An Experimental Study of Router Buffer Sizing for Mixed TCP and Real-Time Traffic

Gajendra Hari Prakash Theagarajan, Sivakumar Ravichandran and Vijay Sivaraman*
School of Electrical Engineering and Telecommunications
University of New South Wales, Sydney, NSW 2052, Australia
{z3147984@student., z3147910@student., vijay@}unsw.edu.au
**Vijay Sivaraman is also affiliated with the ICT Centre, CSIRO, PO Box 76, Epping, NSW 1710, Australia.*

Abstract— **Recent research results on Internet router buffer sizing suggest that when TCP traffic is well-paced, Internet routers need as few as 20-30 packet buffers to realise near-maximum link throughputs, independent of link capacities and number of TCP flows. However, these studies have ignored non-TCP traffic, on the grounds that TCP traffic predominates in the Internet. In this paper we evaluate this assumption via practical experiments on a long-haul Australian network. Specifically, for different volumes of long range dependent real-time streaming video traffic in the network, we quantify end-to-end TCP throughput and real-time traffic loss as router buffer sizes at bottleneck and non-bottleneck links vary. Our results indicate that even in the presence of 5-15% bursty real-time traffic, TCP requires larger router buffers to achieve a given fraction of its saturation throughput. However, larger buffers lead to increased losses for real-time traffic. This suggests that TCP and non-TCP traffic can negatively impact each other, and their performance trade-offs need to be considered when sizing router buffers.**

Keywords: ***Internet router, buffer size, TCP throughput, real-time traffic loss.***

I. Introduction

Internet routers buffer packets during periods of congestion. An important question concerns the sizing of these buffers. An overflow from the buffer causes packet loss, directly impacting application performance, while an underflow causes idling and wastage of link bandwidth, degrading network throughput. Router manufacturers use a rule-of-thumb attributed to [1], in which the buffer size is determined by end-host TCP dynamics. Specifically, the rule-of-thumb requires a router to have enough buffers such that when the buffer overflows, causing TCP to react by reducing its transmission rate, there are enough packets stored to keep the output link of the router utilised till TCP increases its transmission rate back up, thus ensuring that link capacity is not wasted. Stated mathematically, the rule-of-thumb mandates a buffer size of $B = T \times C$ where T denotes the average round-trip time of a TCP flow through the router, and C the capacity of the link. For a typical T of 250 msec, a router with a $C = 40$ Gbps link would thus require 10 Gigabits of buffering.

The large buffering requirement mandated by the rule-of-thumb above poses serious concerns in the design of high-speed Internet routers. Electronic router buffers are built from commercial memory devices such as DRAM and SRAM, neither of which can individually support the combined size and speed requirements. SRAM chips allow fast access, but are small in size (of the order of 36 Mbits), necessitating tens or hundreds of chips. DRAM chips are larger, but have access times of 50 nsec, which is insufficient to cope with packets arriving every 8 nsec on a 40 Gbps link. Modern routers use combinational techniques [2] where multiple parallel DRAM chips provide the required aggregate memory bandwidth while an SRAM cache maintains the mapping of packets to their locations in DRAM. However, such complex configurations require a lot of board space and data pins, making routers large, expensive, and hot. The large buffer requirement also makes it infeasible to move to optical packet switching, where buffering is a very difficult operation. Spools of fibre are unweildy and unscaleable, while emerging integrated opto-electronic chips [3] offer no more than a few dozen packets of buffering. The large buffer size mandated by the rule-of-thumb poses a signifcant impediment to the design and deployment of high-speed electronic and optical switches needed by the next-generation Internet.

Researchers from Stanford first challenged the rule-of-thumb on buffer sizing in 2004. They showed that the buffer requirement can be reduced by a factor proportional to the square-root of the number of TCP flows sharing the bottleneck link, with negligible impact on link throughput [4]. This implies that an Internet router carrying $10,000$ TCP flows need only buffer $10,000$ packets instead of one million. While this is a dramatic improvement, the buffer size is still too large to allow a move to optics. Very recently, the Stanford group has shown theoretical and experimental evidence that as few as 20-30 packet buffers can provide acceptable link throughputs [5], [6], independent of link speeds and number of TCP flows sharing the link. Though the result is remarkable, it requires various assumptions, in particular that the traffic is exclusively TCP, and further that end-hosts pace their packet transmissions to have a certain minimum inter-packet gap.

