A study of the behavior of TCP in variable-bandwidth environments

Authors: Left blank for double-blind review

Abstract

Most of the studies on TCP have assumed than the bottleneck bandwidth remains constant over time. In wired networks, the available bandwidth for best-effort traffic is usually constant but major changes are foreseen as many telco operators and Internet providers (ISP) are beginning to deploy Quality of Service (QoS) features with reservation-like or priority-like mechanisms in their networks. These new technologies, along with the strong desire to provide bandwidth-on-demand features, may turn wired networks into highly variable-bandwidth environments (VBE) for the best-effort traffic. In this paper we present a preliminary study of TCP in VBE and show simulation results to better understand the behavior and the performances of TCP in such environments. Both sine-based and step-based bandwidth variations models are used.

1 Introduction

TCP (Transmission Control Protocol) defined in RFC 793 is used by more than 80% of the traffic on the Internet. One key point of TCP is its congestion control mechanism which regulates the millions of flows of the Internet to provide both efficiency and fairness. TCP's performances have been extensively studied by the research community this last decade, both theoretically [16, 3, 11, 17, 18] and experimentally [2, 7], and many enhancements and optimizations to the original proposition have also been made in order to support a wider range of network conditions ([4, 12, 9, 14, 15] to name a few).

Most of the studies have assumed than the bottleneck bandwidth remains constant over time. This assumption is not true in wireless networks because of the natural dependency of bandwidth to the transmission quality (which is very subject to climatic and interference phenomenon). Hand-overs may also introduce bandwidth variations if the user moves to a lower capacity cell. In wired networks, the available bandwidth for best-effort traffic is usually constant and is set to the link capacity most of the time (because the Internet is mainly a best-effort network). However, major changes are foreseen as many telco operators and Internet providers (ISP) are beginning to deploy Quality of Service (QoS) features with reservation-like or priority-like mechanisms in their networks (both access and backbone networks) to guarantee a given QoS level. Note that some network technologies may also introduce dynamic bandwidth features [5, 10]. These new technologies, along with the strong desire to provide bandwidth-on-demand features, may turn wired networks into highly variable-bandwidth environments (VBE) for the best-effort traffic.

Dutta and Zhang [8] showed in a preliminary that TCP (NewReno) behaves quite badly when the bandwidth is varied over time. In this paper, we try to better understand the behavior of TCP in VBE and make some fundamental observations for TCP in VBE. For this purpose, we use both sine-based and step-based variations to study the general behavior of TCP and then use simulations to validate the impact of several system parameters such as router's buffer size and the RTT (Round Trip Time). The impact of queue management strategies will be left for future works. The paper is organized as follows. Section 2 presents the bandwidth variation models. Section 3 investigates the behavior of TCP in VBE and presents the main characteristics we identified. Section 4 presents simulation results that validate the previous investigations. Section 5 concludes the paper and gives future work directions.

2 Bandwidth variation models

Two variation models are used in this paper to investigate the performance of TCP: a continuous variation and a discrete variation model. In the rest of the paper, *a guaranteed TCP flow or traffic* will be referred to as a TCP flow or traffic with negotiated resources enabling strong QoS guarantees. We do not assume how these resources are negotiated and assigned to the flow, but only assume that the resulting effect on the best effort traffic is a decrease in the available bandwidth.

2.1 Continuous variations

So-called continuous variation models are those where the increase or decrease of the available bandwidth for the

best effort traffic does not present significant jumps as time is varied. In the real world, pure continuous variations (in the mathematical meaning) do not exist as the granularity of bandwidth is at the packet level, however, if the change in bandwidth is not significant when Δt is small then we will refer such variations to as continuous.

Figure 1 shows a 200Mbps link with both a pure sine-based bandwidth variations from 200Mbps to 20Mbps with a period of 30s (dash line) and a more complex variation model (plain line). The y-axis represent the bandwidth available for best-effort traffic over time (therefore the bandwidth reserved for guaranteed traffic is 200-y). This bandwidth variation model could represent a large number of guaranteed TCP flows coming in and out, resulting in variations for the best effort traffic.

