ANALYSIS AND EXPERIMENTAL EVALUATION OF THE TRANSMISSION CONTROL PROTOCOL CONGESTION CONTROL ALGORITHMS IN A WIRELESS MULTIHOP ENVIRONMENT

RAJAN S[1], NANDHINI T[2], RAMKUMAR D[3], PUVIYA M[4], KAVYA R[5]

[1]Professor, Dept. of ECE, Velalar College of Engineering and Technology, Thindal, Erode, Tamilnadu, India
. [2,3,4,5] B.E Final Year Students, Dept. of ECE, Velalar College of Engineering and Technology, Thindal, Erode, Tamilnadu, India.

---***---

Abstract - *In today's world wireless network plays a major important role because of their characteristics such as increased mobility, installation speed, reduced cost of ownership, and wider reach of the network. Hence the number of wireless network user increases which results in high internet traffic. In order to avoid this kind of traffic on the internet, Transmission control protocol (TCP) from the transport layer of network architecture is used. TCP provides various congestion control algorithms(TCP variants). Each of them has different features but their main objective is to provide maximum throughput. The purpose of this paper is to analyze and experimentally evaluate TCP variants such as TCP cubic, TCP hybla, TCP scalable, TCP Vegas, and TCP Westwood for their throughput versus the number of nodes. Therefore this analysis of TCP variants provides a solid foundation for a robust TCP that can adapt to a highly dynamic multi-hop wireless environment.*

Key Words: Transmission Control Protocol (TCP), congestion control, TCP Cubic, TCP Hybla, TCP Scalable, TCP Vegas and TCP Westwood.

1. INTRODUCTION

Now days, demand for wireless communication system is increasing. But it becomes difficult to maintain the data rate because of multiple competing senders [14]. It may lead to degradation of throughput in case of congestion which is highly possible in wireless network. [16]

TCP congestion control algorithm from the transport layer [4] is used to maintain high network utilization by preventing network congestion. It is done by monitoring the network for congestion and adjusts the data rate accordingly. When TCP detects congestion, it reduces transmission rate to prevent further congestion. Likewise increases transmission rate when congestion subsides. There are various kinds of TCP congestion control algorithms [1] have been evaluated which is named as TCP variants.

The performance of some of the selected TCP variants such as TCP cubic, TCP hybla, TCP scalable, TCP vegas and TCP westwood are analyzed and experimentally evaluated [2]. It provides a strong foundation for creating robust TCP that can adapt to highly dynamic multi-hop wireless environment [5]. The simulation work is done on network simulator 2. [3]

2. PROPOSED SYSTEM

The proposed system's flow diagram is shown in Figure 1 below

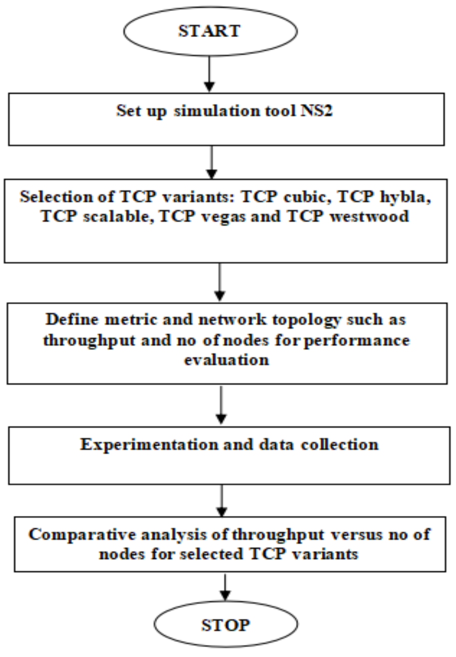
START
Set up simulation tool NS2
Selection of TCP variants: TCP cubic, TCP hybla, TCP scalable, TCP vegas and TCP westwood
Define metric and network topology such as throughput and no of nodes for performance evaluation
Experimentation and data collection
Comparative analysis of throughput versus no of nodes for selected TCP variants
STOP

Figure 1: Flow Diagram of Proposed System

In this proposed system, analysis and experimental evaluation of TCP variants [10] such as TCP cubic, TCP hybla, TCP scalable, TCP vegas and TCP westwood is presented. These variants are used to control congestion in network traffic for wireless environment. All these TCP variants can be differentiated using the metrics such as throughput versus number of nodes. From this comparative analysis [13] we can understand their performance for multiple nodes in wireless environment. We also examine their better performance for different number of nodes which is used to provide strong foundation for robust TCP that can adapt to dynamically multihop wireless environment.

