A Deterministic Loss Model Based Analysis of CUBIC

Rodolfo I. Ledesma Goyzueta, Yu Chen

Department of Electrical and Computer Engineering, Binghamton University, Binghamton, NY 13902
Email: {rledesm1, ychen}@binghamton.edu

Abstract – **Effective congestion control is one of the most critical issues in the utility efficiency of network resources. Because of better scalability and higher flexibility, CUBIC has become a widely deployed TCP congestion control protocol in high-speed long-delay networks and it is the current default algorithm implemented in the Linux kernel. However, the behavior of CUBIC is not fully understood. In this paper, a deterministic loss model has been proposed to analyze the characteristics of the concave region in the congestion avoidance state. This paper aims to provide deeper insight on the function and mechanism of CUBIC protocol. Through extensive mathematical analysis and simulation experimental study, this work verified that the CUBIC protocol effectively improved the bandwidth utility efficiency in high-speed long-delay networks.**

Keywords: CUBIC, TCP, congestion control, Markov chain.

1. Introduction

High-speed long-delay networks, also known as long fat networks, are characterized by a high bandwidth-delay product (BDP). BDP is the maximum amount of data that a sender can send into the network before any acknowledgement comes back from a receiver. It is also used to evaluate the efficiency of bandwidth utility. Ideally, to achieve a 100% resource utility efficiency, the BDP tells the amount of data that a sender should keep within the network.

However, in practice, network links are underutilized and a lot of efforts have been reported to improve the efficiency. One of the obstacles that prevent the senders from injecting more data packets into the network is the limited capabilities of congestion control protocols. In order to avoid congestions that lead to packet dropping and re-transmission, senders are requested to start with small window size and increase either additively or multiplicatively, behaving as specified by such as additive-increase multiplicative-decrease (AIMD) or multiplicative-increase multiplicative-decrease (MIMD) model.

Congestion control protocols (either AIMD, MIMD, or other derived protocols such as NewReno [1], [2], HTCP [3], HSTCP [4], STCP [5]) reduce the sending window size drastically when packet dropping happens. Consequently, the network throughput is also cut down significantly, and the link remains underutilized for a period of time. After the window size cutting, recovery and congestion avoidance are carried out by each protocol striving to occupy the link fully. Additionally, protocols generally also set a threshold of impending loss region.

To achieve better link utilization, researchers have proposed enhanced congestion control mechanisms such as FAST [6], Vegas [7], Westwood [8], BIC [9], and CUBIC [10]. Among them, CUBIC has been widely deployed in high-speed long-delay networks because of its better scalability and higher flexibility. CUBIC is adopted by about 45% of GNU/Linux servers [11], and currently it is the default algorithm implemented in the Linux kernel [12], [13].

The stability of CUBIC lies in the fact that the flow has a slow growth near the threshold's neighborhood, between the concave and convex regions. CUBIC also has a low multiplicative decrease parameter in comparison with other congestion control protocols. Meanwhile, CUBIC protocol possesses two important features. First, it follows a cubic growth function, and second, it was designed with a TCP-friendly region.

Although CUBIC has been accepted widely and its operation has impacted the performance of WANs [14], its behavior is not fully understood yet. While some researchers have tried to study the behavior of CUBIC through intensive experiments, it is still desired to analyze this protocol in a more precise and stricter way, mathematically.

In this paper, we present an analytical model of CUBIC, particularly focusing on the concave region. Considering the window growth process as a random process, this model is derived using Markov chain. In addition, the window size, the round-trip time, and the number of packets before packet dropping happens are considered as variables involved in the Markov chain's states. Through extensive simulation experiments we have verified our model.

The remainder of this paper is organized as follows. Section 2 presents a brief review of related studies on CUBIC. Section 3 describes the principle and operation of the CUBIC window adjustment algorithm, including discussions about both its growth function and its parameters. In Section 4, a deterministic loss model is derived. Section 5 analyzes the mechanism of fast convergence. Section 6 presents the results of the simulation experiments. Section 7 wraps up this paper with some discussion and conclusions.