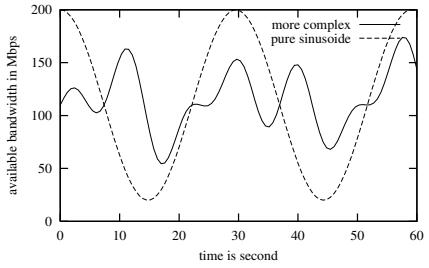


Figure 1. Sine-based continuous bandwidth variation model.

Dutta and Zhang [8] used a pure sine-based variation model to study TCP's performances. They did not claim that such a simple function could represent closely the real variations that could be found in the Internet, but rather that such a model could be used to evaluate TCP in a VBE. We totally agree with this statement and use sine-based variations to model continuous variations and believe that such patterns (especially the more complex one) could be used to evaluate the behavior of TCP in core network's links where the aggregation level is very high.

2.2 Discrete variations

So-called discrete variation models are those where the increase or decrease of the available bandwidth for the best effort traffic does present significant jumps (with discontinuity) as time is varied. Figure 2 illustrates the step model we use in this paper to represent a discrete variation model. When compared to the previous model, the discrete model tries to represent large and sudden variations of the available bandwidth. Such variation patterns could model dynamic bandwidth provisioning of guaranteed traffic, especially in the access networks where the aggregation is not so high.

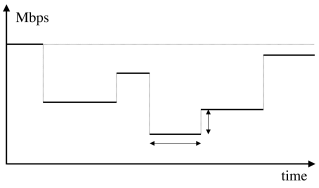


Figure 2. Step-based discrete bandwidth variation model.

3 Understanding TCP on variable-bandwidth environments

3.1 A quick review of TCP Tahoe/Reno/NewReno

TCP Tahoe is the TCP protocol proposed by Jacobson after the congestion collapse observed in 1986. In TCP Tahoe, the sender maintains a congestion window (usually noted $cwnd$) that limits the amount of packets that it is allowed to send into the network without waiting for acknowledgments. The congestion control mechanism is a 2-phase control: a slow-start phase and a congestion avoidance phase. Following is a very simple description of the TCP Tahoe/Reno/NewReno protocols, readers are invited to read the original papers ([13], RFC 2581, RFC 2582) for the detailed and full description of the algorithms. TCP Reno/NewReno are evolutions of TCP Tahoe.

TCP Tahoe: the TCP sender dynamically increases/decreases the congestion window size according to the congestion level of the network, which is conjectured by packet losses. Switching from the first phase to the second phase depends on the slow-start threshold (usually noted $ssthres$): starting with $cwnd = 1pkt$, if $cwnd < ssthres$ then $cwnd$ is doubled at every RTT (1, 2, 4, 8,...), otherwise, it is increased by 1 pkt. When a timeout occurs or more than 3 duplicated ACKs are received (indicating either severe losses or maybe a single segment loss), $ssthres$ is divided by 2 and $cwnd$ is set to 1 in order to start a new slow-start phase. $ssthres$ could initially be set as high as possible. Note that fast retransmit can be used to retransmit the lost packet (and only this one) when duplicated ACKs are received without waiting for the timeout. This congestion control follows the Additive Increase Multiplicative Decrease (AIMD) principle to achieve fairness with additive increase of 1 packet in the congestion avoidance phase and a multiplicative decrease of 1/2 when a congestion occurs.

TCP Reno: when more than 3 duplicated ACKs are received, fast retransmit is used as in TCP Tahoe but a new fast recovery mechanism is implemented by dividing $ssthres$ by 2 and setting $cwnd = ssthres$, followed with

a congestion avoidance phase (and not a slow-start phase as in TCP Tahoe, slow-start is only triggered by a timeout).

TCP NewReno (RFC 2582): in case of multiple losses within the same window of data, TCP NewReno uses the first acknowledgment received after the fast retransmit procedure to decide if more retransmissions should be performed. TCP NewReno is the most deployed version of TCP on the Internet.

TCP achieves good fairness among other flows but known problems of TCP are low efficiency and high oscillations on high $bandwidth.delay$ product networks.

