2016 30th International Conference on Advanced Information Networking and Applications Workshops

Large-Sample Comparison of TCP Congestion Control Mechanisms over Wireless Networks

Kevin Ong, David Murray, Tanya McGill
School of Engineering and Information Technology
Murdoch University
Perth, Australia
{k.ong, d.murray, t.mcgill}@murdoch.edu.au

***Abstract*—As new congestion control mechanisms are developed, their performance relative to existing mechanisms needs to be understood; in particular over wireless networks. This study aimed to evaluate existing TCP congestion control mechanisms using a comprehensive and reproducible methodology designed to be representative of real world usage of wireless networks. The study sought to investigate whether any existing mechanism could provide significant performance benefits over CUBIC and be recommended for adoption.**

The findings of this study showed that YeAH demonstrated an increase in throughput of 3%–5% over CUBIC, with no penalty to latency. While this small improvement may assist applications requiring the highest available performance, it is unlikely that it will provide a significant improvement over existing congestion control mechanisms. As such, it is the conclusion of this study that use of alternate congestion control mechanisms would not provide noticeable improvements in performance in most applications.

I. Introduction

TCP congestion control is intended to prevent TCP senders from introducing congestion to the path being used for transmission, as well as alleviating the need to drop packets in response to congestion [1]. New congestion management algorithms have been proposed and developed with the aim of improving TCP performance in specific scenarios, including over wireless networks where the suboptimal performance of TCP has been recognised for some time [2].

Traffic originating from a wireless device surpassed 50% of Internet traffic in 2014, and is expected to reach 81% by 2019 [3]. As such, the performance of TCP over wireless networks needs to be better understood and improved upon. Numerous studies evaluating the effectiveness of TCP congestion control mechanisms have been undertaken since the development of TCP Tahoe in 1988 [1]. These include both preliminary evaluations carried out by the developers proposing new mechanisms [4], [5], as well as studies comparing existing mechanisms [6], [7], [8].

Prior studies evaluating the performance of TCP congestion control mechanisms have utilised simulation or production networks for data collection. Of these approaches, simulated networks modelled using software tools such as `ns-2` are commonly employed as they allow researchers to evaluate modifications to TCP congestion control relatively quickly and under a wide range of scenarios that are not feasible to reproduce [9]. This approach is typically favoured as a means for quickly evaluating newly proposed congestion control mechanisms.

Internet-based testing allows researchers to remove the possibility that assumptions made in the network model will bias the results [9]. However, results produced using this approach are often difficult to reproduce, due to constant changes to the topology of the Internet [10]. It is also impossible to analyse how traffic generated by a new congestion control mechanism impacts other traffic traversing a specific link [9].

This study aimed to evaluate existing TCP congestion control mechanisms — specifically, Westwood, Hybla, Illinois, YeAH, and LP — using a comprehensive and reproducible methodology designed to be representative of real world usage of wireless networks. In doing so, it sought to investigate whether any existing mechanism can provide noticeable performance benefits over CUBIC.

II. Related Work

Research into TCP congestion control mechanisms aims to improve the performance of existing congestion control mechanisms. In relation to congestion control, performance can be quantified using a variety of metrics including throughput, latency, and fairness. The development of congestion control mechanisms, however, involves a process of refinement to balance these characteristics.

For example, CUBIC [11] was designed to address the aggressiveness its predecessor, BIC-TCP, would employ in utilising available bandwidth. However, this change resulted in CUBIC being slower to utilise available bandwidth.

Similarly, delay-based mechanisms such as Westwood [4] attempt to detect network congestion before packets are dropped. While this approach allows such mechanisms to avoid packet loss events, prior studies have typically shown delay-based mechanisms to have poor performance relative to conventional loss-based congestion control [7].

Recent mechanisms such as Low Priority (LP) [12] and LEDBAT [13] aim to utilise excess bandwidth with minimal impact on other traffic. Such mechanisms are most useful in situations where increased transaction time would be least noticeable. However, a study by Callegari *et al.* [7] has found that LP performs comparably to other mechanisms.

To assess the merit of different approaches to congestion control, the performance of these mechanisms must be quan-

978-1-5090-2461-2/16 $31.00 © 2016 IEEE
DOI 10.1109/WAINA.2016.88

tified. The metrics commonly used in the evaluation of TCP congestion control are discussed below.

