IMPROVED TRANSMISSION CONTROL PROTOCOL CONGESTION CONTROL TECHNIQUE FOR HIGH BANDWIDTH LONG DISTANCE NETWORKS

USMAN AHMAD

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Doctor of Philosophy (Computer Science)

Faculty of Computing
Universiti Teknologi Malaysia

JANUARY 2017

TO MY FATHER (SHEIKH ATTIQUE AHMAD)

ACKNOWLEDGEMENT

Thanks to Almighty ALLAH for providing me the knowledge, guidance and patience to achieve this goal. My deepest gratitude is to my respected supervisor, Associate Professor Dr. Md Asri bin Ngadi. I had been amazingly fortunate to have an advisor who gave me the freedom to explore on my own, and at the same time, provide me the guidance, when my steps faltered. His patience and support helped me to complete this thesis. Next, I would like to thank my friends Mudassar Ahmad and Umar Khan for their encourage and guidance. Last but not least, I would like to express my sincerest and heartiest thanks to my parents and my gorgeous wife, for their encouraging attitude unfailing patience and excellent cooperation throughout my research.

Usman Ahmad

ABSTRACT

Transmission Control Protocol (TCP) is responsible for reliable communication of data in high bandwidth long distance networks. TCP is reliable because of its congestion control technique. Many TCP congestion control techniques for different operating systems have been developed previously. TCP Compound and TCP CUBIC are current congestion control techniques being used in Microsoft Windows and Linux operating systems respectively. TCP Reno is Standard TCP congestion control technique. TCP CUBIC does not perform well in high bandwidth long distance networks due to its exponential growth and less reduction in congestion window size. This leads to burst packet losses, unfair allocation of unused link bandwidth, long convergence time, and poor TCP friendliness among competing flows. The aim of this research work is to develop an improved congestion control technique based on TCP CUBIC for high bandwidth long distance networks. This improved technique is based on three components which are Congestion Control Technique for Slow Start (CCT-SS), Congestion Control Technique for Loss Occurrence (CCT-LO), and Enhanced Response Function of TCP CUBIC (ERFC). CCT-SS is proposed which increases the lower boundary limit of congestion window, which in turn, decreases the packet loss rate. CCT-LO is proposed which introduces a new congestion window reduction parameter in order to achieve fairer and quicker allocation of link bandwidth among the competing flows. ERFC is proposed which reduces the average congestion window size of TCP CUBIC in order to improve the TCP friendliness. As a conjunctive result of this research work, an improved congestion control technique is developed by combining the CCT-SS, CCT-LO and ERFC components. Network Simulator 2 is used to evaluate the performance of the proposed congestion control technique and to compare it with the current and other congestion control techniques. Results show that the performance of the proposed congestion control technique outperforms by 8.4% as compared to current congestion control technique.

ABSTRAK

Protokol Kawalan Penghantaran (TCP) bertanggungjawab untuk komunikasi data yang boleh dipercayai dalam rangkaian jalur lebar tinggi jarak jauh. TCP boleh dipercayai kerana teknik kawalan kesesakannya. Banyak teknik kawalan kesesakan TCP untuk sistem operasi yang berbeza telah dibangunkan sebelum ini. TCP Compound dan TCP CUBIC adalah teknik kawalan kesesakan semasa yang digunakan dalam Microsoft Windows dan sistem operasi Linux masing-masing. TCP Reno ialah teknik kawalan kesesakan bagi Standard TCP. TCP CUBIC tidak beroperasi dengan baik dalam jalur lebar tinggi rangkaian jarak jauh disebabkan oleh pertumbuhan eksponennya dan kurang pengurangan saiz tetingkap kesesakan. Ini membawa kepada kerugian paket pecah, peruntukan yang tidak adil terhadap pautan jalur lebar yang tidak digunakan, masa penumpuan yang panjang, dan keramahan TCP yang tidak adil dikalangan aliran yang bersaing. Tujuan penyelidikan ini adalah untuk membangunkan satu teknik kawalan kesesakan yang dipertingkatkan berdasarkan TCP CUBIC untuk jalur lebar tinggi rangkaian jarak jauh. Teknik yang dipertingkatkan ini adalah berdasarkan kepada tiga komponen iaitu Teknik Kawalan Kesesakan untuk Permulaan Perlahan (CCT-SS), Teknik Kawalan Kesesakan untuk Kerugian Peristiwa (CCT-LO), dan Peningkatan Fungsi Respons TCP CUBIC (ERFC). Cadangan CCT-SS adalah dengan meningkatkan had sempadan bawah tetingkap kesesakan, yang akan mengakibatkan, menurunnya kadar kehilangan paket. Cadangan CCT-LO adalah dengan memperkenalkan satu parameter baru tetingkap pengurangan kesesakan untuk mencapai peruntukan yang lebih adil dan lebih cepat bagi pautan jalur lebar di kalangan aliran yang bersaing. Cadangan ERFC adalah dengan mengurangkan purata saiz tetingkap kesesakan bagi TCP CUBIC untuk memperbaiki keramahan TCP. Sebagai satu hasil penghubungan penyelidikan ini, penambah-baikan teknik kawalan kesesakan ini telah dibangunkan dengan menggabungkan komponen CCT-SS, CCT-LO dan ERFC. Network Simulator 2 digunakan untuk menilai prestasi teknik kawalan kesesakan yang dicadangkan dan untuk membandingkannya dengan teknik-teknik kawalan kesesakan semasa yang lain. Keputusan menunjukkan bahawa prestasi teknik kawalan kesesakan yang dicadangkan adalah 8.4% lebih baik berbanding teknik kawalan kesesakan semasa.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS

ABE	–	Adaptive Bandwidth Estimation
ACK	–	Acknowledgement
AFSS	–	Adaptive Fast Slow Start
AIMD	–	Additive Increase Multiplicative Decrease
ALSS	–	Adaptive Limited Slow Start
A START	–	Adaptive Start
ASS	–	Alternative Slow Start
AFRICA	–	Adaptive and Fair Rapid Increase Congestion Avoidance
BIC	–	Binary Increase Congestion
BDP	–	Bandwidth Delay Product
BW	–	Bandwidth
CTCP	–	Compound TCP
CCT-LO	–	Congestion Control Technique for Loss Occurence
CCT-SS	–	Congestion Control Technique for Slow Start
cwnd	–	Congestion window
DS-TCP	–	Dynamic Scalable TCP
ECN	–	Early Congestion Notification
ERE	–	Eligible Rate Estimation
ERF	–	Explicit Router Feedback
ESSE	–	Early Slow Start Exit
ERFC	–	Enhanced Response Function of CUBIC
FTP	–	File Transport Protocol
HCC	–	Hybrid Congestion Control
HTCP	–	Hamilton TCP
HTTP	–	Hypertest Transfer Protocol
HyStart	–	Hybrid Start
HS-TCP	–	HighSpeed TCP
IP	–	Internet Protocol
LAN	–	Local Area Network
SS	–	Slow Start
LSS	–	Limited Slow Start