3.2 TCP on variable-bandwidth environments

On a steady-state, TCP is most of the time in the congestion avoidance phase and therefore most of the studies have focused on evaluating TCP's performances when the congestion window is increased by 1 packet every RTT. As the RTT increases, the throughput increment per second is decreased. When RTT is not too small, we can use a simple relation to link the sender instantaneous throughput (T) to the congestion window size ($cwnd$) as follows:

$$T(t) = \frac{cwnd(t) * 8 * MSS}{RTT} \quad (1)$$

where T is expressed in bits/s and MSS the maximum segment size in bytes (usually a value of 1024 is used). Since we consider the congestion avoidance phase, $cwnd(t)$ could be expressed as:

$$cwnd(t) = cwnd(t_a) + \frac{(t - t_a)}{RTT} \quad (2)$$

and T could be expressed as a function of t with a linear function as follows:

$$T(t) = \frac{T_b - T_a}{t_b - t_a}(t - t_a) + T_a$$

where T_a and T_b are respectively the throughput at time t_a and t_b. When $cwnd(t_a)$ is known this relation could be rewritten as follows:

$$T(t) = \frac{8 * MSS}{RTT^2}(RTT * cwnd(t_a) + (t - t_a)) \quad (3)$$

by using equations (2) and (1) to express T_a and T_b as a function of $cwnd(t_a)$. This very simple relation links the throughput increase slope to the RTT. In the rest of this section, we will use these relations to either get $cwnd$ in function of T, or T in function of $cwnd$.

3.2.1 Sine-based bandwidth variations

Under sine-based variations, figure 3 depicts a simple view of a TCP connection and shows as an example a throughput increase slope of about 0.02 (dashed line) in the congestion avoidance phase (meaning that throughput is increased at the rate of 0.02Mbps per second) when there are no packet losses. We have then represented with the dashed-arrows one possible evolution of the sender throughput (this is a simple view since slow-start has not been taken into account): when the throughput line crosses the bandwidth variation curve while it is decreasing (it is actually the only possibility), there are drops and $cwnd$ will eventually restart from 1. There may be several re-initialization phases until the bandwidth curve increases again. When this happens, if the throughput increase slope is much smaller than the bandwidth increase slope then the efficiency is very low (when the RTT is small, it is possible to have a higher throughput increase slope). The amount of wasted bandwidth is the surface captured by the bandwidth variation curve and the throughput increase line, between time t_a and time t_b.

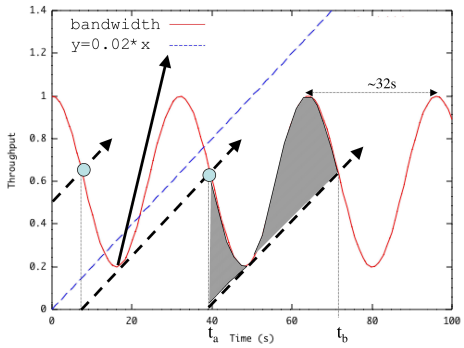


Figure 3. Simple view of a TCP connection.

We could expressed the inefficiency between time t_a and time t_b with the following relation:

$$I_{ab} = 1 - \frac{T_{ab}}{B_{ab}} \quad (4)$$

where B_{ab} is the available bandwidth and T_{ab} the achieved throughput between time t_a and time t_b. B_{ab} and T_{ab} could be expressed by:

$$\begin{cases} B_{ab} = \int_a^b B(t)dt \\ T_{ab} = \int_a^b T(t)dt \end{cases}$$

$T(t)$ is known and has been expressed in equation 3. If we get back to the simple sine variation depicted in figure 3 where we used a sine-based B(t) function defined as follows:

$$B(t) = (max - \frac{max - min}{2}) + \frac{max - min}{2} * sin(\frac{2\pi}{\tau}t + \frac{\pi}{2})$$

with $max = 1Mbps$, $min = 0.2Mbps$ and $\tau = 32s$; then it is not difficult to obtain the inefficiency in this case, especially if we set $cwnd(t_a) = 0$ in equation (3).

