Evaluation of FAST TCP in Low-Speed DOCSIS-based Access Networks

Lachlan L. H. Andrew
ARC Special Research Centre for
Ultra-Broadband Information Networks
University of Melbourne
Vic. 3010, Australia
l.andrew@unimelb.edu.au

Irena Atov, David Kennedy
Centre for Advanced Internet Architectures
Swinburne University of Technology
PO Box 218, Vic. 3122, Australia
{iatov,dkennedy}@swin.edu.au

Bartek Wydrowski
Networking Lab
California Institute of Technology
Mail Code 256-80, CA 91125, USA
bartek@caltech.edu

***Abstract*— There is strong evidence that the efficiency of the Internet is limited by its existing TCP congestion control system. A replacement, FAST, has been shown to improve performance in high-speed networks. In order to achieve widespread acceptance and standardisation, it must also be tested in environments more typical of the existing Internet. This paper experimentally evaluates the performance of FAST over a typical access link, with bandwidths of around 0.5-3 Mbps. Links both using the DOCSIS cable modem medium access control (MAC) protocol and simple low rate links were investigated. It is shown that the random delay introduced by MAC protocol of the cable modem does not appear to interfere significantly with FAST's ability to set the congestion window size to its target. However, the cable modem does appear to introduce consistent additional delays when the link is highly, but not fully, utilised. These unexplained delays mean that a larger congestion window is required, and must be taken into account when setting FAST's parameters, notably the target queue size, alpha.**

I. Introduction

The Internet started as a research tool to allow resources to be shared between universities. It is now indispensable in many areas of the daily life, and increasingly underpins the economy and everyday infrastructure. One of the key technologies behind the Internet is the Transmission Control Protocol, TCP, which governs the rate at which data is transmitted between hosts on the network. The current standard version of TCP, NewReno (RFC 2528), is based closely on TCP Reno (RFC 793), which was designed two decades ago. It reflects the prevailing network technology, and best understanding of network dynamics and congestion control at that time [1], [2]. As the Internet has evolved to encompass a more diverse range of network data rates and packet loss/latency characteristics, standard TCP is increasingly a limiting factor in network performance [3], [4].

FAST (Fast AQM Scalable TCP) [5], [6] is emerging as a strong candidate for a new IETF TCP standard. It was designed from the ground up to work efficiently with modern networks, especially high speed networks, and those with long propagation delays. Unlike proposals such as BIC [3] and High-Speed TCP [7], which control flows' rates in response to packet loss, FAST TCP follows the approach of TCP Vegas [8] and responds to queueing delay. This allows the equilibrium queue size to be orders of magnitude smaller than the buffer size, and avoids the waste incurred by packet losses. Unlike Vegas, FAST does not suffer from instability, even for very large networks.

To date, FAST has been tested by Caltech and independent groups such as SLAC (Stanford Linear Accelerator Center) and CERN (The European Particle Physics Laboratory) in a wide range of high speed environments [9]. Given the benefits of FAST in these environments, it would be desirable for it to be deployed widely. However, this requires that it be evaluated in a diverse range of networks, including low rate access networks typical in the existing Internet. The current and medium term future of access networks is in the 1-10 Mbps range, using such technologies as xDSL and cable modems. FAST's behaviour in these environments must also be understood.

This paper experimentally evaluates the performance of FAST in a typical access network involving DOCSIS (Data Over Cable Service Interface Specification) cable network [10]. We perform experiments using a CISCO DOCSIS cable system [11].

The cable modem environment is of particular interest as it has a MAC (medium access control) protocol, and it is reasonable to expect the delay this introduces interact with the delay based control of FAST. The way the MAC layer delays packet transmission in case of congestion is different from the queueing at a switch; a characteristic dependence of latency on the offered traffic load has been observed [12].

The particular findings of the present paper are:

- The latency of a cable modem increase when the traffic rate is high. This results in the need for a congestion window larger than the bandwidth delay product.
- This increase in latency depends on the capacity of the link, and is more pronounced when the capacity increases.
- The "alpha" parameter, which FAST uses to control the number of packets that are queued on the data path, must be set large enough to allow for the additional packets stored in the cable modem link.
- Although DOCSIS also introduces unpredictable delays, these do not appear to interfere with FAST's ability to estimate the queueing delay.

