International Journal For Technological Research In Engineering
Volume 2, Issue 10, June-2015 *ISSN (Online): 2347 - 4718*

IMPROVING CONGESTION CONTROL FOR END- TO –END DELIVERY IN WIRELESS NETWORKS

Satishkumar D. Prajapati[1], Dhaval J. Varia[2]
[1]PG Student, [2]Assistant Professor, Computer Science and Engineering
Government Engineering College, Modasa, Gujarat, India

Abstract: TCP is the main traffic in the internet today both in wired and wireless network. In wireless network it is difficult to decide whether the cause of packet loss is congestion or link error. That will unnecessarily reduce the congestion window and degrade the throughput of TCP. To overcome we suggest a novel approach to use TCP Westwood bandwidth estimation to decide congestion window (cwnd). New parameter available bandwidth ratio is introduced to decide when to increase or reduce the congestion window. It also controls the aggressiveness of slow –start and congestion avoidance phase as well.
Index Terms: TCP, Cwnd, ssthresh, slow-start, congestion avoidance, Bandwidth ratio. Westwood.

I. INTRODUCTION

TCP is the main protocol in the TCP/IP suite of internet protocol. Most of the traffic in the Internet today is TCP[1-3] One of the main task of Transmission control protocol is congestion control which is done by adapting the sending rate according to available capacity in the network. Early TCP versions included a method for the receiver to control the rate at which the sender was transmitting but no algorithms for handling dynamic network congestion which was the main reason why the networks suffered from congestion collapses. Numbers of algorithms were proposed by Jacobson which were incorporated in the Tahoe version [2]. This TCP regulates its sending rate by maintaining a set of windows, sending window (swnd), congestion window (cwnd), and the receivers advertised window (rwnd) where swnd is the minimum of cwnd and rwnd and it determines the sending rate. To find an appropriate value for cwnd, TCP continuously probes for available bandwidth by increasing the sending rate. Two events we are focused on during the congestion control mechanisms are Timeouts and n-dupack.

A. Timeouts: When acknowledgement is not received in defined time duration the timeout is occurred. This event is best measure to packet lost in the network as well as congestion detection.
B. n- dupacks: Before the time out we are receiving the acknowledgements but they are not in defined order. As we receive the duplicate acknowledgements it seems something wrong happened with the data packet or segment. If n-duplicate packets are received the event is called n-dupack event. This is also helpful to measure what's happening in network. Whether the segment lost or congestion is there in network.[15]

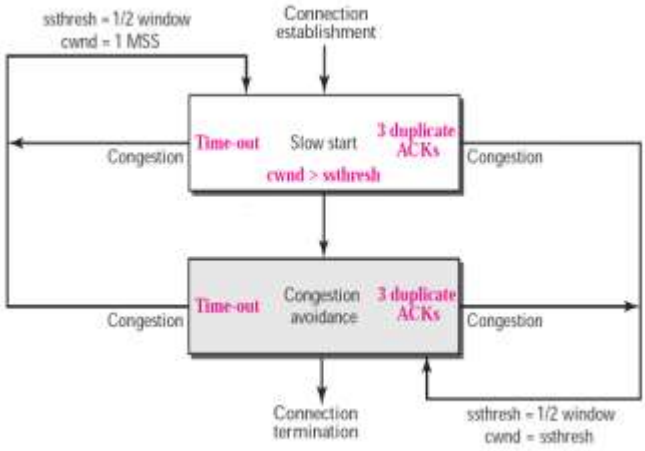

Figure1: Congestion Control Flow Chart

Figure 1 illustrate the consequences of congestion in both slow-start and congestion avoidance phase. After connection establishment sender enter into slow-start phase. Cwnd has its initial value 1 MSS(Maximum segment size) which is doubled in every round trip time. When congestion is detected by timeout in slow-start phase , ssthresh is set to half of the current window and cwnd is set at its initial value 1. But when congestion is detected by n-duplicate acknowledgements in our case it is 3, then it goes into early congestion avoidance phase by putting ssthresh at half of the current window and cwnd at ssthresh. In congestion avoidance phase cwnd increament is linear rather than exponential as in slow-start phase. When time out occure in CA state then it is considered as strong indicaton congestion and it enters into slow –start phase with initial cwnd and half ssthresh of current window size. 3-dupack in CA will lead to in CA phase with half value of current ssthresh. This the conventional mechanism of congestion control in TCP. Various variants are proposed to improve it.[14]