However, this model assumes that the packet losses are detected immediately by the sender who then react by halving or reinitializing its congestion window. The real system is a bit different because intermediate routers do have some buffers to store packets. In this case, 2 scenario are possible when the bandwidth decreases: (i) the buffer is almost empty and (ii) the buffer is almost full. In the first scenario, buffers at the bottleneck link would fill up and compensate in some extend the bandwidth decrease. Eventually some packets will be dropped when the buffer becomes full (if Drop Tail router are considered). In the second scenario, the buffer saturates quickly and packets are rapidly dropped, however the receiver still receives in-sequence packets (those still in the queue) and thus keeps sending outstanding ACK packets. Both cases give a similar result: the behavior of TCP looks more like the one depicted in figure 4 where we identified 3 phases. In phase a, the throughput curve matches the bandwidth curve, at least until the packet losses are detected at the receiver (with a gap in the sequence number) which would then send duplicated ACK making the throughput to decrease.

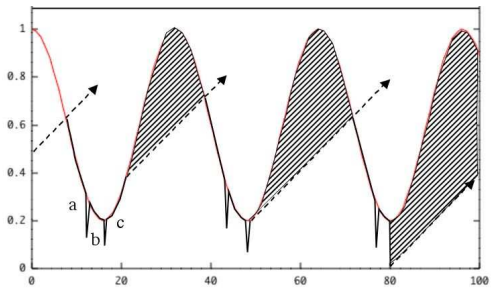


Figure 4. More accurate TCP behavior with sine-based variations: 3 possible scenario.

Depending on the number of packet losses, either fast retransmit/fast recovery would be enough to handle the losses, or a timeout would be triggered. Actually, when the capacity of the buffer is small, timeout are not likely to occur both because packet losses are more frequent (so $cwnd$ can not increase very much) and because loss indications arrive faster with fewer packets in the backlog queue. Figure 4 shows the case where several packet losses are handled with the fast retransmit/fast recovery procedure (phase b). The throughput could go up quickly if the retransmitted packets fill the gap at the receiver. This explains why the throughput curve matches the bandwidth curve again. When the available bandwidth increases again, depending on the amount of packets left in the buffers, the throughput curve could match for a while the bandwidth curve until being regulated by the sender throughput increase rate (which greatly depends on the RTT: small RTTs give high throughput increase rate): this is phase c. Several attempts to model the router's buffer occupancy under TCP traffic have been done ([19] for instance) but very complex hypothesis on the arrival pattern must be made. This is beyond the scope of this paper to build such a model. In order to reduce the inefficiency, the throughput curve should be close to the bandwidth curve when bandwidth is increasing. To do so, timeout should be avoided when the bandwidth curve is at its minimum in order to not start from too "low" (an example of high inefficiency is shown in the right part of figure 4). In summary, high inefficiency can be expected when both RTT and buffers are large.

3.2.2 Step-based bandwidth variations

For step-based variations, the main difference with sine-based variations is that the available bandwidth can drop significantly in a very short amount of time (possibly in several times). We illustrate in figure 5 the 2 cases that can be found in step-based variations models.

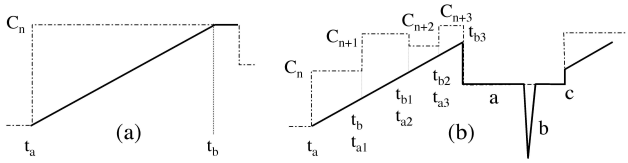


Figure 5. Step-based variations scenario.

Figure 5a shows an increase of the throughput to the maximum value C_n before bandwidth is decreased to a smaller amount. In this case, the inefficiency defined by equation 4 can be expressed by:

$$I_{ab} = 1 - \frac{T_{ab}}{C_n * (t_b - t_a)} = \frac{(t_b - t_a)(\frac{C_n + T_a}{2})}{C_n * (t_b - t_a)} =$$

$$\frac{4 * MSS}{RTT^2 * C_n}(t_b - t_a) - \frac{1}{2}$$

which is obtained by using equations (2) and (3) to express T_a as a function of C_n and t_b. In figure 5b, the bandwidth is decreased before the throughput could reach the maximum value. We show the general case where the ith variation makes the throughput to decrease while several harmless bandwidth variations could happen before. Again, we identified 3 phases in this case which are identical to those of the sine-based variation model explained previously. In this case, which we believe is more frequently encountered, the inefficiency could be exactly computed if all the variations are known (which is practically not possible). However, if only the mean available bandwidth is known (that we note by C_{ab}) then the inefficiency could simply be written as a

