Annals of DAAAM for 2012 & Proceedings of the 23rd International DAAAM Symposium, Volume 23, No.1, ISSN 2304-1382
ISBN 978-3-901509-91-9, CDROM version, Ed. B. Katalinic, Published by DAAAM International, Vienna, Austria, EU, 2012
Make Harmony between Technology and Nature, and Your Mind will Fly Free as a Bird
Annals & Proceedings of DAAAM International 2012

COMPARISON OF WESTWOOD, NEW RENO AND VEGAS TCP CONGESTION CONTROL

MACURA, A[rijana]; MISSONI, E[duard] & KORDIC, Z[oran]

Abstract: *TCP congestion control has been designed to ensure Internet stability along with fair and efficient allocation of the network bandwidth. During the last decade, many congestion control algorithms have been proposed to improve the classic Tahoe/Reno TCP congestion control. This paper aims at evaluating and comparing three control algorithms, which are Westwood+, New Reno and Vegas TCP. Simulation scenarios are carefully designed in order to investigate goodput, fairness and friendliness provided by each of the algorithms. Results show that Westwood+ TCP is friendly towards New Reno TCP and improves fairness in bandwidth allocation whereas Vegas TCP is fair but is not able to grab its bandwidth share when coexisting with Reno or in the presence of reverse traffic because of its RTT-based congestion detection mechanism. Finally results show that Westwood+ remarkably improves utilization of wireless links that are affected by losses not due to congestion.*

Key words: *congestion control, algorithm, Westwood, New Reno, Vegas*

1. INTRODUCTION

Internet stability is still largely based on the congestion control algorithm proposed by Van Jacobson at the end of the eighties, which is known as TCP Tahoe, on its first modification, which is known as TCP Reno and other variants described in [2]. The Van Jacobson congestion algorithm has been designed by following the end-to-end principle and has been quite successful from keeping the Internet away from congestion collapse Two variables, congestion window (cwnd) and slow-start threshold (ssthresh), are used to throttle the TCP input rate in order to match the network available bandwidth. All these congestion control algorithms exploit the Additive-Increase/Multiplicative-Decrease (AIMD) paradigm, which additively increases the cwnd to grab the available bandwidth and suddenly decreases the cwnd when network capacity is hit and congestion is experienced via segment losses, i.e. timeout or duplicate acknowledgements. AIMD algorithms ensure network stability but they don’t guarantee fair sharing of network resources.

After the introduction of the Van Jacobson algorithm research on TCP congestion control become very active and several end-to-end congestion control algorithms have been proposed since then to improve network stability, fair bandwidth allocation and resource utilization of high-speed networks and wireless networks [3]. In fact, today TCP is not well suited for wireless links since losses due to radio channel problems are misinterpreted as a symptom of congestion by current TCP schemes and lead to an undue reduction of the transmission rate. Thus, TCP requires supplementary link layer protocols such as reliable link-layer or split-connections approach to efficiently operate over wireless link [3]. Congestion control concerns controlling traffic entry into a telecommunications network, so as to avoid congestive collapse by attempting to avoid oversubscription of any of the processing or link capabilities of the intermediate nodes and networks and taking resource reducing steps, such as reducing the rate of sending packets. It should not be confused with flow control, which prevents the sender from overwhelming the receiver. This paper aims at comparing Westwood+, New Reno and Vegas TCP. New Reno TCP has been considered because it is leading Internet congestion control protocol [7]. Vegas TCP has been considered because it is also proposes, as Westwood+, a new mechanism for throttling the congestion window that is based on measuring the network congestion status via RTT measurements. Moreover, Vegas TCP provides the basic ideas behind the new fast TCP congestion control algorithm, which has been recently proposed by researches at Caltech. In particular, Westwood TCP estimates the available bandwidth by counting and filtering the flow of returning ACKs and adaptively sets the cwnd and the ssthresh after congestion by taking into account the estimated bandwidth. The original bandwidth estimation algorithm fails to work properly in the presence of ACK (acknowledgement) compression.

2. TCP WESTWOOD+

TCP Westwood is a new congestion algorithm that is based on end-to-end bandwidth estimate [8]. The estimate is obtained by filtering the stream of returning ACK packets and it is used to adaptively set the control windows when network congestion is experienced.

