Experimental Evaluation of Congestion Avoidance

David Murray
Murdoch University
School of Engineering and IT
D.Murray@murdoch.edu.au

Kevin Ong
Murdoch University
School of Engineering and IT
K.Ong@murdoch.edu.au

***Abstract*—The Internet is currently undergoing a significant change. The majority of Internet transfers have historically occurred between wired network devices. The popularity of WiFi, combined with the uptake of smartphones and tablets, has changed this assumption and the majority of Internet connections will soon utilise a wireless link. Congestion avoidance controls the rate at which packets leave a TCP sender. This research re-evaluates different congestion avoidance mechanisms over real world wired and wireless links. This paper does not assert that present mechanisms are poor, nor that wireless has never been a consideration, but that the switch to a mobile and wireless majority necessitates a review, with wireless characteristics at the forefront of design considerations. The results of this study show that TCP Cubic and TCP Hybla perform similarly and generally outperform Veno and Westwood in both wired and wireless scenarios. All tested algorithms featured large delays in 3G links. The results suggest that queuing on the 3G links added between 370ms and 570ms of delay. It is suggested that additional research into congestion avoidance and buffering mechanisms over wireless links is needed.**

I. Introduction

Between 2011 and 2016, Cisco predict that the percentage of TCP connections using a wireless link will increase from 45% to 61% [1]. The end-to-end Internet architecture was built on the assumption of a reliable wired link and thus, with the increasing prevalence of wireles, the current protocols should be re-evaluated. The current Linux default, TCP Cubic [2], was designed in a period where the majority of end-to-end connections were wired. This study will investigate what currently available congestion avoidance scheme will provide the best end-to-end performance over modern wireless links. Before discussing the unique characteristics of our modern wireless links, we will review the role and operation of congestion avoidance algorithms.

A. The Role of Congestion Avoidance

TCP congestion avoidance algorithms control end-to-end speeds across the Internet. The aim is to release packets, from the TCP sender to the TCP receiver, at the perfect rate. The perfect rate is the fastest possible end-to-end speed whilst preventing any unnecessary delays and fairly sharing the links with competing traffic flows. This is clearly a very complex task and one that is impossible to perfect without omniscient knowledge of all transmissions over the Internet. The optimal end-to-end protocol will appropriately balance speed, delay and TCP friendliness.

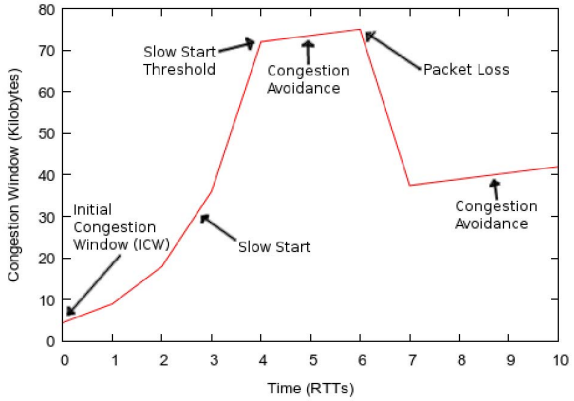

Fig. 1: TCP Slow Start and Congestion Avoidance

B. The Evolution of Congestion Avoidance

The fundamentals of modern TCP congestion avoidance mechanisms are based on TCP Reno, introduced in Jacobson and Karel's 1988 paper [3]. End-to-end speeds are regulated by the TCP congestion window, which stipulates how many packets may be released unacknowledged from the TCP sender to the TCP receiver.

The TCP sender begins with an Initial Congestion Window (ICW). Initially, the growth of this window is exponential, to probe the potential end-to-end bandwidth. This phase is known as slow start, which continues until a packet is lost; a signal of congestion on the Internet. After this packet loss, the congestion window is returned to the ICW and the slow start threshold is set to half of the previous congestion window size. The slow start threshold is used as a demarcation point. The congestion window will restart exponential growth, until it reaches the slow start threshold. Once the congestion window is larger than the slow start threshold, window growth will continue at a linear rate, known as the congestion avoidance phase. Every packet loss will reset the slow start threshold and the congestion window.

The description thus far describes the operation of TCP Reno. These elements, which effect the congestion window, are shown in Fig 1. The diagram does not represent the exact behaviour of all congestion avoidance algorithms but is presented as a guide and reminder of the principles.

TCP NewReno is a modification of Reno, which permits

1550-445X/15 $31.00 © 2015 IEEE
DOI 10.1109/AINA.2015.158

faster recovery after a congestion event such as packet loss. After the loss of a packet, Reno requires retransmission of the lost segment before regrowing the TCP congestion window. TCP NewReno is an enhancement which uses the arriving duplicate acks, received after the notification of packet loss, to continue regrowing the window. This helps to maintain link utilisation and reduce the occurrence of complete TCP timeouts.

BIC superseeded NewReno and hastened recovery after a packet loss but was quickly superseded by Cubic, to address fairness between transactions with different RTTs. TCP Cubic [2] was designed in a era when wired connections were the predominant means of access and has been the default Linux congestion avoidance mechanism since kernel 2.6.19 [4].