Concerns regarding the implications of the above buffer sizing recommendation on network and application performance have already been raised [7]. In particular, the authors note that even though the Stanford model provides enough buffering to allow near-full utilisation of link bandwidth, end-to-end TCP throughput can be significantly lowered.

This was also noted in our own prior work [8], where our preliminary investigation quantified the impact of small buffers on TCP performance in the context of optical packet switched networks. Other concerns raised in [7] regarding the Stanford model include the choice of performance objective, the "saturability" of links, the estimation of the number of "long" TCP flows, and whether the traffic load is independent of network state or not.

In spite of the numerous concerns raised in prior work listed above, an important issue that has been ignored concerns the impact of non-TCP traffic on buffer sizing, particularly when the non-TCP traffic exhibits burstiness on multiple time-scales. Though long range dependent (LRD) models have been studied extensively by the queueing theory research community, we are not aware of any prior work that has considered the impact of bursty real-time traffic on TCP performance, particularly when network buffers are small, as in the Stanford model above. Both the Stanford model itself, and subsequent work [7], have ignored non-TCP traffic on the grounds that TCP accounts for more than 90% of Internet traffic.

In this paper, we evaluate this premise via experiments over a long-haul Australian network. Specifically, we test how the router buffer size that achieves a certain TCP performance is influenced by small volumes of real-time traffic. We send TCP traffic over links with a large round-trip time, and measure the impact of router buffer size at bottleneck as well as non-bottleneck links on end-to-end TCP throughput, in the presence of varying volumes of real-time streaming video traffic. Our results show that the bursty nature of real-time traffic partly nullifies the benefits of pacing TCP traffic at the network ingress, and increases the TCP buffering requirements at both bottleneck and non-bottleneck links. More surprisingly, larger buffers can severly impact loss performance for real-time traffic, suggesting that buffers sizing needs to consider the performance trade-offs when TCP and real-time traffic are mixed in the network.

The rest of this paper is organised as follows: in Section II we discuss our set-up, experiments and results for bottleneck links, while Section III studies performance at non-bottleneck links. We summarise our conclusions and point to directions for future work in Section IV.

II. Sizing Bottleneck Link Buffers

A. *Topology*

In the first part we study the impact of buffer size at bottleneck links. The topology used for this study is shown in figure 1, and broadly models a network with access, metro, and core links. Four PCs running FreeBSD version 6.0 are used to generate TCP traffic. The Ethernet interfaces on each PC operates at 10 Mbps, and can be thought to model TCP traffic from 10 home users, each with a 1 Mbps broadband connection. The traffic from the four PCs is aggregated in the metro using Cisco Catalyst 3750 switches, feeding into the core router over 100 Mbps metro links. Also feeding into the core router is a PC running Windows XP that generates streaming video traffic, representative of real-time traffic in the metro network originating from say an organisation or university with high-speed links, or from an aggregation of many home users running streaming applications. The PCs, switches, and router mentioned so far are housed in Sydney.

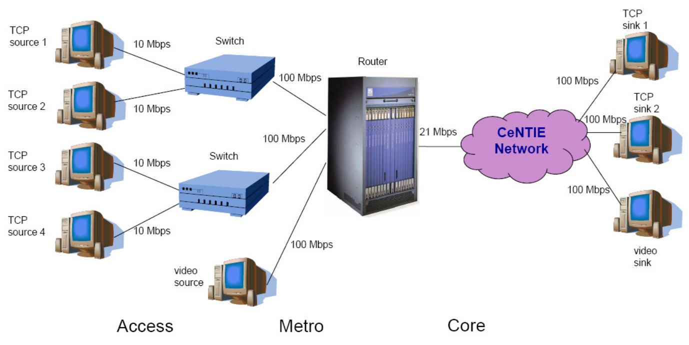

Fig. 1. Topology for bottleneck link experiments

The core router is a Cisco 7204VXR, with three input interfaces operating at 100 Mbps each and one output interface whose rate we control in software. The output link connects to the CeNTIE network, a trans-continental Australian research network [9] with MANs in Sydney, Canberra, Melbourne, and Perth. The TCP and real-time traffic we generate in Sydney are sinked at Perth, with two PCs running FreeBSD sinking the TCP traffic and one Windows XP machine sinking the video traffic.