2. Related Work

There are a lot of reported efforts in Internet congestion control or congestion avoidance mechanisms. This section presents a brief overview on studies that are closely related to CUBIC. Readers who are interested in this area are referred to more comprehensive surveys such as [15], [16].

Leith *et al*. presented some experimental results for CUBIC [17]. The topology employed was dumbbell topology. They tested some features of CUBIC, such as its slow convergence and RTT fairness. However, some information and conclusions presented had led to a rebuttal

by Rhee, one of the designers of CUBIC [18]. In [19], Bateman *et al*. provided comprehensive comparison study through results obtained from both simulation and testbed emulation experiments. The behavior and performance of CUBIC have been compared with multiple congestion control schemes, such as TCP Reno with BIC, Scalable, HighSpeed, and Hamilton.

Using FTP traffic traces, an analysis of performance under multiple scenarios is presented in [20]. The authors used *ns*-2 and compared the performance of CUBIC with different TCP algorithms such as TCP Reno, HSTCP, STCP, and Compound TCP.

In [21], an experimental evaluation of TCP CUBIC was conducted. The implementation was on top of NetFPGA board with small buffer setup. The authors' motivation was to project future network which might possess large capacity and small buffers. Recently, Bao *et al*. reported a slightly different point of view from the above mentioned papers, modeling CUBIC over wireless networks [22]. The authors derived a Markov chain model for performance analysis over wireless networks such as 3GPP, LTE, and WiMAX.

The aforementioned papers presented some performance results. They did not provide more strict analysis or an analytical model, except for [22]. However, the work reported in [22] was conducted in the context of wireless networks. In this paper, our model focuses on wired, long fat network environments.

3. CUBIC Congestion Control

This section presents a brief introduction to the CUBIC scheme. Basically, CUBIC employs ACK-clocking to adjust its window size [10], it does not change the fast recovery and fast retransmission algorithm adopted by TCP NewReno [2] and TCP SACK [23]. The window growth rate function of CUBIC is shown in Figure 1, which displays the cubic function curve and its inflection point in W_{max} (vertical axis).

In case there is packet dropping event when the congestion window size is W_{max}, CUBIC starts work. It sets W_{max} as the maximum window size. Then, CUBIC decreases the congestion window size multiplicatively with a rate β. After that, CUBIC enters the congestion avoidance phase, in which its window growth procedure can be divided into three regions, considering W_{max} as reference value [10].

CUBIC is allowed to maintain a slow growth near the last maximum value, where packet dropping has occurred.

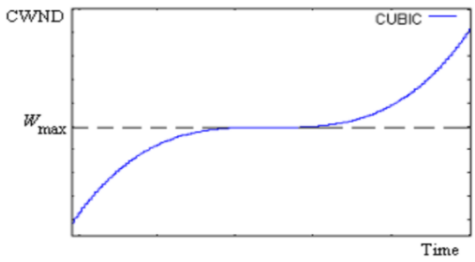


Figure 1. CUBIC window growth function.

The window adjustment function of CUBIC is [10]:

$$W(t) = C(t-K)^3 + W_{\max} \quad (1)$$

where C is a constant that determines the window growth rate (aggressiveness) in high BDP networks, t is the elapsed time from the last packet dropping, and K is the estimated time period that would take to reach W_{max}. Disregarding further packet loss, K is computed as follows:

$$K = \sqrt[3]{\frac{W_{\max}\beta}{C}} \quad (2)$$

3.1 Regions

Let `cwnd` and $W_{TCP}(t)$ be the current congestion window size and standard TCP window size in a certain time, respectively. $W_{TCP}(t)$ is defined by standard TCP window growth function. Then, CUBIC is within the TCP-friendly region if `cwnd` < $W_{TCP}(t)$. Thus, `cwnd` is set to $W_{TCP}(t)$ upon receiving each ACK.