In particular, TCP Westwood estimates the available bandwidth by counting and filtering the flow of returning ACKs and adaptively sets the cwnd and the sshtresh after congestion by taking into account the estimated bandwidth.TCP Westwood, is a sender-side-only modification to TCP New Reno that is intended to better handle large bandwidth-delay product paths (large pipes), with potential packet loss due to transmission or other errors (leaky pipes) and with dynamic load (dynamic pipes). TCP Westwood+ is an evolution of TCP Westwood, in fact it was soon discovered that the Westwood bandwidth estimation algorithm did not work well in the presence of reverse traffic due to ACK compression. Westwood+ is friendly towards TCP Reno and fairer than Reno in bandwidth allocation.

2.1 The algorithm

This section describes the Westwood + congestion control algorithm. The Westwood+ algorithm is based on end-to-end estimation of the bandwidth available along the TCP connection path. The estimate is obtained by filtering the stream of returning ACK packets and it is used to adaptively set the control windows when network congestion is experienced. In particular, when three DUPACKs (duplicate acknowledgement) are received, both the congestion window (cwnd) and the slow start threshold (ssthresh) are set equal to the estimated bandwidth (BWE) times the minimum measured round trip time (RTT_{min}); when a coarse timeout expires the ssthresh is set as before while the cwnd is set equal to one. The pseudo code of Westwood+ algorithm is reported below:

1. on ACK reception: cwnd is increased accordingly to the Reno algorithm; the end-to-end bandwidth estimate BWE is computed;
2. when 3 DUPACKs are received: ssthresh=max (2,(BWE*RTTmin)/seg_size);cwnd=ssthresh;
3. when coarse timeout expires: ssthresh=max (2, (BWE*RTTmin)/seg_size); cwnd=1;

From the pseudo code reported above, it turns out that Westwood+ additively increases the cwnd as Reno, when ACKS are received. On the other hand, when a congestion episode happens, Westwood employs an adaptive setting of cwnd and ssthresh so that it can be said that the Westwood+ follows an Additive-Increase/Adaptive – Decrease paradigm. It is worth noting that the adaptive decrease mechanism employed by Westwood+TCP improves the stability of the standard TCP multiplicative decrease algorithm. In fact, the adaptive window shrinking provides a congestion window that is decreased enough in the presence of heavy congestion and not too much in the presence of light congestion or losses that are not due to congestion, such as in the case of unreliable radio links.

Moreover, the adaptive setting of the control windows increases the fair allocation of available bandwidth to different TCP flows. This result can be intuitively explained by considering that the window setting of Westwood+ TCP tracks the estimated bandwidth, so that, if this estimate is a good measurement of the fair share, then the fairness is improved. Alternatively, it could be noted that the setting $cwnd=BxRTT_{min}$ sustains a transmission rate $(cwnd/RTT)=(BxRTT_{min})/RTT$ that is smaller than the bandwidth B estimated at the time of congestion: as a consequence, the Westwood+TCP flow clears out its path backlog after the setting thus leaving room in the buffers for coexisting flows, which improves statistical multiplexing and fairness.

3. TCP NEW RENO

TCP New Reno improve retransmission during the past recovery phase of TCP Reno [8]. During fast recovery, for every duplicate ACK that is returned to TCP New Reno, a new unsent packet from the end of the congestion window is sent, to keep the transmit window full. For every ACK that makes particular progress in the sequence space, the sender assumes that the ACK points to a new hole, and the next packet beyond the ACKed sequence number is sent. Because the timeout timer is reset whenever there is progress in the transmit buffer, this allows New Reno to fill large holes, or multiple holes, in the sequence space-much like TCP SACK.

Because New Reno can send new packets at the end of the congestion window during fast recovery, high throughput is maintained during the hole-filling process, even when there are multiple holes, of multiple packets each. When TCP enters fast recovery it records the highest outstanding unacknowledged packet sequence number. When this sequence number is acknowledged, TCP returns to the congestion avoidance state. TCP New Reno is an improved version of Reno that avoids multiple reductions of the cwnd when several segments from the same window of data get lost [6].

4. TCP VEGAS

Until the mid 1990s, all of TCP`s set timeouts and measured round-trip delays were based upon only the last transmitted packet in the transmit buffer. University of Arizona introduced TCP Vegas, in which timeouts were set and round-trip delays were measured for every packet in the transmit buffer. In addition, TCP Vegas uses additive increases in the congestion window [4]. TCP Vegas was the first attempt to depart from the loss-driven paradigm of the TCP by introducing a mechanism of congestion detection before packet losses. TCP Vegas ensures network stability but is not able to grab its own bandwidth share when interacting with algorithms that systematically hits network queue capacity as Reno. In particular, TCP Vegas computes the difference between the actual input rate and the expected rate, where RTT is the Round Trip Time and RTT_{min} is the minimum measured round trip time to infer network congestion. In particular, if the difference smaller than a threshold α then the cwnd is additively increased, whereas if the difference is greater than another threshold ß then cwnd is additively decreased; finally if the difference is smaller than ß and greater than α, then the cwnd is keep constant. TCP Vegas ensures network stability but is not able to grab its own bandwidth share.