The core link, namely the link between the router and the CeNTIE network, is made the bottleneck for the experiments in this section, and the effect of buffer size at the router interface on TCP performance is studied for different volumes of video traffic. The core link bandwidth was set at 21 Mbps, and the buffer size at the core link was varied using the technique of [4], [8], namely by employing egress link shaping and adapting the token bucket size. An unfortunate limitation of this approach on the Cisco 7204VXR is that the token bucket size cannot be any lower than 10msec worth of data; thus at the shaping rate of 21 Mbps used in our experiments, the minimum buffering is 210 Kbits, which for 1500 byte packets corresponds to approximately 18 packets. For this reason our plots in this section do not include points corresponding to less than 18 packets of buffering.

B. *TCP Traffic Generation*

The four PCs in Sydney running FreeBSD generate TCP traffic using the Iperf tool version 1.7.0. Each PC generates 10 TCP flows (Iperf was found to be unstable for a larger number of TCP threads). Since the Ethernet interfaces on each PC operates at 10 Mbps, each TCP flow on average gets 1 Mbps, representative of home users with broadband connections. Note that setting the access links to a speed lower than the bottleneck core link provides some measure of "link-pacing", namely successive packet from a TCP flow will be spaced when they reach the core link. The TCP packet sizes were set at 1500 bytes. The round-trip time (RTT) for

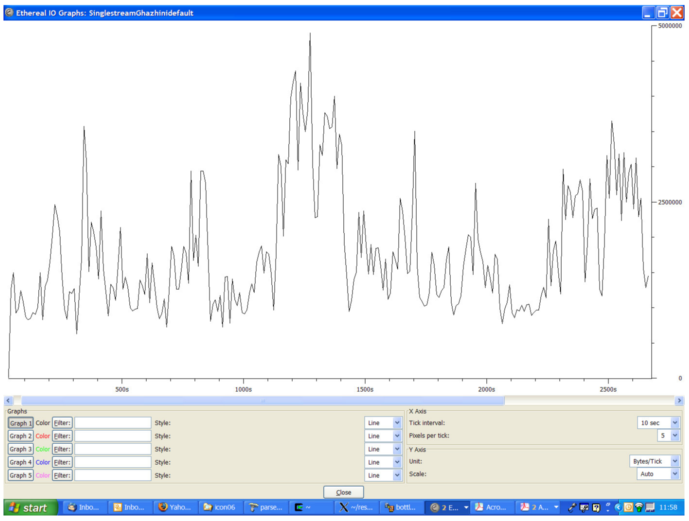

Fig. 2. Video traffic burstiness profile as captured by Ethereal.

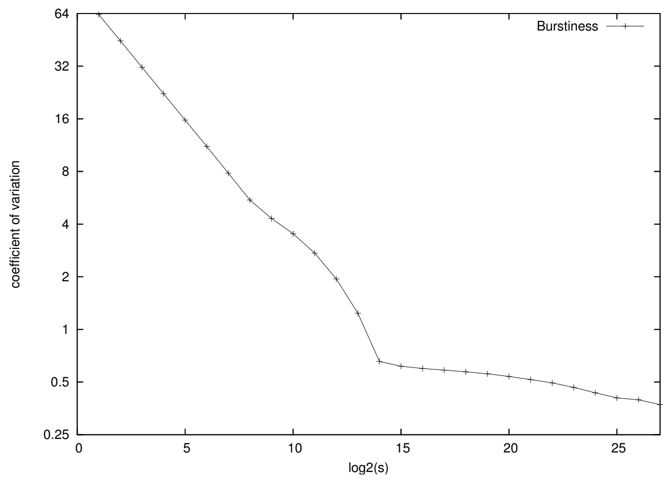

Fig. 3. Video traffic burstiness measured by coefficient of variation over various scaling window sizes.

the TCP traffic between Sydney and Perth was measured to be around 117msec, representative of real Internet delays. The TCP window sizes for all our experiments were set to be larger than the product of the RTT and the bottleneck link bandwidth, to ensure that TCP throughput is not restricted by the window size.

