Asian Journal of Information Technology 5(6): 598-602, 2006

STCP-Timestamp: Bandwidth Estimation Algorithm to Improve TCP Performance in Integrated Wireless Communication Networks

Ezil Sam Leni A.
Sathyabama Institute of Science and Technology, Chennai-119,
TamilNadu, India.600041 and Dr.S.K.Srivatsa, Professor, M.I.T-Chennai

Abstract: As wireless networks with high data rate get widely deployed, improving the performance of TCP over these networks plays vital role. Wireless link losses have dramatic adverse impact on TCP performance due to the difficulty in distinguishing the congestion losses from wireless link losses. We studied the various existing Bandwidth Estimation Algorithms. Then we proposed a new technique to estimate the available bandwidth using the timestamp option of the TCP Header. The proposed system is evaluated in the integrated wireless environment and the simulated results are also shown.

Key words: Bandwidth estimation, wireless links, congestion, wireless link errors, congestion errors

INTRODUCTION

Transmission Control Protocol (TCP)[1] is a reliable connection oriented protocol that is widely used in wired networks and wireless and mobile networks and the performance over wireless links have been studied for the last several years. The TCP has proved efficient in wired networks, showing an ability to adapt to modern, high-speed networks and in Internet. One of the big challenges is, improving the TCP performance in Internet Traffic, including the traffic generated by web accesses, e-mails, bulk data transfers, remote terminals and multimedia traffic. Internet Protocol (IP)[2] is a network layer protocol, which is connectionless, best effort and variable length packet delivery. IP does not guarantee the reliable, timely and in-order delivery of packets between end stations. On the other hand, TCP is a Transport layer protocol that uses the basic IP services to provide end-to-end connection oriented services with reliable and ordered delivery of data.

The performance degradation of TCP in wireless networks, as in much research[3,4], is mainly due to its lack of ability to differentiate the packet losses caused by network congestion from the losses caused by wireless link errors. Therefore the TCP congestion control mechanism reduces the transmission rate, even when not necessary. The existing versions of TCP, TCP-Reno, TCP-New Reno, TCP-Veno, TCP-Vegas, TCP-Westwood, are studied with their performance in integrated wireless networks.

CHARACTERISTICS AND CHALLENGES IN TCP

Characteristics: The mobile connectivity provided by the wireless networks allows users to access the information anytime anywhere. The basic characteristics of wireless networks are described below.

Bandwidth: Bandwidth need of wireless networks increases with increase in data transfer rate. Mobile users share the bandwidth within a cell and can move to another cell with higher or lower bandwidth, leading to variable wireless link data rates.

High bit error rate: As wireless link errors occurs in wireless networks, the bit error rate varies between 1%-10% of transmitted data, even though retransmission algorithms are used.

Varying delay: Delay changes due to the deviation of *RTT (*Round Trip Time) causes spurious TCP Timeouts.

Asymmetrical transmission link: Wireless networks have asymmetric uploading and down loading rates. In most TCP schemes packet loss occur in the forward link. B. Challenges that TCP Faces

ACK compression: In TCP, arrival of ACK packets in the sender side advances the congestion window thus increasing the data transfer rate. Instantaneous arrival of bursts of ACK in the reverse path degrades the Data Transfer rate because of more number of retransmission. This forces, the forward path to carry heavy traffic and exacerbates the forward path conditions.

Correlated errors: In wireless networks, as the transmission technology is Radio transmission the error process is caused by path loss, noise, fast fading, slow fading and interference from other devices. Moreover there is no linear relationship between bit error and the distance.

Rerouting: During a TCP connection, if the routing-path is changed it is not informed to the hosts and varies the *RTT* resulting discrepancies to the sender.

Clustering: In TCP, the packets from different TCP connections can share the same link. So to correctly estimate the bandwidth in use, a TCP source must observe its own link utilization for a longer time than for the entire cluster transmission. The observation time depends on how many connections share the link and the cluster size. The cluster size depends on Bandwidth-delay Product (BDP).

BANDWIDTH ESTIMATION

Bandwidth Estimation schemes are used to estimate the Bandwidth available for the sender to transmit the data with better and fairer utilization of network resources. The literature proposes several Bandwidth estimation algorithms for TCP congestion control. In this section, first we study about the various Existing algorithms pointing with their performance.