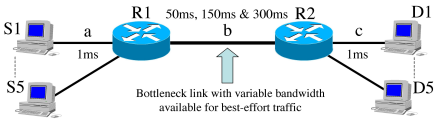
function of T_a (T_a is usually the value of the previous available bandwidth) using once again equations (1) and (2):

$$I_{ab} = 1 - \frac{T_{ab}}{C_{ab} * (t_b - t_a)} = 1 - \frac{\{T_a + \frac{4*MSS}{RTT^2}(t_b - t_b)\}}{C_{ab}}$$

We can see that the larger C_{ab} and RTT, the higher I_{ab}. However, inefficiency can be low in phase a, b and c when fast retransmit/fast recovery can be used instead of timeouts, which is the case when the buffer size is large.

4 Simulations results

The network model on which we simulate TCP is shown in figure 6. The bottleneck link is link b which propagation delay is either 50ms, 150ms or 300ms (i.e. RTT of 100ms, 300ms and 600ms respectively). 2 configurations are used which are detailed in the figure as case 1 and case 2.



	a	b	c	R1	R2
Case 1	10 Mbps	1 Mbps	10 Mbps	5…50 pkts	5…50 pkts
Case 2	100 Mbps	50 Mbps	100 Mbps	200, 500 1000 pkts	200, 500 1000 pkts

Figure 6. Network model for ns simulations.

R1 and R2 refer to the number of buffer slots (expressed in packets) that are available for each incoming link in the router. The optimal buffer size usually follows the rule of thumb of $bandwidth.rtt$ product, e.g. for a 1Mbps link with RTT of 300ms (i.e. one way delay of 150ms), the buffer in number of 1Kbytes packets should be around 36. For 50Mbps, it is about 1831. For case 2, the buffer size is clearly not optimal but reflects the reality. In case 1, we vary the buffer size from 5 packets to 50 packets. All routers use the Drop Tail queue management strategy.

We use both pure sine-based variations with period of about 30s and step-based variations with changes every 30s. The variation range is 0.2Mbps-1Mbps (that is the bandwidth available for best-effort traffic) and 10-50Mbps for respectively the 1Mbps-link and the 50Mbps-link case. For the step-based variation, we draw a random evolution and used it in all simulations. The ns simulator [1] is used for all the simulations. All graphics show the bottleneck link delay, therefore the RTT is twice this value. The throughput is measured by monitoring the ACK packets sent back by the receiver so packet losses causing duplicated ACKs make the throughput to decrease.

4.1 No bandwidth variations

Figure 7 shows the TCP throughput with network configuration 1 (1Mbps) when there are no bandwidth variations. The no variation case is included in this paper in order to serve as a reference. On the 1Mbps link, the window size advertised by the receiver is increased from 20 (default value in ns) to 256 in order to not being limited by the receiver window. Without bandwidth variations standard TCP on a rather low-speed 1Mbps link is very efficient and can grab all available bandwidth provided that sufficient buffer slots are available in routers (10 buffer slots are enough to considerably limit packet drops).

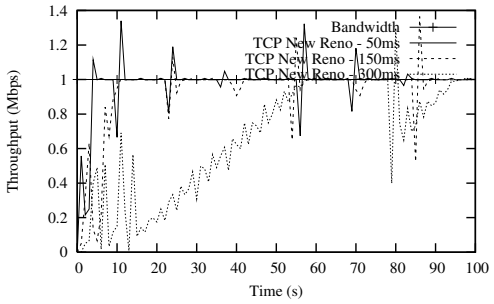


Figure 7. TCP with no variations.

4.2 Sine-based variations

Figure 8 shows for a 1Mbps link the throughput of the TCP sender. As could be expected, when RTT is high TCP is not able to follow the bandwidth variations. For small to medium RTT values (100ms and 300ms), the throughput curve follows quite closely the bandwidth variation curve because the router's buffers can compensate in some extent the bandwidth variations. Figure 8 also shows that the throughput is more stable when the buffer size is small (figure 8(a,b,c)). The reason is that small buffers introduce more frequent packet losses that limit the growth of the sender $cwnd$. If we plot $cwnd$, we would see that the TCP sender always uses fast retransmit/fast recovery. On the contrary, when the buffer size is larger (30 packets and more), it compensates for the bandwidth variations and lets $cwnd$ to grow. Eventually, with a large $cwnd$, the router's buffer will become full and severe losses, and then timeout, would occur which would trigger slow-start phases (this behavior has been verified when we plotted $cwnd$ for the 30-packet case). In figure 8(d,e,f) we can clearly see the behavior described in figure 4.