A. Performance Metrics

Research into TCP congestion control mechanisms aims to improve the performance of existing congestion control mechanisms, usually measured based on throughput [7], [14], goodput [15], [8], or link utilisation [16]. As different throughput metrics have been utilised across these studies, direct comparisons between the results of studies utilising throughput, goodput and link utilisation are infeasible, the performance of congestion control mechanisms is examined, relative to others evaluated by the same study.

Despite studies demonstrating the potential for significant differences in performance between congestion control mechanisms, prior studies have found little difference in the throughput, goodput or link utilisation achievable by CUBIC, Westwood+, and Hybla over 802.11g, 3G and satellite links [17], [6], [7], [8].

While maximising the rate of data transmission is a significant measure of the effectiveness of TCP congestion control mechanisms, increasing the delay experienced by individual data segments can impact the usability of interactive applications [18]. Although several metrics exist to quantify delay [18], the round trip time (RTT) experienced by data transmitted using each congestion control mechanism is most commonly reported metric [8], [14]. However, many studies do not measure or report delay.

'Flow rate fairness' or 'RTT fairness' refer to the impact of a TCP stream to the transmission capacity of other TCP streams operating over the same communications medium [9], [11]. As with delay, many studies do not measure or report on the ability of new congestion control mechanisms to fairly share available bandwidth. However, of studies that do, fairness is typically reported using either Jain's Fairness Index [7], [8], or as a proportion of link utilisation [11], [5].

The results of a study by Callegari *et al.* [7] found that, with the exception of Westwood and Illinois, all TCP variants integrated into the Linux kernel scored above 0.9 on Jain's Fairness Index for long connections. However, Hybla, and YeAH TCP score above 0.9 for shorter connections. Westwood consistently scored the lowest fairness index (0.74 for long lived flows, and 0.7 for short flows).

As with intra-protocol fairness, many studies that evaluate TCP congestion control mechanisms do not consider the ability of new congestion control mechanisms to co-exist with existing — and commonly used — mechanisms such as Reno, NewReno, CUBIC and Compound TCP [19], [14]. Prior studies that have evaluated this metric report friendliness based on the throughput of the new congestion control mechanism, as compared to that of the existing mechanism [11], [7], [5].

However, Callegari *et al.* [7] suggest that Reno is more aggressive, and typically appropriates a greater share of available bandwidth than CUBIC for short and long-lived TCP connections. YeAH TCP and TCP-LP are able to achieve similar throughput to Reno on longer connections, although Reno grabs the largest share of available bandwidth on shorter connections.

III. Methodology

To address the lack of realism in previous studies, and the inability to definitively identify better performing congestion control mechanisms, the study described in this paper was designed to operate on live networks. Fig 1 depicts the network topology used for the experiment.

To best emulate the use of Content Delivery Networks (CDNs), virtual servers hosted by Amazon Elastic Compute Cloud (EC2) in Sydney, Australia were used to host files of varying sizes to be downloaded as part of this study. The use of EC2 allowed the experiment to measure TCP performance over the same network infrastructure that is utilised by the Amazon Simple Storage Service (S3) used as a CDN.

Files to be transferred as part of the experiment were hosted on a t2.small instance in the Sydney datacentre. The image ran Ubuntu 14.04.1 and the Apache 2.4.7 web server. An identical instance was used for testing TCP Friendliness. Data transfers were carried out using three commonly used wireless networking technologies: 802.11n (WiFi), High-Speed Packet Data Access (HSDPA, also known as '3G'), and Long Term Evolution (LTE, commonly referred to as '4G').

Testing over WiFi was carried out using 802.11n operating at 2.4GHz. The wireless network was connected to the Internet via a university network. A Cisco Meraki Z1 acted as the wireless access point and gateway in this scenario. Use of the Z1 allowed the scenario to emulate a typical home network, while use of the university gigabit network ensured that the WiFi link was the bottleneck.

Testing over HSDPA and LTE was carried out using the Telstra mobile network using the Nexus 7's inbuilt modem. Telstra was previously a government-owned telecommunications provider for Australia, and provided the greatest LTE coverage at the time this study took place.

A Google Nexus 7 LTE tablet running Android 4.4.4 was used as the client for this experiment. The tablet was selected due to the inclusion of integrated radios for WiFi, HSDPA, and LTE. Use of these integrated radios allowed each wireless technology to be tested in isolation. Firefox for Android (version 34.0.1) was used for file transfers.