Bandwidth Estimation algorithms are classified into Reactive schemes and proactive schemes. TCP Reno, TCP New Reno, TCP SACK uses reactive mechanism and TCP Veno, TCP Westwood are the proactive mechanism. The algorithms are explained with their own characteristics and advantages.

TCP reno: It is the simple and efficient algorithm for wired networks. TCP Reno uses Additive Increase Multiplicative Decrease (AIMD) as the congestion control mechanism which uses three various phases namely, Slow-start phase, congestion Phase and Congestion Avoidance phase and maintains two state variables to regulate the transmission rate: congestion window *(cwnd)* and slow start threshold *(ssthresh)*. The *ssthresh* sets *cwnd* that discriminates the slow start phase and congestion avoidance phase. At the beginning of transmission (slow start phase), the source increases the *cwnd* exponentially until congestion occurs. Once congestion is recognized ssthresh is set to one half of the bytes in flight. When *cwnd* reaches the new *ssthresh* value, TCP enters a congestion avoidance phase during which *cwnd* is increased linearly as shown in Fig. 1.

TCP Reno is incapable of handling multiple packet losses in one transmission window, which is the very likely situation in wireless links. In TCP Reno it uses fast recovery algorithm for retransmission. After one packet drop is discovered, Reno terminates the fast recovery algorithms. So multiple packet losses make the fast recovery algorithm again and again and slow down the

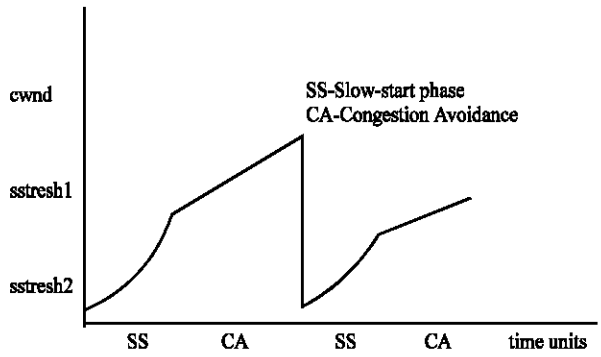

Fig. 1: TCP congestion control alogrithm

recovery of the lost packet. To overcome this difficulty TCP New Reno is implemented

TCP new reno: The TCP New Reno is a variation of TCP Reno. In this, the fast recovery algorithm of TCP Reno is modified to cope with multiple losses from a single window. The modified fast recovery algorithm doesn't terminate until multiple losses indicated by the partial acknowledgements from one window are recovered. It cannot distinguish the cause of the packet loss and so more effective fast retransmission algorithm cannot be used.

TCP veno: TCP Veno[5], estimates the backlogged packets in the buffer of the bottleneck link and uses this estimation to differentiate the random error losses from the congestive losses. According to this estimation, if the number of backlogged packets is below a threshold, the loss is considered to be random and above a threshold, the loss is considered to be congestive loss. Even though this algorithm work well in some situation, the indication is irrelevant under some critical circumstances.

TCP vegas: TCP Vegas is the more sophisticated estimation scheme. TCP Vegas computes the difference between the expected and the actual flow rate that are defined by *cwnd/RTT* min (min value measured by the TCP source) and *cwnd/RTT*, respectively. In this, when the network is not congested the actual flow rate is close to the expected one. When the network is congested actual rate is smaller than the expected flow rate. It computes the quantity using the Equation Eq. (1):

$$\text{diff} = (\text{expected\_rate-actual rate}).\text{RTT min}. \quad (1)$$

Because of the minimum RTT value it suffers from rerouting and persistent congestion problems that are stated in Section 2. Due to this limitation it is not introduced in Internet.