In figure 8f, we can see an interesting behavior at time 80s: the throughput curve matches the bandwidth curve indicating that when bandwidth increases again, the buffer is drained out to make use of all the available bandwidth. This is the result of having large buffers.

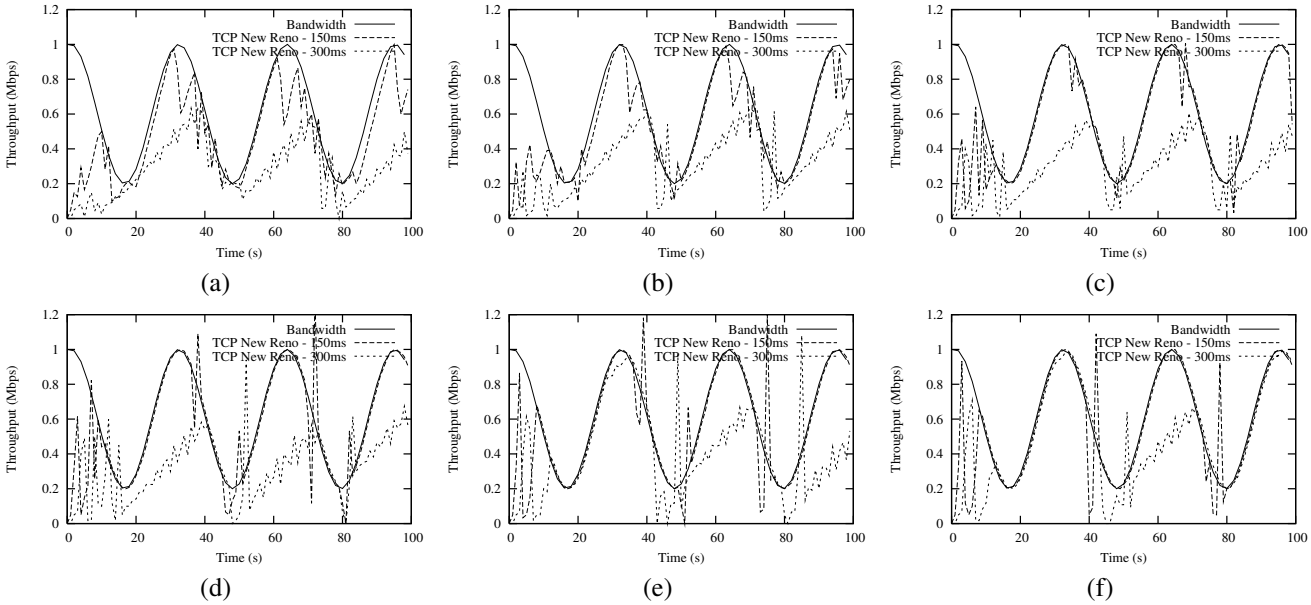


Figure 8. Sine variations, 1Mbps link. Buffer size (a) 5, (b) 10, (c) 20, (d) 30, (e) 40, (f) 50.

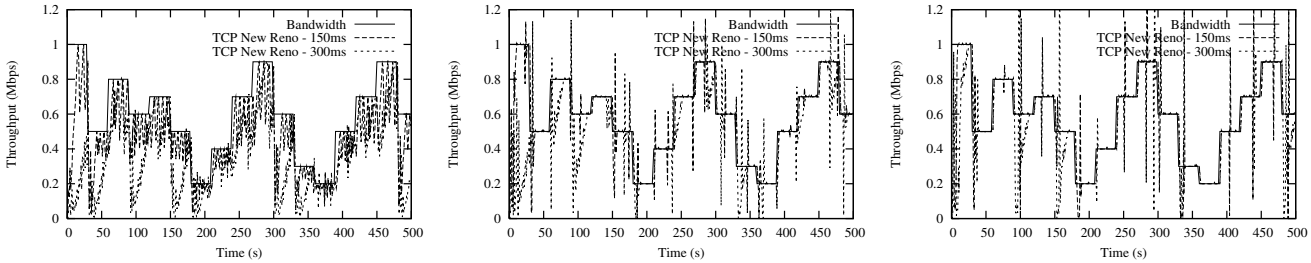


Figure 9. Step variations, 1Mbps link. Buffer size 5, 30 and 50 (left to right).

