Research Article

Optimized TCP Congestion Control Over a Wired Network

Jehan Dastagir, Muhammad Amir*, Bilal Ur Rehman, Shahid Hameed and Majad Ashraf

Department of Electrical Engineering, University of Engineering and Technology, Peshawar, Khyber Pakhtunkhwa, Pakistan.

Abstract: Congestion control in networks is prioritized for maintaining high and error free data rates. The work in this paper directly investigates possible attainment of optimized TCP congestion control over wired networks through a proposed modification done to one of the existing TCP protocols. Firstly, the proposed approach looks into various queue limits e.g. Drop Tail and RED. Evidently, a comparison of these two queue limits vividly demonstrates that RED demonstrates a better performance when it comes to attainment of optimized TCP congestion control. This investigation is naturally followed by simulating different TCP versions such as TCP Tahoe, TCP Sack, TCP NewReno, TCP Reno and TCP Westwood for checking and comparing their effective resource utilization such as network Throughputs, comparative bandwidths, retransmission rates and window sizes. Simulations in this regard were carried through using NS-2 (Network Simulator-2) simulation software. From simulation results, as evident in this paper, TCP Westwood was found to be the best candidate for innovation out of the mentioned flavors based on its output performance. The most active objective of the proposed research presented in this paper was to modify the already chosen TCP Westwood protocol and to come up with a new variant of it proposed here namely as "TCP New-Westwood" for providing a higher degree of TCP congestion control.

Received: March 11, 2021; **Accepted:** June 20, 2021; **Published:** June 28, 2021
***Correspondence:** Muhammad Amir, Department of Electrical Engineering, University of Engineering and Technology, Peshawar, Khyber Pakhtunkhwa, Pakistan; **Email:** amir@nwfpuet.edu.pk
Citation: Dastagir, J., M. Amir, B.U. Rehman, S. Hameed and M. Ashraf. 2021. Optimized TCP congestion control over a wired network. *Journal of Engineering and Applied Sciences,* 40(1): 8-14.
DOI: https://dx.doi.org/10.17582/journal.jeas/40.1.8.14
Keywords: TCP, Congestion control, Queuing algorithms, New-westwood, NS-2

Introduction

This research can be traced back to Van Jacobson whom in 1988 introduced the TCP congestion control in internet (Wischik *et al*., 2011). Internet was suffering from congestion problems at that time due to which a need was felt for such a protocol which could avoid traffic congestions in networks (Chesson, 2010). Due to slow start, congestion avoidance and fast retransmit, TCP Tahoe brought much improvement in computer networking (Ghassan *et al*., 2012). By introducing new versions of TCP, congestion control is improved further and further with the passage of time. Just two years after implementation, Jacobson modified the TCP Tahoe to TCP Reno by adding through the algorithm fast recovery with fast retransmit (Tom *et al*., 2012). Ghassan in 2012 suggested a slightly modern version of TCP named as TCP Vegas with a difference in basic congestion avoidance algorithm from that of TCP Reno (Ghassan *et al*., 2011). Another modified form of TCP Reno is the TCP Sack that is capable of addition of selective acknowledgements to TCP (Waghmare *et al*., 2011). For the sack of clarity and as a consequence of this research, it is proposed that simple modifications to bandwidth estimation during congestion avoidance phase of TCP Westwood considerably improves network's congestion control as well as metrics such as Throughput and Goodput. Initially, the research presented in this paper examined various congestion control algorithms inside TCPs such as Vegas, Reno, Sack and Westwood in relation to different network parameters as described categorically in the following sections and sub-sections. For example, it will check the TCP stack for:

The measurement of use of network bandwidth

Bandwidth is the maximum capacity measured in bits per second for available or consumed information traveling through a channel. The available bandwidth has a standing position in congestion control. Several TCP flavors persistently estimate and measure bandwidth all over the communication. Network Bandwidth might be tangled with Throughput or Goodput. Throughput is the amount of data that is actually transmitted through a channel. For example, a channel with *X* bits per second data rate may not transmit all the data at *X* bits per second due to the fact that channel width can be affected i.e. narrowed by different overheads such as the protocol's algorithm stack, data encryption overhead and induced user defined latency.

Changes in window size

The TCP window size is the maximum amount of data (in byte) that a receiver device willing to receive at any point in time. TCP window size is a key feature in network troubleshooting or executing an application model. Sliding window (windowing) is used by Transmission Control Protocol (TCP) to control the flow rate of data between two different hosts.