If CUBIC is not in the TCP-friendly region and `cwnd` > $W_{TCP}(t)$, then, the protocol is in the concave region. In this region, `cwnd` increases following Eq. (1) until W_{max} is reached. Once `cwnd` ≥ W_{max}, the protocol is in the convex region. The window growth function remains the same for both concave and convex regions. As aforementioned, the growth rate is slow at the beginning near the W_{max} neighborhood.

3.2 Fast Convergence

CUBIC introduces a heuristic into the protocol to improve convergence rate. This heuristic allows existing CUBIC flows to release (share) bandwidth to incoming flows. With this released bandwidth, incoming flows have room to grow. If the fast convergence mechanism is enabled, when a packet loss event occurs, the algorithm compares the previous W_{max} with the current W_{max}. If the current W_{max} is less than the previous W_{max}, then it indicates that there are fewer available resources in the link. The algorithm below explains the heuristic:

```
if (W_max < W_last_max){
        W_last_max = W_max;
        W_max = W_max*(2-β)/2;
} else
        W_last_max = W_max;
```

Here, *W_max* (integer) is the current maximum window size, *W_last_max* (integer) is the last maximum value registered, and β is the multiplicative decrease factor.

4. Deterministic Loss Model

Assume the packet loss probability is p and the packet losses are random and independent to each other. In addition, consider a long-lived CUBIC flow that operates in the concave region of the congestion avoidance phase. To make our mathematics easier, the model is simplified by

considering its performance in the concave region, and by using the variables' expected value. The window growth process is considered as a Markov chain.

Suppose the initial window size is w, three metrics are defined as follows:

$W(r, w)$: the window size after r packets are acknowledged;

$T(r, w)$: the time taken by r packets to be sent and acknowledged;

$R_{max}(w)$: the number of packets acknowledged before a packet loss happens.

Also, assume that packet loss happens due to buffer overflow or other damaging phenomenon that prevents the receiver from acknowledging a received packet correctly. Furthermore, let us assume only one packet loss occurs at one time, and the congestion control protocol takes action for that packet loss event.

Let us define $q=1-p$ as the probability that a packet is acknowledged successfully. Assume the probability of the number of packets lost follows a geometric law. Considering that $(r-1)$ packets ($\leq R_{max}$) have been acknowledged, let us define the probability mass function for R, such that R is an independent and identically distributed random variable (i.i.d.) for the number of packets. Thus,

$$P[R=r]=(1-p)^{r-1}p \tag{3}$$

where $r=1,2,\cdots,R_{max}+1$.

The period of a cycle is $T(R, w)$, and the window size when the cycle ends is $W(R, w)$.

Considering a time-homogeneous Markov chain consisting of the states $\{w_n\}$, where n is the index for each RTT. Let us define the evolution of w_n and T_n:

$$w_{n+1}=W(R_n, w_n) \tag{4}$$

$$T_{n+1}=T(R_n, w_n) \tag{5}$$

Eqs (3), (4), and (5) define the transition probabilities for the Markov chain. $W(r, w)$ and $T(r, w)$ are i.i.d. random variables. The distribution of $T(r, w)$ has an expected value $E[T]$. Thus, the throughput x is calculated considering the expected values of the random variables R and T:

$$x=\frac{E[R]}{E[T]} \tag{6}$$

CUBIC does not depend on RTT, but continuous time that would take to increase $w=(1-\beta)\omega$ up to $W_{max}=\omega$. Furthermore, the average period is K, knowing that when w reaches $W_{max}=\omega$, it goes from the state w_i to the state w_{i+1}, and then, the cycle starts at $w=(1-\beta)\omega$ again after $R_{max+1}=R_{i+1}=1/p$ packets:

$$K=\sqrt[3]{\frac{\omega\beta}{C}} \tag{7}$$

Thus, it gives:

