TCP Congestion Control Macroscopic Behaviour for Combinations of Source and Router Algorithms

Mulalo Dzivhani[1,2], Dumisa Ngwenya[3,4], Moshe Masonta[1], Khmaies Ouahada[2]
Council for Scientific and Industrial Research (CSIR), Meraka Institute[1], Pretoria
Faculty of Engineering and Built Environment, University of Johannesburg[2], Johannesburg
SENTECH[3], Johannesburg
University of Pretoria[4], Pretoria
South Africa
MDzivhani@csir.co.za[1, 2], ngwenyaDW@sentech.co.za[3, 4], MMasonta@csir.co.za[1], kouahada@uj.ac.za[2]

***Abstract*—The network side of Transmission Control Protocol (TCP) congestion control is normally considered a black-box in performance analysis. However, the overall performance of TCP/IP networks is affected by selection of congestion control mechanisms implemented at the source nodes as well as those implemented at the routers. The paper presents an evaluation of macroscopic behaviour of TCP for various combinations of source algorithms and router algorithms using a Dumbbell topology. In particular, we are interested in the throughput and fairness index. TCP New Reno and TCP Cubic were selected for source nodes. Packet First-in-First-out (PFIFO) and Controlled Delay (CoDel) mechanisms were selected for routers. The results show that TCP New Reno performs well, in terms of throughput, in a low BDP scenario. However, as expected in high BDP scenario, TCP New Reno deteriorates and TCP Cubic performs better. CoDel in the network side further deteriorates TCP New Reno flows in high Bandwidth-Delay Product (BDP) scenario, while considerably improving TCP Cubic. PFIFO deteriorates both TCP Cubic and TCP New Reno in high BDP. Almost in all cases CoDel seems to improve fairness.**

***Index Terms*—TCP Congestion Control, TCP New Reno, TCP Cubic, CoDel, Droptail, Throughput, Fairness.**

I. Introduction

The rapid rise in use of the Internet, due to increase in new computing and communications applications such as video streaming, online gaming, desktop and video conferencing, has resulted in an increase in demand for bandwidth and fast access [1]. TCP is the widely used protocol in the Internet for traffic flow control and congestion control. Therefore, the performance of the Internet depends on how well TCP works. TCP guarantees reliability and in-order delivery of packets in TCP/IP networks [2]. In addition, it implements a congestion-control mechanism, which regulates how fast the transport layer sends data to prevent the sender from overloading the network [3]. TCP congestion control algorithms are typically implemented at the source. Each source uses some congestion control feedback or signal by the network to estimate allowable sending rate. The network is also important in congestion control and network based algorithms are usually implemented in routers to control buffer queues. Most widely used TCP variant over the decades has been TCP New Reno, evolving from TCP Tahoe and TCP Reno. TCP New Reno has proved to be efficient in fixed low speed networks and networks with low BDP. However, as shown by [1], TCP as implemented is showing performance in high-speed wide-area networks, in wireless links and for real-time flows. For this reason, a number of source based algorithms have been proposed in the past two decades. TCP Cubic is one of those that has gained popularity for high speed networks. Congestion control at the sources is not enough to curb congestion. The routers mechanisms are required to manage buffers and queue in the network. Most common router mechanism is DropTail [4] [5] [6]. DropTail is simplistic and passive. Although it is implemented with minimal computation overhead, it provides unsatisfactory performance in highly demanding scenarios [6] [7]. For this reason, a large number of Active Queue Management (AQM) algorithms have been proposed. Among these is the Controlled Delay (CoDel), which addresses the problem of buffer float in high speed networks. The performance of TCP congestion control is depended on both source-based algorithms as well as router-based algorithms. The purpose of this study is to evaluate macroscopic behaviour of TCP/IP networks for combinations of source based and router based algorithms. For source based algorithms we consider TCP New Reno and TCP Cubic, and router based we consider PFIFO and CoDel. The paper looks at the impact of the bandwidth and delay on throughput and fairness for the said combinations. The remaining sections in this paper are structured in the following way. Section II takes a look at the related work or literature review. Section III describes the Source algorithms and the Router-based algorithms looked at in this study. Section IV presents the network configuration, traffic models and congestion scenarios used to carry out the simulations. In Section V, the simulation results are discussed, with Section VI, discussing the conclusion and future work.

II. Related Work

In this section, we provide an overview about the closely related work that was looked at in this work.