II. PROBLEMS IN WIRELESS NETWORKS

It is difficult to identify in wireless environment whether the loss of packet is because of congestion or due to link error. Congestion control algorithm takes every loss of packet as cause of congestion and degrade the congestion window as well as early enter in congestion avoidance phase. Multiple losses may repeatedly reduce the slow-start threshold and cause the slower congestion avoidance phase to take over immediately which leads to large throughput degradation. Mobility of nodes and handoff also causes the link failures which result in unnecessary throughput degradation.[10-11].

III. BACKGROUND WORKS

In vast literature survey, we come across different variants of the TCP congestion control. Among them bandwidth estimation based algorithms proves better in heterogeneous wireless environment. We utilize TCP westwood bandwidth estimation in our proposed work. TCP Westwood is the sender side modification of the TCP congestion control algorithm that improves upon the performance of TCP Reno in wired as well as wireless networks. The improvement is most significant in wireless networks with lossy links. An important distinguishing feature of TCP Westwood with respect to previous wireless TCP "extensions" is that it does not require inspection and/or interception of TCP packets at intermediate (proxy) nodes. TCPWestwood gives almost as well as localized link layer protocols such as the well known Snoop scheme, without incurring the overhead and specifications of a spported link layer protocol[5]

Available Bandwidth Estimation:

Suppose that an Acknowledgement is received at the source at time tk, which notifies that dk bytes have been delivered at the TCP receiver. Sample bandwidth of the connection is measured by the equation shown as bk = dk/Δk, where Δk = tk − tk-1 and tk-1 is the time the previous ACK was arrived[westwood]

$$B_i = \alpha_i \cdot B_i + (1 - \alpha_i)\left(\frac{b_i + b_{i-1}}{2}\right) \quad \text{.....(1)}$$

where αi is a co-efficient for Tustin approximation,
Bi-1 is previous bandwidth estimation value and bi-1 is previous sample bandwidth value. Other details for this can be found in[5] Now this bandwidth estimation is utilized to decide the ssthresh when congestion is detected. Pseudo code for time out event and n-dupack are given:

Steps	Westwood Timeout event
1	if (coarse time out occures)
2	ssthresh=(BWE * RTTmin) / Seg_size ;
3	If(ssthresh < 2)
4	Ssthresh = 2;
5	endif ;
6	cwin = 1;
7	endif

Steps	Westwood n-dupack event
1	if (n DUPACKs are received)
2	ssthresh=(BWE * RTTmin) / Seg_size ;
3	If(cwin > ssthresh)
4	cwin = ssthresh;
5	endif ;
6	endif

Note that seg_size identifies the length of the payload of a TCP segment in bits. During the congestion avoidance phase we are probing for extra available bandwidth. Therefore, when n DUPACKs are received, it means that we have reach the network capacity (or that, in the case of wireless links, one or more segments were dropped due to sporadic losses). In either case, the estimated bandwidth BWE is recognized to be a "feasible", i.e., achievable bandwidth (in fact, it was just measured at packet loss time). The goal is to achieve this bandwidth (more appropriately rate) with the minimum possible window, the "ideal window" in order to avoid bottleneck congestion. By definition, the ideal window is equal to the pipe size when the bottleneck buffer is empty, i.e., w = BWE * RTTmin. After the reception of n DUPACKs, TCPWestwood sets both slow start threshold and congestion window equal to the ideal window BWE * RTTmin. The standard Fast Retransmit/Fast Recoveryfollows. It should be noted that the value RTTmin is set to the smallest RTT sample observed over the duration of the connection. This setting allows the queue to be drained after a congestion episode. Also, note that after ssthresh has been set, the congestion window is set equal to the slow start threshold only if cwin > ssthresh. The rationale behind using BWE to set the slow start threshold is that TCP exploits the slow start phase to probe for available bandwidth; it thus seems natural to set ssthresh to the value we believe represented the available bandwidth at the time of congestion. In[9] proposed that the use of bandwidth estimation in the congestion avoidance phase of the TCP westwood algorithm so that in place of linear increment it can improve increment rate. For that it took bandwidth estimation at every acknowledgement and compare it with the previous bandwidth estimation by taking the ratio of both values. If ratio is high it will increment cwnd by two segments in place of conventional 1 segment. So high cwnd values can be achieved to increase the throughput in congestion avoidance phase. If the ratio is lower than the 0.5 in many cases then algorithm will not increase it's traffic. So no increment is there in the congestion window value.Beside this paper propose refinement over the retransmission time value setting as well.By combining effect of this to modification testing is done over the topology and compared with westwood as well as other variants of TCP like reno , vegas and sack. With the use of bandwidth estimation ratio it is easier to react on the loss because of the link error or because of the congestion. So It gives improved throughput than other variants. In TCP LIAD[6] uses to different threshold to define whether the we are in slow start phase or in congestion avoidance phase. More focus on improving transition on the state change from the slow start to congestion avoidance phase. Uses logarithmic increase as well as adaptive decrease in congestion avoidance phase.