2.1 TCP congestion control algorithm

The Additive Increase Multiplicative Decrease (AIMD) congestion control strategy is used by TCP. The fundamental principle of AIMD is to gradually and steadily raise the sending rate of packets until congestion is identified. When congestion is identified it decreases the sending rate of packets. AIMD functions as follows:

Additive increase: Initially, TCP starts with a low sending rate and progressively raises it by gradually increasing the sending rate by a little amount for each successful transmission, until congestion is identified. Due to the little addition it makes to the transmitting rate, this increment is called as the additive increase.

Multiplicative Decrease: When congestion is found (for instance, as a result of packet loss), TCP decreases the sending rate by more than the current sending rate by multiplying it by a factor less than 1. As it lowers the transmission rate by a greater percentage, this reduction is known as the multiplicative decrease.

Slow Start: When a new TCP connection is created, TCP begins by transferring data at a low value of congestion window(cwnd) to determine network's available capacity and then exponentially raises it until congestion is detected. The congestion window is the maximum numbers of bytes that can be transmit at any given time. This initial phase of the AIMD algorithm is called the slow start.

Congestion avoidance: once congestion window reaches a certain threshold, sender then enters the congestion avoidance phase. Here sender increases the congestion window value linearly instead of exponentially. This increase is very small to avoid causing congestion in network.

Fast Retransmit/Fast recovery: If TCP detects a lost transmission, it retransmits the packet without waiting for a timeout (for instance, because of network congestion). This technique, known as fast retransmit, lessens the delay brought on by timeouts. In addition sender enters fast recovery phase where the value of congestion window halved from its current value and then enters congestion avoidance mode.

Timeout: If no acknowledgement is received by the sender for the transmitted packet for certain period of time, it assumes packet has been lost and enters timeout phase. Here congestion window value is reduced to one and slow start phase gets restart. It ensures that sender is not overwhelming the network with too much traffic.

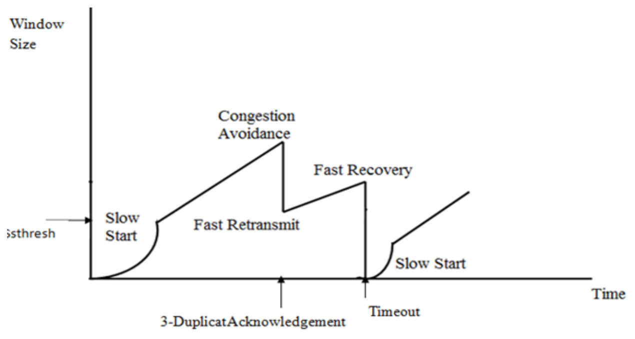

Figure 2 Behaviour of TCP congestion control algorithm

2.2 TCP cubic

The performance of TCP on high-speed and long-distance networks is improved by using TCP cubic. It is used to estimate network's available bandwidth and adaptively adjusts the sending rate of data packets for better utilization of network resources. It aims to increase the sending data rate aggressively when the available bandwidth is high, and reduces the sending rate more conservatively when congestion is detected.

Cubic function [7] is used to determine the congestion window size, where high amount of data may be transferred as quickly as possible without having to wait for a response from the receiver. The following is the mathematical formula:

$$C_w = C(t - K)^3 + W_{max}$$

Where C_w is congestion window size, C is constant that determines the rate of increase of the congestion window size, k is constant that represents time at which congestion window size was last reduced, W_{max} is the maximum window size allowed.

Starting with a small initial congestion window size, the TCP Cubic algorithm gradually expands it until congestion is noticed. When congestion is noticed, the size of the congestion window is reduced, and the algorithm starts again from the reduced size. The value of the constants C and K are dynamically adjusted based on the network conditions to optimize the performance of the algorithm.