Early work on interactions of TCP flows and AQM routers was presented by Misra, Gong and Towsley [8]. The authors in [8] applied their model (MGT model) to TCP flows with Random Early Detection (RED) AQM policy. The MGT model has been used to guide designs of AQM schemes to alleviate

Internet congestion [9]. Recent work has been presented in [9], where the MGT model was modified to cater for heavy traffic conditions and applied to Proportional-Integral (PI), Random Early Marking (REM) and a Robust Active Queue Management Scheme (RaQ).

Performance analysis of CoDel and other AQM schemes with TCP flows has been presented in [10] and [11]. The above work shows that the behaviour of most recently developed AQM schemes, including CoDel, may lead to underutilisation over time. Several modifications of the popular AQMs are presented in the cited work.

Most of the cited work does not consider the performance as a consequence of the combination of source algorithm and router algorithm. In this paper we attempt to provide an experimental analysis that discovers the relationship between end-to-end application performances of TCP congestion avoidance algorithms, as observed by an Internet user.

III. Source Algorithms and Router Algorithms

Source algorithms considered in the paper are TCP New Reno and TCP Cubic. These are classical loss-base mechanisms deployed in almost all important systems [12]. TCP Cubic is currently used by default in Linux kernel [13]. TCP New Reno represents a classical congestion control mechanism for low bandwidth networks and TCP Cubic for high bandwidth networks.

Most loss-based source algorithms, such as TCP New Reno and TCP Cubic, consist of two phases or strategies, slow-start and congestion avoidance phases.

A. Slow start

During slow-start phase the congestion window (cwnd) is doubled every round-trip-time (RTT), resulting in near exponential growth that is, the growth starts slowly and then accelerates with time. The increase profile per packet acknowledgment (per ACK) is given by (1)

$$cwnd_{i+1} = cwnd_i + 1 \tag{1}$$

Which results in near doubling every RTT.

B. Congestion Avoidance for TCP New Reno

The congestion avoidance phase depends on TCP variant used. In TCP New Reno the cwnd is increased by one for every RTT, resulting in a linear growth as shown in (2). The increase profile per packet acknowledgment (per ACK) is given by:

$$cwnd_{i+1} = cwnd_i + \frac{1}{cwnd_i} \tag{2}$$

Which results nearly in one segment every RTT. During congestion avoidance, Cubic uses a cubic window growth function as follows given by (3)

$$cwnd_t = C * (t - K)^3 cwnd_i + cwnd_{max} \tag{3}$$

Where K is given by (4):

$$K = \sqrt[3]{\frac{cwnd_{max} * (1 - \beta)}{C}} \tag{4}$$

$cwnd_{max}$ is the maximum cwnd just before packet loss; β is a coefficient of multiplicative decrease in Fast Recovery; C is a constant determining the aggressiveness of the window growth; t is the elapsed time since packet loss. In Internet routers, AQM is the intelligent drop of network packets inside a buffer associated with a network interface controller (NIC), when that buffer becomes full or gets close to becoming full, often with the larger goal of reducing network congestion. The router algorithms chosen for this study are Controlled Delay (CoDel) and PFIFO (DropTail).

C. CoDel

CoDel queue management algorithm is a scheduling algorithm for the network scheduler that was developed by Van Jacobson and Kathleen Nichols [14]. They developed CoDel to overcome buffer float in network links by setting limits on the delay network packets that suffer due to passing through the buffer being managed by CoDel. When CoDel's estimator comes across a persistent delay which is above the maximum acceptable persistent queue delay, the controller then enters the drop state whereby a packet is dropped early so to maintain the minimum sojourn delay [15].

Another aspect of CoDel, when the delay is high or exceeds the maximum persistent queue delay, CoDel controls the delay while insensitive to round-trip delays link rates and traffic loads [14].

D. PFIFO vs DropTail

PFIFO queueing is the most basic form of congestion control. The frames in this algorithm are queued in First-In, First-Out (FIFO) order and queue build-up can continue until all buffer memory is exhausted which is very like how Tail Drop algorithm handles congestion [16].

In DropTail AQM, once the queue is filled up, the router then begins discarding every additional datagram by dropping the tail of the sequence of packets.

The loss of packets from the drop causes the TCP sender to enter slow-start, which reduces throughput in that TCP session until the sender begins to receive ACKs again and increase the cwnd. In accordance to the regard of how similarly PFIFO works to DropTail, we consider PFIFO as DropTail in our study.