A sending host always see to the acknowledgement of receiver device, so that, it can change the window size according to the receiver demand. TCP always desires to approve all the transmitted data by acknowledgment of the receiver and practices sliding window technique to reduce the probability of packet drop.

Queue algorithm effects

Various ways for improving fairness between different delay connections exist. Here, the effect of different queuing algorithms is concentrated, i.e., Random Early Detection (RED), Droptail (First come, first service) (Partha and Dovrolis, 2010). Simulations show that fairness of the network is dependent on the buffer size parameters (Tianji *et al*., 2011). Droptail can be used to improve the fairness between two connections by increasing buffer size in router. The smaller the buffer size, the faster link occupies the buffer.

RED is used for congestion avoidance (Misra *et al*., 2010). The two main requirements are threshold and maxthreshold. When the average Queue size goes beyond the threshold and remains smaller than maxthreshold, it drops packets with particular probability with respect to queue size.

Congestion control

Congestion is currently a vast area of research and serious issue for network researchers. It occurs if the channel is overloaded, i.e., the number of packets does not handle by the network. Congestion control is the series of techniques to avoid congestion in order to prevent the network from overloading (Charalambos *et al*., 2013). The three basic phases or algorithm and of this process are as follows:

Slow start algorithm

As the name suggests, a slow start algorithm carries out its process in an exponentially increasing way. Slow start algorithm states that new packets should be injected according to acknowledge returned from receiver. In this phase, at first, connection is established with a maximum segment size (MSS) (Borman, 2012).

Initially the data rate is very slow but it increases exponentially. After every acknowledgment received, the window size is increased by one MSS and in such a way the window size becomes doubled for every round trip timer.

Congestion avoidance

Congestion avoidance causes an *additive* increase in congestion window size instead of *exponential* increase each time a segment is received, the window size increases by an integral one time rather than an exponential increment (Wu *et al*., 2013). The process is continued until congestion is deleted. In case of retransmit time out, congestion avoidance assumes it as losses of packets. Consequently, ssthresh (*Slow Start Threshold*) which determines the deactivation of slow start is reduced to half of the current window size and restarts slow start (Mozilla, 2021).

Fast retransmit

Fast retransmit algorithm assumes duplicate acknowledgment as packet loss (Blanton *et al*., 2012). It tells the sender that there was missing somewhere in segments and they were received in an orderly manner. It then helps understand what actual number is required. If it receives three DUPK's (*Duplicate Keywords*) which means and indicates that packets have been lost and retransmission must have occurred. As a consequence, to that, the window size is reduced to half and as already explained before is known as threshold.

Fast recovery

Fast recovery recovers the segment that was missing in fast retransmit. This algorithm improves the performance by permitting high Throughput under moderate congestion (Wischik *et al*., 2011). When three acknowledgements are received it means that there is no congestion and only a segment is missing somewhere. The lost segment is started in buffer and has left the network. TCP does not tend to adopt slow start as the data is still exchanged between sender and receiver.

TCP westwood mechanism

TCP Westwood congestion control algorithm has the ability to estimate bandwidth at the sender side and actually it is the modified version of TCP Reno. Moreover, it has changeable window size in slow start phase while at congestion avoidance it remains constant. TCP Westwood focuses on Bandwidth estimation in order to set a cwnd *(Congestion Window)* (Stackpath, 2021) and to set the ssthresh *(Slow Start Threshold)* after congestion has occurred (Henderson *et al*., 2012). In TCP Westwood, the sender's focus is to calculate a shared bottle neck whose bandwidth changes in accordance to the data sent to the receiver, i.e., bandwidth is equal to data received that was delivered by sender.

End to end bandwidth measurement

The TCP Westwood protocol mainly assesses the consistency of data packets received from sender in a TCP based connection through rate of acknowledgement received back by the sending node (Shimaa and El-Sayed, 2012). This in turn, makes the slow start and congestion control algorithms more efficient and well performed. If the data received is d2 by receiver in t2 time, it means that the source node has received the ACK in time t2, i.e., b2=d2 (t2-t1), where t1 indicates previous acknowledgement (i.e. that constitutes one sample). For bandwidth availability measurement in TCP Westwood, mean or average of such samples is taken.

Setting cwnd and ssthresh in TCP Westwood

Here in this sub-section, setting up of ssthresh and cwnd in TCP Westwood is presented. Let's suppose a sender node in network has successfully done the bandwidth evaluation after signal has been given that a packet has been lost in first step. A gradual increase is well noted in cwnd at the slow start phase while if noted, the congestion avoidance is like TCP Reno as shown in the following descriptive non-commented code listing.