Section II will now describe the FAST and DOCSIS protocols and explain the DOCSIS protocol affects data throughput. The experimental setup described in Section III is then used to study the behaviour of a single FAST flow, and two concurrent FAST flows. The results are presented and analysed in Sections IV and V respectively.

II. Background and Objectives

A. The FAST Algorithm

Most congestion control algorithms since TCP Tahoe, and its popular successor TCP Reno, adjust a source's transmission rate based on the rate at which packets are lost, interpreting packet loss as an indication of congestion. FAST follows from TCP Vegas [8] in adjusting flow rates in response to the measured delay. These algorithms adjust a source's window size to attempt to maintain a constant number of its own packets, α, queued in nodes along its path. The queueing delay is estimated as the difference between the mean round trip time, denoted D, and the minimum round trip time observed by any packet, d.

FAST updates the window size according to [5], [6]

$$w(t+1) = \left\lfloor \frac{1}{2}\left(w(t) + \frac{d}{D} w(t) + \alpha \right) \right\rfloor . \qquad (1)$$

The focus of this paper is how to set the α parameter in two different low-speed environments. A value of $\alpha = 3$ has been recommended for Vegas [8]. In high speed links, this has been found to cause insufficient queueing for reliable measurement of the resulting delays [13]. For high speed links, it has been recommended that α be set to cause a given small queueing delay, such as 2 ms [14]. This paper shows that this rule of thumb gives insufficient queueing for low speed networks, especially when DOCSIS is used.

B. DOCSIS — Cable Modem Links

DOCSIS coaxial or hybrid-fiber/coax (HFC) cable links are among the most popular broadband IP access technologies.

A typical DOCSIS cable network, illustrated in Fig. 1, consists of two key components: the Cable Modems (CM) located at the customer premises, and a Cable Modem Termination System (CMTS) located in the service provider's (SP) network. Transmission over the downstream and upstream channels is controlled by the CMTS. The upstream channel is a multipoint-to-point channel shared by all the cable modems (CM) using a time-slot structure. A centralized MAC protocol based on a reservation scheme, also known as a *Request-and-Grant cycle*, controls the access to the upstream channel.

The CMTS regularly sends control signals in the downstream channel to all CMs in the network, which contain MAP (MAC allocation and management) messages to describe the allocation of upstream bandwidth. The MAP message transmission interval can be dynamic or fixed (our Cisco CMTS used a fixed interval defaulting to 2 ms). The upstream channel can be regarded as a stream of time slots (called mini-slots) with duration, in symbols, determined by the CMTS. These slots can be of the request or data type. Request slots are individual mini-slots used to send request messages. Data slots consist of adjacent mini-slots used to carry the data packets. A CM, upon receiving a packet to transmit, must request an allocation from the CMTS specifying the number of slots needed to transmit the packet, and its service identification (SID). In this process the CM contends for access during periods specified in the MAP messages. Once the CMTS receives the request it allocates a proper portion of the upstream bandwidth for the CM in a future MAP message. The maximum burst size of data that a CM can send upstream per MAP opportunity is limited as specified in the *max-burst size* parameter contained in SID for the CM (we use the default upstream max-burst size of 1600 bytes, which accommodates all Ethernet packets).

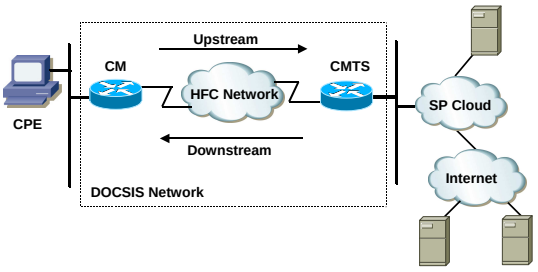

Fig. 1. A typical DOCSIS-based cable network

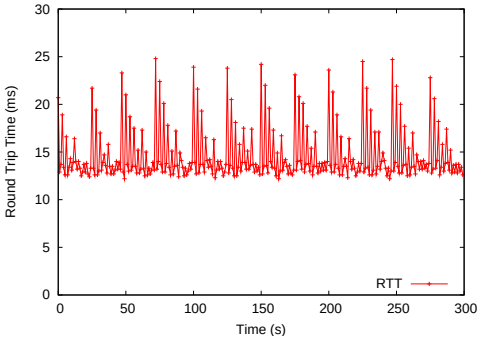

Fig. 2. Round trip time for an idle path containing DOCSIS

The way in which the DOCSIS protocol governs communication between the CM and the CMTS introduces additional and unpredictable latency into the network's performance [12], [15]. Figure 2 shows the round trip time measured by "ping" for a path containing a DOCSIS link, with no other traffic present. Selected packets experience additional delays of up to $25 - 12 = 13$ ms.