C. Accommodating Wireless Links in the Internet

This paper asserts that congestion avoidance protocols require fresh consideration because wireless is becoming the predominant mode of access [1]. This section describes why our modern wireless links differ from traditional wired links.

Wireless links suffer from varying link conditions with higher bit error rates and dropped packets. Collisions are also responsible for packet loss in some wireless link types.

The current, widespread mechanism, used to deal with lossy wireless links is Automatic Repeat Request (ARQ). This data-link layer mechanism successfully confirms the delivery of all packets. ARQ will continually retry failed transmissions at the link layer. As the purpose of ARQ is to hide link losses from the TCP sender, the end-to-end effect is increasing delays and reduced link performance [5].

WiFi data rates are generally higher than 3G/4G networks. However, they use a shared transmission queue. In WiFi networks, users with poor radio conditions and high retransmissions may decrease the predictability of service times for all users. In a study by Rodrig et al, the loss rate in a real world WiFi network was 28% [6]. The use of ARQ to recover these packets means that these losses are hidden from TCP. The consequence of ARQ is often incresed delays [7].

3G/4G technologies often offer lower data rates than WiFi networks, but the service intervals are more predicable and an individual queue is maintained for each user. Similar to WiFi, 3G/4G links may drop packets and ARQ will be used to recover them. As with WiFi networks, the effect on the end-to-end protocols, may simply be fluctuations in the link speed and increased delays [5], [7].

D. Approaches to Congestion Avoidance

Fluctuations in link speeds and suddenly increasing delays are uncommon in wired networks, and therefore, current congestion avoidance protocols may be poorly suited for wireless links [5], [7]. This study will examine currently available congestion avoidance mechanisms. TCP Cubic [2], Veno [8], Westwood [9] and Hybla [10] will all be experimentally tested.

Recent studies [5], [11] have suggested that the current default Internet protocols keep the end-to-end pipe full, at the expense of higher end-to-end delays. Other research has suggested that the excessive buffers in cellular networks may void the loss based congestion control mechanisms [7]. This paper will evaluate which currently implemented congestion avoidance scheme is best suited for these new conditions.

The following section chronologically reviews the congestion avoidance mechanisms evaluated in this paper. While we are aware of a few recently proposed mechanisms [5], [11], at the time of writing they were not integrated into the Linux kernel. This paper only evaluates congestion avoidance algorithms available in the Linux kernel at the time the study took place.

1) Westwood: TCP Westwood was specifically designed for lossy networks and wireless links [9]. Traditional TCP algorithms blindly halve the congestion window upon packet loss, without consideration for whether the loss was due to a link error. Westwood monitors the returning rate of acknowledgements to derive a bandwidth estimate. When a congestion event, for example a packet loss, is detected the TCP congestion window is reset to the bandwidth estimate. TCP Westwood was designed in 2001.

2) Veno: Similar to TCP Westwood, Veno [8] is based around the observation that wireless links may drop packets due to link errors and that blindly halving the TCP window is an inappropriate reaction. TCP Veno is a conceptual combination of TCP Vegas and TCP Reno. TCP Veno tries to differentiate between the losses that occur randomly and may not indicate congestion and the losses that are legitimate signs of congestion. Changes in the RTT are used to estimate the cause of the packet loss. TCP Veno was published two years after Westwood, in 2003.

3) Hybla: As TCP performance improves under low RTT conditions, many mechanisms unfairly penalise large RTT connections [10]. TCP Hybla was specifically designed to address the performance unfairness imposed on TCP connections with long round trip times. In TCP Hybla, TCP growth increases inline with a reference RTT. TCP Hybla was published in 2004.

4) Cubic: TCP Cubic is named after the curve of a cubic function changes the growth rate to be a function of the last congestion event and therefore independent of the RTT [2]. The cubic shaped growth only occurs during the congestion avoidance phase, where the congestion window is greater than the slow start threshold.

When a congestion event occurs, the algorithm will record the congestion window. Although the congestion window will be heavily reduced, window re-growth will follow a cubic function, initially very quickly then tapering off as the congestion window approaches the previous maximum congestion window. Cubic is the current default in Linux operating systems [4] and represents a significant shift away from the standard TCP Reno congestion avoidance phase, which increases the window by 1 segment size per RTT. CUBIC is more aggressive and fairer between TCP flows with disparate RTTs [2].

E. Experimental Evaluations

There are numerous recent academic studies that evaluated congestion avoidance algorithms. De Cicco et al. found no discernible difference between different congestion control mechanisms over wireless links [12]. Sefanos et al. found that, TCP Hybla and Scalable TCP, which were designed for data centres, performed better than delay based mechanisms such as Westwood and Veno [13]. Callegari et al. found that TCP Hybla and Cubic offered the best performance in wired networks [14]. Another conclusion was that the delay based algorithms, designed with wireless links in mind, performed poorly. The lack of consensus has prompted this investigation into the optimal congestion control approach for an increasingly wireless dominated Internet.