When 3 Duplicate ACKS received, If 3 Duplicate ACKS received, Set slow start thresh = (Bandwidth E*Round TripTimer min) /segment-size; Similarly if cwnd > slow start thresh then made congestion cwnd = slowstart thresh; and congestion avoidance started. When Timeout Occurred: then slow start thresh = (Bandwidth E*Round Trip Timer min) /segment_size; if (slow start thresh < 2) slow start thresh =2; end if ; congestion win = 1; end if slow start take place.

Materials and Methods

Modified westwood mechanism (New-Westwood)

As known, TCP Westwood is the modification of TCP New Reno only towards the sender side. The purpose of this modification is to have a better handling of large bandwidth delay products. Now in TCP Westwood, ack packets received help in order to better control the congestion control parameters which are (a) slow start threshold (ssthresh) and (b) the congestion window (cwnd). In order to have better control and performance a new version of TCP Westwood is proposed in this paper i.e. New-Westwood and only the congestion avoidance phase for this protocol is modified.

In order to obtain a new proposed variant mechanism i.e. *New-Westwood*, the congestion avoidance mechanism of TCP Westwood algorithm was modified in a way described through commented code as following and also shown through a flowchart in Figure 1.

Congestion avoidance phase

When slow start become ended cwnd > slow start thresh /slow start, After Acks: Bandwidth Estimated = Bandwidth E and Bandwidth E = BW now; Bandwidth-ratio = BWnow/BW before; If (1.7>Bandwidth-ratio >= 1); cwnd = cwnd + 1/ cwnd //congestion avoidance phase If (Bandwidth-ratio >= 1.7) ; cwnd = cwnd + 2/ cwnd; Else if (Bandwidth-ratio < 1), cwnd = cwnd + 0 up to (3 Duplicate ACKS or timeout).

Note: Westwood cannot estimate BW current more than 1.7BW previous. Beyond this range the simulation results remain constant.

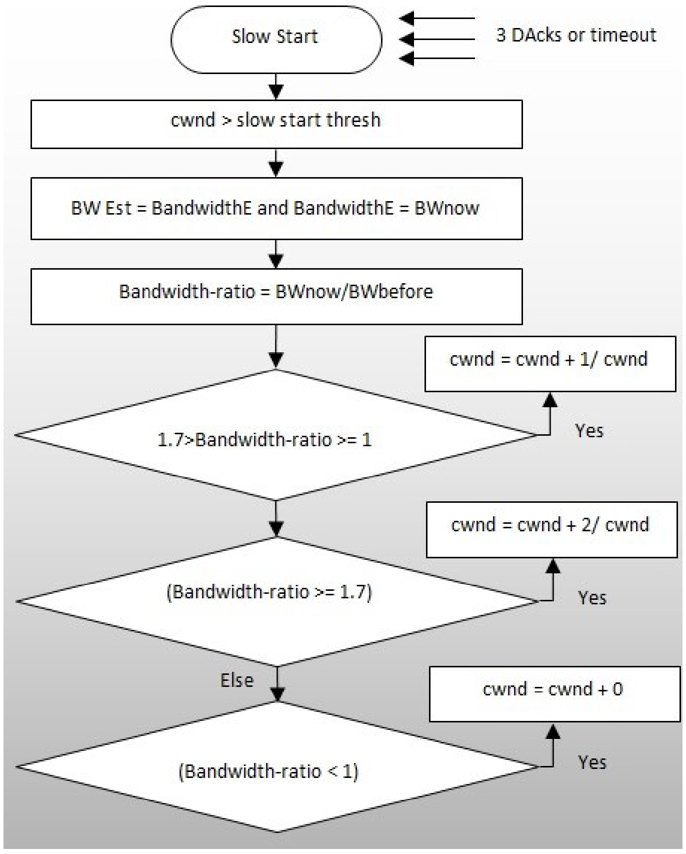

Figure 1: *Flowchart for congestion avoidance phase.*

The results shown in Figures 3 and 4 later also clearly depict that the above modification provides better results as compared to other variants of TCP when analysis is done for queuing algorithms and Throughput.