III. Experimental Setup

We have experimentally evaluated the performance of FAST over two different access networks, each with a single bottleneck link. One contained a DOCSIS cable modem, and the other was a simple rate-limited link. We considered static scenarios where the bottleneck link carried one or two FAST flows, and no other traffic. (The DOCSIS link also transmitted low-rate keep-alive messages.)

Our aim was to investigate the impact of downstream (DS) and upstream (US) bandwidth limits of the bottleneck link on the overall FAST TCP performance, and investigate how the FAST and DOCSIS parameters should be set to ensure good performance.

Figure 3 shows the equipment used for our experiments. The setup containing a DOCSIS link is shown; the differences the simple link experiments are described later. The sender (TCP server) runs Linux with Caltech's FAST patches and the receiver runs standard Linux. In addition to the DOCSIS cable network (CMTS and CM), a router running dummynet [16] under FreeBSD and a standard Ethernet switch are used to

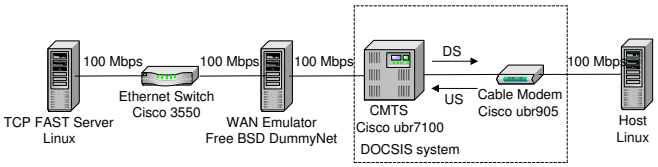

Fig. 3. Test setup

emulate a typical ISP network. The sender, the receiver and the dummynet router are 2.4 GHz Intel Celerons with 256 MB of RAM and 100 Mbps Ethernet cards. The switch is a Catalyst 3550, the CMTS is a Cisco ubr7100 and the CM is a Cisco ubr905, which also acts as a router.

All the links in the network except for the bottleneck link have capacity 100 Mbps. The dummynet was configured to emulate a high-speed WAN path of 100 ms Round Trip Time (RTT) without imposing any limitation on the downstream and upstream channel capacities. Additional constant delays, notably in the DOCSIS link, make the total RTT approximately 115 ms when no traffic is present. The dummynet used a buffer size of 2048 kbytes (involving two pipes in series, each of 1024 kbytes). This resulted in no packet losses in the core network.

In all of our experiments the maximum buffer size on the bottleneck link was 512 ms, the default value of the Cisco CMTS [17] configuration. The experiments consisted of running one or two iperf flow(s) with 1500-byte packets on the downlink.

For the experiments involving a simple low-speed link, the DOCSIS system was bypassed. Instead, the same dummynet that emulated the WAN delay was also configured to emulate the bottleneck capacity limits, in both the downstream and upstream, and the limited buffering. The dummynet RTT was still set to 100 ms.

We tested FAST for configurations involving the following downstream/upstream capacity pairs: 256 kbps/128 kbps, 512 kbps/256kbps, 1.54 Mbps/512 kbps, 2 Mbps/512 kbps, 2.5 Mbps/512 kbps and 3 Mbps/512 kbps.

IV. Single Flow Results

Figure 4 shows the throughput obtained by FAST as a function of the parameter α for a downlink/uplink speed of 3 Mbps/512 kbps. The throughput results are averaged over 100 runs. This demonstrates that $\alpha = 4$ is sufficient for almost full utilisation on a simple 3 Mbps link, but that a much larger value is required on a 3 Mbps DOCSIS link. The slight reduction from full capacity is mostly accounted for by the 2.5% overhead of TCP and IP headers (20+20 bytes out of 1500).

At 3 Mbps with 1500-byte packets, $\alpha = 4$ corresponds to a delay of 16 ms. This is much larger than is necessary to obtain accurate timing estimates. The reason that such a large queueing delay is required on simple and DOCSIS links will now be discussed in turn.

A. FAST on a simple slow link

There are two major effects at work causing utilisation to be low for $\alpha \leq 3$. The first of these, burstiness due to delayed acknowledgements, also affects Vegas, while the second, integer arithmetic, is specific to the FAST algorithm (1).