II. Methodology

A. Testing Setup

The aim of this study was to experimentally test, using a range of different networks, which TCP algorithm provides the best performance. What constitutes the best performance, from a TCP algorithm, is complex and may vary depending on the needs and requirements of the application. For example, in some applications lower throughputs may be acceptable if a reduction in latency is possible. This study evaluated congestion control mechanisms based on throughput, latency and friendliness. For all the experiments Cubic, Hybla, Westwood and Veno samples were interleaved and obtained over multiple days.

1) Throughput: This study measured the throughput from slow start (1MB transfer) and steady state (20MB transfer) respectively. Each result documented in this study is the average of 60 test runs, the lowest 10% were truncated to remove any outliers. The results show the min, max, mean and interquartile ranges.

2) Latency: A sender side TCPdump was used to capture representative samples to measure RTT. These were analysed using TCP trace before averaging. The results show the mean and standard deviation. The latency of an ICMP message, transmitted over an unloaded link, from Singapore to Perth was approximately 54ms.

3) Friendliness: Any new algorithm must also operate fairly with the current TCP congestion control algorithm, TCP Cubic. To measure TCP friendliness, Veno, Westwood and Hybla flows were started alongside a TCP Cubic flow. As bash scripts perform operations linearly, the ordering that flows were initiated was fairly balanced.

B. Scenarios

This experiment tested four different TCP congestion control mechanisms over five different networks. These five networks, shown in Fig 2, and are listed below:

1) A Wired University Internet Connection
2) A WiFi Eduroam University Internet Connection
3) A Wired ADSL Internet Connection
4) A WiFi ADSL Internet Connection

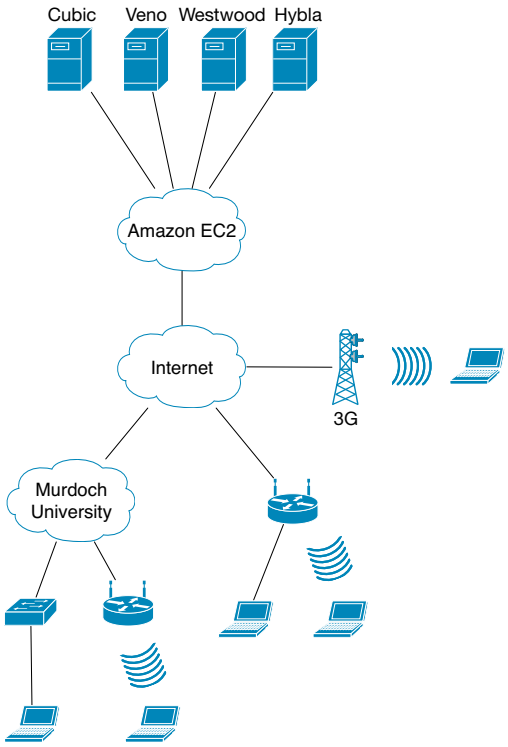

Fig. 2: Experimental Setup

5) A 3G Internet Connection

This study used four Amazon EC2 M1 instances, running in Singapore, as the TCP senders. Each of these four identical servers used Cubic, Veno, Westwood and Hybla respectively. The TCP senders used Ubuntu 12.04 images with the 3.2.0-29-generic kernel.

The TCP receiver was a MacBook Air running the Ubuntu 13.04 and the 3.8.0-27-generic kernel. We believed that the OS and hardware on the receiving device would have very little impact on the overall performance. To demonstrate this, the results on the receiver were sampled against a Dell laptop running Ubuntu 12.04 kernel 3.2.0-31 and a Macbook Air running MacOSX Darwin Kernel Version 12.4.0. As no significant differences were obseved, we believe that the results are applicable to many TCP receivers on the Internet.

III. Results

A. Wired University Internet Connection

The results for the Wired University Internet Connection are shown in Fig 3. In this scenario, the TCP receiver was connected to the LAN. Compared with the other tested networks, the available bandwidth is large, but connections will be competing with many other simultaneous TCP flows. The university has a 1Gb/s fibre optic Internet connection and internal reports suggest that it was rarely used to capacity.

The results suggest that Cubic provided the highest throughputs and Westwood offered the lowest performance. RTTs