Performance evaluation

In this research, performance of the proposed modified algorithm i.e. New-Westwood against other existing network protocols such as Westwood, NewReno, Sack, Reno and Tahoe was evaluated using parameters such as Throughput, Network Delay, Packet Loss, Congestion Window and Advertised Window. Such parameters are commonly described as: (1) Throughput is the amount of packets delivered by sender and admitted by receiver in bits per second (bps), (2) the time used (measured in microseconds) for a bit of data to move from one node to another is called Network Delay, (3) Packet Loss is generally number of packets lost as a result of congestion, (4) the window imposed by sender which prevents the routers or switchs from congestion and overloading is termed as Congestion Window while (5) is the Advertised Window referring to the data flow control imposed by any receiver inside the network.

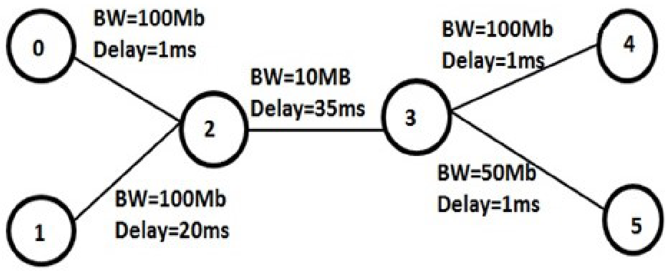

Figure 2: *Network topology.*

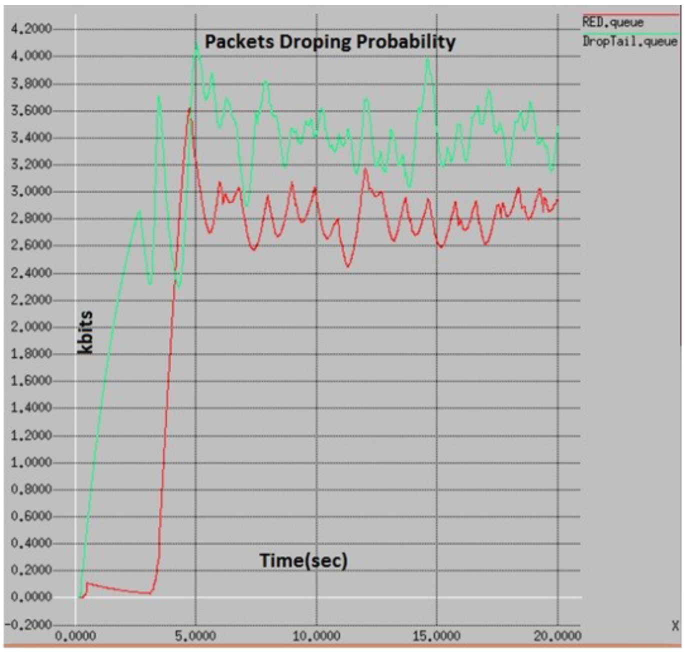

Figure 3: *Packet dropping probability of RED and droptail.*

Simulation setup

The simulation setup used here for performance evaluation evaluates the working behavior of various protocols in computer networking. The considered network topology shown in Figure 2 consists of 6 nodes and 5 TCP connections. Node_0 in Figure 2 is source node while Node_4 is designated as sink node.

Results and Discussion

Simulation results

Results obtained from simulating the setup shown in

Figure 2 are elaborated as following.

Queue algorithm effects

The influence of different queue algorithms on congestion is also investigated through this proposed research. Simulations depicted in Figure 3 show us the rate of packets dropping probability and different queue algorithm on behavior of two connections. Droptail can quickly adapt to small buffer size as it needs much volume for just a few packets. While RED queuing algorithm has a threshold level. When the average queue size became larger than (maxthreshold= buffer size), it started dropping packets with a particular probability. This packet dropping probability of RED is less than Droptail as shown in Figure 3.