A. Congestion Control Mechanisms

In order to maintain a manageable scope, the study was limited to mechanisms that aim to address the suboptimal performance of TCP over wireless networks. In addition to CUBIC, five such congestion control mechanisms were identified: Westwood, Hybla, Illinois, YeAH, and LP.

B. Test Data

As of December 2014, websites had been found by HTTP Archive [20] to transfer an average of 1958KiB of data to the end-user. Individual non-image elements varied in size from 59KiB to 298KiB, with a mean non-image transfer of 117.17KiB. As such, file transfers of 100KiB and 2MiB were

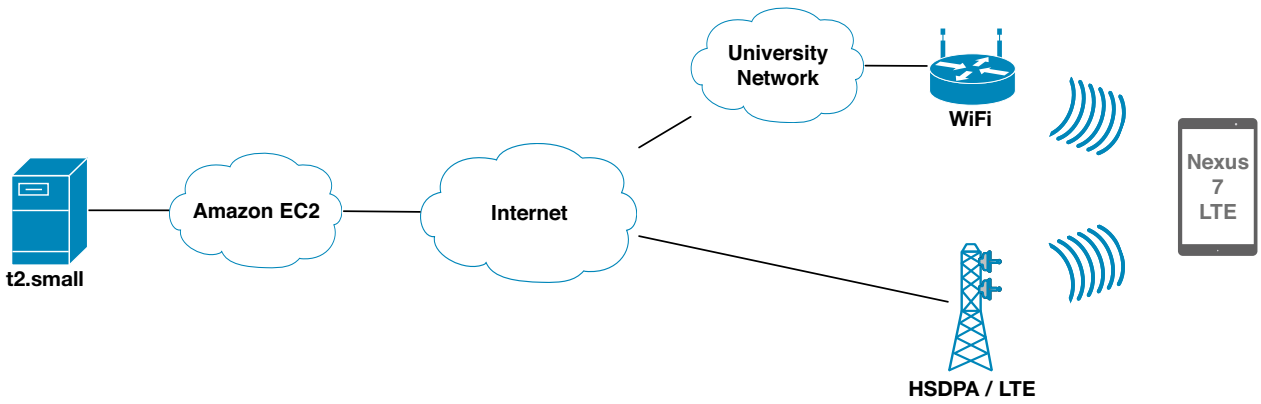

Fig. 1: The experimental network topology.

included in this study. In addition to the smaller 100KiB and 2MiB file transfers, larger file transfers of 10MiB were included to allow for an examination of congestion control mechanism performance over a longer period of time.

C. Metrics

While previous research evaluating the effectiveness of TCP congestion control mechanisms has utilised a wide range of performance metrics, four primary metrics were considered for this study: throughput, latency, intra-protocol fairness, and inter-protocol fairness. However, due to the limited differences found in fairness during previous studies, intra-protocol and inter-protocol fairness were not ultimately included. Values for throughput and latency were calculated using a Python script developed for the data analysis of this study which utilises the Wireshark command-line interface (see Section III-E).

1) Throughput: For the purpose of this study, throughput was calculated based on the transfer size (using the sum of values stored in frame.len), and duration (using frame.time_relative) recorded by the packet capture based on the formula:

$$throughput = \sum (frame.len - 26)/duration$$

2) Latency: The Wireshark RTTs for TCP segments not subject to TCP offload were aggregated. Any RTTs calculated for retransmitted segments were excluded from these calculations.

D. Data Collection

To address the lack of repeatability encountered in Internet-based testing, data collection was undertaken over a 24 hour period for each permutation of testing. This approach ensured that data collected would include captures during times of low and high network usage.

Tests for transfers of 100KiB, 2MiB, and 10MiB over WiFi, HSDPA, and LTE were included in this evaluation. Each of these permutations constitutes a single round of testing, with a minimum of 100 samples captured per congestion control mechanism.

Each sample consists of a packet capture of all packets sent and received by the EC2 instance over a two minute period. Packet captures should contain all traffic resulting from the file transfers initiated by the tablet. This process was mostly automated through the use of shell scripting.

E. Data Analysis

The resulting packet captures were automatically processed by a Python script developed to analyse packet captures, with the results stored using a SQLite database. This Python script utilises the Wireshark packet analysis engine via the `tshark` command-line interface.

The validity of packet captures was also checked to ensure that results were not biased by invalid tests. Packet captures were examined based on the following criteria:

- Traffic was only captured for the number of TCP connections expected for each test.
- Traffic was only captured for the two endpoints used in each test.
- Sufficient traffic was captured to account for the file transferred for the test.

