ENHANCED BANDWIDTH ESTIMATION ALGORITHMS IN THE TCP CONGESTION CONTROL SCHEME

Antonio Capone, Luigi Fratta, Fabio Martignon
DEI, Politecnico di Milano
Piazza L. da Vinci 32, 20133 Milan, Italy
capone@elet.polimi.it, fratta@elet.polimi.it, martignon@elet.polimi.it

Abstract The use of enhanced bandwidth estimation procedures within the congestion control scheme of TCP has been recently proposed as a way to improve the performance over links affected by random losses. In this paper we propose TIBET (Time Intervals based Bandwidth Estimation Technique), a new bandwidth estimation scheme that can be implemented within the TCP congestion control procedure by modifying only the sender-side of a connection. TIBET allows TCP sources to achieve higher performance over wireless links, while guaranteeing a good fairness level with TCP Reno over wired networks. We analyze and compare the performance of the new algorithm with that of previously proposed schemes.

Keywords: TCP, Bandwidth Estimation, Wireless Links, Performance Model

Introduction

The Transmission Control Protocol (TCP) has shown to be efficient in classical wired networks, proving its ability to adapt to modern high-speed networks and new scenarios for which it was not originally designed. However, the extraordinary success of modern wireless access networks, such as cellular networks and wireless local area networks, poses new challenges to the TCP congestion control scheme.

The present versions of TCP, as Reno or NewReno, experience heavy throughput degradation over channels with high error rate such as wireless channels. The main reason of this poor performance is that the TCP congestion control mechanism cannot distinguish between random losses that occur in wireless channels and true congestion signals due to

high network load. Therefore, the TCP congestion control mechanism reduces the transmission rate even when not necessary [1].

A possible approach to this problem is to modify the TCP congestion control scheme implementing explicit bandwidth estimation schemes. The idea is to reduce the impact of losses due to transmission errors on the throughput performance by considering not only the bytes in flight when a loss is detected, as TCP Reno does, but also the past history of the connection [2, 3].

Various bandwidth estimation algorithms have been proposed in the literature [4, 5]. More detailed descriptions of that implemented by TCP Vegas is in [6], and of that adopted by TCP Westwood is in [3].

In this paper we first discuss the problem of end-to-end bandwidth estimation for TCP and point out the issues that may affect the estimation accuracy and its impact over TCP tunable parameters. Then we focus our analysis over the estimation algorithm proposed for TCP Westwood, showing how it may over-estimate the available bandwidth. This leads to an aggressive and unfair behavior that prevents a smooth introduction of the new TCP versions in the Internet.

To obtain more accurate, unbiased and stable bandwidth estimates, needed for a fair sharing of the network resources, we propose a new algorithm, TIBET (Time Intervals based Bandwidth Estimation Technique). TIBET, as the algorithms used in TCP Vegas and TCP Westwood, only requires modifications at the sender-side since it needs no cooperation from the peer TCP.

To show the benefits of the proposed scheme in networks affected by independent or correlated losses, typical of a wireless environment, we compare TIBET with TCP Reno performance. Moreover, we examine the behavior of TCP with an ideal bandwidth estimation in order to find an upper bound to the performance of all possible schemes based on different bandwidth estimates. We show how TIBET behaves close to the ideal algorithm, and we provide an empirical model to get some insight on the behavior of the ideal scheme.

The paper is structured as follows. In Section 1 we study the problem of bandwidth estimation performed at the sender-side and we review the existing estimation techniques. In Section 2 we present TIBET and we show by simulation that it efficiently copes with the problems shown in Section 1. In Section 3 we study TIBET performance in presence of wireless links, affected either by independent and correlated losses. An empirical model is also proposed for TCP with ideal bandwidth estimate. Finally, Section 4 concludes the paper.

1. Bandwidth Estimation

The more information is available, the better the TCP protocol can estimate the bandwidth available to a connection, thus resulting in a better and fair utilization of the network resources. Following this principle several bandwidth estimation techniques have been proposed in the literature.