IV. Simulation Methodology

In this section, we present a case study whereby Network Simulator—3 (NS3) [17] is used for realization of two scenarios, which are discussed below in sub-section A and B, and tested using NS3.27. Some modifications were done on NS-3.27 to incorporate one of a number of technologies that have not been incorporated in NS-3. One of the networking technologies that have not been incorporated in NS3 is TCP Cubic, the default TCP congestion control algorithm in the Linux kernel and one of the most widely deployed variants

of TCP. For this study some modifications were done on TCP Cubic's Internet module intended for NS3.26 done by [18] and obtained from [19]. In this study two simulation scenarios were considered, the first simulation environment was carried out using an End-to-End network topology which consisted of a single source node with two routers and a single receiver node. The second simulation scenario consisted of several source and receiver nodes which were connected by two routers in a dumbbell topology.

A. Scenario One: Bandwidth Delay Product

The simulations experimenting on the impact of delay used an End-to-End topology as shown by Fig. 1. The simulation environment consisted of a single flow and other simulation parameters shown in Table I. For each TCP variant and AQM algorithm, several simulations were conducted varying the delays in each simulation and keeping other parameters unchanged. The main attempt of the configuration is to compare TCP New Reno and TCP Cubic's average throughput ratio when both use CoDel and Droptail on the router side with varying the delay to determine the impact the delay has on the network.

Fig. 1. End-to-End topology used for simulating the impact of delay.

Below are the simulation parameters used for setting up the simulation environment for determining the impact of delay in a network

TABLE I
DELAY BANDWIDTH IMPACT PARAMETERS

Parameter	Value
TCP variants	TCP New Reno and TCP Cubic
AQM Algorithm	CoDel and PFIFO DropTail
Link capacity	100Mbps for the source node and 20Mbps for bottleneck
Link delay	0.1µs for the source node
Bottleneck delay	5ms, 10ms, 20ms, 40ms, 80ms
Simulation time	200 sec

The average throughput was calculated and analysed to determine the impact of delay in terms of bandwidth for the connections when different source and router algorithms are used with varying delays.

Average Throughput: TCP throughput, which is the rate that data is successfully delivered over a TCP connection, is an important metric to measure the quality of a network connection [20]. In this study average throughput given by (5) refers to the throughput per unit of time, whereby the average amount of data received is distributed by the receiver per unit time, regardless of whether the data is retransmission or not.

$$T_{avg} = \frac{Rbytes * 8}{t * 1024} \tag{5}$$

where Rbytes is the total bytes received throughout the simulation and t is the per unit time elapsed during the simulation.

B. Scenario Two: Fairness Analysis

In the second simulation environment, performance of TCP New Reno and TCP Cubic with different propagation delays is analysed. The topology used in these experiments is shown by Fig. 2. The simulation consisted of 10 connections or flows where the Fairness of the algorithms was analysed. The simulation parameters as shown in Table II were used to conduct the experiment and all other parameters were kept the same other than the delay which was changed from 5ms to 30ms for each simulation conducted.

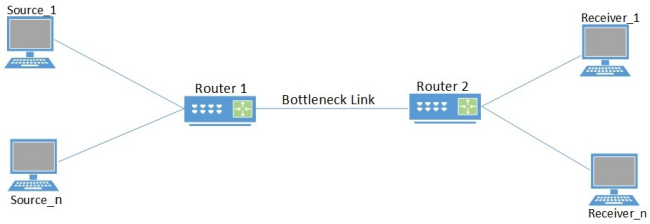

Fig. 2. Dumbbell network topology with a single bottleneck link.

TABLE II
FAIRNESS COMPARISON SIMULATION PARAMETERS

Parameter	Value
TCP Variants	TCP New Reno, TCP Cubic
Flow control algorithm	PFIFO DropTail, CoDel
Link capacity	100Mbps for the source node and 100Mbps for bottleneck
Link delay	0.1µs for the source node
Bottleneck delay	5ms, 30ms
Number of flows	10
Simulation time	200 sec

We simulated the activity of multiple flows with bandwidth sharing between the multiple TCP flows. It was assumed that the windows of the connections will vary in a synchronized manner.

Fairness Index: Fairness refers to the relative performance of several connections under the same TCP variant. To provide a numerical measure for the criterion, we use Jain's Fairness index represented by (6) [21]. Intuitively, each connection or flow in a network should get no more than its demand, the excess if present should be equally shared.

$$f(x_1, x_2, x_3, \ldots x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{(n \sum_{i=1}^{n} x_i)^2} \tag{6}$$

where x_i is the normalized throughput (in Mbps) of the $i-th$ TCP flow and n is the number of connections or flows.