C. Video Traffic Generation

The real-time traffic in our experiments was generated using the VLC media player [10] version 0.8.4a running on a Windows XP PC. The Tamil movie "Ghazini" was streamed, and found to have a mean data rate of around 1.3 Mbps. The stream was captured using the Ethereal packet capture tool and the profile plotted. Figure 2 shows a screen capture of the data rate plotted over 10-second intervals, and shows that the video traffic exhibits substantial burstiness. We analyse the burstiness of the stream by computing the coefficient of variation $\beta(s)$ of the traffic volume measured over time intervals of length s. Log-log plots of $\beta(s)$ versus s are routinely used to indicate self-similarity of traffic traces, and the Hurst parameter H can be estimated by equating the slope of the curve to $-(1 - H)$. Figure 3 plots $\beta(s)$ versus s (in μsec) on log-log scale for the captured video stream, and shows a curve with two near-linear pieces. At time-scales above 2^{14} μsec or so (roughly 16 msec), the video trace exhibits self-similarity with a Hurst parameter of around 0.96, which is high but consistent with other studies of video traffic traces reported in the literature.

D. Experiments and Results

Three sets of experiments were run based on the configuration described above. First, only TCP traffic was generated and the aggregate throughput of the 40 TCP flows sharing the bottleneck core link operating at 21 Mbps was profiled as a function of the bottleneck buffer size (in packets). The throughput was recorded every 10 seconds over a 10 minute interval, and the top curve in figure 4 shows the mean aggregate TCP throughput, as also errorbars corresponding to the standard deviation. It is observed that the TCP throughput approaches near-saturation very quickly as the buffer size increases; achieving 98% of the saturation throughput (of 20.08 Mbps) requires no more than around 27 packets of buffering. The rapid rise to saturation is consistent with the behaviour of paced TCP observed in [6, Fig.1], and can be attributed to the natural pacing of TCP packets when they arrive at the core link due to lower access link speeds.

The middle curve in figure 4 corresponds to the second experiment in which a single video stream is now added in at the core router, and shares the output link (and buffers) with the TCP streams. Thus, at the bottleneck link, the video traffic constitutes around 6% of the total traffic in bytes. The TCP throughput in this case is again measured at 10 second intervals over 10 minutes, and exhibits standard deviation indicated by error-bars on the curve. The high variance is not unexpected, since the video traffic intensity fluctuates widely over time; however, in spite of the large variance, the mean TCP throughput plotted in the curve shows a smooth trend. The maximum achievable aggregate TCP throughput in this case is 19.32 Mbps, and to achieve 98% of this saturation throughput, around 48 packet buffers are required, which is much larger than the 27 needed earlier in the absence of the video stream. The shape of the curve also shows that the rise of aggregate TCP througput to saturation is not as fast as was the case with TCP traffic only, indicating that the presence of real-time traffic can partly negate the pacing of TCP packets, making TCP require larger buffers to obtain the same fraction of maximum throughput.

The bottom curve in figure 4 corresponds to the third experiment in which two video streams are added in at the core router, contributing to around 12% of total traffic at the bottleneck link. The aggregate TCP throughput is plotted in the curve, with error-bars indicating standard deviation. The maximum achievable aggregate TCP throughput is 18.76 Mbps, and achieving 98% of this saturation throughput