$$W(\frac{1}{p}, w)=W_{max} \tag{8}$$

Considering that the growth rate function is subject to the packet loss frequency, the following expression may be derived: $E[T]=K$. The expected value of packets sent is:

$$E[R]=\sum rP[R=r] \tag{9}$$

As the number of packets sent before a packet loss event is already established, and assuming that a single packet is lost in each cycle of the growth rate function, it yields $1/p=E[R]$. According to Eq. (1), we may write the protocol dynamics in each RTT with the following growth function:

$$W(t)=C\left(tRTT-\sqrt[3]{\frac{\omega\beta}{C}}\right)^3+\omega \tag{10}$$

Thus far, it is known the period of each cycle and the fact that the protocol works only in the concave region of the congestion avoidance state. Therefore, the amount of transmitted packets in each cycle is:

$$\frac{1}{p}=\int_0^{\frac{K}{RTT}}W(t)dt \tag{11}$$

The Eq. (11) calculates the number of packets per cycle. As it is shown above, the growth function has been modified. It depends on RTT, and the reference time unit is RTT as well. This modification does not affect main further calculations. For the default value of the protocol, $\beta=0.2$; Figure 2 displays the curve plotted according Eq. (10).

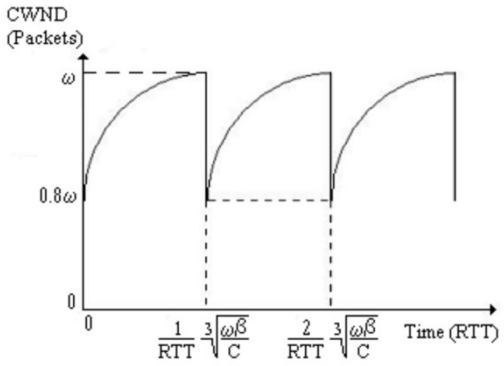


Figure 2. CUBIC window growth function under periodic loss events.

The expected window size is equal to the amount of packets transmitted during an elapsed time (taking RTT as the time unit) until a packet loss event occurs. Thus, the following expression is obtained:

$$E[W_{CUBIC}]=\frac{RTT}{Kp} \tag{12}$$

Solving Eq. (12), the expression of the average window size is calculated:

$$E[W_{CUBIC}]=\sqrt[4]{C\left(\frac{RTT}{P}\right)^3\left(\frac{4-\beta}{\beta}\right)} \tag{13}$$

Additionally, consider that packets travel on an available link for each flow and arrive to a queue as a Poisson process with a mean throughput x. This parameter x depends on Eq. (13). Therefore,

$$x = \frac{E[R]}{E[T]} = \frac{E[W]}{RTT} \tag{14}$$

5. Analysis of Fast Convergence

The fast convergence mechanism, as indicated in Section 3.2, adds heuristic in the protocol. In turn, fast convergence provides room for incoming flows into the network. Likewise, it is considered that all conditions are met to run the mechanism of fast convergence.

However, the bandwidth release has a time limit. When a flow suffers a packet loss event, it might be operating in two modes: under fast convergence or not under fast convergence. Then, after an elapsed time, the flow will reach a threshold where the window sizes of both modes are the same. Beyond this threshold, the growth rate of the flow that works under fast convergence becomes more aggressive than the same flow if it had not used fast convergence.

Consider K_1 and K_2 as estimated time periods using fast convergence mode and not using it, respectively. ω_1 and ω_2 are the reference maximum window sizes of the cubic functions using fast convergence mode and not using it, respectively. Thus, the time threshold may be calculated, knowing that at that point, the window sizes are equal:

$$C(t-K_1)^3 + \omega_1 = C(t-K_2)^3 + \omega_2 \tag{15}$$

Solving Eq. (15) gives the elapsed time to reach the threshold, and in turn, we can obtain the time limit of fast convergence. In this fashion,

$$at^2 + bt + c = 0 \tag{16}$$

where

$$\begin{aligned} a &= 1 \\ b &= -(K_1 + K_2) \\ c &= \frac{1}{3}(K_1^2 + K_1K_2 + K_2^2) - \frac{\omega_2 - \omega_1}{3C(K_2 - K_1)} \end{aligned} \tag{17}$$