Burstiness refers to the artificially high queueing observed by the second and later packets transmitted in a burst sent at a rate larger than the bottleneck rate. This causes the average queue size observed by the packets to be greater than the true mean queue size. FAST attempts to minimise burstiness, but is hindered by TCP delayed acknowledgements [2]. TCP is only required to acknowledge every second packet; when an acknowledgement for two packets arrives, it causes two back-to-back packets to be sent. This will cause the mean queueing delay to be overestimated by an entire packet time when the window size is even; if every second packet is acknowledged, then acknowledgements are all for even-numbered packets, while each acknowledgement causes the release of an odd-numbered packet followed immediately by an even-numbered packet. Thus, even for very low utilisation, delayed ACKs alone can allow D to be up to $d + t_p$, where t_p denotes the packet delay. Thus FAST requires

$$\alpha \geq 1 \tag{2}$$

to achieve full utilisation.

The second reason for requiring large α is the floor operation in (1). Without this operation, the update rule

$$w(t+1) = \frac{1}{2}\left(w(t) + \frac{d}{D}w(t) + \alpha\right) \tag{3}$$

satisfies the equilibrium relationship

$$w = \alpha\frac{D}{D-d} \tag{4}$$

for a given base round trip time, d, and average round trip time D. However, at the equilibrium of (1), we only know

$$w \leq \frac{1}{2}\left(w + \frac{d}{D}w + \alpha\right) < w + 1,$$

giving

$$(\alpha - 2)\frac{D}{D-d} < w \leq \alpha\frac{D}{D-d}, \tag{5}$$

so that an equilibrium window size can be as small as that predicted by (3) using $\alpha' = \alpha - 2$. (In practice, $w \geq 1$ is enforced.)

In the absence of burstiness or random delays, $D - d$ would be zero if the link were underutilised, resulting in an infinite equilibrium window; thus the utilisation would be 100%. However, if $D-d$ is bounded away from zero, then the rounding in (1) can cause the utilisation to be as low as is predicted by (3) under the substitution $\alpha' = \alpha - 2$. Combined with (2), which requires $\alpha' \geq 1$, we expect that $\alpha \geq 3$ for full utilisation.

Note that the impact of integer arithmetic depends on the amount of rounding at the particular equilibrium point. We will later see that full utilisation can be achieved with $\alpha = 1$ in some circumstances.

Figure 4 shows that full utilisation is still not achieved even for $\alpha = 3$. This may be due to some additional burstiness in addition to the unavoidable burstiness due to delayed acknowledgements, or it may be due to the jitter of up to 1 ms introduced by dummynet.

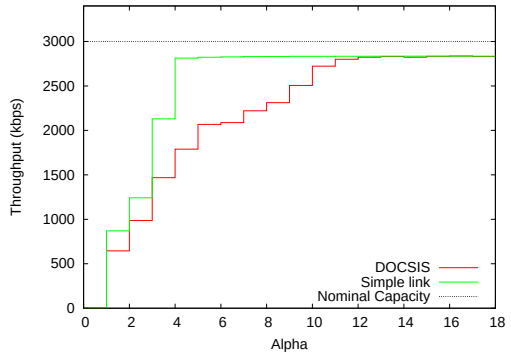

Fig. 4. Throughput vs. alpha for DS=3Mbps, US=512Kbps, DOCSIS and simple link.

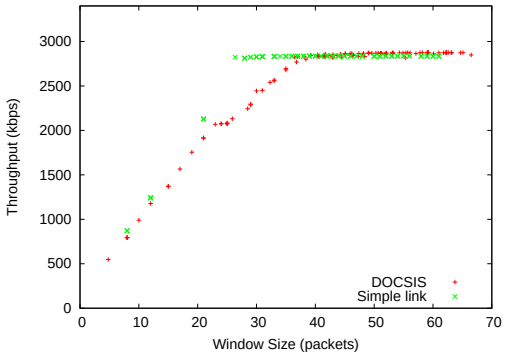

Fig. 5. Utilisation vs. window size, DOCSIS and simple link.