MSS – Maximum Segment Size

MIMD – Multiplicative Increase and Multiplicative Decrease

NS-2 – Network Simulator 2

QTCP – Quick TCP

RTT – Round Trip Time

RTO – Re-transmission timeout

RFC – Request for Comments

rwnd – receiver congestion window

SACK – Selected Acknowledgement

SSP – Safe Switch Point

ssthresh – Slow start threshold

SIRENS – Simple Internet Resource Notification Scheme

STCP – Scalable TCP

TCP – Transmission Control Protocol

TCP/IP – Transmission Control Protocol/Internet Protocol

TCP-UB – TCP University of Bridgeport

WAN – Wide Area Network

YeAH-TCP – Yet Another HighSpeed TCP

–

LIST OF SYMBOLS

α – Increase parameter of congestion window

β – Decrease parameter of congestion window

γ – Slow start constant of TCP Vegas in slow start phase

δ_k – Inter interval time between packets

B – Unused link bandwidth in HyStart

$cwnd$ – Congestion window

$\hat{B}$ – Unused link bandwidth in CCT-SS

C – Scaling and safe exit point in TCP CUBIC

D_{min} – Minimum one delay in HyStart

iT – The ith RTT of a flow in CCT-LO

K – Time peroid of congestion window of TCP CUBIC

$\hat{K}$ – Time peroid of congestion window of CCT-LO technique

p – Loss probability of TCP CUBIC

R_A – Actual throughput in TCP Vegas

R_E – Expacted throughput in TCP Vegas

S – Available buffer size in HyStart

Θ – Pre defined scalling factor in ABE TCP

$\hat{S}$ – Available buffer size in CCT-SS

RTT_k – k^{th} packet RTT in ESSE

Δt_k – Inter arrival time in ESSE

N – Used to adjust the size of congestion window in TCP Veno

ρ – normalized round trip time of TCP Hybla

t – Elapsed time from last congestion window reduction

W_{last} – Last minimum congestion window size in TCP CUBIC

W_{max} – Maxiimum congestion window size in TCP CUBIC

$W_{(t)}$ – Average congestion window in TCP CUBIC

w – Sum of congestion window and advertised window in TCP CUBIC

Q – Number of packets buffred by the flow in YeAH TCP

C_{f1} – Flow 1 for calculation of convergence time

C_{f2} – Flow 2 for calculation of convergence time

T_1 – Start time of flow 1 for convergence time

T_2 – Start time of flow 2 for convergence time

T – Convergence time between competing flows

Δ – A variable used in CCT-LO

δ – A variable used in ERFC

Q_{max} – Maximum number of packets in YeAH TCP

$diff$ – Difference of response functions of TCP CUBIC and ERFS

G – Goodput in YeAH TCP

$Q1$ – Predefined threshold of TCP Illinois

L – Level of congestion in the network

S_{max} – Constant parameter of TCP BIC

S_{min} – Constant parameter of TCP BIC

n – Number of flows

b_k – Available bandwidth in TCP Westwood

d_k – Amount of data sent in TCP Westwood

$dwnd$ – delay based component CTCP

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Overview

Transmission Control Protocol (TCP) is responsible for reliable data transmission among different networks by using its congestion control technique. Inside this technique, there are four supporting algorithms: slow start, congestion avoidance, fast retransmit and fast recovery. To control the data transmission, first two algorithms are used whereas, for retransmission of lost packets and for recovery of packets, last two algorithms are used. TCP Tahoe, TCP Reno, and TCP Vegas are the legacy congestion control techniques. TCP Compound is being used in Microsoft Windows and TCP CUBIC congestion control technique in Linux-based operating systems. The users of Linux-based operating systems are increasing as it is a free open source operating system that provides relatively better reliability, security, and consistency than most of other propriety operating systems i.e. Microsoft Windows. The motivation behind this research work is to address the problems of TCP CUBIC regarding its performance, which is current congestion control technique inside Linux-based operating systems. The aim of this research work is to develop an improved congestion control technique based on TCP CUBIC. This research work specifically targets the use of aforesaid congestion control technique in high bandwidth long distance networks. A high bandwidth long distance network is referred to as a network having long Round Trip Time (RTT) and high bandwidth connections among the source and destination.

1.2 Background of the Problem

TCP was adopted to deal with message flow and error correction during data transmission over the network (Postel, 1981). TCP works on the top of Internet Protocol (IP) to ensure reliable data transmission. TCP is reliable due to its congestion control technique which is responsible for controlling congestion in a network and reacting accordingly. Enormous and rapid development in worldwide network infrastructure is resulting in expansion of the Internet accompanied by the extensive use of TCP/IP (Abed *et al.*, 2012). The techniques of TCP congestion control is considered a significant factor in improving the performance of TCP. As Internet users are rising at fast pace, Internet congestion is much anticipated and is one of the core problem in computer networks (Abed *et al.*, 2012). Congestion in any network can be defined as, the phase in which the demand of network resources exceeds the available resources which lead to the loss of information or data packets. This scenario results in data retransmission (Papadimitriou *et al.*, 2011; Abed *et al.*, 2011a). TCP congestion control technique is based on packet conversation principal (Jacobson, 1988). According to this principal, a data packet will not be transmitted in the network until last transmitted data have been acknowledged or lost. This is an ideal case for TCP to transmit data. However, changes in network resources and multiple TCP connections almost always tend to deviate TCP from this ideal situation (Kelly, 1985; Cai *et al.*, 2005).

Traditional TCP like TCP Reno (also known as Standard TCP) is the major congestion control technique used by applications like HTTP and FTP over the Internet and also the trademark for congestion control techniques (Kushwaha, 2014). With the passage of time, other congestion control techniques are introduced to cope with new network scenarios. Such congestion control techniques are TCP Compound (Tan *et al.*, 2006) and TCP CUBIC (Ha *et al.*, 2008). According to Yang *et al.* (2014) among 30,000 Web servers, 3.31% to 14.47% of web servers are still using TCP Reno congestion control technique, 46.92 % web servers are using TCP CUBIC and 14.5 % to 25.66 % are using TCP Compound congestion control technique. These measurements show that very few TCP flows are controlled by TCP Reno and majority of TCP flows are controlled by TCP CUBIC. Furthermore, this TCP CUBIC is the recent congestion control technique used in Linux-based system (Yang *et al.*, 2014).

TCP breaks data into segments and transmits these segments to the receiver over the network. After receiving these segments, the receiver node creates an acknowledgment (ACK) and sends back to sender assuring that the segment is received. To avoid congestion, sender limits the amount of data for data transmission by using a variable called congestion window (Dukkipati *et al.*, 2010). This variable decides how much data needs to be sent by using alpha (α) and beta (β) parameters. These α and β are increase and decrease parameter of congestion window respectively. At the beginning of the connection, the alpha (α) parameter is used to increase the size of congestion window for data transmission and beta (β) parameter is used to avoid the congestion by decreasing the size of the congestion window. The receiver also updates the sender about its buffering size by using a variable called receiver window. By using these two variables, TCP sender decides to transmit data that is not more than the minimum size of congestion window and receiver window (Petrov and Janevski, 2013; Molia and Agrawal, 2014).

At the start of the connection between sender and receiver, slow start technique is used to find out the available link bandwidth of the network. Slow Start technique increases the size of congestion window exponentially to probe the unknown equilibrium state of the network. The sender side increases the capacity of congestion window by one per ACK. Then the size of congestion window doubles after receiving each round trip time (RTT). This process is denoted in Equations 1.1 and 1.2 respectively (Abed *et al.*, 2012; Alrshah *et al.*, 2014).

$$ACK : cwnd \leftarrow cwnd + 1 \tag{1.1}$$

$$RTT : cwnd \leftarrow cwnd \times 2 \tag{1.2}$$

Figure 1.1 shows the exponential increment in the size of congestion window after each RTT during the slow start phase. TCP Compound, TCP CUBIC, and TCP Reno, all use the same exponential increment for the size of the congestion window. After receiving a successful ACK, the size of congestion window is increased by 1 as denoted in Equation 1.1 and the size of congestion window is doubled after each RTT as denoted by Equation 1.2. In high bandwidth long distance networks, the flows having the long RTTs cause big size of congestion window, even during the early period of the connection. This type of exponential increment of congestion window size creates congestion, which in turn, cause the very high rate of packet loss and decrease the performance of network (Ha and Rhee, 2011; Abed *et al.*, 2011b; Lar and Liao, 2012; Alrshah *et al.*, 2014).

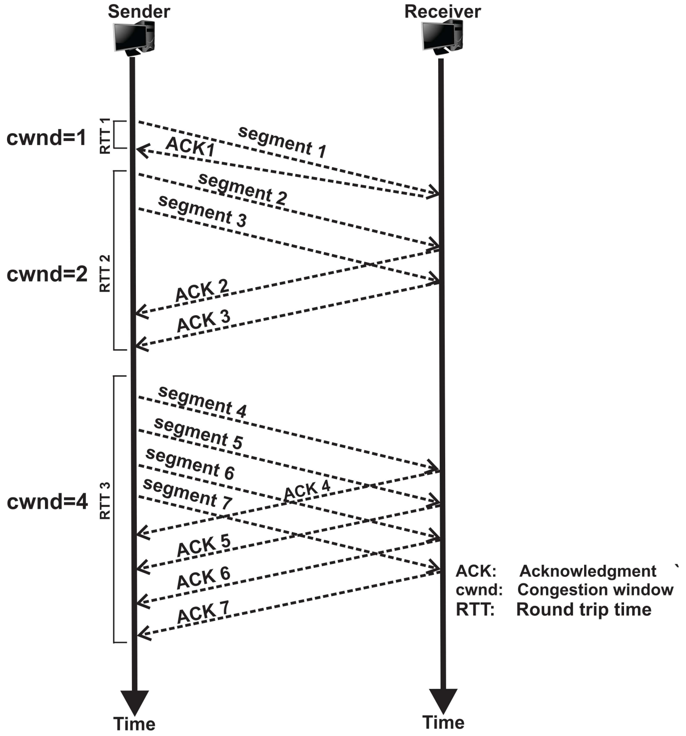

Figure 1.1: Exponential increment in congestion window size during slow start phase (Welzl and Normann, 2012)