5. SIMULATION-BASED COMPARISON

5.1 A simple single-connection scenario

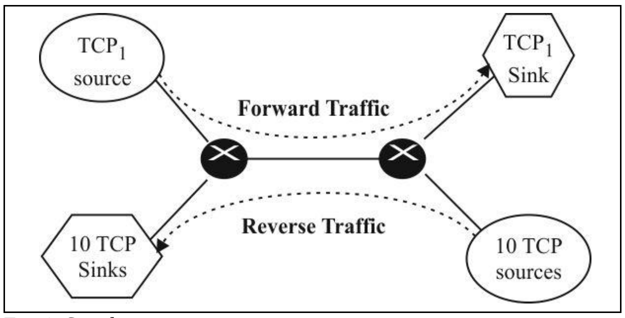

Fig. 1. Simulation scenario

In this section we evaluate and compare Westwood+, New Reno and Vegas TCP. Simple scenarios are considered in order to illustrate the fundamental features of the considered protocol dynamics whereas more complex topologies are considered to test the protocols in more realistic settings. In order to analyze the fundamental dynamics of the considered TCP congestion control algorithms, we start by considering the single connection scenario depicted in Fig. 1. The TCP_1 connection is persistent and sends data over a 2Mbps bottleneck link. The RTT (Round Trip Time) is 250 ms. Ten ON-OFF New Reno TCP Senders inject traffic along the ACK path of the single TCP connection, i.e. they generate reverse traffic for the TCP connection on the left side of Fig. 1. Reverse traffic aims at provoking congestion along the ACK path at exciting ACK compression, which is important to be considered since it exacerbates the bursty nature of the TCP.

The TCP_1 scenario starts at t=0. The 10 TCP connections on the backward path follow an OFF-ON-OFF-ON pattern in order to investigate the effect of reverse traffic. In particular, the reverse traffic is ON during the intervals [250s, 500s] and [750s,1000s] and is silent during the intervals [0s, 250s] and [500s, 750s].

Table 1. reports the goodputs that have been measured during each interval.

	[0s, 250s] (Mbps)	[250s, 500s] (Mbps)	[500s, 750s] (Mbps)	[750s, 1000s] (Mbps)
New Reno	1.86	1.62	1.99	1.64
Vegas	1.97	0.48	1.97	0.51
Westwood+	1.86	1.68	1.99	1.69

Tab. 1. Goodputs of a single TCP connection

The Goodput has been computed as follows:

$$\text{Goodput} = \frac{(\text{sent\_data-retransmitted \_data})}{\text{transfer\_time}} \quad (1)$$

When the reverse traffic is OFF, goodputs of all considered TCPs approaches the bottleneck capacity. However, when the reverse traffic is ON Vegas provides the worst goodputs whereas Westwood+ obtains a slightly better goodput with respect to TCP New Reno.

5.2 Single bottleneck scenario

The scenario depicted in Fig. 2, where M TCP sources with different RTTs share a 10Mbps bottleneck link, is particularly suited for evaluating goodput and fairness in bandwidth allocation. The M TCP flows are persistent and controlled by the same algorithm in order to evaluate the intra-protocol fairness. RTTs are uniformly spread in the interval [20+230/M, 250]ms, with M ranging from 10 to 200, to investigate the fairness with respect to the RTT. Simulations last 1000s during which all the TCP sources send data. In order to obtain ACK compression, 10 TCP New Reno senders inject traffic along the ACKs path of the M connections. The total goodput is defined as the sum of the goodputs of all the M TCP connections on the forward path.This phenomen is due to the TCP traffic on the backward path, which has a significant impact on Vegas TCP.