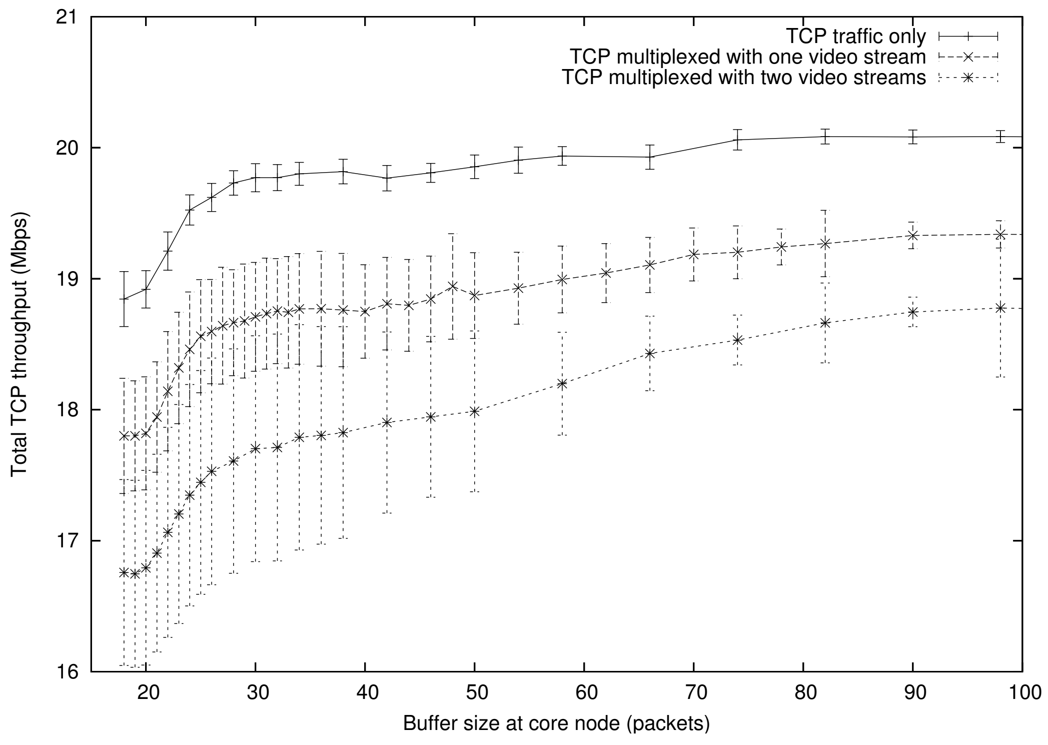

Fig. 4. TCP throughput vs. bottleneck buffer size when TCP is multiplexed with zero, one, and two video streams.