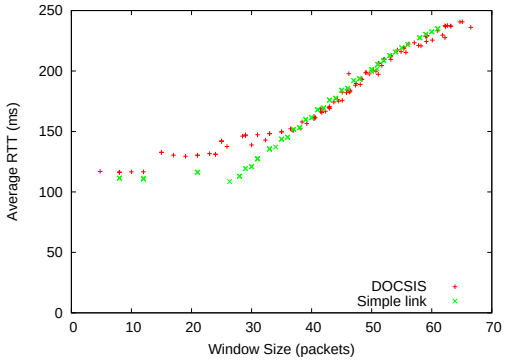

Fig. 6. Average RTT vs. window size, DOCSIS and simple link.

B. *FAST on a DOCSIS link*

Figure 4 shows that the throughput achieved by a single FAST stream is much less in a DOCSIS access system than an equal-rate system not running DOCSIS. In particular, at 3 Mbps FAST requires $\alpha = 13$ to obtain full utilisation using DOCSIS, compared with $\alpha = 4$ on a simple link. At this speed, $\alpha = 13$ corresponds to a target queueing delay of 52 ms.

There are several possible reasons for this discrepancy. As illustrated in Figure 2, the DOCSIS cable system introduces latency fluctuations. It is possible that these interfere with FAST's estimates of the queueing in the network, resulting in the congestion window being too low. This interference may be a direct result of the additional delay, or it may be that these fluctuations induce burstiness in the pattern of packet transmissions; as discussed in the previous section, this also causes the mean queue size to be overestimated. A second possibility is that the actual window size required to achieve full utilisation in a DOCSIS system is larger than the bandwidth-delay product. Let us first consider the second possibility.

A bottleneck link carrying a single flow in a purely deterministic network will be fully utilised if the flow's window size is at least the "bandwidth delay product", d times the link capacity. For a 100 ms (or 115 ms) path with a bottleneck link of 3 Mbps, this is 25 (or 28) packets of 1500 bytes. For smaller windows, the throughput reduces in proportion to the window size.

Once equilibrium was reached, FAST usually maintained a constant window size, even though different experiments yielded different constant values for a given value of α. This means that it is reasonable to talk of "the" window size for a given experiment.

Figure 5 plots the observed throughput against the window size for 140 experiments using α values from 1 to 35. The expected behaviour is observed for a simple link, and for DOCSIS systems with small windows. However, for window sizes between 20 and 37 DOCSIS consistently yields lower utilisation than predicted. (Close inspection shows that this is also true for window size between 13 and 20.) Thus, even if FAST correctly sets the window size to the bandwidth delay product plus α, full utilisation will not be achieved unless $\alpha \geq 37 - 28 = 9$ packets. As indicated in the previous section, the integer arithmetic of (1) can require α to be increased by 2, yielding a requirement of $\alpha \geq 11$ packets for full utilisation. This is close to what was observed in Figure 4, suggesting that FAST's window size is not adversely affected by the randomness of the delay at this operating point.

After the kink at a window size of around 20 packets, the utilisation is again approximately proportional to the window size, but with a constant of proportionality indicating a RTT of 145 ms. This can be observed in the plot of mean RTT, D against window size in Figure 6. The mean RTT is equal to the propagation delay plus the mean queueing delay. The queueing delay should be zero when the link is underutilised, and be proportional to the difference between the window size and the bandwidth delay product when the link is fully utilised. This is indeed observed for the simple link, but DOCSIS shows a higher delay for windows between about 13 and 37 packets, with a value of approximately 145 ms between about 20 and 37 packets, in agreement with Figure 5.

This phantom delay cannot be attributed to burstiness, since burstiness does not reduce the throughput, denoted x, for a given window size W, while Figure 5 demonstrates that this phantom delay does. That is, it is not simply FAST's estimate of the RTT that has increased, but rather the actual RTT, W/x. This is supported by RTT measurements from ping and consistent with the findings of [12].

Similarly, if the additional delay were purely due to burstiness, it would be expected that at least some packets every round trip time would observe approximately the true propagation delay. This is shown not to be the case by Figure 7, which plots the average of the minimum value of RTT observed in successive 4 s measurement intervals.