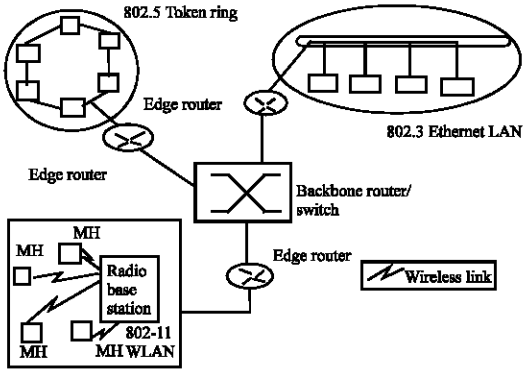

Fig. 2: Intigrated wireless communication networks

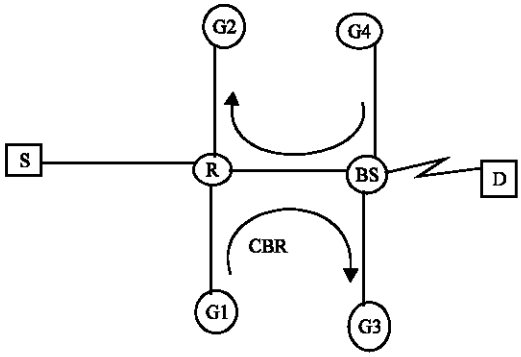

Fig. 3: Network structure

TCP westwood: TCP Westwood[6] gives end-to-end bandwidth estimation in the sender side by measuring the rate of acknowledgements. This rate is used to set *ssthresh* and *cwnd* after congestion events ie the receipt of three duplicate ACKs. This avoids reducing the data transfer rate to half in TCP Reno after packet losses and achieves fairer link utilization. TCP Westwood algorithm used for bandwidth estimation with filtering is given by

Algorithm westwood

if (ACK is received)
sample_BWE [k]=(acked*pkt_size* 8)/
(tcurrent - last_ackt);
BWE [k]=(1-β)*(sample_BWE [k]+
sample_BWE [k-1])/2
+β*BWE [k-1];
endif

Where acked is the number of segments acknowledged by the last ACK, pkt_size is the segment size in bytes, tcurrent is the current time, last_ackt is the time the previous ack was received, BWE is the Bandwidth Estimated to adjust the congestion window, k and k-1 indicates the current and previous value of the variables and β is the pole used for filtering.

PROPOSED SYSTEM

A new technique TCP-Timestamp, to estimate the bandwidth using timestamp option in the TCP Header Format. It is a 10-byte option with the following format

Timestamp option format

Code(4)	Length(4)
	Timestemp value (32)
	Timestemp echo reply (32)

The first two fields define the code and length of the option field. The timestamp value field is used by the source, when a segment leaves the source it stores current time in this field. The destination receives the segment and stores the timestamp value. When the destination sends the acknowledgement, it stores the previously stored value in the timestamp echo reply field. Both the fields are of 32 bits. When the source receives the acknowledgement, it checks the current time versus this value and is taken as the Round Trip Time (RTT). This value can be used by TCP to dynamically define the retransmission time. By computing the RTT value directly from the TCP header's optional field, no additional overhead is needed to compute the RTT value. In the Integrated Wireless Networks, we use wired and wireless networks as shown below.

In the Network topology, we have taken 802.3 Ethernet LAN, 802.5 Token Ring LAN and 802.11 WLAN as the various networks, which are integrated using edge routers in the Backbone switch. Edge routers receive data from the respective Local Area Networks and forward them to the respective destination via the Backbone Router. In the 802.11 WLAN, the Mobile Hosts (MH) are connected to the Radio Base Station using wireless Links. To study the behavior of the proposed Bandwidth estimation technique, we use the following algorithm. We have divided the Bandwidth Estimation into two steps. At first, if there is congestion in the network, it is notified. In the next step, the available Bandwidth is estimated and based on that the two parameters cwnd and ssthresh values are changed to avoid congestion caused by the traffic flow.

Congestion is notified if the sender receives three consecutive Acknowledgements. The proposed algorithm is a variation of TCP Westwood and is shown below.

Algorithm 1

if (sender receives ACK)

t1 = timestamp echo reply value;
RTT = (current_time – t1);

```
        ps = segment_size *8;
        BWE [k] = (RTT x BWE [k-1]*ps)/
                (tack [k]-tack [k-1]);
endif
```

Where ACK is the acknowledgement for the data transmitted, ps is the size of the segment, BWE is the Bandwidth Estimated, RTT is the computed Round Trip Time using Timestamp option field, tack is the time at which acknowledgement is received, k and k-1 are the current and previous indices.

Once the Available Bandwidth is estimated, in order to avoid congestion the following procedure is applied.