Packet captures that failed any of these tests were discarded. The remaining results were examined for outliers. Potential outliers were identified based on the rule that values that are more than three interquartile ranges above the 75th quartile, or below the 25th percentile are probable outliers. Such outliers are identified by the SPSS statistics package as 'extreme outliers' [21]. This test was applied to the aggregate throughput, per-connection throughput, and median latency for all tests. Any test for which one of these results was identified as a probable outlier was excluded from the set of results.

F. Pilot Testing

In order to ensure the data collection methodology, as well as automation and data analysis techniques functioned

as expected, a pilot study was undertaken. Data collection for the pilot study was limited to one hour, capturing five samples per congestion control mechanism.

IV. Evaluation Results

Tables I, II, and III present a summary of the results of data transfers over a single TCP connection for each link technology included in this study. These results are ranked in order of throughput.

TABLE I: Mean throughput and latency for transfers over a single TCP connection via WiFi.

Mechanism	Tput	Latency
YeAH	338.02	74.17
LP	337.56	74.01
Illinois	337.34	75.39
Westwood	337.21	75.44
Hybla	336.60	74.65
CUBIC	330.55	73.61

(a) For 2MiB transfers.

Mechanism	Tput	Latency
Hybla	1051.58	171.68
Illinois	1050.21	172.05
Westwood	1043.40	169.62
YeAH	1034.44	92.34
CUBIC	1032.23	171.17
LP	1031.03	132.18

(b) For 10MiB transfers.

TABLE II: Mean throughput and latency for transfers over a single TCP connection via HSDPA.

Mechanism	Tput	Latency
YeAH	261.96	129.67
Hybla	261.95	123.44
LP	260.93	124.35
Westwood	260.61	123.81
Illinois	260.48	126.71
CUBIC	250.01	126.36

(a) For 2MiB transfers.

Mechanism	Tput	Latency
Hybla	1346.07	81.40
YeAH	1300.08	76.48
CUBIC	1294.23	79.20
Illinois	1280.61	78.53
LP	1277.82	76.97
Westwood	1253.56	78.91

(b) For 10MiB transfers.

TABLE III: Mean throughput and latency for transfers over a single TCP connection via LTE.

Mechanism	Tput	Latency
YeAH	304.17	101.43
LP	302.00	101.38
Illinois	301.99	101.39
Westwood	296.76	101.40
Hybla	294.75	101.38
CUBIC	288.57	101.42

(a) For 2MiB transfers.

Mechanism	Tput	Latency
Illinois	1302.24	111.38
Westwood	1300.93	111.10
YeAH	1297.05	105.55
Hybla	1287.89	111.18
CUBIC	1106.34	104.74
LP	633.27	106.41

(b) For 10MiB transfers.

While a difference between the slowest and fastest mechanisms of 38.68% was noted for 10MiB transfers via LTE, it should be noted this result includes an abnormally low throughput for LP. For all other tests, the differences between the slowest and fastest throughput ranged from 1.95% (for 10MiB transfers via WiFi), and 6.87% (for 10MiB transfers via HSDPA). Due to the volume of data collected, only key results are elaborated on below.

A. Throughput Results

The observed throughput for a small file transfer (100KiB) via WiFi, HSDPA, and LTE showed little difference can be observed in the throughput achieved by the mechanisms tested.

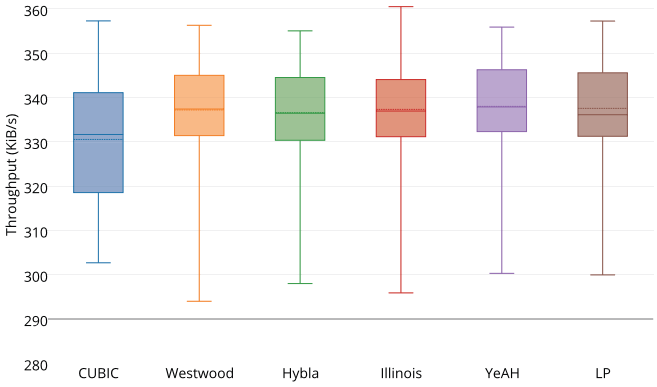

Fig. 2: Throughput for 2MiB transfers over a single TCP connection via WiFi.