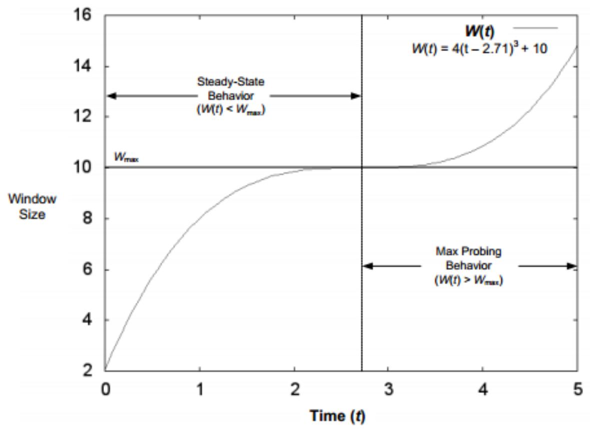

Figure 3 TCP cubic congestion control.

The above Figure 3 shows the graphical representation of TCP cubic congestion control[6].

2.3 TCP hybla

By optimizing the congestion window size based on network conditions, TCP Hybla is used to enhance TCP performance over high bandwidth and long latency networks. By dynamically modifying the congestion window size based on the current network conditions, it seeks to shorten the time needed for TCP flows to converge to a stable and fair bandwidth allocation.

The congestion window size is decreased to a low value at the beginning of the TCP connection. Then it linearly increases until the first congestion event is detected. When congestion occurs, Congestion window size is halved from its present level and size of the new congestion window is calculated using hybla equation,

$$Cc = Cn + \alpha * \Delta * (bw * rtt - Cn)$$

Where Cc is current congestion window size, Cn is new congestion window size, α is a constant value between 0 and 1 that determines the weight of the new sample in the calculation, bw is the estimated available bandwidth, rtt is the estimated round trip time, Δ is a constant value that determines the rate of window increase.

Based on available bw and network's rtt, this equation will alter the congestion window size. It takes into account both the current congestion window size and estimated available bandwidth, and scales the increased rate based on the estimated round trip time. It helps to avoid network congestion while utilizing the available bandwidth effectively.

The size of the congestion window is then raised linearly until it experiences the occurrence of congestion. The above steps were repeated for the duration of the TCP connection.

2.4 TCP scalable

TCP scalable algorithm used to enhance the sender side of standard TCP congestion control algorithm. By using the traditional TCP it improves the performance such as high speed and wide area networks. Using fixed increase and decrease parameters, the congestion window in scalable TCP can be modified. There are two phases to update the congestion window in scalable TCP.

In slow start phase: Congestion window is raised by one packet for each acknowledgement that is received.

In congestion avoidance phase: When no congestion has been identified for at least one round trip time, then window replies to each received acknowledgement with the update.

$$C_w = C_w + \alpha$$

Where C_w is a congestion window, α is constant parameter between 0 and 1. In case if congestion occurs the congestion window is multiplicatively decrease.

$$C_w = \beta.C_w$$

Where β is also a constant between 0 and 1. Typical values for α=0.01 and β=0.875.

2.5 TCP vegas

TCP vegas[8] is used to decrease packet loss and improve throughput. It provides more responsive and fair congestion control compared to standard additive increase and multiplicative decrease algorithm used by TCP. RTT (Round Trip Time) is used here to measure the state of the network.

By using the slow start mechanism, current congestion window size is determined. Then actual and expected throughput is calculated as follows.

$$\text{Actual throughput} = \frac{\text{cwnd}}{\text{RTT}}$$

$$\text{Expected throughput} = \frac{\text{cwnd}}{\text{Base RTT}}$$

$$\text{DIFF} = \text{Expected throughput} - \text{Actual throughput}$$

when DIFF is less than the threshold value, then congestion window increases exponentially. Vegas enters the congestion avoidance phase when the DIFF exceeds the threshold value as a result of the slow start phase's queue building.

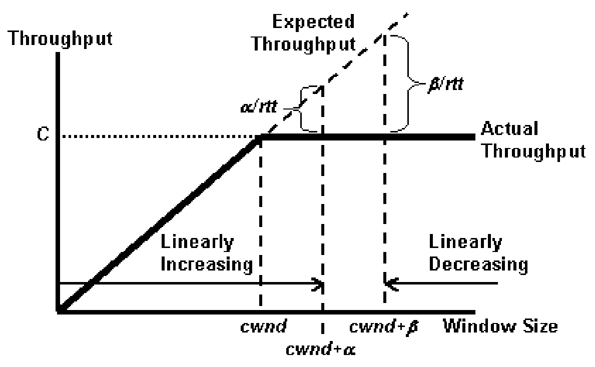

Figure 4 TCP vegas congestion avoidance.