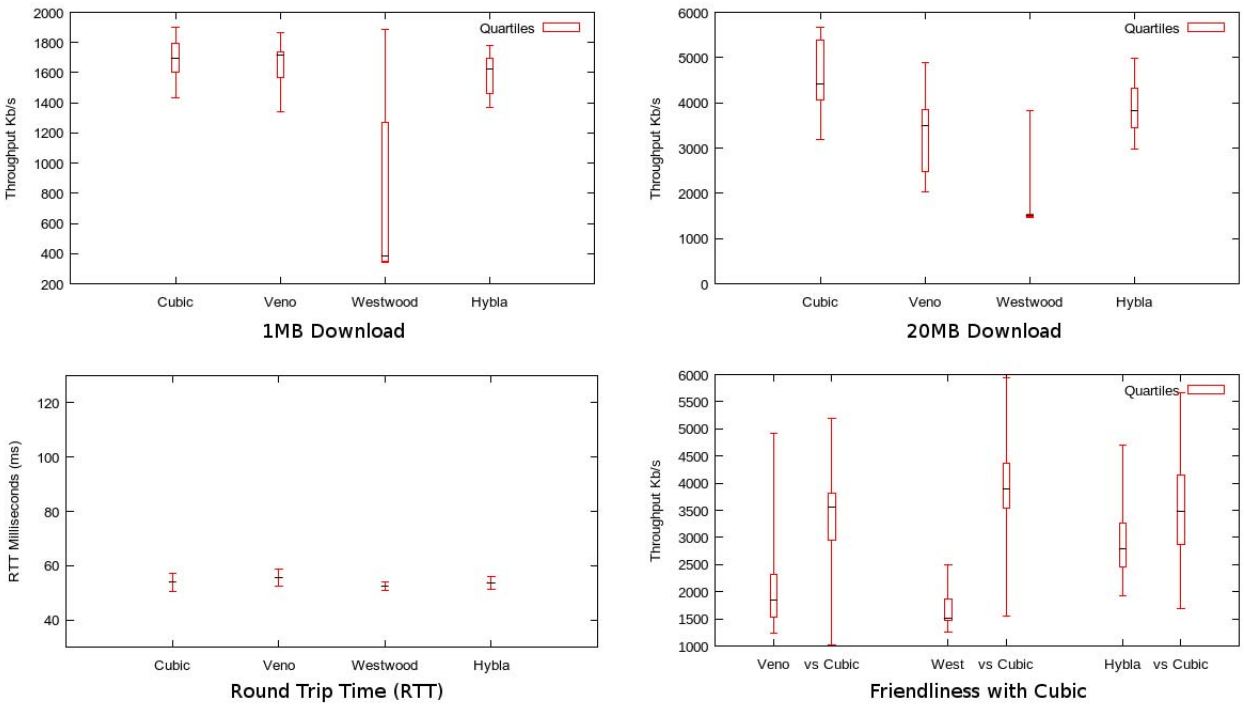

Fig. 3: Wired University Internet Connection

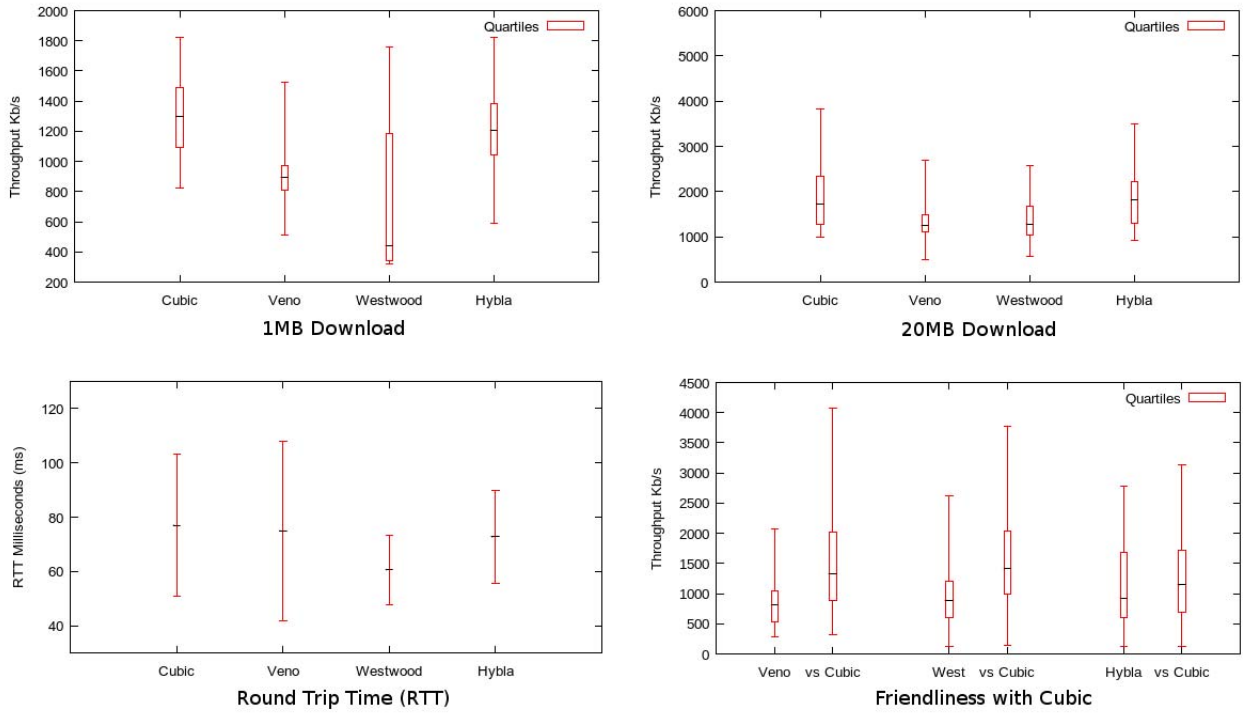

Fig. 4: WiFi Eduroam University Internet Connection

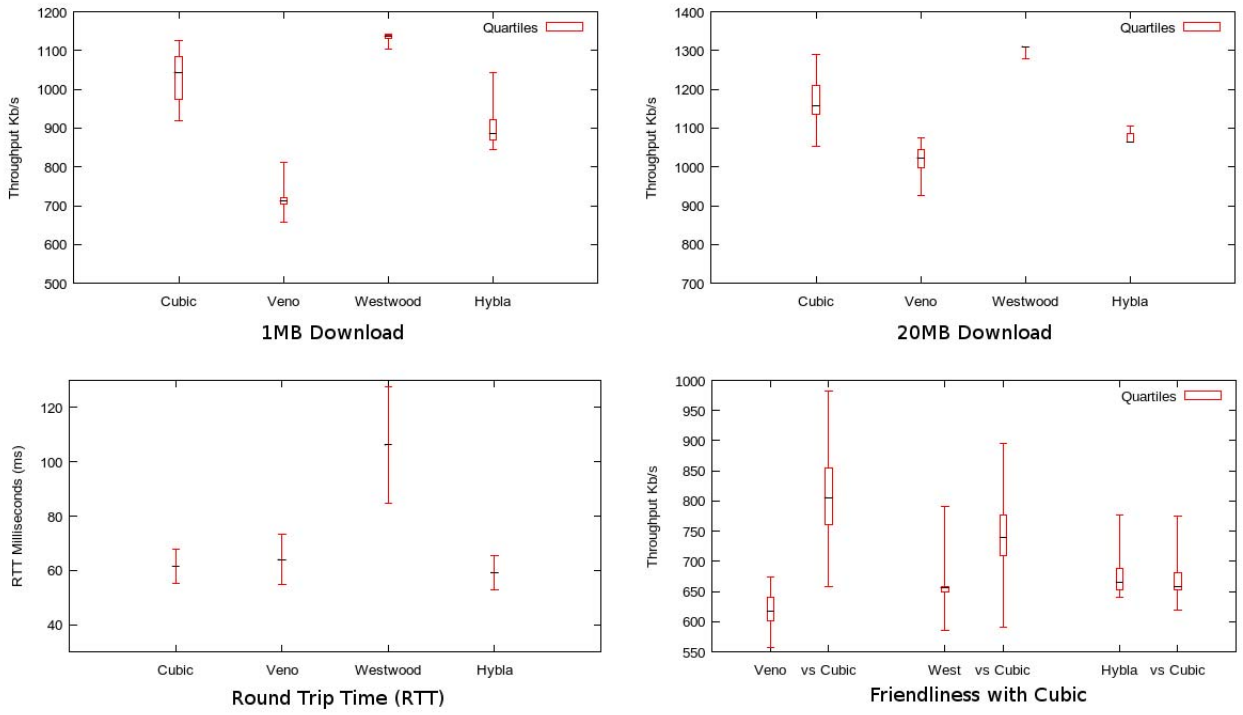

Fig. 5: Wired ADSL Internet Connection

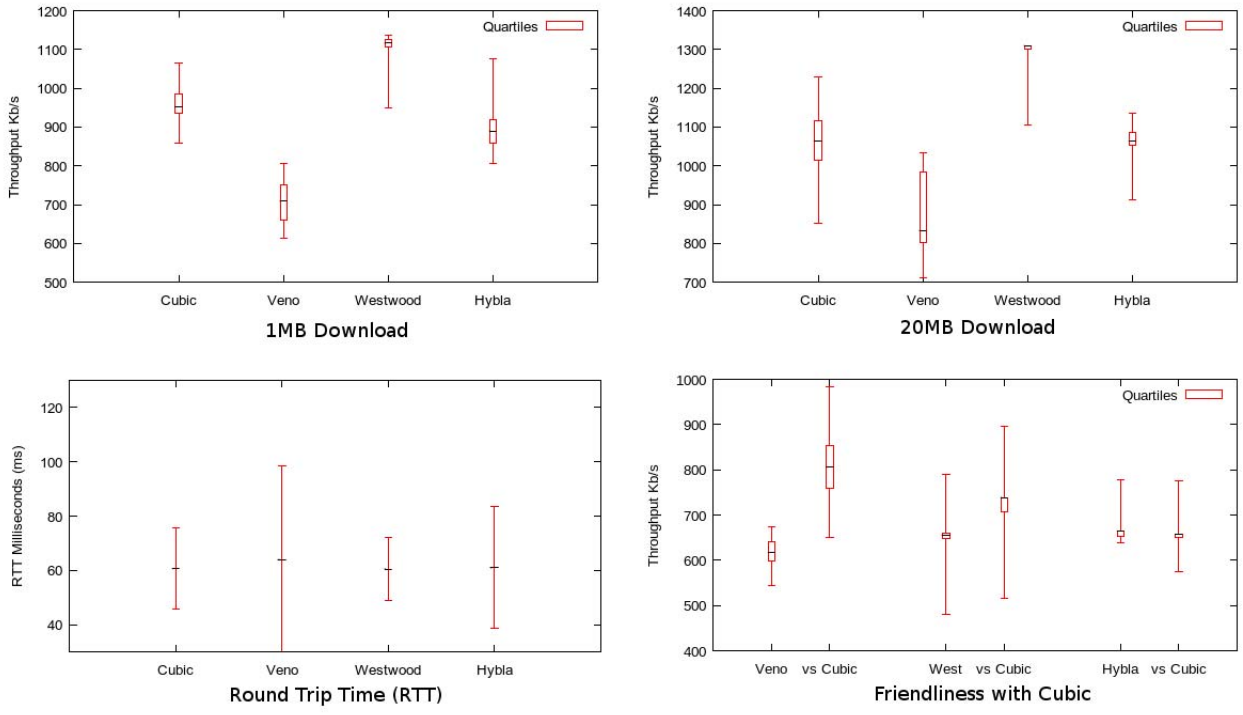

Fig. 6: WiFi ADSL Internet Connection

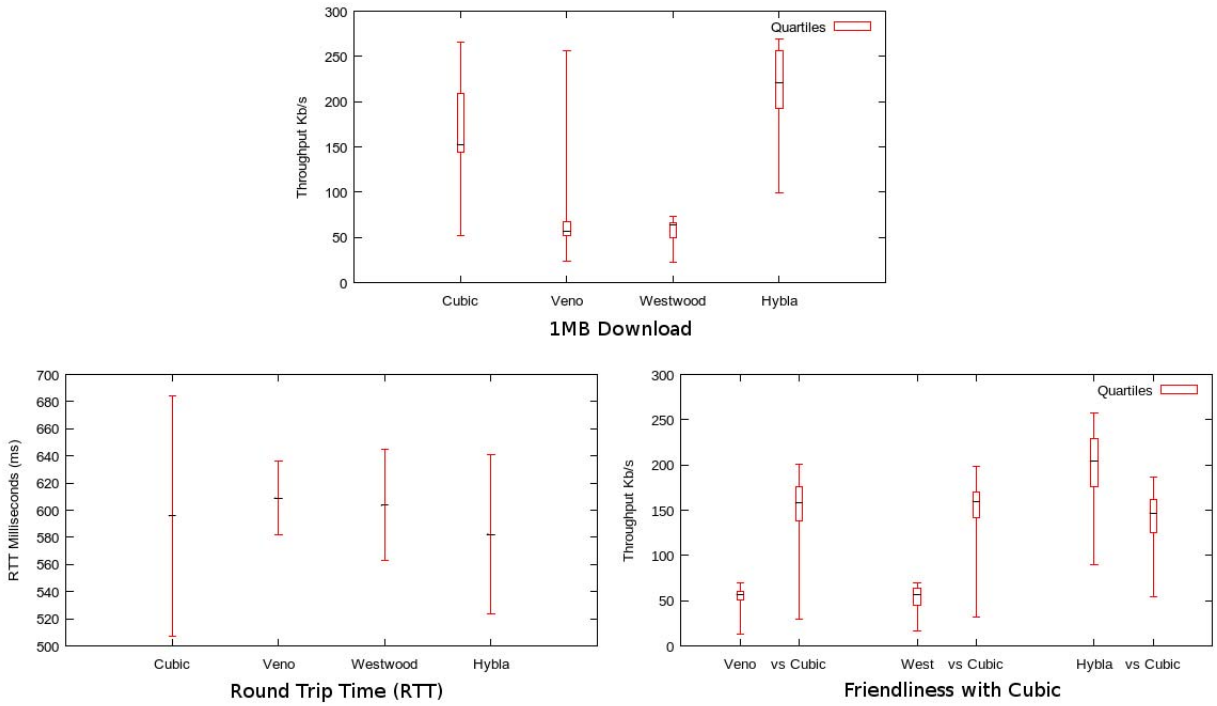

Fig. 7: 3G Internet Connection

and deviations in the RTT were small in all cases. When the algorithms were tested in competition, Cubic consistently obtained the greater share of the bandwidth.

B. WiFi Eduroam University Internet Connection

The results for the Eduroam WiFi University Connection are shown in Fig 4. Eduroam is a WiFi network which operates over the same backbone as the previously tested Wired University LAN. The Eduroam AP, was located in an academic corridor. Average throughput is less than in the previous wired scenario.

Cubic and Hybla provided similar throughput in this experiment. Both Veno and Westwood were clearly slower. Compared with the previous scenario, much larger deviations in RTT were evident. This is likely due to congestion or retransmissions over the WiFi network. Similar to the previously tested scenario, the TCP Friendliness tests show that Cubic achieved a higher proportion of the bandwidth in all tests.

C. Wired ADSL Internet Connection

The results for the Wired ADSL Internet Connection are shown in Fig 5. This scenario tested a 13 Mb/s home broadband connection. Over the duration of the testing period, it was devoid of competing connections. This scenario presents a strong contrast to the busy University Internet connection.