CUBIC demonstrated the lowest median throughput of the mechanisms evaluated in transfers of 2MiB of data, as shown in Fig 2, although the difference between the lowest (CUBIC) and highest (YeAH) median throughputs was only 6.22KiB/s; or 1.8%. The differences in throughput achieved by the tested mechanisms would not be noticeable in real-world usage.

Illinois possessed the widest range of observed throughput values for this test, although CUBIC exhibited the largest interquartile range of all mechanisms evaluated — 22.56KiB/s, compared to 14.33KiB/s and 14.17KiB/s for LP and Hybla, respectively — suggesting less consistent performance overall.

The low throughput achieved by CUBIC became somewhat more evident in larger transfers. In tests of 10MiB transfers over HSDPA, CUBIC demonstrated the lowest median throughput of the six congestion control mechanisms included in this study. However, this result amounts to a difference in throughput of 28.19KiB/s, or 2.1% between CUBIC and LP.

Hybla and Westwood achieved the highest median throughputs in this set of tests. However, these results only account for an increase of 5.8% and 5.7% over CUBIC, respectively.

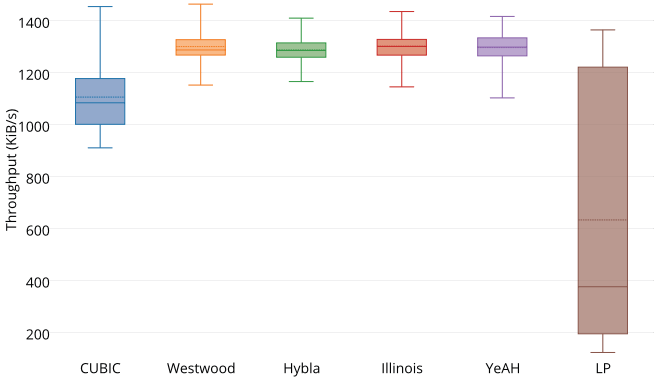

Fig. 3: Throughput for 10MiB transfers over a single TCP connection via LTE.

As depicted in Fig 3, CUBIC was outperformed by Hybla, Westwood, YeAH, as well as Illinois in 10MiB transfers over LTE,. In these tests, a difference of 18.6% was observed between between CUBIC and Hybla.

LP demonstrated the lowest median throughput of 376.02KiB/s. Somewhat interestingly, the mean throughput achieved by LP in these tests was significantly higher than the median at 633.27KiB/s. As LP is known to be a less than best effort congestion control mechanism, this anomaly might be explained by additional network traffic being serviced during the testing period. A slight difference (1.2%) was observed in the median throughput for the remaining four mechanisms evaluated.

B. *Latency Results*

As with throughput, little difference was observed in the median latency achieved for short transfers. In testing over an LTE network the differences in RTT ranged from 17.56ms for LP, to 23.32ms for Hybla.

Differences in latency are more apparent in Fig 4, which presents results from the same test carried out over the Telstra HSDPA network. In these tests, the range of median RTTs varied by as much as 46.61ms. Data transmitted using Westwood experienced the greatest variation in latency, with an interquartile range of 32.14ms. This interquartile range was approximately 64% larger than that of the second largest: 19.57ms experienced by traffic managed by YeAH. Despite these differences, median RTT for Westwood, Hybla, Illinois, YeAH, and LP was relatively uniform although traffic managed by CUBIC experienced slightly more delay.

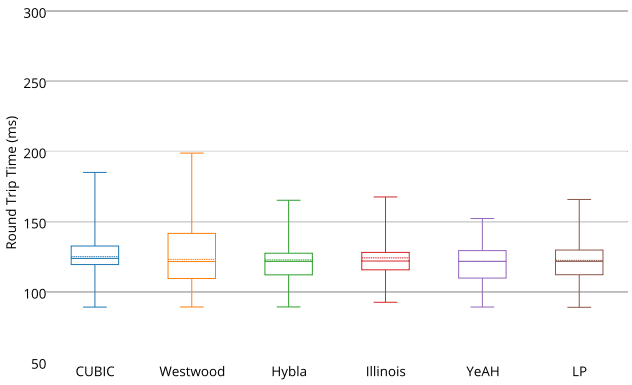

Fig. 4: Latency for 100KiB transfers over a single TCP connection via HSDPA.