To control data transmission, slow start threshold ($ssthresh$) is calculated to decide whether slow start phase or congestion avoidance phase would be used. This $ssthresh$ is a variable that has the estimated value of unused link bandwidth on any given path (Cavendish *et al.*, 2009). If the value of congestion window is less than $ssthresh$ then slow start phase is used and if the value of congestion window is greater than or equal to $ssthresh$, then congestion avoidance phase is used for the growth of congestion window (Jacobson, 1988; Allman and Falk, 1999; Ha and Rhee, 2011). This process is denoted in Equation 1.3. After hitting the $ssthresh$ during the slow start phase, TCP enters in congestion avoidance phase. Different congestion control techniques use different methods to calculate the value of $ssthresh$, which affects the exit point for the slow start phase. TCP uses this value to decide whether slow start phase or congestion avoidance phase should be used for the transmission of data.

$$CongestionControl : \left\{ \begin{array}{lll} cwnd & < & ssthresh \quad Slow\,Start\,Phase \\ cwnd & \geq & ssthresh \quad Congestion\,Avoidance\,Phase \end{array} \right\} \tag{1.3}$$

Congestion avoidance phase controls the increment in the size of congestion window because the source has already calculated the stable state of the network. During congestion avoidance phase, TCP Reno and TCP Compound both, increase the size of congestion window by $\frac{1}{cwnd}$ after receiving each ACK, which enables the source to increase the size of congestion window gradually by only one packet per RTT. This growth is also called the linear growth of congestion window (Patel and Rani, 2016). However in TCP CUBIC, the size of congestion window increases by $cwnd + 1$ after receiving each ACK, instead of RTT during the congestion avoidance phase grabbing the unused link bandwidth very quickly. This growth is also called the exponential growth of congestion window (Ha *et al.*, 2008; Ha and Rhee, 2011). This exponential increment in congestion window size during the congestion avoidance phase can create congestion in the network and cause the high rate of packet loss, which in turn create the problem of performance decrement. A comparison of congestion window growth of TCP Compound, TCP CUBIC, and TCP Reno during the congestion avoidance phase is shown in Equation 1.4 (Ha *et al.*, 2008; Alrshah *et al.*, 2014; Patel and Rani, 2016).

$$ACK : \left\{ \begin{array}{lllll} cwnd & = & cwnd + \frac{1}{cwnd} & Linear & TCP\;\;Reno \\ cwnd & = & cwnd + \frac{1}{cwnd} & Linear & TCP\;\;Compound \\ cwnd & = & cwnd + 1 & Exponential & TCP\;\;CUBIC \end{array} \right\} \tag{1.4}$$

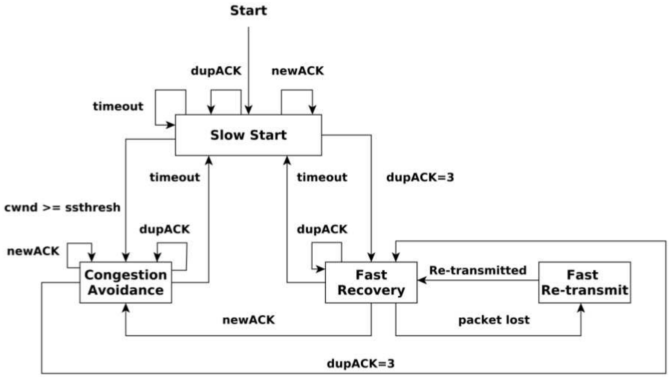

Figure 1.2: Functional interference among slow start, congestion avoidance, fast re-transmit and fast recovery phases (Abed *et al.*, 2012)

Figure 1.2, shows the process flow of the slow start, congestion avoidance, fast retransmission and fast recovery phases. The switching technique among the phases are described as follows:

i. In slow start phase, after receiving a new ACK, a duplicate ACK, or after registering a timeout, the connection does not change its phase.

ii. During the slow start phase, if the value of congestion window is greater than or equal to $ssthresh$ $(cwnd >= ssthresh)$, connection switches to congestion avoidance phase. And then after receiving three duplicate ACKs, it switches to fast recovery phase.

iii. During the congestion avoidance phase, connection remains in the same phase after receiving a new or duplicate ACK. After receiving three duplicate ACKs, connection switches to fast recovery phase and when the timeout occurs, connection switches to slow start phase.

iv. In the fast recovery phase, dropped packets are retransmitted by fast retransmit algorithm as shown in Figure 1.3. In this phase if the receiver receives out of order sequence number of a packet then it will send a duplicate ACK of last ACK packet. Receiving three duplicate ACKs is the indication of dropped packets which is discussed in RFC 2001 (Stevens, 1997). If timeout occurs, connection switches to slow start phase from fast recovery and remains in same phase (fast recovery) if the sender receives duplicate ACK. However, after receiving new ACK, the connection switches to congestion avoidance phase.

For the reliable data transmission between sender and receiver, TCP needs a technique to detect packet losses and retransmit those lost packets. After detection of packet loss by receiving three duplicate ACKs, fast retransmit phase is used to retransmit the dropped packets as shown in Figure 1.3 and fast recovery technique is used to adjust the rate of new segments until a non-duplicate ACK is received by the sender. In Equation 1.5, after detecting each packet loss, TCP Reno and TCP Compound reduce the size of congestion window by 50% of its original size, whereas TCP CUBIC reduces only 20% of its original size by a reduction parameter β, because the value of reduction parameter β has been adjusted at 0.5 in TCP Reno and TCP Compound and 0.2 in TCP CUBIC. Thus, the current TCP CUBIC flows do not release sufficient amount of used link bandwidth for the new incoming flows. As a result, new flows cannot get sufficient amount of link bandwidth for data transmission. Moreover, due to this insufficient release of link bandwidth, issues of unfair share of link bandwidth allocation and problem of long convergence time between competing flows happend (Qureshi *et al.*, 2013; Kozu *et al.*, 2014).

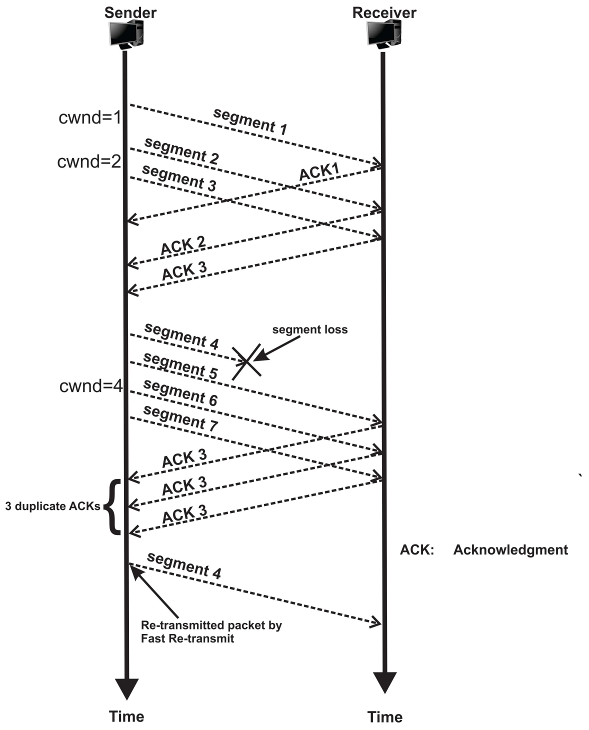

Figure 1.3: Fast retransmission after receiving three ACKs (Welzl and Normann, 2012)

$$ACK : \left\{ \begin{array}{lcll} cwnd & = & \frac{1}{2} \times cwnd & 50\% \, reduction \, in \, cwnd \quad TCP \;\; Reno \\ cwnd & = & \frac{1}{2} \times cwnd & 50\% \, reduction \, in \, cwnd \quad TCP \;\; Compound \\ cwnd & = & \frac{4}{5} \times cwnd & 20\% \, reduction \, in \, cwnd \quad TCP \;\; CUBIC \end{array} \right\} \tag{1.5}$$

TCP CUBIC is the recent TCP congestion control technique used in Linux-based operating systems. It replaces the linear growth function of TCP Reno to cubic growth function and this congestion window growth is RTT independent. Due to this characteristic, the increment in congestion window of TCP CUBIC is more aggressive and flows can grab more unused link bandwidth. However, this behavior also causes the problem of slow convergence between the competing flows i.e. when the new flow shares the common link bandwidth with existing flows, the new incoming flow takes a long time to achieve the fair share of unused link bandwidth due to less reduction of the congestion window size.