IV. PROPOSED SOLUTION

In our whole work we will use the bandwidth estimation technique of TCP Westwood proposed in the equation (1). After having the estimated bandwidth we will take ratio of estimated bandwidth. Using equation (1) we will take bandwidth estimation of the current connection when an ACK arrives. Sender will calculate the ratio of the previous bandwidth estimation(Bi-1) and the current bandwidth estimation i th transmission as Bi. So the ratio can be calculated as,

BR i = B i / Bi-1 ……(2)

Here for the sake of compatibility, B0 = 1 is taken. Bi-1 is considered as previous bandwidth estimation and ratio is taken with current estimated bandwidth Bi .This ratio will reflect the traffic situation from sender to receiver in more precise manner. Based on this ratio we can say whether the network is facing delay or congestion or it is suitable to deal with more traffic load in the network. If the value of BRi ≥ 1 then it clearly shows that network is fluent with current traffic and we can increase in the load in slow start phase by 1 and in congestion avoidance phase linearly by 1/cwnd. If BRi is showing much higher than some threshold, than we can increase the congestion window more aggressively to reach the equilibrium and better utilization of the available bandwidth. If the value of BRi < 1 then it is simple to assume that network is phasing queuing delay and if we increase the offered load in the network than it leads to more delay , packet loss or may be congestion. So at that time we reduce the aggressiveness of the slow- start by reducing the increment factor of the congestion window in slow start. If we are in congestion avoidance phase it is better to slow down or decrement the size of the congestion window linearly. So cwnd = cwnd — 1/cwnd. Now BRi is near some lower threshold it is considered as congestion probe and then take action as per the westwood or reno suggest. This approach is basically improving TCP westwood by utilizing its bandwidth estimation in other standard algorithms like slow start and congestion avoidance to have better through put in wireless environment. Algorithmic steps for proposal is in simple language can be defined as follows:

Step 1 : Take the ratio of the estimated bandwidth as per the equation(2). (BRi)
Step 2 : Check Ratio is greater than or equal to 1 (BRi>=1) else go to step 5
Step 3: if in slow start phase then increase cwnd by 1
Step 4: if in CA phase then increase cwnd by 1/cwnd
Step 5 : BRi is less than 1 so decrease the aggressiveness of the slowstart
Step 6: if in CA phase then some threshold is maintained then perform linear increment
Step 7: other wise decreament the cwnd.
Step 8: End

Now we have to do modification based on proposed scheme in two phase of the congestion control algorithm. We have to modify the slow-start and congestion avoidance phases as per the bandwidth estimation ratio values tell us.