Testing of 2MiB transfers demonstrated similar uniformity in latency to the shorter 100KiB transfers, with data transmitted over WiFi subjected to median delays of between 72.96ms (CUBIC), and 76.93ms (Illinois). Results from the same test with data transmitted via HSDPA, shown in Fig 5, show a similar uniformity in median RTT. However, network traffic managed by YeAH — and to a lesser extent, CUBIC — experiences a much larger variation in delay than the other mechanisms.

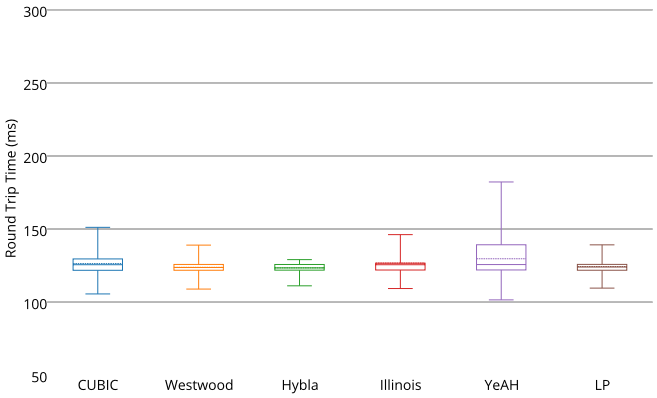

Fig. 5: Latency for 2MiB transfers over a single TCP connection via HSDPA.

Interestingly, median RTT experienced by traffic for all mechanisms transmitted via an LTE network was significantly more uniform than any other result recorded in this study. Given the number of samples considered, as well as the conditions for excluding data points, these results are unlikely to be the result of some random short-term anomaly. A similar level of uniformity was observed in two other sets of results for tests conducted as part of this study.

Some differences in the latency achieved by different congestion control mechanisms become more apparent over longer transfers. In tests involving the transfer of 10MiB over LTE, traffic managed by CUBIC experienced the least delay, achieving a median result of 101.63ms. This delay also appears relatively consistent, with traffic experiencing a variation in RTTs of of 35.65ms, and an interquartile range of 4.14ms.

Traffic transferred while LP was applied experienced the second lowest median delay. This delay was slightly less consistent than that experienced by YeAH.

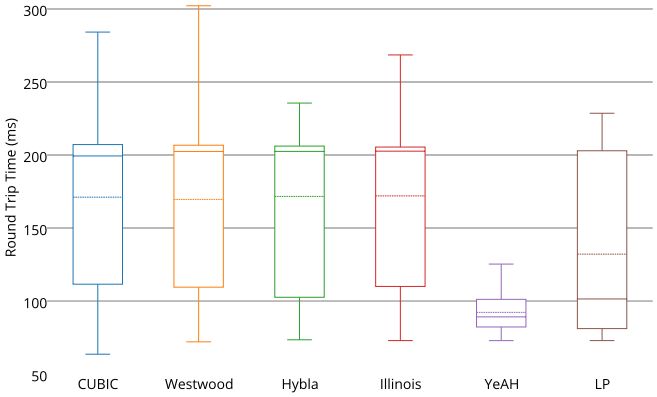

Fig. 6: Latency for 10MiB transfers over a single TCP connection via WiFi.

Traffic transmitted via WiFi experienced the greatest delay of the three wireless networking technologies considered in this study. As shown in Fig 6, with the exception of YeAH and LP, traffic transmitted over a WiFi network was subject to a median delay of 199.37ms–202.66ms. YeAH also achieved much lower variation in delay.

V. Discussion

As new congestion control mechanisms are developed, their performance relative to existing mechanisms needs to be understood. This study evaluated existing TCP congestion control mechanisms using an approach designed to be representative of real world usage of wireless networks in order to attempt to identify one mechanism that would provide significant performance benefits over CUBIC.

The key findings of this study are as follows:

- YeAH demonstrates a mean increase to throughput of 3%–5% over CUBIC across all tests, with as much as a 17% improvement for transfers over LTE.
- Throughput for the six congestion control mechanisms tested was found to be largely consistent.
- CUBIC often achieved slightly lower throughput than the other mechanisms being evaluated.
- LP achieved similar throughput to the other mechanisms evaluated, with the exception of 10MiB file transfers over LTE.
- Latency was relatively consistent across all congestion control mechanisms.

The findings of this study are discussed in greater detail below.

A. Throughput

While prior studies evaluating TCP congestion control mechanisms have shown some variation in results, these studies have found little difference in the throughput achievable through use of different congestion control mechanisms [17], [19], [7], [8]. The results of this study are broadly supportive of these findings, in that the six mechanisms included in this evaluation achieved similar throughput in each test over WiFi, HSDPA, and LTE.