The quick allocation of link bandwidth between competing flows depends on the convergence time. The behavior of convergence is determined by interaction among the competing flows. The rate of convergence relies upon two factors; the first factor is the rate at which the flows gain the unused link bandwidth through an increase parameter called (α) of the congestion window. The second factor is the rate at which the flows release the bandwidth after each packet loss through a decrease parameter called (β) of congestion window (Leith *et al.*, 2008a; Cao and Zhang, 2013). At the beginning of the connection, the flows having higher congestion window increase their congestion window more aggressively as compared to flows having lower congestion window. Therefore, TCP CUBIC reduces its congestion window only by 20% of its current size after detecting packet loss and does not release enough bandwidth for other flows during the transmission of data. Because of this less reduction in the size of congestion window, new incoming flows have a disadvantage when it comes to gain bandwidth and take more time to get the fair allocation of the link bandwidth. This slow allocation of unused link bandwidth results in slow convergence among competing flows and, in turn, causes poor network performance (Leith *et al.*, 2008a; Kushwaha, 2014).

Increment and decrement in the size of congestion window depend on the phase of congestion control technique. The rate of increment and decrement depends on the response function of congestion control technique. Response function depends on the α and β parameters for the increment and decrement in the size of the congestion window. Response function uses Additive Increase Multiplicative Decrease (AIMD) algorithm having the values of $\alpha = 1$ and $\beta = 0.5$ (Jacobson, 1988). TCP Reno uses the AIMD algorithm for the increment and decrement in the size of the congestion window. General form of AIMD algorithm is shown in Equation 1.6, where the $\alpha = 1$ is increase parameter after receiving ACK and $\beta = \frac{1}{2}$ is decrement parameter at each packet loss. In Figure 1.4 the behavior of TCP Reno regarding congestion window is shown, it uses the AIMD algorithm for the increment and decrement in the size of congestion window, where the size of congestion window reduces by 50% of its original size after each packet loss. TCP Compound also uses the same α and β values for the congestion window used by TCP Reno.

$$AIMD : \left\{ \begin{array}{lll} ACK & = & cwnd \leftarrow cwnd + \frac{\alpha}{cwnd} \\ Loss & = & cwnd \leftarrow (1-\beta) \times cwnd \end{array} \right\} \tag{1.6}$$

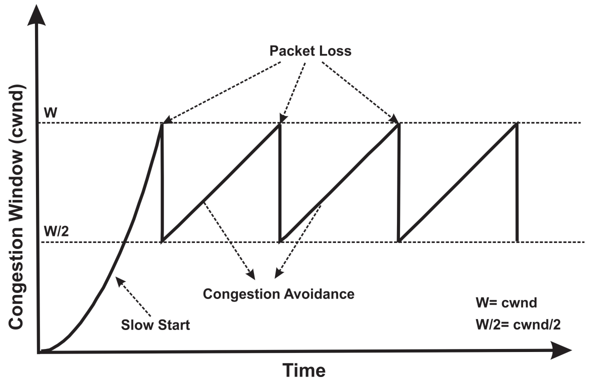

Figure 1.4: Congestion window behaviour of TCP Reno (Ahmad *et al.*, 2015)

TCP CUBIC also uses AIMD algorithm with modified values of α and β parameters. TCP CUBIC uses $\alpha = 0.3$ and $\beta = 0.2$ for the increment and decrement in the size of congestion window. Due to its Beta value $\beta = 0.2$, TCP CUBIC reduces only 20% size instead of 50% of its original size of congestion window as in TCP Compound and TCP Reno. TCP CUBIC flows do not release enough link bandwidth after packet loss for new incoming TCP flows in the same network. Thus, new flows are usually not able to use sufficient amount of link bandwidth for the data transmission. This lack of link bandwidth creates congestion in the network, which causes the decrement in the performance of the network regarding slow convergence time.

The response function of congestion control technique can be defined by the TCP friendliness behavior of congestion control technique. TCP friendliness is referred to as the fair share of available link bandwidth among the flows of TCP Reno and the flows of other congestion control technique (Xu *et al.*, 2004; Masaki *et al.*, 2010). The response function is a measure of average throughput of a single TCP flow of congestion control technique as the level of random packet loss is varied (Li *et al.*, 2007). The throughput of a TCP flow depends on the average congestion window size of congestion control technique. The size of congestion window of TCP flow depends on the usage of available link bandwidth during data communication. So, if a TCP CUBIC flow is using more available link bandwidth as compared to TCP Reno flow, the average congestion window size of TCP CUBIC flow is greater than the TCP Reno flow, means that these both flows are not sharing available link fairly, which refers as low TCP friendliness of TCP CUBIC. It implies that the throughput of TCP CUBIC flow is higher than the standard congestion control technique, i.e., TCP Reno flow. A comparison of TCP friendliness (response function) of TCP Compound and TCP CUBIC with TCP Reno is shown in Figure 1.5. Response function curve of TCP CUBIC is higher than TCP Reno. As a result, TCP CUBIC flows did not share available link bandwidth fairly with TCP Reno flows, which, in turn, causes the problem of TCP friendliness. However, response function curve of TCP Compound is close to TCP Reno. Thus, TCP Compound flows share available link bandwidth fairly with TCP Reno flows.

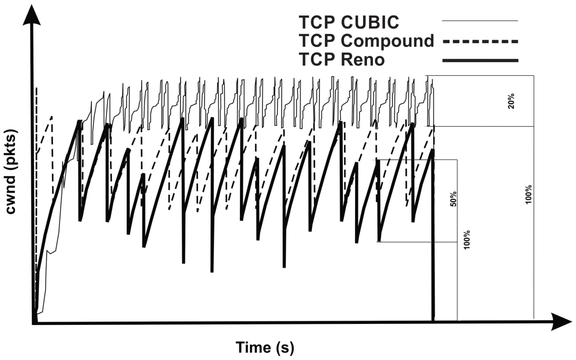

Figure 1.5: The comparison of response functions of TCP CUBIC, TCP Reno and TCP Compound (Ahmad *et al.*, 2015)

Some major problems related to TCP CUBIC congestion control technique are summarized as follows:

i. In slow start and congestion avoidance phase, the increment in congestion window size is exponential which causes the high rate of packet loss (Ha and Rhee, 2011; Alrshah *et al.*, 2014).

ii. After detecting each packet loss, TCP CUBIC reduces its congestion window size only 20% of its existing congestion window. Which means, TCP CUBIC flows did not release much bandwidth for new incoming TCP flows. Thus, new incoming flows of any protocol are not able to grab sufficient amount of link bandwidth for data transmission; causing unfair share of unused link bandwidth and long convergence time among the competing flows (Qureshi *et al.*, 2013; Cao and Zhang, 2013; Kozu *et al.*, 2014).

1.3 Research Gap

The increment in the size of congestion window during slow start and congestion avoidance phase plays a significant role in congestion control techniques. The increment in the size of congestion window should not be too low that TCP flows cannot be able to use available link bandwidth properly, and it should not be too fast that it can create congestion in the network. The growth of congestion window depends on the estimated available link bandwidth. Each slow start algorithm of congestion window technique has rule or function to calculate the available link bandwidth by using different techniques and calculates the size of congestion window accordingly. Today the most commonly used operating systems, such as Microsoft Windows, Linux operating system, and Android, are still using exponential increment in the size of congestion window for slow start phase during the data transmission.

Therefore, many studies have observed that the performance of congestion control technique suffers during slow start phase in high bandwidth long distance networks (Zhang *et al.*, 2012). The increment in the size of congestion window during slow start phase affects the performance of congestion control technique. Thus, the performance of congestion control technique can be enhanced by controlling the increment of congestion window wisely in slow start phase during data transmission (Wang and Williamson, 1998; Floyd, 2004; Ha and Rhee, 2011; Zhang *et al.*, 2012; Alrshah *et al.*, 2014). To avoid congestion in the network, a number of TCP congestion control technique control the increment in the size of congestion window during slow start phase (Kozu *et al.*, 2014).

After detecting each packet loss during data transmission, congestion control technique reduces the size of the congestion window to avoid the congestion in the network. This reduction in the size of congestion window is varying in different congestion control techniques. The minimum reduction rate of congestion window after each packet loss is 12.5%, which is used by Scalable TCP (Kelly, 2003). The maximum reduction rate after each packet loss is 50%, which is the default reduction rate of TCP Reno also known as Standard TCP congestion control technique. The number of congestion control techniques are also configured with 50% reduction in the size of congestion window after each packet loss during data transmission such as TCP Compound (Song *et al.*, 2006), HighSpeed TCP (Floyd, 2003) and Hamilton TCP (Leith and Shorten, 2004). The reduction in the size of congestion window after each packet loss affects the performance of congestion control technique regarding protocol fairness and convergence time among the TCP flows during data transmission. According to Qureshi *et al.* (2013); Alrshah *et al.* (2014), TCP CUBIC congestion control technique is still under development, and there is a need for more evaluation studies for TCP CUBIC. According to their research, TCP CUBIC congestion control technique does not reduce enough size of congestion window after each packet loss for other incoming TCP flows. This less reduction in the size of congestion window after each packet loss creates congestion and reduces the overall performance of the network. Thus, more research is needed on TCP CUBIC by adaptive adjustment of congestion window size during the data transmission.