V. RESULTS AND DISCUSSION

In this paper, we have investigated the impact of delay on bandwidth delay product in the first scenario of the simulation with focus on TCP New Reno and TCP Cubic against Droptail

and CoDel. We then investigated the fairness between TCP New Reno and TCP Cubic against CoDel and Droptail in the second scenario of the simulation.

Scenario One: Bandwidth Delay Product

The Network Delay consist of a number of delays where each of these delays contribute to the round trip time (RTT). More application turns causes more data to be sent over the network which strains the bandwidth of each network link and the routing competency of the network devices as can be seen in the following Figures: Fig. 4 where the delay is 40ms and 80ms, Fig. 5 whereby the delay is highest at 80ms and Fig. 6 at delays 40ms and 80ms. TCP Cubic outperforms TCP New Reno, as the delay increases as shown by Fig. 3 and Fig. 4 respectively, the average throughput of TCP New Reno decreases as the delay increases. TCP Cubic ensures that, its throughput is not lower than the throughput of the standard TCP New Reno, which is done by enforcing the calculated value of TCP New Reno's congestion window whenever the maximum congestion window of TCP Cubic is going lower than New Reno's.

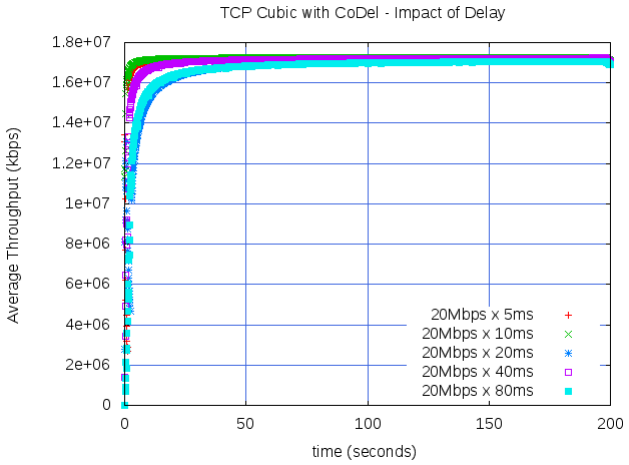

Fig. 3. Impact of delay on TCP Cubic against CoDel.

Scenarion Two: Fairness Analysis

Fairness Analysis: TCP Cubic Fig. 7 shows the fairness index of TCP Cubic and CoDel which shows a fair share of the bandwidth as TCP Cubic is designed to perform well in high bandwidth network. CoDel in this case seems to improve the bandwidth as the fairness can be seen gradually improving from 5 – 40 seconds of the simulation and becomes constant throughout the simulation.

The increase in delay can be seen as a factor for deteriorating the fairness index as can be seen in Fig. 8. Although the slow start in the performance of TCP Cubic, its window size quickly ramps up to the size before the last congestion event, hence the convex growth in Fig. 8, as Cubic now probes for more bandwidth, slowly at first then very rapidly. Only after the network has stabilized then TCP Cubic can look for more

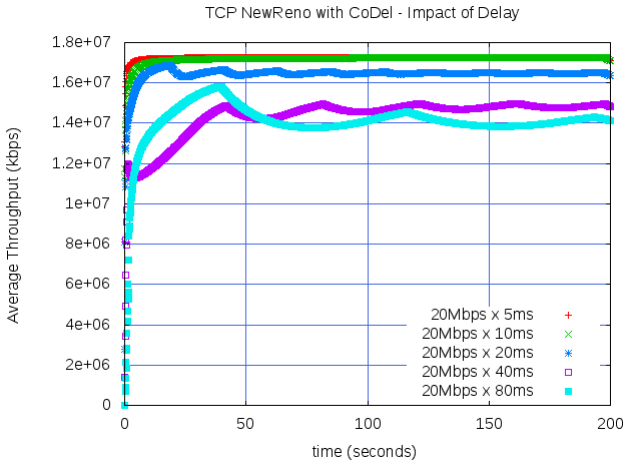

Fig. 4. Impact of delay on TCP New Reno against CoDel.

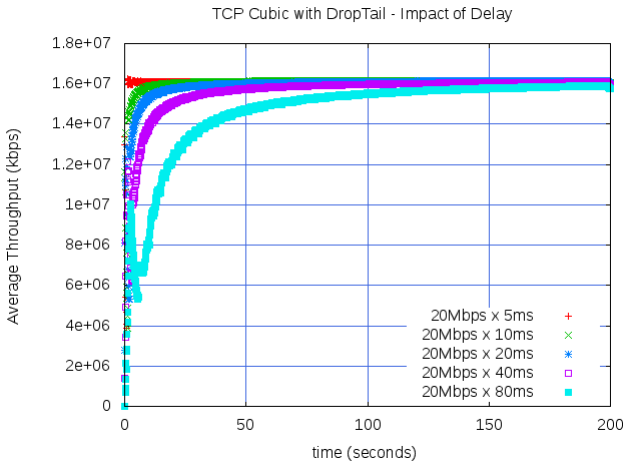

Fig. 5. Impact of delay on TCP Cubic against Droptail.