A first approach that has been proposed in literature, assuming that only the end-systems are in charge of the rate regulation without any explicit support from the network, is to monitor the time spacing between ACK arrivals. If we divide the amount of acknowledged bytes by the interarrival time between consecutive ACKs, we obtain a sample of the ACK bandwidth, i.e. the bandwidth promised by the ACK to the sender [7]. Some filtering techniques can be added on this sequence of samples in order to smooth fast variations and to reduce the impact of random losses due to the channel. As proposed in [4, 3], the TCP slow start threshold (*ssthresh*) is related to the *byte-equivalent* of the estimated bandwidth (Bwe) according to the following relation:

$$Ssthresh = Bwe * RTT_{min} \tag{1}$$

where RTT_{min}, the lowest RTT ever measured by the TCP source, is considered an estimate of the raw propagation delay of the connection.

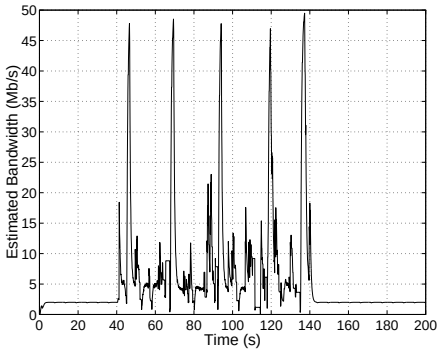
1.1. Estimation Problems

Due to the peculiar structure of the packets injected into the network by the TCP, and to the uncertainty with which a TCP source can measure time intervals and estimate the minimum RTT, the following problems arise:

- Clustering [7, 8], which is experimented by TCP segments belonging to the same connection;
- ACK Compression [7, 8], which alters the spacing between received ACKs due to congested routers in the retourning path, thus causing possible estimation problems;
- TCP coarse-grained clocks [6, 9], i.e. the low precision with which many real TCP implementations take measures of the RTT.

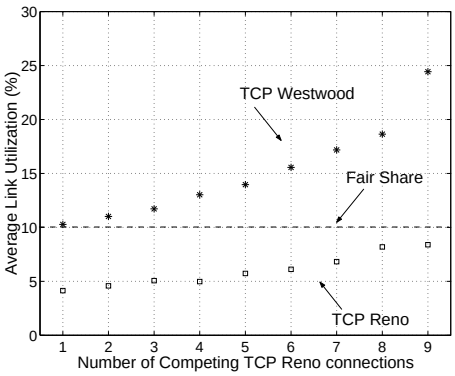
1.2. Estimation Algorithms

Several bandwidth estimation schemes have been proposed in the literature for the TCP congestion control [4, 5, 6]. Here we focus our analysis



(a)

Figure 1. TCP Westwood: estimated bandwidth in presence of ACK compression. 2 connections cross a 2 Mbit/s link in opposite directions



(b)

Figure 2. Link utilization of TCP Westwood and TCP Reno sources sharing the same 10Mb/s link

mainly over TCP Westwood, which was recently proposed in [3]. TCP Westwood performs an estimate of the available bandwidth by measuring the returning rate of acknowledgments, and uses this estimate to set the *ssthresh* and the *cwnd* after congestion events such as the receipt of three duplicate ACKs or coarse timeout expirations.

We have evaluated the performance of TCP Westwood's estimation algorithm by simulations implemented using Network Simulator, 'ns' ver.2. We have considered a scenario with a congested return path to check the algorithm behavior in presence of ACK Compression. Figure 1 shows the time trace obtained considering a 2 Mbit/s link and two TCP Westwood connections which send data packets in the two directions. In the time interval between 40 and 140 seconds the TCP connection in the opposite direction transmits packets and we observe a dramatic increase of the estimated values due to the filtering mechanism adopted.

This has a negative impact on the friendliness of TCP Westwood towards TCP Reno as shown in Figure 1 where the average link utilization for the two mechanisms is presented versus the number of TCP Reno connections. The values shown refer to a 10 Mbit/s link (same parameters as in Figure 2) with 10 overall connections. We observe that TCP Westwood always achieves higher link utilization than the fair share, thus penalizing the TCP Reno sources. This behavior prevents a smooth introduction of TCP Westwood in the Internet.

2. TIBET

TIBET (Time Intervals based Bandwidth Estimation Technique) is a new technique which succeeds in obtaining correct estimates of the bandwidth used by the TCP source even in presence of packet clustering and ACK compression. TIBET is also able to let TCP connections quickly track changes in the available bandwidth.