The response function of congestion control technique is defined by the TCP friendliness behavior of congestion control technique. TCP friendliness behavior can be calculated between the TCP Reno and other congestion control flows which show the fair share of available link bandwidth during data transmission among the flows (Xu *et al.*, 2004). If a TCP flow uses more available link bandwidth as compared to TCP Reno flows that means, its average congestion window size is high.

1.4 Statement of the Problem

The above background of the problem leads this research to address the problem of allocation of unused link bandwidth fairly and quickly among the multiple competing flows of different TCP congestion control techniques transferring data over the same network path, such that, packet loss rate can be mitigated by achieving the maximum protocol fairness, convergence time, TCP friendliness, and stability. The above problem statement leads to the following research questions:

i. How to reduce the packet loss rate during data communication in slow start phase?
ii. How to allocate unused link bandwidth fairly and quickly between the TCP CUBIC flows?
iii. How to allocate the unused link bandwidth fairly between TCP CUBIC and TCP Reno flows?

1.5 Aim of Research

The aim of this research work is to develop an improved congestion control technique based on TCP CUBIC for high bandwidth long distance networks.

1.6 Objectives of Research

The objectives of this research work is described as follows:

i. To design and develop a Congestion Control Technique for Slow Start (CCT-SS) to decrease the packet loss rate.
ii. To design and develop a Congestion Control Technique for Loss Occurrence (CCT-LO) for fair and quick allocation of unused link bandwidth.
iii. To design and develop a Enhanced Response Function for TCP CUBIC (ERFC) to increase the friendliness among TCP CUBIC and TCP Reno flows.

1.7 Scope of Research

The scope of this research work is described as follows:

i. Congestion can be controlled or avoided by two approaches called end-to-end congestion control technique or router based congestion control technique. In this research, end-to-end congestion control approach has been used, and router based approach has not been touched as it is not within the scope of this research.

ii. In end-to-end congestion control approach, there are three types of categories for the indication of congestion during the data transmission. First is based on packet loss, the second one is delay based and third is the combination of loss and delay (hybrid). In current research work, loss based approach has been used for the indication of congestion and other approaches (i.e. delay, hybrid) have not been covered because of the scope of the present study.

iii. There are different types of networks i.e. wired network and wireless network; this research work focuses on the performance of congestion control techniques in wired networks and not wireless networks.

iv. There are four phases inside the congestion control techniques which are the slow start, congestion avoidance, fast retransmission and fast recovery. This research work focuses on the slow start, and congestion avoidance phases and other two phases (fast retransmission and fast recovery) are hence, out of scope.

1.8 Significance of Research

High bandwidth long distance networks are emerging in several continents rapidly, and TCP is one of the data transferring protocol in these networks. This research work positively impacts the field of high bandwidth long distance networks. Implementing the results of present study decrease the congestion in these networks. This is achieved by improving the current congestion control technique for Linux-based operating systems (TCP CUBIC). With improved congestion control technique (proposed in this research), the rate of packet loss of TCP flows could be mitigated, and fair allocation of available link bandwidth between TCP flows could be improved. This research work shows the importance of decreasing the packet loss rate by developing a technique for slow start phase. Moreover, protocol fairness, TCP friendliness and convergence time of TCP flows is improved by implementing the technique for packet loss event and response function respectively. These findings which are the outcome of each proposed technique can have positive effects on the future work of research and hence, it contributes to the continued knowledge formation in the field of high bandwidth long distance networks.

REFERENCES

Abdeljaouad, I., Rachidi, H., Fernandes, S. and Karmouch, A. (2010). Performance Analysis of Modern TCP Variants: A Comparison of CUBIC, Compound and New Reno. In *Proceeding of 25th Biennial Symposium on Communications (QBSC)*. IEEE, 80–83.

Abed, G., Ismail, M., Jumari, K. *et al.* (2011a). A Comparison and Analysis of Congestion Window for HS-TCP, Full-TCP, and TCP-Linux in Long Term Evolution System Model. In *2011 IEEE Conference on Open Systems (ICOS),*. IEEE, 358–362.

Abed, G. A., Ismail, M. and Jumari, K. (2011b). A Survey on Performance of Congestion Control Mechanisms for Standard TCP Versions. *Australian Journal of Basic & Applied Sciences*. 5(12).

Abed, G. A., Ismail, M. and Jumari, K. (2012). Exploration and Evaluation of Traditional TCP Congestion Control Techniques. *Journal of King Saud University-Computer and Information Sciences*. Elsevier, 24(2), 145–155.

Afanasyev, A., Tilley, N., Reiher, P. and Kleinrock, L. (2010). Host-to-Host Congestion Control for TCP. *Communications Surveys & Tutorials, IEEE*. 12(3), 304–342.

Ahmad, M., Ngadi, M. A. and Mohamad, M. M. (2015). Experimental Evaluation of TCP Congestion Control Mechanisms in Short and Long Distance Networks. *Journal of Theoretical and Applied Information Technology*. 71(2).

Allman, M. (1999). TCP Slow start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. *RFC 2581*.

Allman, M. and Falk, A. (1999). On the Effective Evaluation of TCP. *ACM SIGCOMM Computer Communication Review*. ACM 29(5), 59–70.

Allman, M., Paxson, V. and Blanton, E. (2009). *TCP Congestion Control*. Technical report.

Allman, M., Paxson, V., Stevens, W. *et al.* (1999). TCP Congestion Control.

Alrshah, M. A., Othman, M., Ali, B. and Hanapi, Z. M. (2014). Comparative Study of High-Speed Linux TCP Variants over High-BDP Networks. *Journal of Network and Computer Applications (JNCA)*. Elsevier, 43, 66–75.

Arokkiam, J. A., Wu, X., Brown, K. N. and Sreenan, C. J. (2014). Experimental Evaluation of TCP Performance over 10Gb/s Passive Optical Networks (XG-PON). In *IEEE/ACM Transactions on Networking (ToN)*. 34(8), 2223–2228.

Aydin, I., Iyengar, J., Conrad, P., Shen, C.-C. and Amer, P. (2012). Evaluating TCP-Friendliness in Light of Concurrent Multipath Transfer. *Computer Networks*. 56(7), 1876–1892.

Baiocchi, A., Castellani, A. P. and Vacirca, F. (2007). YeAH-TCP: Yet Another Highspeed TCP. In *Proceeding of Fifth International Workshop for Fast Long Distance Networks (PFLDnet-2007)*, vol. 7. PFLD, 37–42.

Barayyan, T. O. (2013). Alternative Slow Start Algorithm for High Bandwidth. In *Proceeding of International Conference on Computer Applications Technology (ICCAT)*. IEEE, 1–4.

Bateman, M., Bhatti, S., Bigwood, G., Rehunathan, D., Allison, C., Henderson, T. and Miras, D. (2008). A Comparison of TCP Behaviour at High Speeds using Ns-2 and Linux. In *Proceedings of the 11th Communications and Networking Simulation Symposium*. ACM, 30–37.

Brakmo, L. and Peterson, L. (1995). TCP Vegas: End to End Congestion Avoidance on a Global Internet. *IEEE Journal on Selected Areas in Communications,*. 13(8), 1465–1480.

Brakmo, L. S., O'Malley, S. W. and Peterson, L. L. (1994). *TCP Vegas: New Techniques for Congestion Detection and Avoidance*. vol. 24. ACM.

Cai, L., Shen, X., Pan, J. and Mark, J. W. (2005). Performance Analysis of TCP-Friendly AIMD Algorithms for Multimedia Applications. *IEEE Transactions on Multimedia,*. 7(2), 339–355.

Caini, C. and Firrincieli, R. (2004). TCP Hybla: A TCP Enhancement for Heterogeneous Networks. *International journal of satellite communications and networking*. 22(5), 547–566.

Callegari, C., Giordano, S., Pagano, M. and Pepe, T. (2012). Behavior Analysis of TCP Linux Variants. *Computer Networks*. Elsevier, 56(1), 462–476.

Cao, N. and Zhang, W. (2013). CUBIC with Faster Convergence: An Improved CUBIC Fast Convergence Mechanism. In *Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering*. Atlantis Press.

Carter, R. L. and Crovella, M. E. (1996). Measuring Bottleneck Link Speed in Packet-Switched Networks. *Performance Evaluation*. 27, 297–318.