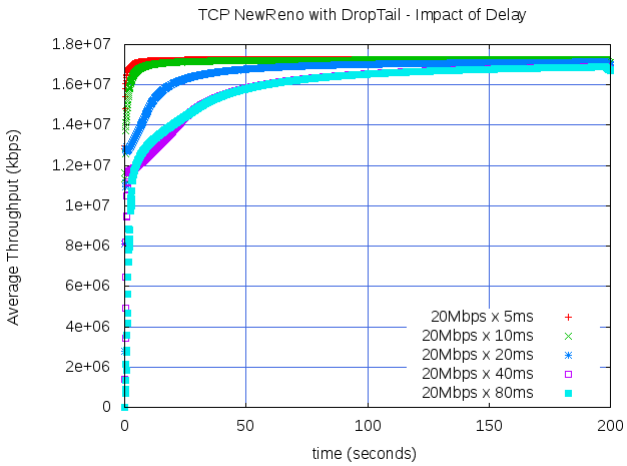

Fig. 6. Impact of delay on TCP New Reno against Droptail.

bandwidth. In Fig. 9, TCP Cubic suffers due to the router algorithm implemented, it can be observed from Fig. 9 that Droptail weakens the performance of the network as compared to TCP Cubic with CoDel were all parameters were similar except for the router based algorithm. However, TCP Cubic improves the networks performance for a short period of time 5- 40 seconds where after the 40 seconds it does not recover any more.

Fairness index of Fig. 10 shows us that although a high bandwidth algorithm is used on the source side, the router based algorithm can reduce the performance of the network especially in cases where the delay is high. In short the serious unfairness is caused by the difference of the congestion control algorithms of the two algorithms and Droptail cannot improve performance when used in high bandwidth delay networks.

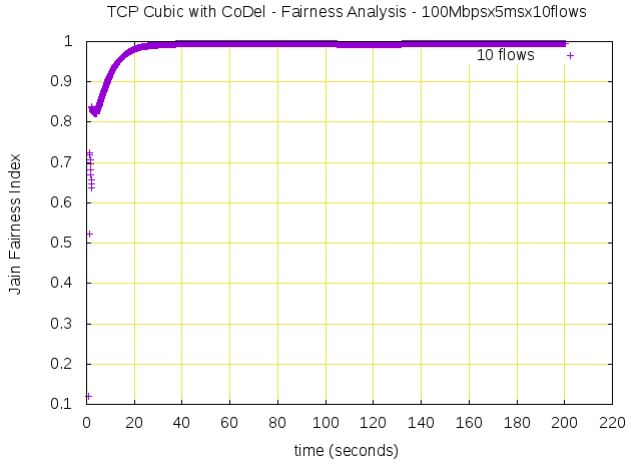

Fig. 7. Fairness Index of TCP Cubic with CoDel having a 5ms delay on the bottleneck link.

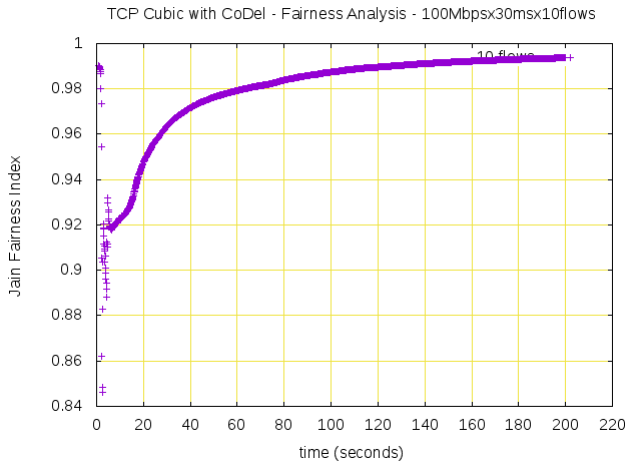

Fig. 8. Fairness Index of TCP Cubic with CoDel having a 30ms delay on the bottleneck link.

Fairness Analysis: TCP New Reno TCP New Reno is a low bandwidth network algorithm and as observed in Fig.