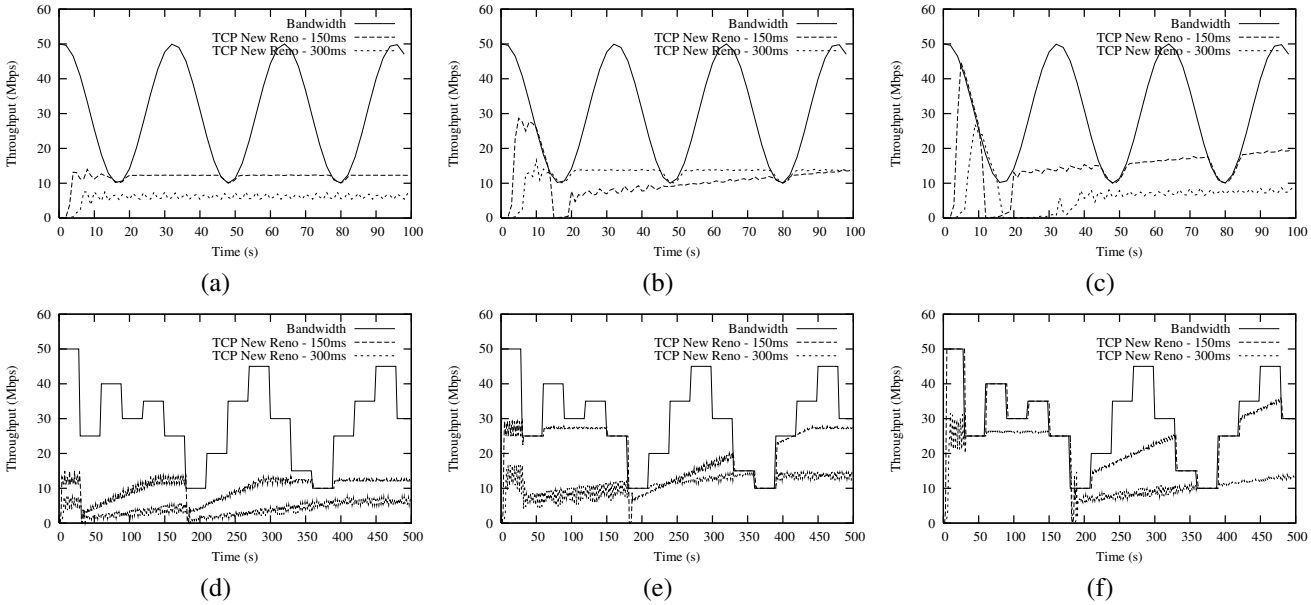


Figure 10. 50Mbps link. Sine: buffer (a) 200, (b) 500, (c) 1000. Step: buffer (d) 200, (e) 500, (f) 1000.

The results presented in figure 8 show a better behavior of TCP on a low-speed link than those previously presented by Dutta et al in [8]. We believe the authors in [8] did not tune the TCP agent in ns (receiver window size for instance), and especially did not increase the router buffer size set by default to 1 packet. With the default TCP agent parameters, we indeed got the same curves than those presented in [8]. However, we believe that our results presented in this paper are more realistic.

Figure 10(a,b,c) show the 50Mbps-link case. Here the behavior of TCP is not different from the TCP problem of inefficiency on a high $bandwidth.delay$ product network: when $cwnd$ is reset to 1, the congestion avoidance phase is too slow to grab the available bandwidth. The only problem specific to VBE is that packet losses are unavoidable.

4.3 Step-based variations

For the step-based variation model, figure 9 shows the sender throughput on a 1Mbps link for 3 buffer sizes: 5, 30 and 50 packets. As the buffer size increases, the number of packet losses decreases (and so the number of timeouts). Once again, we can clearly see the behavior predicted in figure 5b. When the bandwidth reduction is high, phase a is very short and most of the time leads to a timeout instead of a fast retransmit/fast recovery phase. However, inefficiency is rather low as the buffer size increases (50 packets for instance) which was also predicted.