To explain the rationale of TIBET let us refer to the example in Figure 3 where transmissions occurring in a period T are considered. Let n be the number of packets belonging to a connection and $L_1, L_2 ... L_n$ the lengths, in bits, of these packets. The average bandwidth, Bw used by the connection is simply given by $\frac{1}{T}\sum_{i=1}^{n} L_i$, and can be expressed as

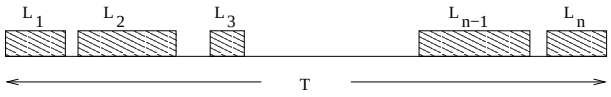


Figure 3. Packet timing structure

$$Bw = \frac{n\overline{L}}{T} = \frac{\overline{L}}{\frac{T}{n}} \tag{2}$$

where $\overline{L} = \frac{1}{n}\sum_{i=1}^{n} L_i$. The basic idea is to perform a run-time sender-side estimate of the average packet length, $\overline{L}$, and the average interarrival, $\frac{T}{n}$, separately.

Our scheme can be applied by measuring and low-pass filtering the length of acknowledged packets and the intervals between ACKs' arrivals. However, since we want to estimate the used bandwidth we can low-pass filter either the packet lengths and their interdeparture times or the bytes acknowledged by the last ACK and its gap from the previous one. Note that interdeparture times can be very short when groups of packets are generated by TCP sources. However, this is not a problem for TIBET since the estimate is performed directly on the interarrival samples and not on bandwidth samples, such as in TCP Westwood, that would be very high and close to peak rate of the host interface.

The pseudo-code of the bandwidth estimation scheme based on transmitted packets that has been implemented in our simulations is the following:

```
if (Packet is sent)
sample_length[k] = (packet_size * 8);
sample_interval[k] = now - last_sending_time;
Average_packet_length[k] = alpha * Average_packet_length[k-1]
```

```
                          + (1-alpha)*sample_length[k];
Average_interval[k] = alpha * Average_interval[k-1]
                    + (1-alpha )* sample_interval[k];
Bwe[k] = Average_packet_length[k] / Average_interval[k]
endif
```

where *packet_size* indicates the segment size in bytes, *now* indicates the current time, *last_sending_time* the time of the previous packet transmission. *Average_packet_length* and *average_interval* are the low-pass filtered measures of the packet length and of the interdeparture times, respectively. *Alpha* is the pole of the two low-pass filters. *Bwe* is the estimated value of the used bandwidth.

In the second alternative, the algorithm using the *received ACKs* is executed each time an ACK is received and is very similar to the previous one. The only differences are in the expressions used to calculate *sample_length* and *sample_interval*:

```
sample_length[k] = (acked * packet_size * 8);
sample_interval[k] = now - last_ack_time;
```

where *last_ack_time* is the time when the last ACK was received, and *acked* indicates the number of segments acked by the last ACK.

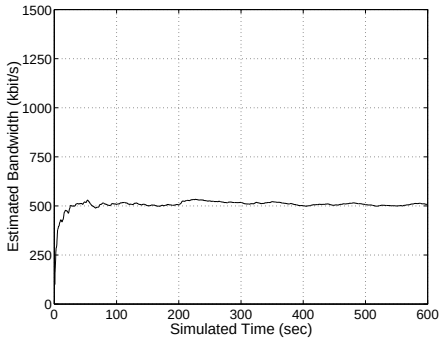
Considering the minimum RTT measured by the TCP source (RTT_{min}) as a good estimate of the end-to-end propagation delay, we set the *ssthresh* according to relation (1) after three duplicate ACK's, or after a coarse-grained timeout expiration.

2.1. Estimation accuracy and fairness

Figure 5 shows how TIBET estimation algorithm is able to correctly estimate the bandwidth available to the TCP source. The simulation scenario considers 10 TCP connections over a 5 Mbit/s link. The bottleneck queue, managed with a drop-tail policy, was designed to contain a number of packets equal to the bandwidth-delay product. The accuracy of TIBET estimation algorithm is evident if we compare this result with that obtained by TCP Westwood, which has been shown in Figure 1.

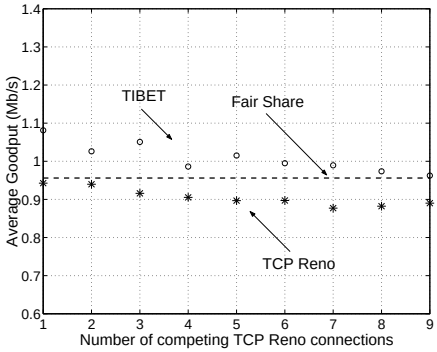
Moreover, since we use the flow of sent packets, the ACK compression effect has no impact over the bandwidth estimation.