Steps	In Slow-start Phase
1	if (cwnd < ssthresh) {
2	If(br >= 1)
3	If(br <= thresh_br)
4	Cwnd = cwnd + br //aggressive SS
5	Else
6	Cwnd = cwnd + 1// normal SS
7	Else
8	If(br < 0.5)
9	Cwnd= cwnd – br
10	Else
11	Cwnd = cwnd + 1
12	}

As described in the above puedocode when we are in slow start phase cwnd< ssthresh then we will check for the estimated bandwidth ratio as per the equation(2). Then check whether it is greater than 1 then aggressive slow-start is applied. The factor of aggression which was missing in [sayed] is given here by BRi .If ratio is less than the certain limit here is 0.5 then decrement it by BRi .So adaptive decerment in LIAD[2] is also in the code.In other condition it will be as conventional slowstart having increment cwnd by 1.

Steps	In Congestion Avoidance Phase
1	if (br > 1)
2	Cwnd = cwnd + br / cwnd
3	Else
4	If(br < 0.5)
5	
6	Cwnd = cwnd – br/cwnd
7	Else
8	Cwnd= cwnd + 1 / cwnd
9	Endif
10	Endif

In congestion avoidance BRi can be utilized to provide higher linear increment of cwnd then conventional increment of unit segment. If it is less than 0.5 then decrement is there in CA phase as well. In other condition BRi between 1 and 0.5 normal congestion avoidance is applied.

V. IMPLEMENTATION AND SIMULATION

We use simulator for implementing the proposed algorithms instead of real life implementation. We have implemented proposed algorithm in Network Simulator NS-2.35 which is a discrete event driven simulator[ns2]. NS 2 is dual language co-ordination simulator. In front end TCL script is there to create topology and scenarios while C++ for protocol level implementation.[13] The topology on which we have conducted our testing is the infrasttuctural wireless network topology with one base station, one source node attached to it with base station and one wireless node to create bottleneck scenario. The wired link has bandwidth of 10Mb and delay of 45 ms between node W(0) and BS(0). While bottle neck is created by wireless link having bandwidth 2Mb and delay 0.1 ms. We run our simulation for 25 seconds and testing is done in two different enrionment. One is with 5% error rate and other is 15% error rate. After that code is saved in .tcl file. In our case it is topo.tcl We run it by ns command in the terminal with admistrative access. .nam and trace files with .tr are generated. In our case window trace file win.tr is also generated.

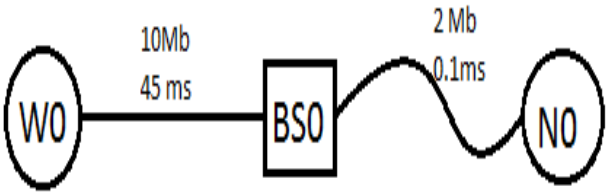

Figure 2: topology over view

The algorithm we have implemented is coded in gecmod.cc file and compiled in NS2 project. Three new variables are introduced bwr_, prev_bwr_ , threshold_br_. In westwood.cc file two core methods has been changed. The openwnd( ) and recv ( ) are updated to code the proposed scheme. Results are taken and graph are created by gnuplot and xgraph utility of NS2[16]

VI. RESULTS AND ANALYSIS

Same topology is tested for two different error model 5% and 15 % error rate.