Algorithm 2

```
if( 3 duplicate ACKs are received)

        ssthresh= BWE  * RTT;
        if (cwnd > ssthresh)
        cwnd=ssthresh;
endif
```

In some cases, due to the wireless link losses there won't be any ACK given to the sender. At that time, the sender waits for the retransmission timer (RTO, Retransmission time out), when the timer expires, the sender is to retransmit all the transmitted packets except those ACKed. This in turn increases the Network traffic resulting in congestion. To overcome from this tragedy, we propose another procedure as below.

Algorithm 3

```
if (RTO expires)

        ssthresh= BWE  * RTT;
        cwnd=1;
endif
```

With these procedures, we have simulated the performance by comparing TCP Westwood and the proposed TCP-TIMESTAMP technique.

SIMULATION RESULTS

To compare the performance of the Proposed technique with TCP Westwood in the integrated environment we have taken the following network setup. In the network setup, the Base Station (BS) and the destination are connected using wireless links. Remaining nodes are connected using wired links. We assumed the bi-directional traffic in the entire network. In the simulation, for all the links except the wireless link the Data Transfer Rate is 100 Mbps and for the wireless link the data transfer rate is 2 Mbps.

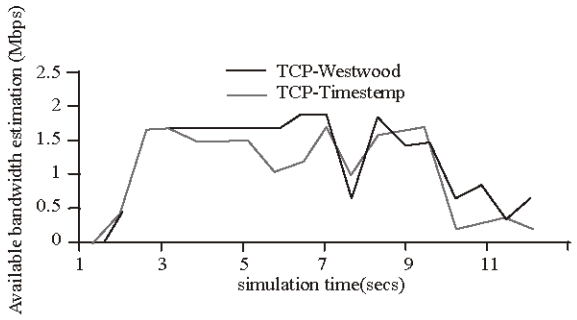

Fig. 4: Rate estimation between TCP-westwood and TCP-timestemp.

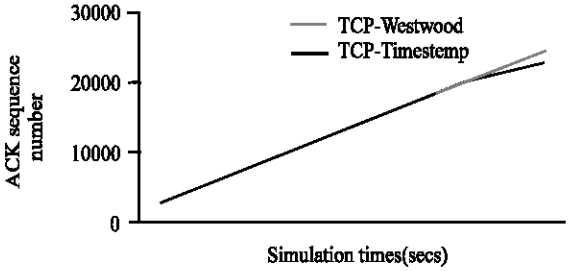

Fig. 5: Goodput comparision between TCP-westwood and TCP-timestemp

The FTP flow is introduced from G4 to G2 and CBR (Constant Bit Rate) flow is introduced between G1 to G3 to simulate the bi-directional traffic. Due to the traffic flow from G4 to G2, the rate Estimator Behavior of the proposed TCP-Timestamp with TCP-Westwood is shown in the Fig. 4

Form the above graph, the proposed technique captures the bottleneck bandwidth most of the time and Westwood exhibits significant over shoots from time to time.

The goodput is defined as the rate at which the data packets are successfully received and ACKed. The measure of the proposed technique TCP-timestamp and TCP-Westwood are shown in the Fig. 5.

With the above seen comparison it shows that TCP-Timestamp method gives good rate estimation and after certain time it gives better goodput compared with TCP-Westwood.

CONCLUSION

In this study, we have studied the various existing Bandwidth Estimation algorithms for TCP congestion control. The proposed algorithm TCP-Timestamp performs well during congested network. Even though the performance is good in normal condition, during fading wireless channels the performance of this algorithm gives degraded throughput and is the limiting factor.

REFERENCES

1. Postel, J., 1981. Transmission Control Protocol, RFC pp: 793.
2. Postel, J., 1981. Internet Protocol, RFC, pp: 791.
3. Mun Choon Chan and Ramachandran Ramjee, 2004. Improving TCP/IP Performance over 3rd Generation Wireless Networks, Infocom.
4. Abouzeid, A.A., S. Roy and M. Azizoglu, 2003. Comprehensive performance analysis of a TCP session over a wireless fading link with queuing, IEEE Transactions on wireless Communications, 2: 344-356.
5. Cheng Peng Fu, 2003. TCP Veno: TCP Enhancement fir Transmission Over Wireless Access Networks. IEEE J. Selected Areas In Communications.
6. Claudio Casetti, Mario Gerla, Saverio Mascolo, M. Yaha Sanaddi and Ren Wang, 2002. TCP Westwood: End- to-End Bandwidth Estimation for Enhanced Transport over Wireless Links, Wireless Networks, 8: 467-479.