A further important test is to verify that TCP sources using TIBET succeed to fairly share network resources with sources using other existing versions of the protocol.



(a)

Figure 4. TIBET: estimated bandwidth with 10 connections sharing a single 5 Mbit/s link



(b)

Figure 5. TIBET fairness towards TCP Reno over a 10 Mbit/s link

Figure 6 shows the results of fairness tests referring to a mixed scenario with TIBET and TCP Reno connections. The scenario is the same as that used in Figure 1, with a total of 10 connections sharing a 10 Mbit/s bottleneck link. The x-axis of Figure 6 represents the number of TCP Reno connections, while the remaining connections use TIBET. The y-axis value represents the average throughput obtained by the two types of algorithms, and the dotted line is the fair-share. We observe that both algorithms achieve a throughput very close to the fair share.

We have also run simulations over different scenarios covering link bandwidths ranging from a few kbit/s to 150 Mbit/s, varying the number of competing connections and using also a more complex topology with multiple congested gateways. The results obtained are almost the same: TIBET obtains the same level of fairness as TCP Reno. Simulation results also show that TIBET improves its performance when the number of connections sharing the bottleneck link increases since the estimate variance reduces. Moreover, the presence of constant rate flows, such as UDP flows for IP telephony or video conference, makes TIBET to perform better as it reduces the size of packet clusters.

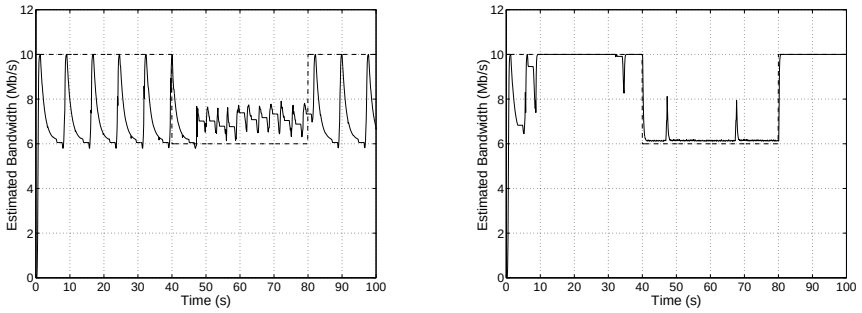
In order to reduce the effect of the coarse-grained clocks problem, described in Section 1, on the TIBET performance we propose the following modification to update RTT_{min} to be used when RTT_{min} is smaller than the clock granularity.

1 *every time the connection experiences a congestion signal (time-out expiration or the receipt of* 3 *duplicate ACKs), a variable* n *is increased by one.*

2 *if n has reached the threshold value of N_{cong}, i.e. $n = N_{cong}$, RTT_{min} is reduced by a factor β and n is reset to zero;*

3 *point 1 is executed once again.*

Figure 6 shows the performance of TIBET with this modification.



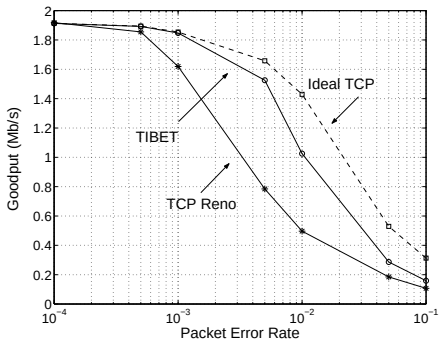
(a) (b)

Figure 6. Bandwidth Estimate executed (a) in absence (b) in presence of RTT_{min} updating algorithm

In this scenario, a single TCP source running the TIBET algorithm transmits over a 10 Mb/s link, with a RTT of 50 ms. In the interval between 40 and 80 seconds an UDP flow having the same priority as TCP transmits with a data rate of 4 Mb/s. The clock granularity is 500 ms, i.e 10 times greater than the actual RTT of the connection, and therefore also the RTT_{min} is set to 500 ms by the TCP source. Figures 6(a) and 6(b) show the bandwidth estimated without and with the RTT_{min} updating algorithm ($N_{cong} = 5$, $\beta = 0.5$), respectively. In both the figures, the dotted line represents the ideal estimate. Without the algorithm the connection is very aggressive and the link is often congested, while with the algorithm the bandwidth estimate is more accurate and the link congestion is controlled in a more effective way. This is confirmed by the average goodput achieved by the TCP connection, which is only 1.2 Mb/s in the scenario of Figure 6(a),while it raises up to 5.8 Mb/s in the scenario of figure 6(b). Similar behaviors have been observed with different scenarios.