The Eq. (16) has the following unique solution:

$$t_{lim} = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \tag{18}$$

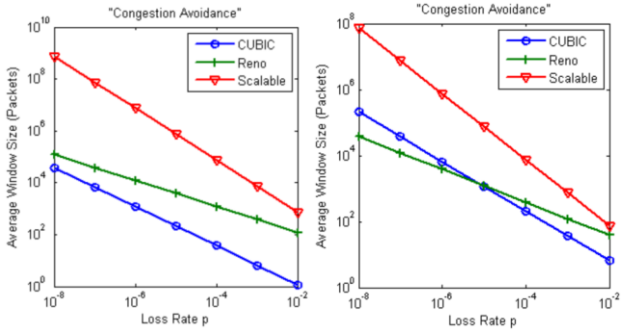
The above result may be negligible if a packet loss event occurs or the data transmission concludes before the time limit of fast convergence. Therefore, Eq. (18) is considered as the time limit t_{lim}, or duration of fast convergence.

6. Validation

This section validates the results of the analysis in the previous sections. The mathematical analysis in previous sections allows obtaining theoretical results and understanding the characteristics of CUBIC.

6.1 Behavior of CUBIC

To analyze the influence of the periodic probability as shown by Eq. (13), two scenarios are considered. Figure 3 compares the evolution of the average window size against loss rate of CUBIC, TCP Reno, and Scalable TCP. Figure 3(a) presents the average window size of a flow over a link with RTT=10 ms. Figure 3(b) displays the behavior of a flow over a link with RTT=100 ms. As shown in Figure 3(b), the mean window size of CUBIC has less aggressive evolution than STCP and TCP Reno. The first one is considerably aggressive, whereas the second one has a more cautious growth, but still amply aggressive.

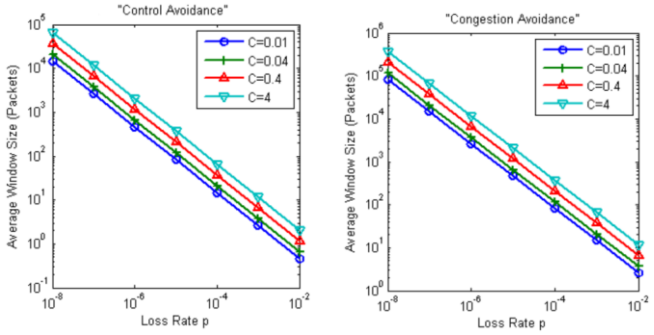


(a) RTT=10 ms. **(b) RTT=100 ms.**

Figure 3. Validation of Eq. (13), with different RTT values.

Figure 3(b) shows a better performance of CUBIC, which has a more aggressive growth in high-delay networks, STCP is even more aggressive, but with less performance than in low-delay networks. Furthermore, TCP Reno has the poorest performance in high BDP networks, with high delay. This easily demonstrates the lack of fairness of both STCP and TCP Reno flows on high-delay links.

The parameter C is a constant defined by CUBIC and denotes the window growth rate aggressiveness [10]. We compared the network response against different values of C. Figure 4 displays a better performance for CUBIC in high-delay networks when RTT=10 ms and 100ms respectively.



(a) RTT=10 ms. **(b) RTT=100 ms.**

Figure 4. Impact of parameter *C*.

Figure 5 illustrates the relationship between the parameters K and C through different values of packet loss

rates p. The chosen delay is RTT= 100 ms., because it is the network scenario where CUBIC has an improved performance as congestion control protocol. For example, when p=10^{-8} and C=0.04 gives approximately K=99 s., which is the estimated time to reach W_{max} without considering further packet loss events. Likewise, for p=10^{-8} and C=0.4, which is the default value of CUBIC, around K=55 s. is obtained. Additionally, for a constant value of C, it is shown that the degree of aggressiveness in window size growth rate is reflected in a smaller value of the estimated time K.