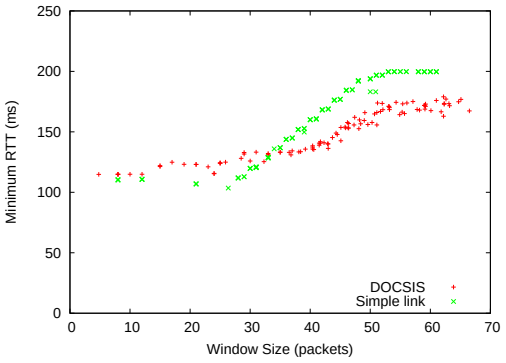

Fig. 7. Minimum RTT vs. window size, DOCSIS and simple link.

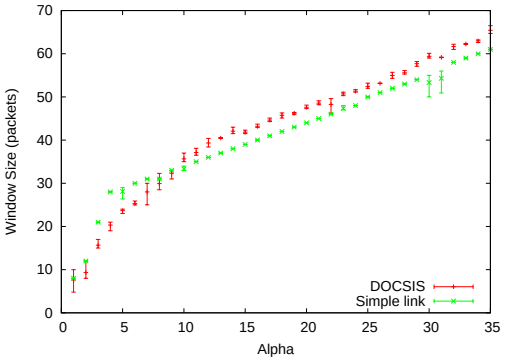

Fig. 8. Window size calculated by FAST for a downlink rate of 3 Mbps.

This indicates that DOCSIS is not work conserving; that is, it may buffer packets even when the link is idle. This may be due to the Request-and-Grant time slot allocation scheme it employs. However this is not clear, since DOCSIS obtains the full 3 Mbps throughput with sufficiently large α, suggesting that this overhead may not be significant.

For windows greater than 37 packets, this phantom delay merges with the queueing delay, so that the total observed delay is the same for DOCSIS and a simple link.

Let us now consider the first reason for requiring large α suggested at the start of this section: that the delay causes FAST to set the window size too small for a given α. Figure 8 shows FAST's window size (with error bars indicating the maximum and minimum) as a function of α for a simple link and a DOCSIS link. For small α, when the link is not fully utilised, FAST underestimates α using DOCSIS, reflecting the phantom delay observed in Figure 6. This exacerbates the underutilisation in this region. However, once the link reaches full utilisation, the phantom delay is "absorbed" into the queueing delay, and the window size is no longer too small. In fact, the window size is approximately 4 packets larger under DOCSIS than the simple link. This corresponds to the difference of $115 - 100 = 15$ ms in the observed base round trip times, d.

These results are summarised in Figure 9, which shows the target queueing delay (α divided by the capacity) that FAST needs in order to obtain full utilisation. For very low rates, full utilisation is achieved with $\alpha = 1$ or 2, but the delay becomes large because the delay from a single packet is large. As the rate increases, the queueing required on a simple link decreases

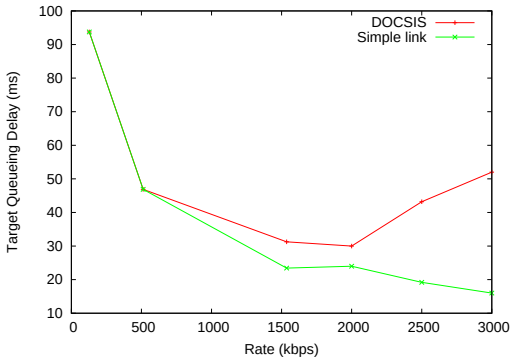

Fig. 9. Total target queueing delay required by FAST as a function of link capacity.

monotonically, and will asymptote to a value which gives a queueing delay just large enough to be reliably detected. In contrast, the queueing delay required over DOCSIS increases again as the rate increases.

The default value of α when FAST operates on a slow link is 20 packets, which is large enough to accommodate the observed range of effects. However, if the trend continues to higher rate DOCSIS systems then α may need to be increased.

V. Two Flow Results

The previous section investigated and compared the performance of FAST when a single TCP connection is established over either a simple low-speed link or a cable modem access link. We now extend this study to consider two FAST TCP connections sharing the system. The two TCP connections were two concurrent iperf sessions from the TCP server to the receiver run. Note that both flows shared the same cable modem, and so again there was no contention at the MAC layer.

The α parameter was set equal for the two TCP connections and varied from 1 to 35. Each experiment was run 10 times, resulting in a total of 350 tests for each considered type of access network.