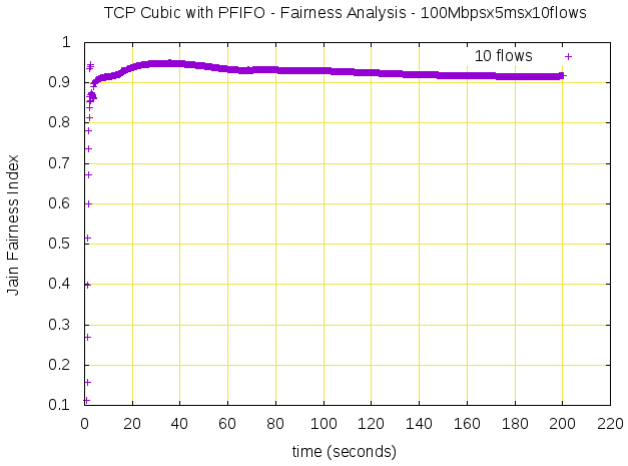

Fig. 9. Fairness Index of TCP Cubic with Droptail having a 5ms delay on the bottleneck link.

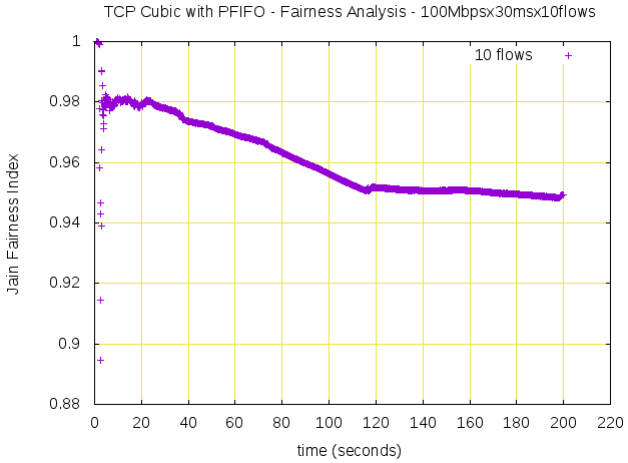

Fig. 10. Fairness Index of TCP Cubic with Droptail having a 30ms delay on the bottleneck link.

11, CoDel improves the utilization of the bandwidth in the network and thus we have a rational share of the bandwidth in the network. In Fig. 12, the fairness keeps weakening and improving and the performance of the network can be blamed on the high delay as also observed in Fig. 8. The router based algorithm however helps the source based algorithm in improving the overall share of resources in the network. Fig. 13 displays a decent performance of the network as the fairness index improves greatly after the simulation has just started, the network performance remains high and the low delay doesn't take much toll on the network and algorithms implemented on both source and router sides. When comparing Fig. 13 and Fig. 14, it can be observed that Droptail deteriorated the networks performance with the help of the delay. When the queue filled up, a number packets were dropped and that caused the degrading of the performance as packets had to be retransmitted again. While the previous sequence of packets has not yet been acknowledged, another sequence

of packets are retransmitted causing more congestion and degrading of the networks performance. In all cases CoDel seems to improve fairness as can be observed on Fig. 7, Fig. 8, Fig. 11 and Fig. 12. Performance of TCP New Reno can be seen deteriorating when delay has been increased to 30ms on Fig. 12 and Fig. 14. Whereas performance of TCP Cubic deteriorates when used with Droptail as observed in Fig. 8 and performance of TCP New Reno when used with Droptail under a high delay further weakens the fairness in the network as observed in Fig. 14.

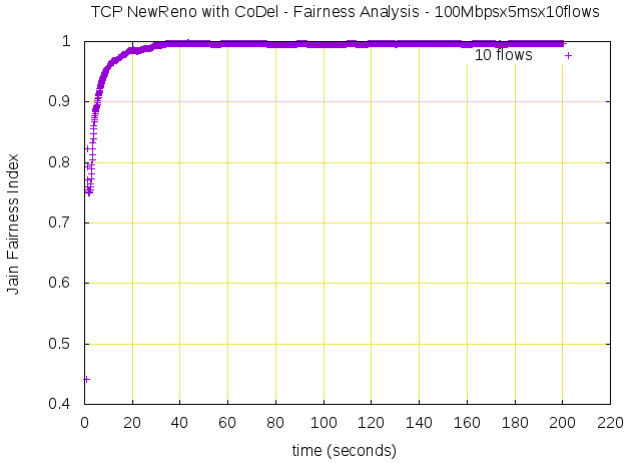

Fig. 11. Fairness Index of TCP New Reno with CoDel having a 5ms delay on the bottleneck link.