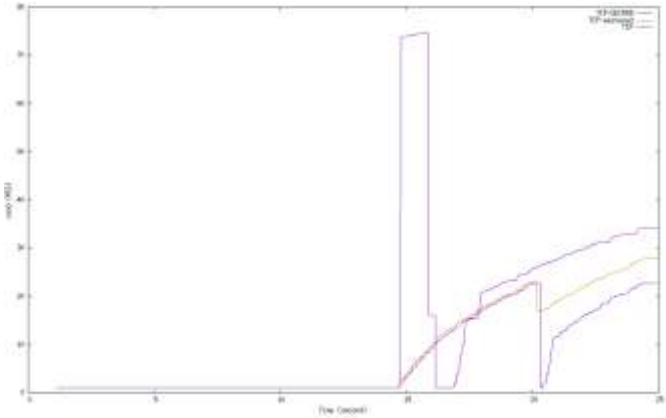

Figure 3: cwnd for 5% error rate

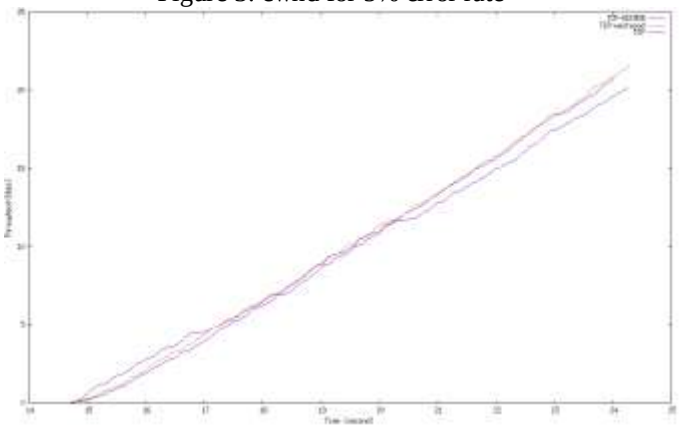

Figure 4: throughput for 5% error rate

As illustrated in figure in initial phase after the connection establishment at 14 th second gecmod scheme shows high value of congestion window. Then it shows sudden decrease because of packet loss and then adaptive to the network capacity. As in figure 4 throughput is similar to the conventional TCP and Westwood TCP.

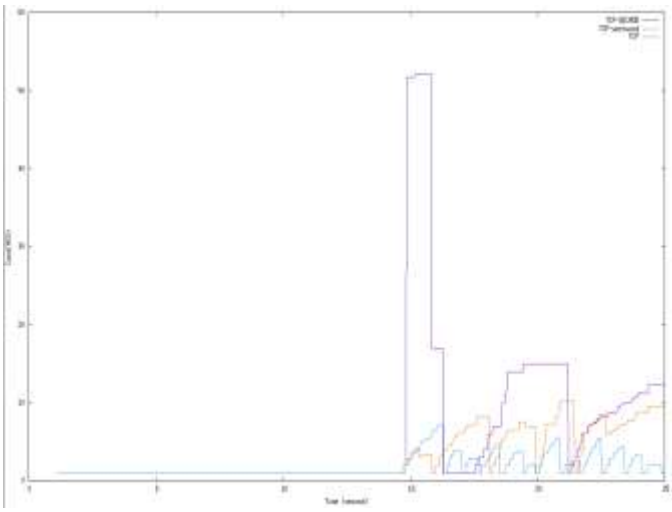

Figure 5: Cwnd for 15 % error rate

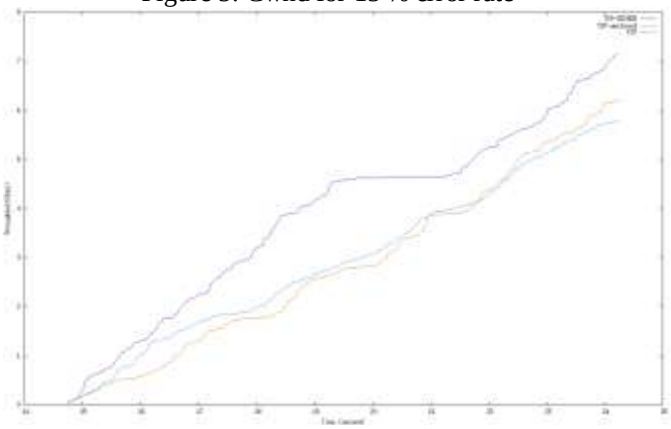

Figure 6: Throughput for 15% error rate

For Higher error rate (as 15% in our case) congestion window behavior of three TCP variants are illustrated in figure 5 . TCP –Gecmod shows better congestion window value than other two variants, TCP and TCP Westwood. While Throughput of TCP- Gecmod is higher around 25%-30% than other two variants. Because decision of decrement in cwnd is taken on the bases of bandwidth ratio, which better distinguish the loss is because of congestion or link error rate. From the graphs it is clear when loss is because of the link error in place congestion the TCP- Gecmod shows better performance.