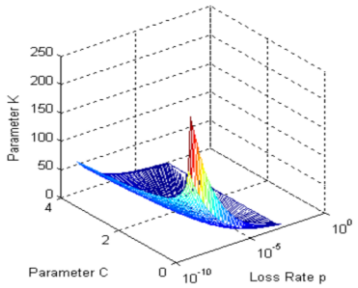


Figure 5. Relationship between *C*, *p* and *K*, RTT=100 ms.

6.2 Case study: fast convergence

The validations in this section are performed based on the time interval Δt, where Δt =[0; t_{lim}]. This interval denotes the duration of fast convergence. Figure 6 displays the available room (in number of packets) versus the duration interval of fast convergence. Available room refers to as available bandwidth shared within the duration of fast convergence so that incoming flows may occupy them, and in turn, be able to grow. As a result, the flows on the link obtain a fairer bandwidth allocation.

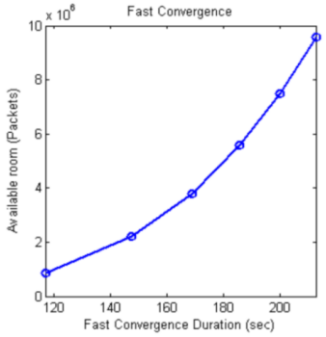


Figure 6. Available room within Δt.

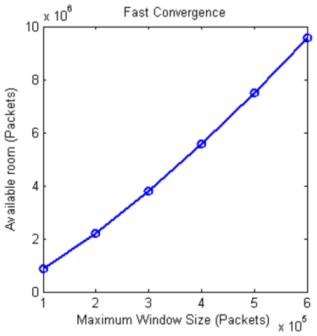


Figure 7. Available room within Δt in relation to W_{max}.

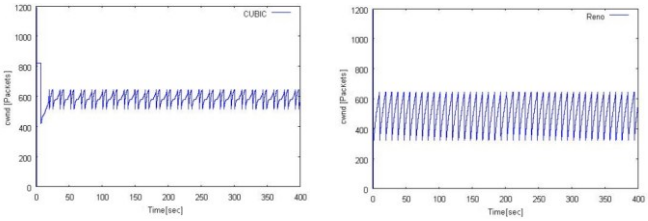
As explained in Section 4, the time limit or duration of fast convergence will be completed if it is assumed no further loss event before this period. Figure 7 displays the available room (in number of packets) versus the maximum window size of the last loss event W_{max}. This provides us a better appreciation of available packets to be taken by other flows from the point of view of the giving flow (and its maximum window size), whose bandwidth or available packet space will be available to incoming flows. As discussed earlier, the heuristic added in the congestion control mechanism detects an incoming flow through comparing the current maximum window size with the last one registered.

6.3. Utilization

The link utilization is an important metric in high-speed networks. This section reports our simulation results obtained using *ns*-2 [24]. The experiments are performed over dedicated links of 150 Mbps for each protocol. The Figure 8 considers the propagation delay of 20 ms, which is a network scenario with low BDP.

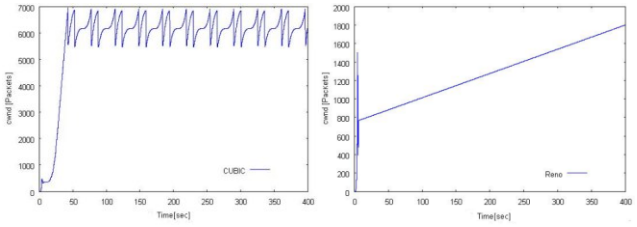
The performance of CUBIC is compared with TCP Reno. As shown in Figure 9, there are periodical packet loss events for both flows. Our simulation experimental results verified that both CUBIC and TCP Reno have a link utilization close to ideal with a network scenario favorable to TCP Reno. The average utilization of the CUBIC flow is of 95.1%, and for the TCP Reno flow is equal to 94.9%.