On a 50Mbps link, the TCP behavior is quite different because of the additional problem of having a high $bandwidth.delay$ product. Once again, we can see the behavior predicted in figure 5b and when a large amount of buffers is available, several bandwidth reductions can be handled without difficulties.

5 Conclusions and future works

In this paper, we studied the behavior of TCP on variable bandwidth environments and compared it to simulations performed with the ns simulator. Both the study and the simulation results show that VBE are challenging networking environments for TCP where packet losses are very frequently encountered. In this case it is tempting to let the TCP congestion control performs the regulation of bandwidth which would limit the growth of $cwnd$, but there is at the same time the need to grab very quickly the bandwidth when it increases. There are new propositions that increases the TCP increase slope in congestion avoidance phase [9, 14] for very high $bandwidth.delay$ product networks, but then a bandwidth drop is very costly and some preliminary simulations of such approaches suggest that the inefficiency problem is still not completely solved. Regarding the buffer size, the results suggest that the larger the buffer, the lower the inefficiency. However, as we did not look at the end-to-end performance issues such as the transmission completion time, it is still early to say that large buffers is better because large buffers also means more timeouts. We are currently performing more simulations to investigate this issue and also fairness issues. We are also looking at the impact of queue management in VBE.

References

[1] The network simulator ns-2. http://www.isi.edu/nsnam/ns.

[2] A. Antony, J. Blom, C. de Laat, J. Lee and W. Sjouw. Microscopic Examination of TCP Flows over Transatlantic Links Future Generation Computer Systems, Volume 19(6), 2003.

[3] Chadi Barakat. TCP modeling and validation. IEEE Networks, vol. 15, no. 3, pp. 38-47, May 2001.

[4] C. Barakat, E. Altman, W. Dabbous. On TCP performance in a heterogeneous network: a survey. IEEE Comm. Mag., January 2000.

[5] C. Bohm et al. Fast circuit switching for the next generation of high performance networks. IEEE JSAC, 14(2), February 1996.

[6] Cerf, V., and R. Kahn. A Protocol for Packet Network Intercommunication IEEE Transactions on Communications, Vol. COM-22, No. 5, pp 637-648, May 1974.

[7] T. Dunigan, M. Mathis, B. Tierney. A TCP Tuning Daemon. Supercomputing 2002.

[8] D. Dutta, Y. Zhang. An active proxy based architecture for TCP in heterogeneous variable bandwitdh networks. GlobeCom 2001.

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

[10] L. Gauffin, L. Hakansson, B. Pehrson. Multi-gigabit networking based on DTM. Computer Networks and ISDN Systems, 24(2):119?130, April 1992.

[11] P. Gevros, J. Crowcroft, P. Kirstein, and S. Bhatti. Congestion control mechanisms and the best effort service model. IEEE Network, vol. 15, pp. 16 - 26, May/June 2001.

[12] G. Hasegawa, M. Murata. Survey on fairness issues in TCP congestion control mechanisms. IEICE Transactions on Communications, January 2001.

[13] V. Jacobson. Congestion avoidance and control. In *ACM SIGCOMM '88*, 1988.

[14] C. Jin, D. X. Wei, S. H. Low. FAST TCP: Motivation, Architecture, Algorithms, Performance. IEEE Infocom 2004.

[15] D. Katabi, M. Handley, C. Rohrs. Congestion Control for High Bandwidth-Delay Product Networks. ACM SIGCOMM 2002.

[16] S. H. Low et al. Dynamics of TCP/AQM and a scalable control. IEEE Infocom 2002.

[17] Matthew Mathis, Jeffrey Semke, Jamshid Mahdavi, Teunis Ott. The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm. Computer Communications Review, Vol. 27(3), July 1997.

[18] J. Padhye et al. Modeling TCP Throughput: A Simple Model and its Empirical Validation ACM SIGCOMM 1998.

[19] Hans-Peter Schwefel. Behavior of TCP-like elastic traffic at a buffered bottleneck router. Infocom 2001.