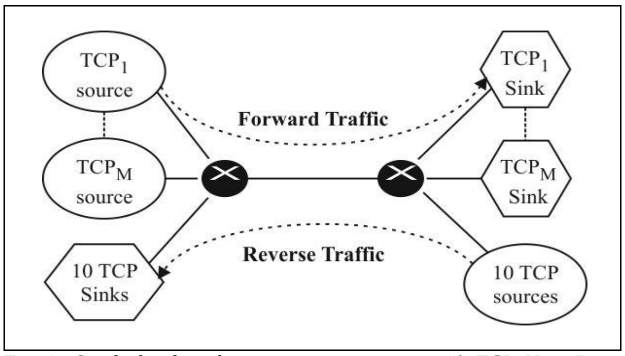

Fig. 2. Single bottleneck scenario compression, 10 TCP New Reno senders inject traffic along the ACKs path of the M connections

5.3. Multi bottleneck scenario

The multi-bottleneck scenario is particularly suited to investigate the inter-protocol friendliness of New Reno, Westwood+ and Vegas TCP. The topology depicted in Fig. 3. Is characterized by: (a) N hops; (b) one persistent connection C_1 going through all the N hops; (c) 2N persistent sources $C_2, C_3, C_4 \ldots C_{2N+1}$ transmitting cross traffic data over every single hop. The capacity of the entry/exit links is 100 Mbps with 20 ms propagation delay. Router queue sizes have been set equal to 125 packets, which corresponds to the bandwidth delay product of a typical RTT of 150 ms. Simulation lasts 1000 s during which the cross traffic sources are always active. The connection C1 is persistent and starts at time t=10 s. Notice that the described scenario is a "worst case" scenario for the source C1 since: (1) C1 starts data transmission when the network bandwidth has been grabbed by the cross traffic sources; (2) C1 has the longest RTT and experiences drops at each router it goes through.

We will consider one scenario: The $C_2, C_3, C_4 \ldots C_{2N+1}$ sources of cross traffic are controlled by New Reno TCP whereas the C1 connection is controlled by New Reno, Vegas or Westwood+, respectively. This scenario aims at comparing New Reno, Vegas or Westwood+, when going through an Internet dominated by New Reno traffic. In other terms, this scenario allows us to investigate the capacity of New Reno, Vegas and Westwood+ to grab network bandwidth when competing with New Reno cross traffic, which is the friendliness of New Reno TCP towards Vegas or Westwood+ TCP.

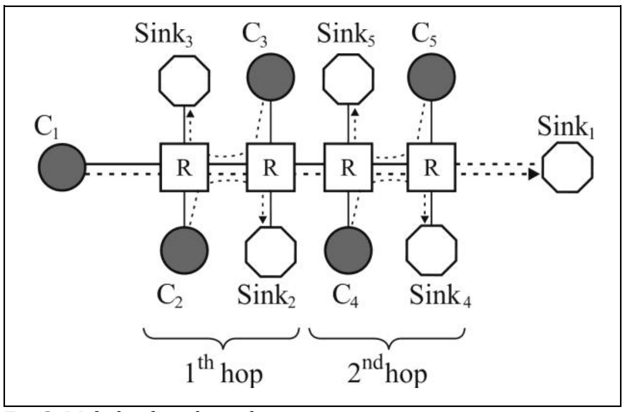

Fig. 3. Multi bottleneck topology

5.5 Mixed wired/wireless scenario

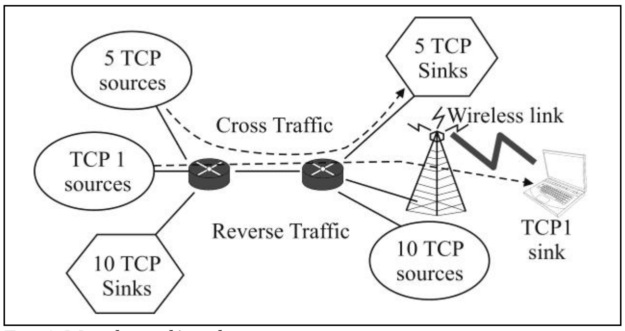

Fig. 4. Mixed wired/wireless scenario

This section aims at investigating the behavior of TCP over wireless links that are affected by losses not due to congestion. This case is particularly interesting since it is well known that protocols that react to losses by multiplicatively decreasing the control windows do not provide efficient utilization of lossy channels. In this scenario we consider Westwood+, New Reno and also TCP SACK (selective acknowledgement) to investigate the efficiency of SACK to recover from sporadic random losses. Since TCP SACK by default does not use delayed ACK, we consider Westwood+ and New Reno with delayed ACK (default case) and without delayed ACK in order to get a fair comparison. The first scenario we consider is the hybrid wired/wireless topology shown in Fig. 4.