In the case of a high BDP network environment, the dedicated link capacity of 150 Mbps is reserved, and the propagation delay is considered equal to 380 ms. For the CUBIC flow, periodic loss events are plotted in Figure 9(a). However, for the TCP Reno flow, Figure 9(b) shows a continuous growth of the window size.



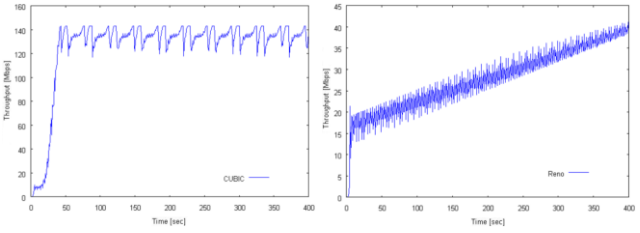
(a) Window size of CUBIC. **(b) Window size of TCP Reno.**

Figure 8. Comparison of window sizes.



(a) Window size, CUBIC. **(b) Window size, TCP Reno.**

Figure 9. Comparison of window sizes.



(a) Throughput of CUBIC. **(b) Throughput of TCP Reno.**

Figure 10. Comparison of throughput.

The case where high BDP is presented, the network environment is favorable to CUBIC. As shown in Figure 10,

the CUBIC throughput achieves higher link utilization than TCP Reno. CUBIC has an average link utilization of 83.5% in comparison with TCP Reno with 18.5%. Therefore, TCP Reno shows a high degree of link underutilization in comparison with CUBIC.

7. Discussions and Conclusions

In this paper, CUBIC congestion control protocol, which is a loss based protocol, has been analyzed. Considering packet loss rate, the CUBIC deterministic model that was developed allows us study its window adjustment features. An analysis of the fast convergence heuristic embedded in the CUBIC congestion control algorithm is presented, which verified that CUBIC improves the bandwidth distribution fairness, considering that shared room will be taken by incoming flows.

Our numerical analysis results also validated that CUBIC performance in steady state improves in high RTT networks. Furthermore, we have analyzed link utilization and compared the dynamics of CUBIC and TCP Reno through simulation experiments using network simulator *ns*-2. Both algorithms are loss based protocols. The simulations showed that the dynamic performance of CUBIC is better than TCP Reno in high RTT networks; hence, there is better bandwidth utilization in high RTT networks by CUBIC.

At present, CUBIC is still a protocol under development. There is a need for more evaluation studies of this nature. For the sake of simplicity, in this paper we assumed that packet losses are not correlated. However, by its nature, the Internet traffic is bursty on different time scales [25], [26]. Some changes are needed to make our model accommodate this property.

Our ongoing efforts also include an extension of this work, which is conducted in order to develop stochastic models of CUBIC based on random losses. Additionally, studies addressing the performance of CUBIC in wireless networks have yet to be conducted.

References

[1] V. Jacobson, "Congestion Avoidance and Control," in *Proceedings of ACM SIGCOMM'88*, Standford, CA, 1988.

[2] S. Floyd, T. Henderson, A. Gurtov, "The NewReno Modification to TCP's Fast Recovery Algorithm," RFC 3782, April 2004.

[3] R. N. Shorten, D J. Leith, "H-TCP: TCP for High-speed and Long-distance Networks," in *Proceedings of the Second PFLDNet Workshop*, Argonne, IL, February 2004.

[4] S. Floyd, "HighSpeed TCP for Large Congestion Windows," RFC 3649, December 2003.

[5] T. Kelly, "Scalable TCP: Improving Performance in Highspeed Wide Area Networks," *ACM SIGCOMM Computer Communication Review*, Vol. 33, Issue 2, pp. 83-91, April 2003.

[6] C. Jin, D. X. Wei, S. H. Low, "FAST TCP: Motivation, Architecture, Algorithms, Performance," in *Proceedings of IEEE INFOCOM*, Hong Kong, March 2004.