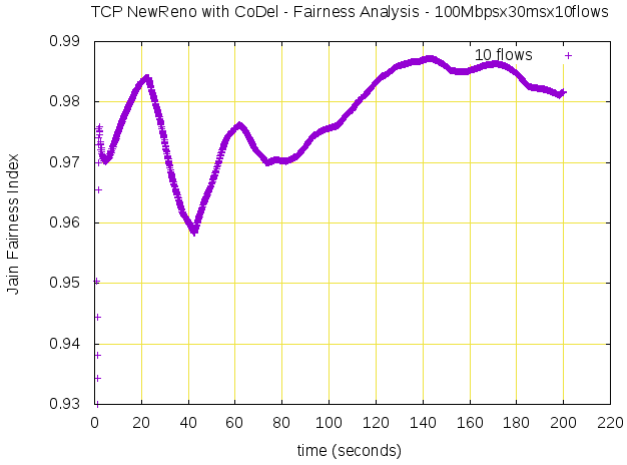

Fig. 12. Fairness Index of TCP New Reno with CoDel having a 30ms delay on the bottleneck link.

VI. Conclusion and Future Work

In regards to our simulation results, it can be concluded that TCP New Reno performs well, in terms of throughput, in low BDP scenario. However, as expected in high BDP scenario, TCP New Reno deteriorates and TCP Cubic performs better. CoDel in the network side further deteriorates TCP New Reno

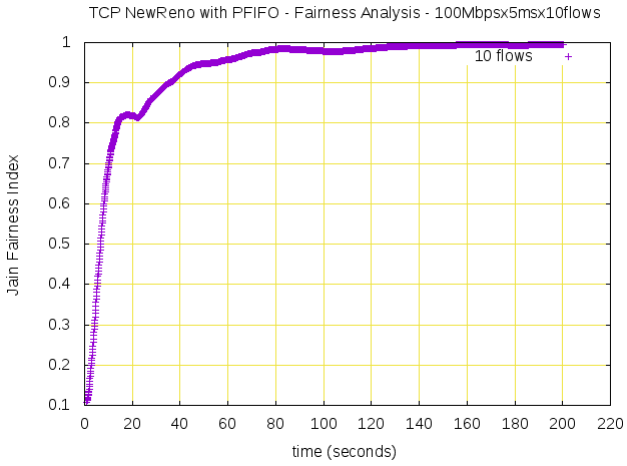

Fig. 13. Fairness Index of TCP New Reno with Droptail having a 5ms delay on the bottleneck link.

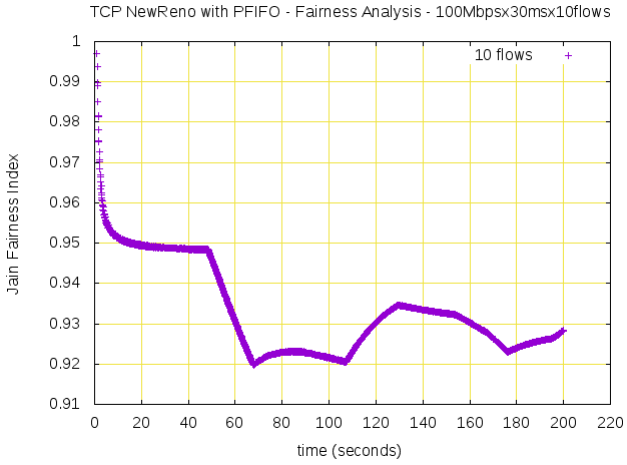

Fig. 14. Fairness Index of TCP New Reno with Droptail having a 30ms delay on the bottleneck link.

flows in high BDP scenario, while considerably improving TCP Cubic. Droptail deteriorates both TCP Cubic and TCP New Reno in high BDP. Almost in all cases CoDel seems to improve fairness.

As the delay varies over time due to delay variations of router queues and congestion avoidance effects, it deteriorates the performance of the network and that lowers the performance of the network.

The performance of TCP New Reno in terms of throughput is affected by high delay. TCP Cubic is an improvement to TCP New Reno and improves TCP throughput and fairness in high BDP networks. On the network side, Droptail introduces delay as a result of increase in buffer size, which adversely affects the performance. CoDel is an active queue management system implemented in the router to keep the delay low.

For further work, more congestion control algorithms can be added and studied under these conditions to have a more insight as to whether other high speed congestion control al-

gorithms will maintain the same characteristics as TCP Cubic and determine if they will behave well or not. Furthermore, other AQM can be considered which will be implemented on the network side.