TCP Westwood offered the highest throughputs in the single flow 1MB and 20MB transfers. Although it was the fastest, Westwood also had a significantly higher average RTT, which suggests that Westwood was flooding the connection and building a standing queue. Cubic and Hybla were second and third in terms of throughput, but also had correspondingly lower RTTs.

When competition was introduced, Cubic obtained a larger share of the bandwidth in all cases except when matched against Hybla. TCP Westwood achieved high throughputs over the ADSL link with no competing TCP connections and therefore very few variations in delay. In the tests conducted over a university Internet connection, with thousands of competing users, TCP Westwood performed very poorly. It is possible that varying queueing delays, when competing flows were present, may have caused Westwood to frequently infer congestion.

D. WiFi ADSL Internet Connection

The results for the WiFi ADSL Internet Connection are shown in Fig 6. The WiFi ADSL scenario was performed on the same testbed, except a wireless link was introduced at the network edge to the TCP receiver. The RF environment of the WiFi AP was relatively free over the duration of the tests.

Westwood offered the highest throughputs over the WiFi ADSL link. Similar to other tests over WiFi, large variations in the RTT were evident. Westwood offered significantly higher throughputs when operating as a standalone TCP flow. Similar to previous tests, a proportionately lower speed was observed when a competing Cubic flow was introduced. This suggests that delays introduced by competing algorithms cause

significant reductions in Westwood's congestion window. This observation is present in all tests.

E. 3G Internet Connection

The 3G results are based on single link, in a single location and are shown in Fig 7. Due to the low speeds, only 1MB tests were performed. TCP Hybla offered the highest throughput over the 3G link. The throughput of Cubic was slightly lower than Hybla, however the delay was also lower to compensate. Veno and Westwood performed poorly in throughput metrics, without any benefits in terms of latency.

The poor performance of TCP Veno and Westwood is interesting because they both make changes to the congestion window based on delay and were designed with wireless links in mind. As discussed earlier, the use of ARQ, particularly in wireless links, works against the original design considerations of Veno and Westwood. In wireless links, with ARQ, lost packets get retransmitted, resulting in increased delay.

Very large RTTs were recorded for all congestion avoidance mechanisms operating over 3G links. On the wired links, the RTT from Singapore to Perth was approximately 54ms. Over a 3G link, ICMP tests reported delays of 130ms, suggesting that 76ms was added by media access, propagation delay and the service provider infrastructure. It is therefore alarming that, when a TCP transaction was started over the 3G links, RTTs ranged between 500ms and 700ms. This means that between 370ms and 570ms were added by queuing delays.

Some reports suggest that 3G links buffer excessively to prevent any packet loss [7], [12], [15]. The interaction of ARQ and large 3G buffers with congestion avoidance is suspected to be the cause. Buffering to hide link layer losses may prevent the necessary signals to the TCP sender, causing erroneous window reductions. Implementations which can independently signal congestion [16] and loss [17], [18] have been avaliable for some time. These TCP extensions enable intermediate devices, such as routers or APs, to specifically signal problems in a connection. Explicit signalling would enable TCP senders to maintain their TCP windows under packet loss, and reduce the window in the case of congestion. It is possible that future work could leverage existing mechanisms to better manage the TCP window but given the adoption rate of ECN over the past decade [19], a dramatic change in the foreseeable future is unlikely.

IV. Friendliness with UDP G711 Impact

This study also aimed to measure the impact on VoIP flows. Five, G711 equivalent, flows were initiated, using iperf, from the receiver while a 20MB TCP download was started. The average jitter, and packet loss rates reported by iperf were recorded. The results of these tests were performed over the University network using a wired connection and the Eduroam WiFi connection. The results are shown in I and II. The results show that WiFi introduces higher losses and jitter however, in all cases, the levels of loss and jitter are well within acceptable recommendations [20].

TABLE I: Wired University VoIP

	Cubic	Veno	Westwood	Hybla
Jitter (ms)	0.3813	0.4559	0.5020	0.412
Loss (%)	0.000302	0.000042	0.000120	0.000065

TABLE II: WiFi University VoIP

	Cubic	Veno	Westwood	Hybla
Jitter (ms)	3.80	2.93	5.15	4.28
Loss (%)	0.014	0.018	0.014	0.011

V. Conclusion and Future Work

The Internet is undergoing a dramatic change from a wired to a wireless majority. These changes necessitate a re-evaluation of numerous protocols used in the Internet. This paper has experimentally evaluated four different TCP congestion avoidance mechanisms across five different scenarios. While it is arguable that the scenarios in this paper are not representative of the entire Internet, the results are largely confirmatory of what has previously been reported in the literature [4], [13]. The following are a list of the key findings from this research, with references to confirmatory studies where applicable.

1) TCP Cubic and Hybla, which modify the congestion window based on packet loss, are generally more successful than Veno and Westwood, which modify the congestion window based on delay [13], [14].
2) TCP Veno was consistently the most poorly performing algorithm.
3) When operating in isolation TCP Westwood built large TCP queues, leading to large RTTs.
4) Tests of concurrent Cubic, Veno, Westwood and Hybla connections along side G711 VoIP connections found that all algorithms interacted acceptably with VoIP.
5) All congestion algorithms introduce large delays in 3G links. Some reports suggest that 3G links buffer excessively to prevent any packet loss [15], [7], [12].
6) The results suggest that congestion avoidance algorithms added an additional 370ms and 570ms of delay over 3G links, a problem which new congestion control algorithms must address.