[7] L. S. Brakmo, S. W. O'Malley, L. L. Peterson, "TCP Vegas: New Techniques for Congestion Detection and Avoidance," in *Proceedings ACM SIGCOMM'94*, pp. 24-35, 1994.

[8] C. Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, R. Wang, "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links," in *Proceedings of ACM Mobicom*, Roma, Italia, July 2001.

[9] L. Xu, K. Harfoush, I. Rhee, "Binary Increase Congestion Control for Fast Long-distance Networks," in *Proceedings of IEEE INFOCOM*, Hong Kong, March 2004.

[10] S. Ha, I. Rhee, L. Xu, "CUBIC: A New TCP-Friendly High-Speed TCP Variant," *ACM SIGOPS Operating System Review*, Vol. 42, Issue 5, pp. 64–74, July 2008.

[11] P. Yang, W. Luo, L. Xu, J. Deogun, Y. Lu, "TCP Congestion Avoidance Algorithm Identification," in *Proceedings of IEEE ICDCS 2011*, Minneapolis, MN, June, 2011.

[12] The Linux Kernel Archives, www.kernel.org

[13] The Linux Cross Reference (LXR), http://lxr.linux.no/#linux+v2.6.28.7/net/ipv4/tcp_ipv4.c

[14] S. Ha, TCP testing results, http://netsrv.csc.ncsu.edu/wiki/index.php/TCP_Testing

[15] G. Hasegawa and M. Murata, "Survey on Fairness Issues in TCP Congestion Control Mechanisms," *IEICE Transactions on Communications*, E84-B(6):1461-1472, June 2001.

[16] J. Widmer, R. Denda, and M. Mauve, "A Survey on TCP-Friendly Congestion Control," *IEEE Network*, May/June 2001.

[17] D. J. Leith, R.N. Shorten, G. McCullagh, "Experimental Evaluation of CUBIC TCP," in *Proceedings of the 6th International Workshop on Protocols for Fast Long-Distance Networks* (PFLDnet 2008), 5-7 March, Manchester, U.K., 2008.

[18] I. Rhee, Rebuttal to "Experimental Evaluation of CUBIC-TCP" by Leith, Shorten and McCullagh, http://www4.ncsu.edu/~rhee/Rebuttal-LSM-new.pdf

[19] M. Bateman, S. Bhatti, G. Bigwood, D. Rehunathan, C. Allison, T. Henderson, D. Miras., A comparison of TCP behaviour at high speeds using ns-2 and Linux, in *ACM Proceedings of the 11th communications and networking simulation symposium* (CNS '08), pp. 30-37, New York, NY, USA, 2008.

[20] H. Jamal, K. Sultan, "Performance Analysis of TCP Congestion Control Algorithms," *International Journal of Computers and Communications*, Vol. 1, Issue 2, 2008.

[21] S. Jain, G. Raina, "An experimental evaluation of CUBIC TCP in a small buffer regime," *National Conference on Communications* (NCC), Bangalore, India, 28-30 Jan. 2011.

[22] W. Bao, V. W.S. Wong, V. C.M. Leung, "A Model for Steady State Throughput of TCP CUBIC," in *Proc. of IEEE Globecom*, Miami, Florida, December 2010.

[23] M. Mathis, S. Floyd, A. Romanow, "TCP Selective Acknowledgment Options," RFC 2018, October 2006.

[24] The Network Simulator ns-2, http://www.isi.edu/nsnam/ns/

[25] A. Botta and A. Pescape, "IP packet interleaving: bridging the gap between theory and practice," the *16th IEEE Symposium on Computer and Communications* (ISCC), Kerkyra (Corfu), Greece, June 2011.

[26] D. X. Wei, P. Cao, and S. H. Low, "Packet Loss Burstiness: Measurement and Implications for Distributed Applications," *IEEE IPDPS*, 2007.