Cavendish, D., Kumazoe, K., Tsuru, M., Oie, Y. and Gerla, M. (2009). CapStart: An Adaptive TCP Slow Start for High Speed Networks. In *2009. INTERNET'09. First International Conference on Evolving Internet,*. IEEE, 15–20.

Chen, Q., Schmidt-Eisenlohr, F., Jiang, D., Torrent-Moreno, M., Delgrossi, L. and Hartenstein, H. (2007). Overhaul of IEEE 802.11 Modeling and Simulation in ns-2. In *Proceedings of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*. ACM, 159–168.

Cheng, R.-S., Lin, H.-T., Hwang, W.-S. and Shieh, C.-K. (2005). Improving the Ramping up Behavior of TCP Slow Start. In *2005. AINA 2005. 19th International Conference on Advanced Information Networking and Applications,*, vol. 1. IEEE, 807–812.

Cui, Y., Wang, L., Wang, X., Wang, Y., Ren, F. and Xia, S. (2016). End-to-end coding for TCP. *IEEE Network*. 30(2), 68–73.

de Santi, J. and da Fonseca, N. L. (2011). Design of Optimal Active Queue Management Controllers for HSTCP in Large Bandwidth-Delay Product Networks. *Computer Networks*. 55(12), 2772–2790.

Dovrolis, C., Ramanathan, P. and Moore, D. (2004). Packet-Dispersion techniques and a Capacity-Estimation Methodology. *IEEE/ACM Transactions on Networking,*. 12(6), 963–977.

Dukkipati, N., Refice, T., Cheng, Y., Chu, J., Herbert, T., Agarwal, A., Jain, A. and Sutin, N. (2010). An Argument for Increasing TCP's Initial Congestion Window. *Computer Communication Review*. 40(3), 26–33.

Elmannai, W., Elleithy, K. and Razaque, A. (2012). A High Performance and Efficient TCP Variant. *Proceeding of ASEE Northeast Section Journal, University of Massachusetts Lowell.*, vol. 2. 331–346.

Eshete, A., Jiang, Y. and Landmark, L. (2012). Fairness among High Speed and Traditional TCP under Different Queue Management Mechanisms. In *Proceedings of the Asian Internet Engineeering Conference*. ACM, 39–46.

Floyd, S. (2003). HighSpeed TCP for Large Congestion Windows. *International Journal of Computer Applications. 48(6), 35-55*.

Floyd, S. (2004). Limited Slow-Start for TCP with Large Congestion Windows. *IEICE Transactions on Communications. 52(8), 156-170.*

Floyd, S. (2008). Metrics for the Evaluation of Congestion Control Mechanisms. *International Journal of Computer Applications. 21(4), 75-95*.

Floyd, S., Allman, M., Jain, A. and Sarolahti, P. (2007). Quick-Start for TCP and IP. *International Journal of Autonomous and Adaptive Communication Systems. 6(2), 55-71*.

Floyd, S. and Fall, K. (1999a). Promoting the use of End-to-End Congestion Control in the Internet. *IEEE/ACM Transactions on Networking (TON)*. 7(4), 458–472.

Floyd, S. and Fall, K. (1999b). Promoting the use of End-to-End Congestion Control in the Internet. *IEEE/ACM Transactions on Networking (TON)*. 7(4), 458–472.

Fu, C. P. and Liew, S. C. (2003). TCP Veno: TCP Enhancement for Transmission over Wireless Access Networks. *IEEE Journal on Selected Areas in Communications,*. 21(2), 216–228.

Fu, X., SUN, L.-j., WANG, R.-c. and FANG, Y.-c. (2011). BIPR: A New TCP Variant over Satellite Networks. *The Journal of China Universities of Posts and Telecommunications*. 18, 34–39.

Giordano, S., Procissi, G., Russo, F. and Secchi, R. (2005). On the Use of Pipesize Estimators to Improve TCP Transient Behavior. In *2005. ICC 2005. 2005 IEEE*

International Conference on Communications,, vol. 1. IEEE, 16–20.

Ha, S. (2009). *Improving TCP Congestion Control for High Bandwidth and Long Distance Networks*. North Carolina State University, vol. 7. 554-573.

Ha, S., Kim, Y., Le, L., Rhee, I. and Xu, L. (2006). A Step Toward Realistic Performance Evaluation of High-speed TCP Variants. In *Proceeding Fourth International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet-2006)*. vol. 35. PFLD, 331–343.

Ha, S. and Rhee, I. (2008). Hybrid Slow Start for High-Bandwidth and Long-Distance Networks. In *Proceeding of Ninth International Workshop on Protocols for Fast Long Distance Networks (PFLDnet-2008),*. vol. 2. PFLD, 1–6.

Ha, S. and Rhee, I. (2011). Taming the Elephants: New TCP Slow Start. *Journal IEEE on Selected Areas in Communications,*. 55(9), Elsevier, 2092–2110.

Ha, S., Rhee, I. and Xu, L. (2008). CUBIC: A New TCP-friendly High-Speed TCP Variant. *Journal IEEE on Selected Areas in Communications,*. 42(5), 64–74.

Hagag, S. and El-Sayed, A. (2012). Enhanced TCP Westwood congestion avoidance mechanism (TCP WestwoodNew). *International Journal of Computer Applications*. 45(5), 21–29.

Hasegawa, G. and Murata, M. (2001). Survey on Fairness Issues in TCP Congestion Control Mechanisms. *IEICE Transactions on Communications*. 84(6), 1461–1472.

Ho, C.-Y., Chan, Y.-C. and Chen, Y.-C. (2006). Gallop-Vegas: An Enhanced Slow-Start Mechanism for TCP Vegas. *Communications and Networks, Journal of*. 8(3), 351–359.

Hoe, J. C. (1996). Improving the Start-Up Behavior of a Congestion Control Scheme for TCP. In *ACM SIGCOMM Computer Communication Review*, vol. 26. ACM, 270–280.

Hu, N. and Steenkiste, P. (2003). Improving TCP Startup Performance using Active Measurements: Algorithm and Evaluation. In *2003. Proceedings. 11th IEEE International Conference on Network Protocols,*. IEEE, 107–118.

Jacobson, V. (1988). Congestion Avoidance and Control. In *ACM SIGCOMM computer communication review*, vol. 18. ACM, 314–329.

Jacobson, V. (1990). Berkeley TCP Evolution from 4.3-Tahoe to 4.3-Reno. In *Proceedings of the 18th Internet Engineering Task Force, University of British Colombia, Vancouver, BC*. 365.

Jacobson, V. (1995). Congestion Avoidance and Control. *ACM SIGCOMM Computer Communication Review*. 25(1), 157–187.

Jain, R., Chiu, D.-M. and Hawe, W. (1998). A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems. *Journal IEEE on Selected Areas in Communications,*, 22(4), 81–92.

Jain, S. and Raina, G. (2011). An Experimental Evaluation of CUBIC TCP in a Small Buffer Regime. In *Communications (NCC), 2011 National Conference.* IEEE, 12(5), 1–5.

Janiszowski, K. B. (1993). A Modification and the Tustin Approximation. *IEEE Transactions on Automatic Control,*. 38(8), 1313–1316.

Kaneko, K., Fujikawa, T., Su, Z. and Katto, J. (2007). TCP-Fusion: A Hybrid Congestion Control Algorithm for High-speed Networks. *Proceeding of Sixth International Wrokshop on Protocol for Fast Long Distance Networks (PFLD-net). 34(5) 34-50*.

Kang, S.-r., Liu, X., Dai, M. and Loguinov, D. (2004). Packet-Pair Bandwidth Estimation: Stochastic Analysis of a Single Congested Node. In *2004. ICNP 2004. Proceedings of the 12th IEEE International Conference on Network Protocols,*. IEEE, 316–325.

Karn, P. and Partridge, C. (1987). Improving Round-Trip Time Estimates in Reliable Transport Protocols. In *ACM SIGCOMM Computer Communication Review,* vol. 17. ACM, 2–7.

Kelly, F. P. (1985). Stochastic Models of Computer Communication Systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 379–395.

Kelly, T. (2003). Scalable TCP: Improving Performance in Highspeed Wide Area Networks. *ACM SIGCOMM Computer Communication Review*. 33(2), 83–91.

King, R., Baraniuk, R. and Riedi, R. (2005). TCP-Africa: An Adaptive and Fair Rapid Increase Rule for Scalable TCP. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE,* vol. 3. IEEE, 1838–1848.