In future work, we intend to increase the level of automation necessary to obtain a larger sample of results. We also plan to further investigate the problems of congestion avoidance over many different mobile broadband technologies. The cause and solution to the very high, latencies, which appear to be directly caused by congestion avoidance, will also be further investigated.

Acknowledgment

The authors would like to acknowledge Amazon for their research grant. This grant was used to provide a consistent and repeatable platform for the TCP senders used in this paper. Any opinions, findings, and conclusion or recommendations expressed in this paper are those of the authors and do not necessarily reflect the view of the Amazon.

REFERENCES

[1] Cisco Systems, "Bandwidth management," *Cisco Visual Networking Index: Forecast and Methodology, 2011 — 2016*, May 2012.

[2] S. Ha, I. Rhee, and L. Xu, "Cubic: a new tcp-friendly high-speed tcp variant," *SIGOPS Oper. Syst. Rev.*, vol. 42, pp. 64–74, July 2008.

[3] V. Jacobson, "Congestion avoidance and control," *SIGCOMM Comput. Commun. Rev.*, vol. 18, pp. 314–329, Aug. 1988.

[4] R. Taank and X.-H. Peng, "An experimental evaluation of sender-side tcp enhancements for wired-to-wireless paths: A real-world home wlan case study," in *Advanced Information Networking and Applications, 2009. AINA '09. International Conference on*, pp. 323–330, 2009.

[5] K. Winstein, A. Sivaraman, and H. Balakrishnan, "Stochastic forecasts achieve high throughput and low delay over cellular networks," in *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, NSDI'13, (Berkeley, CA, USA), pp. 459–472, USENIX Association, 2013.

[6] M. Rodrig, C. Reis, R. Mahajan, D. Wetherall, and J. Zahorjan, "Measurement-based characterization of 802.11 in a hotspot setting," in *Proceedings of the 2005 ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis*, E-WIND '05, (New York, NY, USA), pp. 5–10, ACM, 2005.

[7] H. Jiang, Y. Wang, K. Lee, and I. Rhee, "Tackling bufferbloat in 3G/4G networks," in *Proceedings of the 2012 ACM conference on Internet measurement conference*, IMC '12, (New York, NY, USA), pp. 329–342, ACM, 2012.

[8] C. P. Fu and S. Liew, "TCP Veno: TCP enhancement for Transmission over Wireless Access Networks," *Selected Areas in Communications, IEEE Journal on*, vol. 21, no. 2, pp. 216–228, 2003.

[9] C. Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, and R. Wang, "TCP Westwood: End-to-End Congestion Control for Wired/Wireless Networks," *Wirel. Netw.*, vol. 8, pp. 467–479, Sept. 2002.

[10] C. Caini and R. Firrincieli, "TCP Hybla: A TCP Enhancement for Heterogeneous Networks," *International Journal of SatelliteE Communications and Networking*, vol. 22, 2004.

[11] K. Winstein and H. Balakrishnan, "Tcp ex machina: Computer-generated congestion control," *SIGCOMM Comput. Commun. Rev.*, vol. 43, pp. 123–134, Aug. 2013.

[12] L. D. Cicco and S. Mascolo, "Tcp congestion control over hsdpa: an experimental evaluation," *CoRR*, vol. abs/1212.1621, 2012.

[13] H. Stefanos, S. Nikolaos, and F. Eleni, "An extended evaluation of a collection of TCP Congestion Control Algorithms." Tech Rept, 2013.

[14] C. Callegari, S. Giordano, M. Pagano, and T. Pepe, "Behavior analysis of tcp linux variants," *Comput. Netw.*, vol. 56, pp. 462–476, Jan. 2011.

[15] R. Mahajan, J. Padhye, S. Agarwal, and B. Zill, "High performance vehicular connectivity using opportunistic erasure coding," in *USENIX Annual Technical Conference*, USENIX, 2012.

[16] K. Ramakrishnan, S. Floyd, and D. Black, "The addition of explicit congestion notification (ecn) to ip." IETF RFC 3168, September 2001.

[17] H. Balakrishnan and R. H. Katz, "Explicit Loss Notification and Wireless Web Performance," in *IEEE Globecom Internet Mini-Conference*, 1988.

[18] D. Murray, T. Koziniec, and M. Dixon, "Solving ack inefficiencies in 802.11 networks," in *2009 IEEE International Conference on Internet Multimedia Services Architecture and Applications (IMSAA)*, pp. 7–12, 2009.

[19] D. Murray and T. Koziniec, "The State of Enterprise Network Traffic in 2012," in *18th Asia-Pacific Conference on Communications (APCC 2012)*, 2012.

[20] Cisco, "Quality of Service Design Overview," in *Enterprise QoS Solution Reference Network Design Guide* (Cisco, ed.), Cisco Press, 2004.