November 4, 2003.

[11] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling tcp throughput: a simple model and its empirical validation. In *SIGCOMM '98*, pages 303–314, New York, NY, USA, 1998. ACM Press.

[12] 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.

[13] G. Raina and D. Wischik. Buffer sizes for large multiplexers: Tcp queueing theory and instability analysis. http://www.cs.ucl.ac.uk/staff/D.Wischik/Talks/tcptheory.html.

[14] K. Ramanan and J. Cao. A poisson limit for buffer overflow probabilities. In *INFOCOM*, 2002.

[15] C. Villamizar and C. Song. High performance TCP in ANSNET. *ACM Computer Communications Review*, 24(5):45–60, 1994.