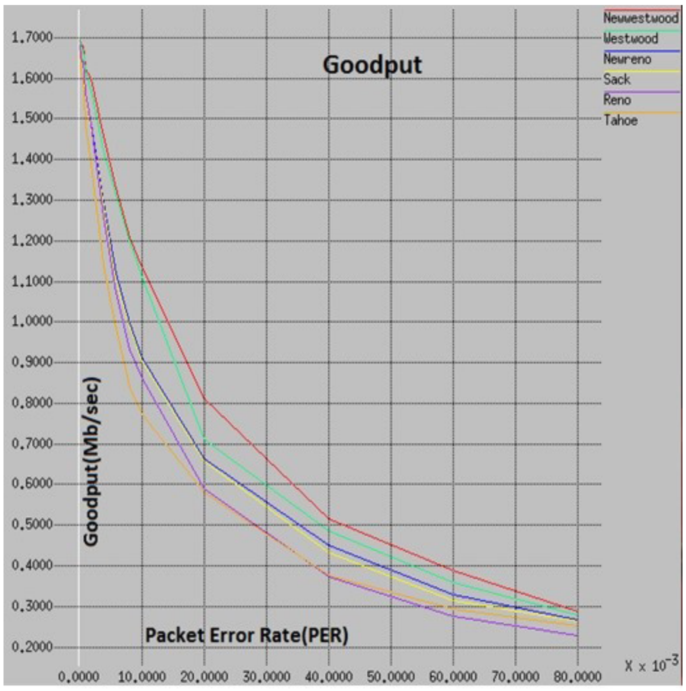

Figure 4: *Performance comparison of throughput.*

New-Westwood

Figure 4 represents the Throughput of TCP Sack, Tahoe, NewReno, Vegas, Sack, Westwood and New-Westwood. According to this simulation, New-Westwood has the highest value of Throughput. It evaluate the network states just by looking to current Bandwidth without concerning the previous network states, Therefore Westwood delivers the same amount of packets irrespective of the network states whether it is higher or lighter whereas TCP New-Westwood depends on states of network, If it is not heavier, it causes an increase in sending packets and resultantly improves the Throughput and for heavier states the rate remains constant.

In order to compare and analyze behaviors of different TCP flavors, an error generator subroutine is inserted into the code. Consequently, the error generator causes packets to drop with a range of probabilities. Goodput levels of different TCP flavors as shown in Figure 5 are the outputs of various probabilistic error-rate values.

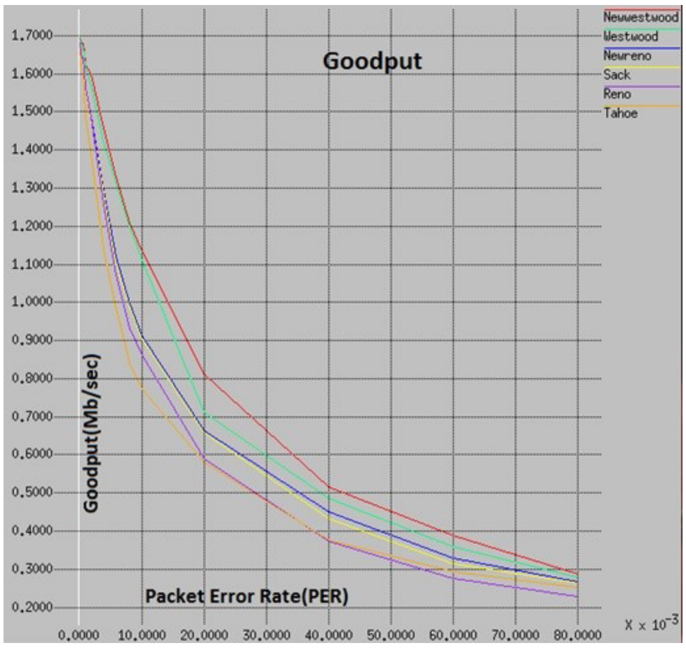

Figure 5: *Comparison of goodputs and packet error rates.*

The performance evaluation shows that TCP New-Westwood has highest Goodput level (Figure 5) using the new bandwidth estimation, while TCP Tahoe achieved the lowest Goodput level. As the error rate on X-axis is increased, the Goodput on Y-axis decreases accordingly.

The simulation results depicted in Figures 3, 4 and 5, in reference to the proposition in abstract has shown definite performance improvement by New-Westwood over others for a network with six (6) nodes and five (5) TCP connections. However, it is of critical importance to ponder what will happen when the proposition is applied to a larger or growing network. Experience suggests that bandwidth estimation and allocation is finite and even with the modification, New-Westwood might struggle with keeping the same performance level but so will others. Further research on the aspect must be carried out in the near future for to evaluate the effects on performance due to network expansion.

Conclusions and Recommendations

The research depicted in this paper has analyzed different queue algorithms from various aspects. Simulations results marked the RED queue algorithm to be the best as portrayed earlier. The main advantage it has over Droptail is the possession of extra buffer size that stores the excessive data and controls congestion. Due to such a quality, RED possesses lower packet dropping probability than DropTail (Figure 3). After simulation analysis of the most common TCP flavors over the same network (Figure 2) and same queue algorithm i.e. RED, a humble conclusion can be reached stating that the proposed TCP New-Westwood can provide better Throughput and Goodput by helping to avoid data congestion to a certain optimized extent in wired computer networks.