The TCP1 connection goes through a wired path terminating with a last hop wireless link. The wireless last hop models a mobile user accessing the Internet using a radio link as in the case of a cellular telephone system. The one way delay of the TCP1 connection is 125 ms with 20 ms delay on the wireless link, which is a 2 Mbps link. RTTs of the 5 cross traffic connections and of the 10 New Reno backward traffic connections are uniformly spread in the intervals [66ms,250ms] and [46ms,250ms], respectively. We consider a wireless link affected by bursty segment losses in both directions. In particular, we assume a segment loss probability equal to 0, when the channel is in the Good state, and equal to 0.1 when the channel is in the Bad state. The permanence time in the Good state is assumed deterministic and equal to 1s whereas the permanence time in the Bad state is assumed also deterministic but this time we consider values ranging from 0.1 ms to 100 ms. When the permanence time in a state elapses, the state can transit to a Good or Bad state with a probability p=0.5. For each considered case, we run 10 simulations by varying the seed of the random loss process. For each value of the BAD state duration we report the maximum, minimum and average goodputs. In order to analyze only the impact of bursty losses on the TCP behavior, we have first turned off the cross and reverse traffic sources. This simple scenario is particularly useful to investigate the effectiveness of the adaptive decrease paradigm when losses not due to congestion are experienced by the TCP.

Westwood + improves the goodput for a large set of channel conditions. In particular, when the delayed ACK option is enabled, Westwood+ increases the utilization of the wireless link from 70% to 230% with respect to New Reno. One further point valuable of investigation is when Westwood+ shared the wired portion of the network with several TCP flows on the forward and backward paths. For that purpose, we turn on the cross and reverse traffic in Fig. 4, and we measure the goodput of the TCP1 connections for various values of the BAD state duration.

6. CONCLUSION

When a new protocol is proposed, it is necessary to collect a large set of simulation and experimental results in order to assess its validity and advantages of its deployment in the real Internet. A detailed evaluation and comparison of Westwood+, New Reno and Vegas TCP congestion has been developed through this paper. This paper compares Westwood+, New Reno and Vegas TCP.

The average goodput of a measurements session is obtained by averaging the goodputs of the session uploads. It turns that Westwood+ provides goodput improvements ranging from 23% to 53% with respect to New Reno. New Reno TCP has been considered because it is the leading Internet congestion protocol. TCP Vegas has been considered because it also proposes, as Westwood+, new mechanism for throttling the congestion window that is based on measuring the network congestion status via RTT measurements. There is the inter-protocol friendliness of Westwood+ and New Reno whereas Vegas is not able to grab its bandwidth share when coexisting with New Reno or Westwood+.

We expect that Westwood + will provide larger goodput improvement. The future researches will continue in direction to discoveries of new mechanisms and creation of new version of TCP protocols aiming in combination of existing version of TCP Westwood+, New Reno and Vegas, which would have superior performances.

7. REFERENCES

[1] Evans, J. &Filsfils, C. (2007) Deploying IP and MPLS QoS for Multiservice Networks: Theory and Practice, Morgan Kaufmann, ISBN 0-12-370549-5, Massauchusetts, USA

[2] Floyd, S.&Fall, K.(1999)Promoting the use of end-to-end congestion control in the Internet, IEEE/ACM Transactions on Networking, Vol.7, No.4, August, 1999., pp. 458-472, ISSN 1063-6692

[3] Krishnan, R.; Allman, M.; Partridge, C.& Sterbenz, J.P.G.: Explicit Transport Error Notification (ETEN) for Error Prone Wireless and Satellite Networks, BBN Technical Report No. 8333, March 22, 2002

[4] Kurose, J. &Ross, K. (2008) Computer Networking – A Top-Down Approach (4th Edition), Addison Wesley, ISBN 978-0136079675, Boston, USA

[5] Mascolo, S.; Casseti, C.; Gerla, M.; Sanadidi, M. &Wang, R.: TCP Westwood: End to End Bandwidth Estimation for Efficient Transport over Wired and Wireless Network, Proceedings of ACM Mobicom, July 2001, Rome, Italy, pp. 287-297, ISBN 1-58113-422-3

[6] Marsan, M.A.; Corazza, G.; Listanti, M.&Rovery, A.: Quality of Service in Multiservice IP Networks, Second International Workshop, QoS-IP 2003, Milano, Italy, February 24-26, 2003, Proceedings Lecture Notes In Computer Science 2601 Springer 2003, ISBN 3-540-00604-4

[7] *** http://en.wikipedia.org/wiki/Congestion_control,(2012) Accessed on: 2012-05-08

[8] *** http://en.wikipedia.org/wiki/TCP_congestion_avoidance (2012) Accessed on: 2012-05-08