De Cicco & Mascolo [8] found a slight difference in the goodput achievable by CUBIC and Westwood+ over 3G networks. This difference between CUBIC and Westwood+ favoured CUBIC — CUBIC was recorded to have achieved goodput of 1.44Mbps, compared to 1.41Mbps for Westwood+. These findings were supported by Callegari *et al.* [7], who found a 0.2Mbps difference in throughput favouring CUBIC. A study by Adami *et al.* [19] also observed small differences in the throughput achieved by both mechanisms over wireless links, broadly supporting the findings of other studies [7], [8]. However, results collected by Adami *et al.* [19] found that CUBIC achieved lower throughput than Westwood. The results of the current study support the findings of Adami *et al.* [19], with CUBIC achieving lower throughput than Westwood in most tests.

The prior evaluation conducted by Callegari *et al.* [7] also found that Illinois and YeAH achieved noticeably lower throughput compared to the other mechanisms tested in this study. However, this finding is inconsistent with the throughput observed in the course of the current study. The throughput results presented in Section IV-A depict minimal differences between the throughput achieved by these mechanisms in tests considering a single TCP connection.

With the notable exception of 10MiB transfers via the Telstra LTE network, LP achieved similar or better throughput when compared to the other six mechanisms included in this study. The finding is consistent with that of Callegari *et al.* [7], which found that LP achieved the greatest throughput over a WiFi link of all the mechanisms included in this study. Adami *et al.* [19] observed that LP achieved the fourth highest throughput for the mechanisms included in this study, ahead of Illinois and Hybla. LP's competitive performance was surprising given that LP was intended to only make use of excess network capacity [12].

B. Latency

In addition to the limited differences observed in the achieved throughput, this study found only small differences in the latency achieved by the six mechanisms tested. The notable exception to this was during tests involving the transfer of 10MiB via WiFi, in which YeAH and LP exhibited significantly lower median latency to the other mechanisms. However, these results were not observed in other tests.

As previously noted, few studies have utilised latency as a metric for evaluating the performance of congestion control mechanisms. Of these studies, none examined two or more of the mechanisms included in this study. However, studies that examined latency found minimal differences between the latency achieved by different mechanisms [6], [8], [14], supporting the findings of this study.

VI. Conclusions and Future Work

In 2014, Internet-bound traffic originating from a wireless device exceeded 50% and is expected to continue increasing [3]. Therefore, performance and efficiency improvements for TCP over wireless networks are needed in response. This study aimed to evaluate existing TCP congestion control mechanisms using a comprehensive and reproducible methodology designed to be representative of real world usage of wireless networks. In doing so, it sought to identify one such mechanism that would provide noticeable performance benefits over CUBIC.

Throughput for the six congestion control mechanisms evaluated as part of this study was relatively uniform within each test. A lack of differentiation in the throughput achieved by different congestion control mechanisms has also been observed in prior studies [17], [19], [7], [8].

In addition, the results of this study demonstrated only small differences in the latency achieved by the mechanisms tested. This finding was broadly supported by those of previous evaluations of congestion control mechanisms [6], [8], [14].

The results of this study demonstrated that YeAH exhibits a performance improvement of approximately 3% over CUBIC, with no penalty to — and in some cases, lower — latency. As such, applications that require or would benefit from any performance improvement could implement YeAH as the TCP congestion control mechanism on their servers.

However, based on the findings of this study, as well as previous studies evaluating the performance of congestion control mechanisms, it is unlikely that significant improvements will be achieved by the widespread adoption of existing congestion control mechanisms. As such, it is the conclusion of this study that use of alternate congestion control mechanisms would not provide noticeable improvements in performance in most applications.

Existing mechanisms are constrained by the necessity of maintaining existing TCP semantics, making incremental modifications to the original congestion control algorithm developed by Jacobson [1]. However, the lack of performance improvements demonstrated by existing mechanisms suggests that more extensive changes to TCP may be required for any significant benefits to be realised.

Recent studies into the development of new congestion control mechanisms have proposed larger changes. One such modification, Remy [5], has been demonstrated to significantly improve performance over wireless networks in preliminary evaluations. While the inclusion of Remy in the evaluation carried out as part of this study was considered, no Linux implementation was available at the time this study took place. As such, future evaluations should assess the availability of new mechanisms for testing.