The above Figure 4 shows the graphical representation of TCP vegas congestion avoidance.

When there is congestion, Vegas[9] determines the product of the DIFF and base RTT. A linear rise in cwnd size occurs if the product is smaller than α. A linear drop in cwnd occurs if the product is greater than β.

Otherwise, cwnd is maintained constant. If it receives a duplicate ACK, it will resend the segment without waiting for the three duplicate ACKs. As a result, segment loss is noticed sooner.

2.6 TCP westwood

The TCP standard algorithm has been enhanced on the sender side by TCP Westwood. It uses an alternative method to estimate available bandwidth. TCP westwood algorithm[11] initializes the congestion window (cwnd) to maximum segment size (MSS), where MSS is the most information that may be delivered in a single TCP segment.

During slow start phase, TCP Westwood exponentially raises the transmission rate. It uses adaptive slow start mechanism that adjusts the rate of increase based on the estimated available bandwidth. It helps to prevent overshooting the available bandwidth and triggering congestion control unnecessarily.

Once TCP westwood detects congestion, it switches to congestion avoidance phase, where it reduces the sending rate using a multiplicative decrease mechanism. Unlike TCP reno it uses alternative method to estimate available bandwidth called westwood+.

Westwood+ algorithm is used to estimate available bandwidth on the network based on the rate of acknowledge (ACK) packets. This provides feedback about the actual throughput achieved by the sender, allowing TCP Westwood to adjust the sending rate more precisely and efficiently than other TCP variants. Based on the most recent estimate of the available bandwidth, Westwood+ algorithm changes the transmission rate by calculating a congestion window reduction factor.

TCP westwood uses the fast recovery mechanism to recover from losses due to congestion. When TCP Westwood is recovering, the congestion window is halved and enters the state as "fast recovery". In this situation, TCP Westwood retransmits dropped packets and still sends new data, although very slowly. TCP Westwood quits rapid recovery after successfully retransmitting all of the lost packets and begins regular functioning.

3. RESULTS AND DISCUSSION

Table 1 simulation parameters

SIMULATION PARAMETERS	
MAC	IEEE 802.11
Routing protocol	AOMDV
Propagation model	shadowing
Simulation area	850m x 850m
Interface queue type	Queue/DropTail/PriQueue
Queue length	50

Above table 1 shows the list of the simulation parameters.

Throughput values for TCP variants such as TCP cubic, TCP hybla, TCP scalable, TCP vegas and TCP westwood has been calculated for number of nodes using network simulator 2[NS2].

The below table 2 shows the values for throughput versus number of nodes.

Table 2 Throughput versus number of nodes.

THROUGHPUT	**NUMBER OF NODES**		
	50	**100**	**150**
cubic	80.27	1993.75	11001.3
vegas	48.87	1675.15	6907.18
westwood	127.1	2057.08	9867.12
scalable	190.61	1956.72	8281.22
hybla	64.28	1993.02	8700.02

This comparative analysis technique is used to determine TCP performance in both crowded and scarce wireless environments. The simulation uses nodes that range in number from 50 to 150.

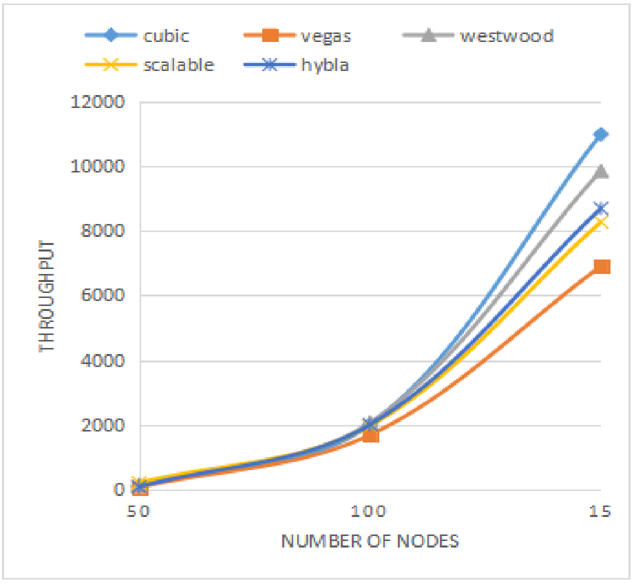

Chart -1: Graph between throughput versus number of nodes

The above Figure 5 displays the image of graph between throughput versus number of nodes. Throughput numbers rise in proportion to the number of nodes.