When n flows share a single bottleneck link, the total queueing at the link is ideally $n\alpha$. Thus, if the only reason to need $\alpha > 1$ were to ensure that the queueing delay was larger than the timing uncertainties, as is the case in high speed networks, we would expect the required α value to scale inversely with n. Specifically, if there are two flows, we would expect the α required for full utilisation to halve. From Section IV-B we know that single flow needs $\alpha = 13$ or target queueing delay of 52 ms to achieve full utilisation on a 3 Mbps DOCSIS link. When two flows are sharing the link one would expect each individual flow to need $\alpha = 7$, which would again give a total queueing delay of 52 ms. In other words, one wouldn't expect the total target queueing delay to change with the increase of the number of TCP flows.

Figure 10 shows the aggregate throughput obtained by the two FAST flows as a function of α for both a simple link with downlink/uplink speed of 3 Mbps/512 kbps and a DOCSIS link of the same speed.

If the reason for needing $\alpha = 4$ in Section IV were simply to allow for delay jitter, we would expect in this case to

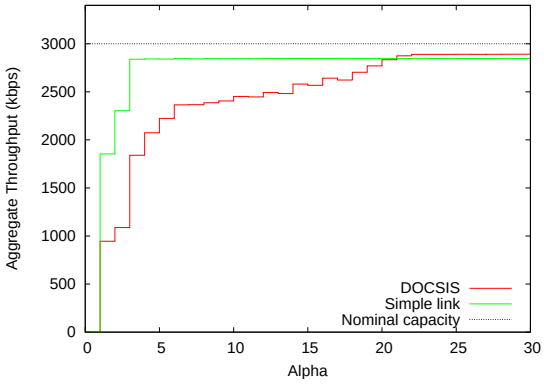

Fig. 10. Rate versus α for two FAST flows for DS=3 Mbps, US=512 kbps, for simple and DOCSIS links.

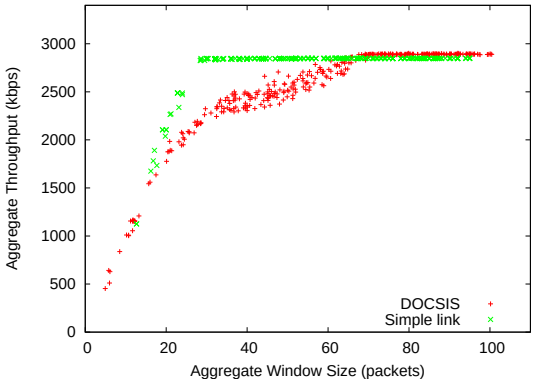

Fig. 11. Rate versus window size for two FAST flows for DS=3Mbps, US=512Kbps, for simple and DOCSIS links.

achieve full bandwidth utilisation with $\alpha = 2$ for each flow. However, Figure 10 shows that $\alpha = 3$ is needed. This is not too unexpected, since Section IV-A argues that it is reasonable to need $\alpha \geq 3$ for reasons other than needing queueing delay.

The more surprising results were for the DOCSIS link. Rather than decreasing by a factor of two to 7, the required α actually increased to 22. That corresponds to a total target queue size of 44 packets or a delay of 176 ms.

To investigate this result, the throughput achieved for a given flow is again plotted against the (aggregate) window size in Figure 11. As in Section IV-B, this indicates that the total window size required for full utilisation is significantly greater than the bandwidth delay product. That is, the need for large α is not due to an inability of the FAST algorithm to set the window size to maintain α packets at the sending node of the bottleneck link.

The trend of superlinear buffer requirements is concerning, in light of the fact that the cable modems had a default "traffic shaping" buffer with maximum delay 512 ms, which can be increased to at most 1028 ms [17]. If the trend continues, then the default buffer could support fewer than $512/(176/2) \approx 6$ flows at full utilisation. Once again, this appears to be intrinsic to DOCSIS, rather than due to FAST.

VI. Conclusions and Future Directions

We have investigated the performance of FAST TCP over low speed links, and in particular links running DOCSIS. FAST is able to achieve almost full utilisation over a low speed link if its target queue size (alpha) is at least three packets. This corresponds to a decreasing queueing delay as the data rate increases. In contrast, DOCSIS introduces additional latency, which requires both alpha and the target queueing delay to increase as the link capacity increases beyond 1 Mbps, up to $\alpha = 13$ for 3 Mbps. These values are below FAST's current default value of $\alpha = 20$ for low speed links, but this phenomenon must be considered if the default values are changed.