Acknowledgement

The authors acknowledge the support of Amazon, Inc through the AWS Research Grant program. This grant provided the EC2 instances used in this study, allowing testing to be carried out on infrastructure used by a wide variety of web-based services.

References

[1] V. Jacobson, "Congestion avoidance and control," in *Symposium proceedings on Communications architectures and protocols*, SIGCOMM '88, (New York, NY, USA), pp. 314–329, ACM, 1988.

[2] H. Balakrishnan, S. Srinivasan, E. Amir, and R. H. Katz, "Improving TCP/IP performance over wireless networks," in *International Conference on Mobile Computing and Networking: Proceedings of the 1 st annual international conference on Mobile computing and networking*, vol. 13, pp. 2–11, 1995.

[3] Cisco Systems, Inc., "The zettabyte era: Trends and analysis," May 2015.

[4] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "TCP westwood: Bandwidth estimation for enhanced transport over wireless links," in *Proceedings of the 7th annual international conference on Mobile computing and networking*, MobiCom '01, (New York, NY, USA), pp. 287–297, ACM, 2001.

[5] K. Winstein and H. Balakrishnan, "TCP ex machina: Computer-generated congestion control," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, (New York, NY, USA), pp. 123–134, ACM, 2013.

[6] L. De Cicco and S. Mascolo, "TCP congestion control over 3G communication systems: An experimental evaluation of new reno, BIC and westwood+," in *Next Generation Teletraffic and Wired/Wireless Advanced Networking* (Y. Koucheryavy, J. Harju, and A. Sayenko, eds.), vol. 4712, pp. 73–85, Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[7] C. Callegari, S. Giordano, M. Pagano, and T. Pepe, "Behavior analysis of TCP linux variants," *Computer Networks*, vol. 56, pp. 462–476, Jan. 2012.

[8] L. De Cicco and S. Mascolo, "TCP congestion control over HSDPA: an experimental evaluation," arXiv e-print 1212.1621, Dec. 2012.

[9] M. Allman and A. Falk, "On the effective evaluation of TCP," *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 5, pp. 59–70, 1999.

[10] S. Floyd and E. Kohler, "Internet research needs better models," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 29–34, 2003.

[11] 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.

[12] A. Kuzmanovic and E. Knightly, "TCP-LP: low-priority service via end-point congestion control," *IEEE/ACM Transactions on Networking*, vol. 14, no. 4, pp. 739–752, 2006.

[13] D. Rossi, C. Testa, S. Valenti, and L. Muscariello, "LEDBAT: The new BitTorrent congestion control protocol," in *2010 Proceedings of 19th International Conference on Computer Communications and Networks (ICCCN)*, pp. 1–6, 2010.

[14] M. Nirmala and R. Pujeri, "Performance of TCP vegas, bic and reno congestion control algorithms on iridium satellite constellations," *International Journal of Computer Network and Information Security*, vol. 4, pp. 40–49, Nov. 2012.

[15] S. Trivedi, S. Jaiswal, R. Kumar, and S. Rao, "Comparative performance evaluation of TCP hybla and TCP cubic for satellite communication under low error conditions," in *2010 IEEE 4th International Conference on Internet Multimedia Services Architecture and Application(IMSAA)*, pp. 1–5, 2010.

[16] J. Chicco, D. Collange, and A. Blanc, "Simulation study of new TCP variants," in *2010 IEEE Symposium on Computers and Communications (ISCC)*, pp. 50–55, 2010.

[17] C. Caini, R. Firrincieli, D. Lacamera, T. De Cola, M. Marchese, C. Marcondes, M. Sanadidi, and M. Gerla, "TCP live experiments on a real GEO satellite testbed," in *12th IEEE Symposium on Computers and Communications, 2007. ISCC 2007*, pp. 523–529, 2007.

[18] S. Floyd, "Metrics for the evaluation of congestion control mechanisms," RFC 5166, RFC Editor, Mar. 2008.

[19] D. Adami, C. Callegari, S. Giordano, M. Pagano, and T. Pepe, "A behavioral study of TCP linux variants over satellite networks," in *2011 IEEE/IPSJ 11th International Symposium on Applications and the Internet (SAINT)*, pp. 474–479, 2011.

[20] S. Souders, "HTTP archive," Oct. 2010.

[21] C. S. Parke, *Essential first steps to data analysis: scenario-based examples using SPSS*. Thousand Oaks, California: SAGE Publications, Inc, 2013.