Acknowledgements

Authors would like to thank the entire academic and support staff of the Department of Electrical Engineering, UET Peshawar, KPK, Pakistan for their academic input and facilitating role.

Novelty Statement

In this paper, the congestion avoidance protocol of an existing and proven traffic control algorithm has been enhanced in capability. Suggesting through implementation research that complex problems can be addressed with simple solutions thus reducing complexity while enhancing system predictability.

Author's Contribution

The original idea and supervision for this research was provided by Majad Ashraf. The reference implementations of TCP protocols was carried out by Jehan Dastagir. Code writing for congestion control and its verification was carried out by Bilal Ur Rehman and Shahid Hameed, respectively while overall implementation results were validated in conjunction with current research by Muhammad Amir.

Conflict of interest

The authors have declared no conflict of interest.

References

Blanton, E., M. Allman, L. Wang, I. Jarvinen, M. Kojo and Y. Nishida. 2012. A conservative loss recovery algorithm based on selective acknowledgment (SACK) for TCP. RFC 6675. https://doi.org/10.17487/rfc6675

Borman, D., 2012. TCP options and maximum segment size (MSS). RFC-6691. https://doi.org/10.17487/rfc6691

Charalambos, S., V. Vassiliou and A. Paphitis. 2013. Hierarchical tree alternative path (HTAP) algorithm for congestion control in wireless sensor networks. AdHoc. Netw., 11(1): 257-272. https://doi.org/10.1016/j.adhoc.2012.05.010

Chesson, G.L., 2010. Google Inc. Method and apparatus to avoid network congestion. U.S. Patent No. 7675857.

Ghassan, A., M. Ismail and K. Jumari. 2011. Characterization and observation of (transmission control protocol) TCP-Vegas performance with different parameters over (Long term evolution) LTE networks. Sci. Res. Essays, 6(9): 2003-2010. https://doi.org/10.5897/SRE11.252

Ghassan, A., M. Ismail and K. Jumari. 2012. Exploration and evaluation of traditional TCP congestion control techniques. J. King Saud Univ. Comput. Inf. Sci., 24(2): 145-155. https://doi.org/10.1016/j.jksuci.2012.03.002

Henderson, T., S. Floyd, A. Gurtov and Y. Nishida. 2012. The NewReno modification to TCP's fast recovery algorithm. RFC 6582. https://doi.org/10.17487/rfc6582

Misra, S., B.J. Oommen, S. Yanamandra and M.S. Obaidat. 2010. Random early detection for congestion avoidance in wired networks: a discretized pursuit learning automata like solution. IEEE Trans. Syst. Man Cybernet., 40(1): 66-76. https://doi.org/10.1109/TSMCB.2009.2032363

Mozilla, 2021. https://developer.mozilla.org/en-US/docs/Glossary/TCP_slow_start

Partha, K. and C. Dovrolis. 2010. Diffprobe: Detecting ISP service discrimination. In the proceedings of IEEE INFOCOM conference, pp. 1-9.

Shimaa, H. and A. El-Sayed. 2012. Enhanced TCP westwood congestion avoidance mechanism (TCP Westwood New). Int. J. Comput. Appl., 45(5): 21-29.

Stackpath, 2021. https://blog.stackpath.com/glossary-cwnd-and-rwnd/

Tianji, L., D. Leith and D. Malone. 2011. Buffer sizing for 802.11-based networks. IEEE/ACM Transactions on Networking (TON), 19(1): 156-169. https://doi.org/10.1109/

TNET.2010.2089992

Tom, H., S. Floyd, A. Gurtov and Y. Nishida. 2012. The New Reno modification to TCP's fast recovery algorithm. No. RFC 6582.

Waghmare, S., P. Nikose, A. Parab and S.J. Bhosale. 2011. Comparative Analysis of different TCP variants in a wireless environment. Proc. 3rd IEEE Int. Conf. Electron. Comput. Technol., (ICECT), 4(1): 158-162. https://doi.org/10.1109/ICECTECH.2011.5941878

Wischik, D., C. Raiciu, A. Greenhalgh and M. Handley. 2011. Design, implementation and evaluation of congestion control for multipath TCP. NSDI. Vol. 11.

Wu, H., Z. Feng, C. Guo and Y. Zhang. 2013. ICTCP: Incast congestion control for TCP in data-center networks. IEEE/ACM Trans. Netw., 21(2): 345-358. https://doi.org/10.1109/TNET.2012.2197411