VII. CONCLUSION AND FUTUTE WORK

According to our observations, the performances of the algorithms are strongly depends on the technique we are adopting to decide the value of cwnd size to control the offered load in the network. There are Various work has been done using different approaches like loss based , delay based and ECN as well as some mix approaches are gaining success like compound TCP and others. But they all are like Static in Nature. Bandwidth estimation based proposal are having better performance in wireless scenarios. But Still need improvements to have better throughput and good performance in high packet loss ratio are there in the network. So the new parameters are needed to decide the size of the congestion window. Novel approache of using

bandwidth ratio in deciding the value of cwnd is proving better in the higher link error rate environment. It increases the throughput in remarkable way. So we can conclude that ratio based algorithm proves better in deciding whether loss is because of the bit error or congestion. The compatibility of the proposed scheme must be tested with other variants. Fairness and friendliness need to be tested when multiple flows of data is there in simulation environment. Other parameters like delay, packet drop ratio, etc need to be tested to make it more robust solution.

REFERENCES

[1] Toni Janevaski, and Ivan Petrov, "Performances of Internet Transport protocols in wireless Environment," TELSIKS, IEEE,Serbia, vol. 9, pp. 126–129, October 2009.
[2] Van Jacobson, "Congestion control and avoidance "ACM SIGGCOMM,Computer Communication Review, vol.18(4) ,1988 , pp.314–329..
[3] Toni Janevaski, and Ivan Petrov, "Improved Slow Start Algorithms" 21 st Telecommunication Forum TELFOR,Serbia, 2013, pp. 26-28.
[4] Saleem-ullah Lar,Xiaofeng Liao, and Songtao Guo, "Modeling TCP New Reno Slow – Start and Congestion Avoidance using Simulation Approach" IJCSNS, International Journal of Coputer Science and Network Security., vol 11, January,2011 pp. 117-124.
[5] Claudio Casetti, Mario Gerla, Saverio Mascolo, M.Y. Sanadidi and Ren Wang, "TCP Westwood : End –to – end Congestion Control for Wired/Wireless networks", Wireless networks , vol 8, Kluwe Academics publications.,Nethereland, 2002, pp. 467-479.
[6] Ben-Jye-Change, Shu-Yu Lin, Jun-yu Jin ,"LIAD : Adaptive bandwidth prediction based Logarithmic Increase Adaptive Decrease for TCP congestion control in heterogeneous wireless networks," ScienceDirect ,Elsevier,Computer network vol. 53, pp. 2566–2585, 2009.
[7] K.I.Oyeyinka, A.O. Oluwatope, A.T. Akinwale, O. Folorunso, G.A. Aderounmu, O.O. Abiona, "TCP window based congestion control slow – start approach" ,Communication and networks, vol.3 pp.85-98, 2011
[8] Diwan G. Raimagia, Charvi N. Chanda, "A novel approach to enhance the performance of TCP westwood on wireless link" IEEE, Nirma University International conference in Engineering(NUiCON) pp. 1-6, 2013.
[9] Sheema Hagag, Ayeman-ul Sayed, "Enhanced TCP Westwood Congestion avoidace mechanism(TCP westwood new)", International Journal of Computer Application,IEEE, Vol. 45, 2012
[10] George C. Polyzos, George Xylomenos, Peteri Mahonen ,Mika Saaranem, " TCP Performance Issues over Wireless Links", IEEE Communication Magazine, Vol 39, pp 52-58 , 2001
[11] Gavin Holland , Nitin Vaidya, "Analysis of TCP Performance over Wireless Adhoc Netwoks", Wireless Networks, Kluwer Academics Publishers,Vol 8 . pp. 275-288, 2002
[12] G.A. Abed, Mahmaod Ismail, Kasmiran Jumari, "Comparision and analysis of Congestion Window for HS-TCP, Full-TCP, and TCP-Linux in Long Term Evolution System model" ,IEEE Conference on Open System pp.25-28, 2011
[13] The Network Simulator-ns-2, http://www.isi.edu/nsnam/ns/
[14] Andrew S. Tanenbaum.,Computer Networks,Fourth edition, pp. 547-556, PHI, New Delhi,2003
[15] W.R. Steven, "TCP/IP illustrated" pp. 579-592, Addision Vasely, New York.
[16] Tiravat Issanyakul,Ikaram Hossain,"Introduction to Network Simulator NS2", Springer.com, 2007