From the graph, it shows with minimum number of nodes scalable provides good throughput followed by westwood, cubic, hybla and vegas. However, when the number of nodes reaches upto 150, cubic provides good throughput followed by westwood, hybla, scalable and vegas.

4. CONCLUSION

In this paper successful analysis and experimentation evaluation of TCP congestion control algorithm has been demonstrated. This analysis was conducted to investigate how the congestion control algorithm behaves in a multi-hop environment. Here we investigate among TCP cubic, TCP hybla, TCP scalable, TCP vegas and TCP westwood in which scalable provides good throughput in sparse environment and cubic provides good throughput in dense environment among the other variants.

The above analysis led to the following recommendations is that the feedback mechanism should be added to the standard queue to notify the sender.

REFERENCES

[1] Allman, V. Paxson and E. Blanton, “TCP Congestion Control”, RFC 5681, 2009.

[2] Adnan Majeed, Nael B. Abu-Ghazaleh, Saquib Razak and Khaled A. Harras (2012), Analysis of TCP performance on multi-hop wireless networks: A cross layer approach, Ad Hoc Networks 10, 586–60.

[3] NS Simulator, http://www.isi.edu/nsnam /ns

[4] “Improving TCP/IP Performance over Wireless Network”, Hari Balakrishnan, Srinivasan Seshan, Elan Amir and Randy H. Katz (hari,ss,elan,randy}@CS.Berkeley.EDU Computer Science Division University of California at Berkeley.

[5] X. Zhang, J. Tang, H.-H. Chen, S. Ci, and M. Guizani, “Cross-Layer-Based Modeling for Quality of Service Guarantees in Mobile WirelessNetworks,” IEEECommunications Magazine, vol. 44, pp. 100–106,January 2006.

[6] T. A. N. Nguyen, S. Gangadhar, and J. P. G. Sterbenz, “PerformanceEvaluation of TCP Congestion Control Algorithms in Data CenterNetworks,” pp. 21–28, 2016.

[7] M. Shivaranjani, R. Shanju, M. Joseph Auxilius Jude , V. C. Diniesh, “ Analysis of TCP's Micro Level Behaviour in Wireless Multi-hop Environment”,2016

[8] Pooja Chaudhary, Sachin Kumar “Comparative Study of TCP Variants for CongestionControl in Wireless Network”International Conference on Computing, Communication and Automation (ICCCA2017)

[9] Harjinder Kaur and Dr. Gurpreet Singh, “TCP Congestion Control and Its Variants”, Advances in Computational Sciences and Technology ISSN 0973-6107 Volume 10, Number 6 (2017) pp. 1715-1723

[10] Pooja Patel; Nirali Sarang; Daizy Daruwala, “performance analysis of tcpvariants protocols forcongestion in wireless ad-hocnetwork”,International Journal of Software &Hardware Research in EngineeringISSN-2347-4890

[11] Amol P. Pande and Dr. S. R. Devane, “Study and Analysis of Different TCP Variants”, IEEE 2018

[12] Girish Paliwal, Dr. Swapnesh Taterh, “Congestion Control Algorithms Review and Analysisfor Structured Wireless Network”,JETIR October 2018

[13] Patrick Michael Hughes, “A comparison of TCP congestioncontrol algorithms for videostreaming over LTE in autonomousvehicle scenarios”

[14] Rajan S, Abinaya S, Dharanidevi V, Dharanitharan V, Dharani B, “Intelligent Shopping Cart with Online Payment for Futuristic Shopping Experience”, International Research Journal of Engineering and Technology, Vol.7, Issue 5, May 2020, pp:405-409.

[15] Girish Paliwal, Kanta Prasad Sharma, Swapnesh Taterh, Saiyam Varshney, “A New Effective TCP-CC Algorithm PerformanceAnalysis”, 2019 4th International Conference on Information Systems and

Computer Networks (ISCON)GLA University, Mathura, UP, India. Nov 21-22, 2019

[16] S.Jeneeth Subashini, D. Guna Shekar, C.Harinath Reddy, M. Manikanta , "Implementation of Wired and Wireless Networks, Analysis Simulation and Result Comparison Using Ns2", International Journal of Electrical and Electronics Research ISSN 2348-6988.