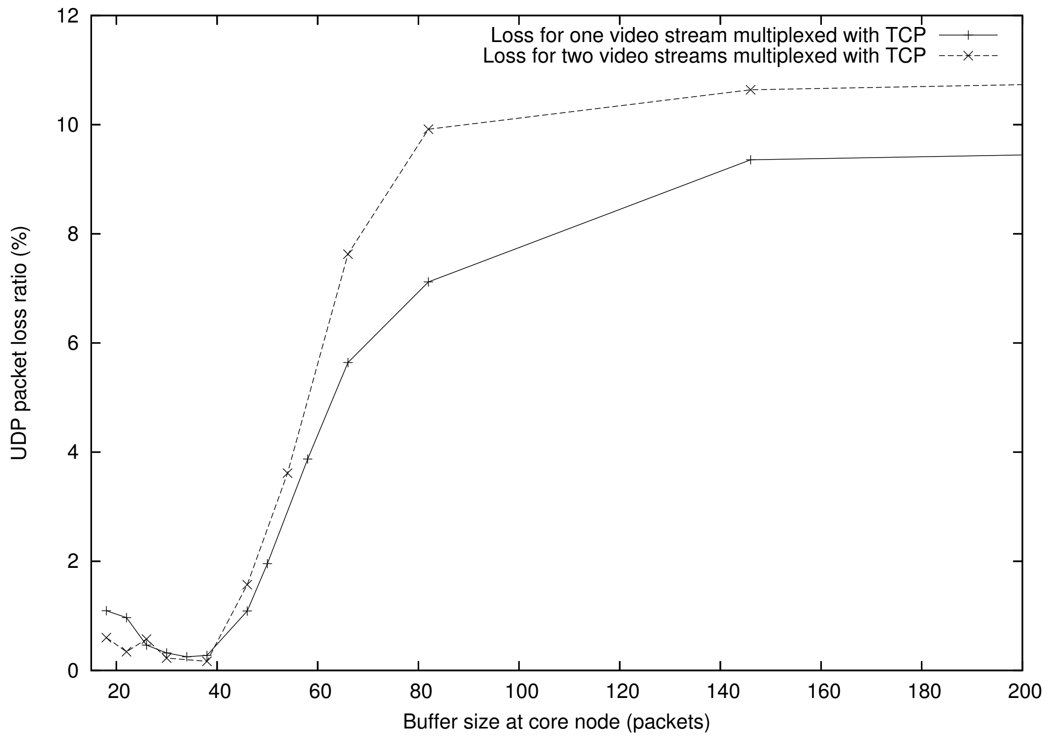

Fig. 5. UDP loss vs. bottleneck buffer size when TCP is multiplexed with one and two video streams.

requires approximately 62 packet buffers. It is evident that as the ratio of non-TCP traffic increases, the TCP throughput rises to its saturation value less rapidly as a function of bottleneck buffer size. This suggests that as bursty real-time traffic increases in the network, TCP traffic, though paced at the network acess links, requires larger buffers to achieve

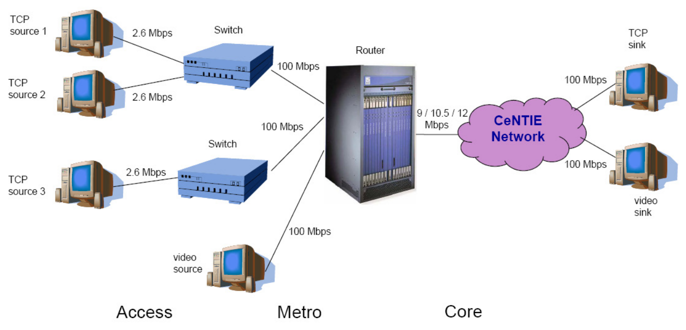

Fig. 6. Topology for non-bottleneck link experiments

close to saturation throughput.

While TCP benefits from larger buffers, let us see how the UDP (video) traffic is affected by buffer size. Figure 5 plots the fraction of video packets lost as a function of buffer size, for the cases when one and two video streams are multiplexed with TCP on the bottleneck link. For small buffer sizes, UDP losses reduce with increasing buffers, as expected. However, at buffer size of around 35 packets (corresponding to the end of the phase in which TCP's throughput rises rapidly), UDP losses reach their lowest value, and surprisingly, subsequent increase in buffer size increases UDP loss. This can be explained by the interaction of TCP dynamics with UDP traffic. As buffers get bigger, the round-trip-time (RTT) for the TCP flows gets larger, and TCP takes a longer time to detect and react to congestion losses arising from buffer overflows at the bottleneck link. This keeps the queue at the bottleneck link close to full for longer durations, in turn increasing loss for UDP traffic.

In summary, larger buffers can improve TCP throughput, but at the expense of increased losses for real-time traffic. Buffer sizing therefore needs to consider these trade-offs in environments where TCP and real-time traffic co-exist. For the above experiment, an operating point of 35 buffers might be suitable, since larger buffers offer little throughput advantages to TCP in exchange for increased losses for UDP.

III. Sizing Non-Bottleneck Link Buffers

A. Topology and Traffic Generation

It can be argued that most core links in the Internet are over-provisioned, and unlikely to be bottlenecks. In other words, the offered load may not be high enough to fill the capacity of the link over long periods of time, even if sufficiently large buffers are available. If high-capacity backbone links are indeed not bottlenecks, they need only very small number of buffers to absorb transient bursts of packet arrivals. Nevertheless, we are interested in evaluating if this small number of buffers required at non-bottleneck links is affected by the inclusion of bursty real-time traffic.

The topology we set up for experiments with a non-bottlenecked core link is shown in figure 6. The core link should be high capacity (so as not to be bottlenecked) while having very small buffers. Unfortunately, as observed in section II-A, the token bucket on the Cisco 7204VXR can be made only as small as 10msec worth of buffers at the configured shaping rate. This implies that to make the buffer as small as say 8-10 packets, the link rate has to be set no higher than 9-12 Mbps (for 1500-byte packets). Our experiments therefore set the core link speed to between 9 and 12 Mbps, as described in the next subsection.