REFERENCES

[1] D. Papadimitriou, M. Welzl, M. Scharf, and B. Briscoe, "Open research issues in internet congestion control, rfc 6077," Tech. Rep., 2011, [Accessed: Jan. 6, 2018].

[2] J. Postel, "Rfc 793: Transmission control protocol, september 1981," *Status: Standard*, vol. 88, 2003.

[3] E. Aboelela, "2009" [Accessed 2016-02-01]. [Online]. Available: http://www.vlebooks.com/vleweb/product/openreader?id=noneisbn=9780123852113uid=none

[4] A. Ceco, N. Nosovic, and K. Bradic, "Performance comparison of active queue management algorithms," in *2012 20th Telecommunications Forum (TELFOR)*, Nov 2012, pp. 230–232.

[5] S. Patel, P. Gupta, and G. Singh, "Performance measure of drop tail and red algorithm," in *2010 2nd International Conference on Electronic Computer Technology*, May 2010, pp. 35–38.

[6] C. Brandauer, G. Iannaccone, C. Diot, T. Ziegler, S. Fdida, and M. May, "Comparison of tail drop and active queue management performance for bulk-data and web-like internet traffic," in *Proceedings. Sixth IEEE Symposium on Computers and Communications*, 2001, pp. 122–129.

[7] "White Paper on TCP and Queue Management," Tech. Rep., 2008, [Accessed: Jan. 07, 2018]. [Online]. Available: https://pdfs.semanticscholar.org/b971/ca90f89a77a4557c1e7eb9b708b506eff2e8.pdf

[8] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of aqm routers supporting tcp flows with an application to red," *SIGCOMM Comput. Commun. Rev.*, vol. 30, no. 4, pp. 151–160, Aug. 2000. [Online]. Available: http://doi.acm.org/10.1145/347057.347421

[9] Q. Xu, F. Li, J. Sun, and M. Zukerman, "A new tcp/aqm system analysis," *Journal of Network and Computer Applications*, vol. 57, pp. 43–60, 2015.

[10] F. Schwarzkopf, S. Veith, and M. Menth, "Performance analysis of codel and pie for saturated tcp sources," in *2016 28th International Teletraffic Congress (ITC 28)*, vol. 01, Sept 2016, pp. 175–183.

[11] T. Høiland-Jørgensen, P. Hurtig, and A. Brunstrom, "The good, the bad and the wifi: Modern aqms in a residential setting," *Computer Networks*, vol. 89, pp. 90–106, 2015.

[12] I. Abdeljaouad, H. Rachidi, S. Fernandes, and A. Karmouch, "Performance analysis of modern tcp variants: A comparison of cubic, compound and new reno," in *2010 25th Biennial Symposium on Communications*, May 2010, pp. 80–83.

[13] "Linux tuning: Expert guide," [Accessed: Jan 06, 2018]. [Online]. Available: https://fasterdata.es.net/host-tuning/linux/expert

[14] K. Nichols and V. Jacobson, "Controlling queue delay," *Queue*, vol. 10, no. 5, pp. 20:20–20:34, May 2012. [Online]. Available: http://doi.acm.org/10.1145/2208917.2209336

[15] [Accessed: Jan. 07, 2018]. [Online]. Available: https://tools.ietf.org/id/draft-ietf-aqm-codel-08.html

[16] "Tail drop algorithm," [Accessed: Jan. 08, 2018]. [Online]. Available: http://www.brocade.com/content/html/en/configuration-guide/$\mathrm{NOS}_6 00_L AYER2/GUID-3B53320A-94F4-40FA-920B-054342FBB091.html$

[17] "Ns-3," [Accessed: Dec. 10, 2017]. [Online]. Available: https://www.nsnam.org/ns-3-27/

[18] B. Levasseur, M. Claypool, and R. Kinicki, "A tcp cubic implementation in ns-3," in *Proceedings of the 2014 Workshop on Ns-3*, ser. WNS3 '14. New York, NY, USA: ACM, 2014, pp. 3:1–3:8. [Online]. Available: http://doi.acm.org/10.1145/2630777.2630780

[19] ——, "Cubic implementation download," [Accessed: Nov. 17, 2017]. [Online]. Available: http://perform.wpi.edu/downloads/cubic

[20] "Throughput," [Accessed: Jan. 08, 2018]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Throughputoldid=815171739

[21] R. K. Jain, D.-M. W. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination," *Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA*, 1984.