Koyama, T. and Aoki, K. (2015). Slow Start Algorithm for Mobile Broadband Networks including Delay Unrelated to Network Congestion. In *2015 International Conference on Computing, Networking and Communications (ICNC),*. IEEE, 148–152.

Kozu, T., Akiyama, Y. and Yamaguchi, S. (2014). Improving RTT Fairness on CUBIC TCP. *International Journal of Networking and Computing*. 4(2), 291–306.

Kristoff, J. (2002). TCP Congestion Control. *Technical Notes.*, 21(3), 177–287.

Kumari, D., Tahiliani, M. P. and Shenoy, U. K. K. (2014). Experimental Analysis of CUBIC TCP in Error Prone MANETs. In *2014 Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT),*. IEEE, 256–261.

Kuo, F.-C. and Fu, X. (2008). Probe-Aided MulTCP: An Aggregate Congestion Control Mechanism. *ACM SIGCOMM Computer Communication Review*. 38(1), 17–28.

Kurose, J. F. and Ross, K. W. (2006). Computer Networking Complete Package with Study Companion, 1–733.

Kushwaha, V. (2014). Research Issues in TCP Based Congestion Control for High Speed Network. *International Journal of Computer Science & Communication Networks*. 4(1), 18.

Lar, S. and Liao, X. (2012). An Initiative for a Classified Bibliography on TCP/IP Congestion Control. *Journal of Network and Computer Applications. 12(4) 423-436*.

Le, T. A., Hong, C. S. and Lee, S. (2011). MPCubic: An Extended cubic TCP for Multiple Paths over High Bandwidth-Delay Networks. In *2011 International Conference on ICT Convergence (ICTC),*. IEEE, 34–39.

Leith, D. and Shorten, R. (2004). H-TCP: TCP for High-Speed and Long-Distance Networks. In *Second International Workshop on Protocol for Fast Long-Distance Networks (PFLDnet-2004)*, vol. 2004. 111–131.

Leith, D., Shorten, R. and McCullagh, G. (2008a). Experimental Evaluation of CUBIC-TCP. *Journal of Hamilton Institute Ireland*, JACE, 44(3), 212–232.

Leith, D. J., Andrew, L. L., Quetchenbach, T. and Shorten, R. N. (2008b). Experimental Evaluation of Delay/Loss-Based TCP Congestion Control Algorithms. In *Proceedings of the 6th International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet 2008),*.

Li, Y.-T., Leith, D. and Shorten, R. N. (2007). Experimental Evaluation of TCP Protocols for High-Speed Networks. *IEEE/ACM Transactions on Networking,*. 15(5), 1109–1122.

Liu, S., Başar, T. and Srikant, R. (2008). TCP-Illinois: A Loss-and Delay-Based Congestion Control Algorithm for High-Speed Networks. *IEEE/ACM Transactions on Networking (ToN)*. 65(6), 417–440.

Lv, W. and Zhang, J. (2012). Research of TCP Optimization Technology for Long-Distance and High Bandwidth-Delay Private Network. In *2012 International Conference on Computer Science and Information Processing (CSIP),*. IEEE, 381–384.

Man, C. L. T., Hasegawa, G. and Murata, M. (2003). A New Available Bandwidth Measurement Technique for Service Overlay Networks. In *Management of Multimedia Networks and Services*. (pp. 436–448). Springer.

Marfia, G., Palazzi, C. E., Pau, G., Gerla, M. and Roccetti, M. (2010). TCP Libra: Derivation, Analysis, and Comparison with Other RTT-Fair TCPs. *Computer Networks*. 54(14), 2327–2344.

Marrone, L. A., Barbieri, A. and Robles, M. (2013). TCP Performance: CUBIC, Vegas & Reno. *Journal of Computer Science & Technology*, vol. 13, 145–156.

Masaki, J., Nishantha, G. and Hayashida, Y. (2010). Development of a High-Speed Transport Protocol with TCP-Reno Friendliness. In *2010 The 12th International Conference on Advanced Communication Technology (ICACT),*, vol. 1. IEEE, 174–179.

Mascolo, S., Casetti, C., Gerla, M., Sanadidi, M. Y. and Wang, R. (2001). TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*. ACM, 287–297.

Mbarek, R., Othman, M. T. B. and Nasri, S. (2008). Performance Evaluation of Competing High-Speed TCP Protocols. *IJCSNS*. 8(6), 100.

Mirkovic, J., Hussain, A., Fahmy, S., Reiher, P. and Thomas, R. K. (2009). Accurately

Measuring Denial of Service in Simulation and Testbed Experiments. *IEEE Transactions on Dependable and Secure Computing,*. 6(2), 81–95.

Molia, H. K. and Agrawal, R. (2014). A Conceptual Exploration of TCP Variants. In *Communication and Networking (ET2ECN), 2014 2nd International Conference on Emerging Technology Trends in Electronics,*. IEEE, 1–6.

Molnár, S., Sonkoly, B. and Trinh, T. (2009). A Comprehensive TCP Fairness Analysis in High Speed Networks. *Computer Communications*. 32(13), 1460–1484.

Nabeshima, M. and Yata, K. (2004). Improving the Convergence Time High-Speed TCP. In *12th IEEE International Conference on Networks, 2004.(ICON 2004). Proceedings.*, vol. 1. IEEE, 19–23.

Nagle, J. (1984). Congestion Control in IP/TCP Internetworks. *International Journal of Autonomous and Adaptive Communication Systems.*, 17(5), 344–359.

Nakauchi, K. and Kobayashi, K. (2007). An Explicit Router Feedback Framework for High Bandwidth-Delay Product Networks. *Computer Networks*. 51(7), 1833–1846.

Nemoto, Y., Ogura, K. and Katto, J. (2013). An Adaptive TCP Congestion Control having RTT-Fairness and Inter-Protocol Friendliness. In *Consumer Communications and Networking Conference (CCNC), 2013 IEEE*. IEEE, 178–183.

Olsén, J. (2003). *On Packet Loss Rates Used for TCP Network Modeling*. Technical report. Citeseer, Request for Comments.

Oura, R. and Yamaguchi, S. (2011). Fairness Analysis among Modern TCP Congestion Avoidance Algorithms Using Actual TCP Implementation and Actual Network Equipments. In *2011 Second International Conference on Networking and Computing (ICNC),*. IEEE, 297–299.

Oura, R. and Yamaguchi, S. (2012). Fairness Comparisons among Modern TCP Implementations. In *2012 26th International Conference on Advanced Information Networking and Applications Workshops (WAINA),*. IEEE, 909–914.

Pan, X.-z., Su, F.-j., Lü, Y. and Ping, L.-d. (2006). CW-HSTCP: Fair TCP in High-Speed Networks. *Journal of Zhejiang University SCIENCE A*. 7, 172–178.

Papadimitriou, D., Welzl, M., Scharf, M. and Briscoe, B. (2011). *Open Research Issues in Internet Congestion Control*. Technical report.

Patel, S. and Rani, K. (2016). Comparative Performance Analysis of TCP-Based Congestion Control Algorithms. *International Journal of Communication Networks and Distributed Systems*. 17(1), 61–75.

Paxson, V., Allman, M., Chu, J. and Sargent, M. (2011). *Computing TCP's Retransmission Timer*. Technical report.

Peng, Q., Walid, A., Hwang, J. and Low, S. H. (2016). Multipath TCP: Analysis, design, and implementation. *IEEE/ACM Transactions on Networking (ToN)*. 24(1), 596–609.

Petrov, I. and Janevski, T. (2013). Improved TCP Slow Start Algorithm. In *21st Telecommunications Forum (TELFOR-2013)*. IEEE, IEEE, 121–124.

Poojary, S. and Sharma, V. (2011). Analytical Model for Congestion Control and Throughput with TCP CUBIC Connections. In *Proceeding of ISSc Bangalore GLOBECOM,*. vol. 4. 1–6.

Poshyvanyk, D., Gethers, M. and Marcus, A. (2012). Concept Location using Formal Concept Analysis and Information Retrieval. *ACM Transactions on Software Engineering and Methodology (TOSEM)*. 21(4), 23.

Postel, J. (1981). Transmission Control Protocol. *Information Sciences Institute, University of Southern California, USA*, 1–85.

Qureshi, B., Othman, M. and Hamid, N. (2009). Progress in various TCP variants (February 2009). In *2009. IC4 2009. 2nd International Conference on Computer, Control and Communication,*. IEEE, 1–6.

Qureshi, B., Othman, M., Sabraminiam, S. and Wati, N. A. (2011). QTCP: An Optimized and Improved Congestion Control Algorithm of High-Speed TCP Networks. In *Software Engineering and Computer Systems*. (pp. 56–67). Springer.