The low core link speed forces us to reduce the access link speeds so as not to saturate the core link and make it a bottleneck. Since the speeds on the PC Ethernet interfaces cannot be reduced below 10 Mbps, we generate periodic UDP traffic (using Iperf) from each PC that consumes a constant 7 Mbps of the access link capacity, leaving around 2.6 Mbps for the TCP streams from each PC. This ensures that the TCP streams are bottlenecked at the access links. Note that the UDP streams used to bottleneck the access link are sinked at a PC in the metro, and are not carried across the core link. Three PCs in Sydney generate 4 TCP streams each, for a total of 12 streams that multiplex at the core link. As before, the video traffic is introduced at the core router, and models an aggregate of real-time traffic from the metro networks. The TCP and video traffic are sinked at PCs in Perth across the CeNTIE network.

B. Experiments and Results

In the first experiment, only TCP traffic is mutliplexed at the core link. The TCP streams are bottlenecked to approximately 2.5 Mbps at the access links, totaling 7.6 Mbps by the time they reach the core link, which is set at 9 Mbps (this corresponds to a load of around 85%; it was found that with lower loads the buffer size needed to be smaller than the $10s$msec minimum permitted on the Cisco 7204VXR). The top curve in figure 7 shows that 9-10 buffers are needed for an TCP aggregate throughput of 7.4 Mbps.

In the second experiment, one video stream is introduced at the router. The video stream has an average bit-rate of 1.3 Mbps, and to accomodate for this extra traffic, the core link bandwidth is increased by 1.5 Mbps to maintain the link load. The middle curve in figure 7 plots the aggregate TCP throughput for various buffer sizes (the standard deviation is indicated by error-bars), and shows that more buffers are needed to reach the saturation throughput (of 7.6 Mbps as before). For example, achieving 7.4 Mbps in this case requires 11-12 buffers, which is more than when the core link carried only TCP traffic.

In the third experiment, two video streams were multiplexed with the TCP traffic at the core link. Correspondingly the core link rate was increased to 12 Mbps to maintain a loading of around 85%. The bottom curve in figure 7 depicts the TCP throughput for various buffer sizes. Achieving throughput of 7.4 Mbps requires around 13 buffers in this case, indicating that the buffer requirement goes up as the video traffic increases.

The difference in buffering requirement is not as pronounced at non-bottleneck links as was observed at bottleneck links. This is not surprising, given that very short buffers