This paper has not considered the mechanisms by which DOCSIS introduces the additional delays. The next stage in this research will be to derive an analytic model of the interaction between DOCSIS and FAST.

Acknowledgements

We thank Cisco Systems Australia for donating equipment, and L. Stewart of CAIA for assistance setting up the network. CUBIN is an affiliated programme of National ICT Australia. This work was supported by the Australian Research Council.

References

[1] J. Postel, *Transmission Control Protocol*, STD 7, IETF RFC 793, September 1981. Available at http://www.ietf.org/rfc/rfc793.txt.
[2] M. Allman, V., Paxson, and W. Stevens, *TCP Congestion Control*, IETF RFC 2581, April 1999. Available at http://www.ietf.org/rfc/rfc2581.txt.
[3] L. Xu, K., Harfoush, and I. Rhee, "Binary Increase Congestion Control for Fast Long-Distance Networks", in *Proc. of IEEE INFOCOM 2004*, pp. 2514-2524, Hong Kong, March 2004.
[4] T. Kelly, "Scalable TCP: Improving Performance in Highspeed Wide Area Networks", *ACM SIGCOMM Computer Communication Review*, Vol.33, No. 2, pp. 83-91, 2003.
[5] C. Jin, D. X., Wei, and S. H. Low, "FAST TCP: Motivation, Architecture, Algorithms, Performance", in *Proc. of IEEE INFOCOM 2004*, pp. 2490-2501, Hong Kong, March 2004.
[6] J. Wang, D. X., Wei, and S. H. Low, "Modeling and stability of FAST TCP", in *Proc. of IEEE INFOCOM 2005*, Miami, FL, March 2005. Available at http://netlab.caltech.edu/pub/papers/FASTstability-infocom05.pdf.
[7] S. Floyd, *HighSpeed TCP for Large Congestion Windows*, IETF Internet Draft, draft-floyd-tcp-highspeed-02.txt, 2002.
[8] L. S. Brakmo and L. L. Peterson, "Vegas: End-to-End Congestion Avoidance on a Global Internet", *Journal on Select Areas in Communications*, Vol.13, No. 8, pp. 1465-1480, 1995.
[9] H. Bullot, R. L. Cottrel, and R. Highes-Jones, "Evaluation of Advanced TCP Stacks on Fast Long-Distance Production Networks", *Journal on Grid Computing*, Vol.1, No. 4, pp. 345-359, Dec 2003.
[10] CableLabs, *Data-Over-Cable Service Interface Specifications Radio Frequency Interface Specification SP -RFIv1.1-I01-990311*,1999.
[11] Broadband Access Research Testbed, Centre for Advanced Internet Architectures, Swinburne University of Technology, http://caia.swin.edu.au/bart.
[12] T. T. T., Nguyen and G. J. Armitage, "Experimentally derived interactions between TCP traffic and service quality over DOCSIS cable links", in *Proc. of IEEE GLOBCOM 2004*, Texas, USA, November 2004.
[13] S. Hegde, D. Lapsley, B. Wydrowski, J. Lindheim, D. Wei, C. Jin, S. Low, H. Newman, "FAST TCP in High-Speed Networs: An Experimental Study, *Proc. of GridNets*, San Jose, CA, October 2004.
[14] C. Jin, D. Wei, and S. H. Low, "FAST TCP for High-Speed Long-Distance Networks," internet draft draft-jwl-tcp-fast-01.txt, [Online]. Available http://netlab.caltech.edu/pub/papers/draft-jwl-tcp-fast-01.txt.
[15] (2004) Understanding Data Throughput in a DOCSIS World. [Online]. Available: http://www.cisco.com/en/US/tech/tk86/tk168/ technologies_tech_note09186a0080094545.shtml
[16] L. Rizzo, "Dummynet", [Online]. Available: http://info.iet.unipi.it/∼luigi/ip_dummynet/.
[17] Cisco uBR7100 Series Software Configuration Guide. [Online]. Available: http://www.cisco.com/en/US/products/hw/cable/ps2211/products_configuration_guide_chapter09186a00801b355a.html#wp1021916.