Qureshi, B., Othman, M., Subramaniam, S. and Wati, N. A. (2013). QTCP: Improving Throughput Performance Evaluation with High-speed Networks. *Arabian Journal for Science and Engineering. Springer,*. 38(10), 2663–2691.

Rachedi, A., Lohier, S. p., Cherrier, S. and Salhi, I. (2012). Wireless Network Simulators Relevance Compared to a Real Testbed in Outdoor and Indoor Environments. *International Journal of Autonomous and Adaptive Communications Systems*. 5(1), 88–101.

Rani, R. U. and Narayanan, T. L. (2014). Innovative Congestion Control Algorithm for High Speed Networks. *International Journal of Research*. 1(5), 293–306.

Reddy, K. S. and Reddy, L. C. (2008). A Survey on Congestion Control Mechanisms in High Speed Networks. *IJCSNS-International Journal of Computer Science and Network Security*. 8(1), 187–195.

Rosberg, Z., Matthews, J. and Zukerman, M. (2010). A Network Rate Management Protocol with TCP Congestion Control and Fairness For All. *Computer Networks*. 54(9), 1358–1374.

Sansa-Otim, J., Rai, I., Hulst, J. *et al.* (2010). F-TCP: A Delay-Based Protocol with Fair Co-existence. In *2010 Ninth International Conference on Networks (ICN),*. IEEE, 34–41.

Sarolahti, P. and Kuznetsov, A. (2002). Congestion Control in Linux TCP. In *USENIX Annual Technical Conference, FREENIX Track*. 49–62.

Scharf, M., Hauger, S. and Kögel, J. (2008). Quick-Start TCP: From Theory to Practice. In *Proc. 6th Intern. Workshop on Protocols for FAST Long-Distance Networks (PFLDnet), Manchester*.

Shirazi, H. (2009). Smart Congestion Control in TCP/IP Networks. *Journal of Information and Communication Technology*. 2(2), 73–78.

Song, K. T. J., Zhang, Q. and Sridharan, M. (2006). Compound TCP: A Scalable and TCP-Friendly Congestion Control for High-Speed Networks. *Proceedings of PFLDnet 2006*.

Stanojević, R., Shorten, R. N. and Kellett, C. M. (2006). Adaptive Tuning of Drop-Tail Buffers for Reducing Queueing Delays. *Communications Letters, IEEE*. 10(7), 570–572.

Stevens, W. R. (1997). TCP Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery Algorithms. *International Journal Electronics Communication and Computer Engineering (IJECCE)*, vol. 33(4). 334–401.

Su, F.-j., Pan, X.-z., Wang, J.-b. and Wan, Z. (2006). An Algorithm for Reducing Loss Rate of High-Speed TCP. *Journal of Zhejiang University SCIENCE A*. 7(2), 245–251.

Tan, K., Song, J., Zhang, Q. and Sridharan, M. (2006). A Compound TCP Approach for High-Speed and Long Distance Networks. *25th IEEE INFOCOM 2006*.

Umeyama, S. (1991). Least-Squares Estimation of Transformation Parameters Between Two Point Patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*. (4), 376–380.

Utsumi, S., Zabir, S. M. S. and Shiratori, N. (2008). TCP-Cherry: A New Approach for TCP Congestion Control over Satellite IP Networks. *Computer Communications*. 31(10), 2541–2561.

Vallamsundar, B., Zhu, J., Ponnambalam, K., Huang, C., Srinivasan, A. and Cheng, B. (2007). Congestion Control for Adaptive Satellite Communication Systems using Intelligent Systems. In *2007. ISSSE'07. International Symposium on Signals, Systems and Electronics*,. IEEE, 415–418.

Wang, G., Ren, Y. and Li, J. (2012). DSTCP: An Improved TCP to Increase Scalable TCP's Friendliness and Stability. In *2012 IEEE 14th International Conference on Communication Technology (ICCT)*,. IEEE, 546–549.

Wang, H. and Williamson, C. (1998). A New Scheme for TCP Congestion Control: Smooth-Start and Dynamic Recovery. In *Sixth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 1998. Proceedings*. IEEE, 69–76.

Wang, J., Gao, F., Wen, J., Li, C., Xiong, Z. and Han, Y. (2014a). Achieving TCP Reno Friendliness in Fast TCP over Wide Area Networks. In *Proceeding of International Conference Computing, Networking and Communications (ICNC)*. IEEE, 445–449.

Wang, J., Wen, J., Han, Y., Zhang, J., Li, C. and Xiong, Z. (2013). CUBIC-FIT: A High Performance and TCP CUBIC Friendly Congestion Control Algorithm. *Communications Letters, IEEE*. 17(8), 1664–1667.

Wang, J., Wen, J., Han, Y., Zhang, J., Li, C. and Xiong, Z. (2014b). Achieving High Throughput and TCP Reno Fairness in Delay-based TCP over Large Networks. *Frontiers* *of Computer Science*. 8(3), 426–439.

Wang, Q. and Yuan, D. (2010). An Improved TCP Congestion Control Mechanism with Adaptive Congestion Window. In *2010 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS),*. IEEE, 231–235.

Wang, R., Pau, G., Yamada, K., Sanadidi, M. and Gerla, M. (2004). Tcp Startup Performance in Large Bandwidth Networks. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 2. IEEE, 796–805.

Wang, Z. and Crowcroft, J. (1991). A New Congestion Control Scheme: Slow Start and Search (Tri-S). *ACM SIGCOMM Computer Communication Review*. 21(1), 32–43.

Wanxiang, C., Peixin, S. and Zhenming, L. (2001). Network-Assisted Congestion Control. In *2001. Proceedings. ICII 2001-Beijing. 2001 International Conferences on Info-tech and Info-net,*, vol. 2. IEEE, 28–32.

Wei, D., Jin, C., Low, S. and Hegde, S. (2006). FAST TCP: Motivation, Architecture, Algorithms, Performance. *IEEE/ACM Transactions on Networking (ToN)*. 14(6), 1246–1259.

Wei, D. X. and Cao, P. (2006). NS-2 TCP-Linux: an NS-2 TCP Implementation with Congestion Control Algorithms from Linux. In *Proceeding from the 2006 workshop on ns-2: the IP network simulator*. ACM, 9.

Welzl, M. and Normann, R. E. (2012). A Client-Side Split-ACK Tool for TCP Slow Start Investigation. In *Computing, Networking and Communications (ICNC), 2012 International Conference on*. IEEE, 804–808.

Xu, L., Harfoush, K. and Rhee, I. (2004). Binary Increase Congestion Control (BIC) for Fast Long-Distance Networks. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 4. IEEE, 2514–2524.

Xu, W., Zhou, Z., Pham, D., Ji, C., Yang, M. and Liu, Q. (2011). Hybrid Congestion Control for High-Speed Networks. *Journal of Network and Computer Applications*. 34(4), 1416–1428.

Xue, L., Kumar, S., Cui, C. and Park, S.-J. (2014). A Study of Fairness among Heterogeneous TCP Variants over 10Gbps High-Speed Optical Networks. *Optical Switching and Networking*. 13, 124–134.

Yamamoto, H., Komada, G., Nakamura, K., Takahashi, T. and Yamazaki, K. (2011). Performance Comparison of Enhanced TCPs over High-Speed Internetworking Satellite (WINDS). In *2011 IEEE/IPSJ 11th International Symposium on Applications and the Internet (SAINT),*. IEEE, 274–278.

Yang, P., Shao, J., Luo, W., Xu, L., Deogun, J. and Lu, Y. (2014). TCP Congestion Avoidance Algorithm Identification. *IEEE/ACM Transactions on Networking,*. 22(4), 1311–1324.

Yasin, W., Ibrahim, H., Hamid, N. A. W. A. and Udzir, N. I. (2011). Performance Analysis of Transport Control Protocol Flavours in the Existence of Packet Reordering Phenomena. In *Digital Enterprise and Information Systems*. (pp. 569–579). Springer.

Yue, Z., Zhang, X., Ren, Y., Li, J. and Zhong, Q. (2012). The Performance Evaluation and Comparison of TCP-Based High-Speed Transport Protocols. In *2012 IEEE 3rd International Conference on Software Engineering and Service Science (ICSESS),*. IEEE, 509–512.

Zhang, Y., Ansari, N., Wu, M. and Yu, H. (2012). AFStart: An Adaptive Fast TCP Slow Start for Wide Area Networks. In *2012 IEEE International Conference on Communications (ICC),*. IEEE, 1260–1264.

Zhou, Y. and Yan, J. (2012). Experimental Evaluation of TCP Implementations on Linux/Windows Platforms. In *2012 21st International Conference on Computer Communications and Networks (ICCCN),*. IEEE, 1–5.