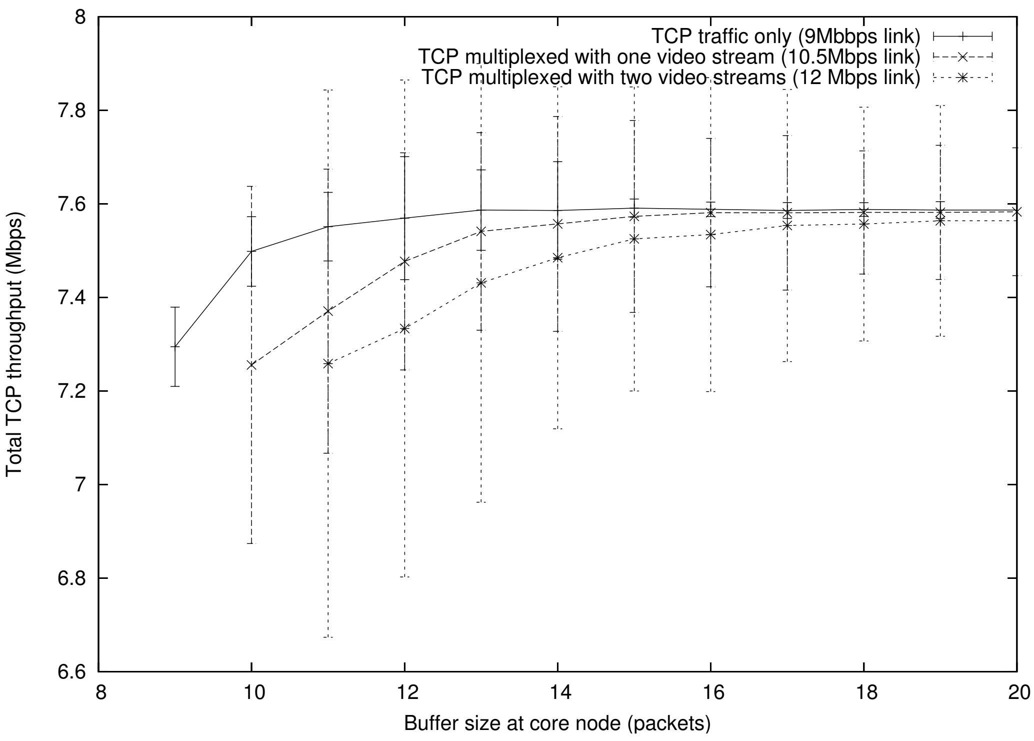

Fig. 7. TCP throughput vs. non-bottleneck buffer size when TCP is multiplexed with zero, one, and two video streams.

are needed when the link is not saturated. Nevertheless, the shape of the curves does clearly indicate that real-time traffic impacts TCP performance and increases buffering requirements even at non-bottleneck links.

IV. Conclusions and Future Work

In this paper we studied experimentally the impact of router buffer size on end-to-end performance for mixed TCP and real-time traffic. Our study indicates that even small quantities of bursty real-time traffic can interact with well-behaved TCP traffic (paced either by end-hosts or by slow access links) to make the latter bursty within the network, thereby increasing the buffers required by TCP to realise near-maximum throughput. Further, experimental results show that the increase in TCP throughput from larger buffers comes at the expense of increased UDP losses, and this trade-off needs to be considered when sizing router buffers.

Our future work will involve running more extensive experiments with larger numbers of TCP flows and diverse real-time applications so as to understand the interaction of TCP and UDP better. We would also like to explore the efficacy of techniques that pace traffic *within* the network; in the optical packet switching context, our work in [11], [8] develops such techniques that efficiently pace packets at optical edge nodes. We would like to quantifying the resulting impact on buffer requirements for good end-to-end TCP and real-time traffic performance.

V. Acknowledgements

This work was funded under the UNSW Early Career Researcher (ECR) grants scheme projects PS08904 and PS08804. The authors would like to thank Craig Russell of CSIRO for configuring the CeNTIE network switches for the experiments conducted in this paper.

References

[1] C. Villamizar and C. Song, “High performance TCP in ANSNet,” *ACM SIGCOMM Computer Communications Review*, vol. 24, no. 5, pp. 45–60, 1994.

[2] S. Iyer, R. R. Kompella, and N. McKeown, “Analysis of a memory architecture for fast packet buffers,” in *Proceedings of IEEE HPSR*, Dallas, TX, May 2001.

[3] 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)*, Santa Barbara, CA, Aug 2004.

[4] G. Appenzeller, I. Keslassy, and N. McKeown, “Sizing router buffers,” in *Proceedings of SIGCOMM 2004*, Portland, OR, Aug-Sep 2004.

[5] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden, “Part III: Routers with very small buffers,” *ACM SIGCOMM Computer Communications Review*, vol. 35, no. 3, pp. 83–90, Jul 2005.

[6] ——, “Routers with very small buffers,” in *Proceedings of IEEE Infocom*, Barcelona, Spain, Apr 2006.

[7] A. Dhamdhere and C. Dovrolis, “Open issues in router buffer sizing,” *ACM SIGCOMM Computer Communications Review*, vol. 36, no. 1, pp. 87–92, Jan 2006.

[8] V. Sivaraman, H. ElGindy, D. Moreland, and D. Ostry, “Packet pacing in short buffer optical packet switched networks,” in *Proceedings of IEEE Infocom*, Barcelona, Spain, Apr 2006.

[9] T. Percival, “An Introduction to CeNTIE,” Presentation, *http://www.centie.org/docs/CeNTIE-web-intro.ppt*.

[10] “VLC media player,” *http://www.videolan.org/vlc/*.

[11] V. Sivaraman, D. Moreland, and D. Ostry, “A Novel Delay-Bounded Traffic Conditioner for Optical Edge Switches,” in *IEEE HPSR 2005*, Hong Kong, May 2005.