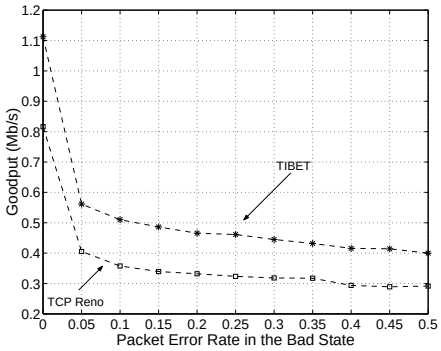
3. Wireless Links

So far we have proved the accuracy of TIBET and shown that TCP sources using this algorithm behave fairly. However, since the main goal



(a)

Figure 7. TIBET and TCP Reno throughput vs packet error rate over a 2 Mbit/s link



(b)

Figure 8. TIBET and TCP Reno goodput achieved over a 2 Mb/s link affected by correlated losses, as a function of the error rate in the Bad state

for which this algorithm has been designed is to achieve high throughput in links affected by sporadic losses, we have simulated the operation of TIBET over a link with either random independent errors or correlated errors to account for multipath fading typical in wireless environments

3.1. TIBET vs. Reno

In Figure 7 we compare the throughput achieved by a connection running the TIBET algorithm with that of a TCP Reno connection over a link with random independent errors. The link has a capacity of 10 Mbit/s and a propagation delay equal to 50 ms. As usual the queue can contain a number of packets equal to the bandwidth-delay product. For all packet error rates, TIBET provides an higher throughput which is almost twice that given by TCP Reno. This improvement is due to the filtering process which keeps in account also the past history of the connection avoiding most of the times to confuse real network congestion signals due to queue overflows with those due to link errors.

We have also investigated the behavior of TIBET in presence of links affected by correlated errors, to account for the effects of multipath fading typical of wireless environments. For this case, according to the existent literature [10], we have modeled the wireless link state (*Good* or *Bad*) with a two-state Markov chain, with average durations of good and bad states equal to 8 and 4 seconds, respectively. The packet error rate in the good state is constant and equal to 1%, while the one in the bad state varies from 0 to 50% to take into account various levels of fading. The results in Figure 8 show that, also in this case, TIBET

obtains higher goodput than TCP Reno for a wireless link capacity equal to 2 Mb/s.

3.2. Throughput upper bound

Once proved the advantage of TIBET over Reno, we address the issue of how far is the performance of TIBET from the upper bound of all possible schemes exploiting the new bandwidth estimation approach. Such an upper bound is obtained by assuming an *Ideal TCP* that sets the *ssthresh* having knowledge of the exact bandwidth available along the path. The throughput achieved by this ideal scheme is shown in Figure 7. We notice that the TIBET performance is reasonably close to the bound. The difference is mainly due to the estimation errors and to the fact that TIBET estimates the used bandwidth instead of the available one. The throughput degradation measured even in the Ideal TCP at high error rates proves that it is unavoidable and does not depend on the estimation algorithm used.

To get more insight of this behavior, we developed a simple semi-empirical model of the Ideal TCP that enables to evaluate the throughput of a single TCP connection over a wireless link with random independent errors. In this model the *ssthresh* is always set equal to the link bandwidth, and the packet error rate is assumed to be high enough to prevent the congestion window, *cwnd*, to reach the *ssthresh*. Basically, the model focuses on links affected by a packet error rate ranging between 0.1% and 10%, typical values for real Wireless LAN systems and cellular networks, as shown in [11].

Here we only report the final approximate expression of the goodput achieved by this TCP source, which has been obtained by dividing the average number of bits sent between two loss events and the average duration of the period itself:

$$Goodput \propto \frac{MSS}{RTT} \frac{1}{p \cdot log_2(\frac{1}{p})} \tag{3}$$

where p represents the packet loss probability, RTT is the Round Trip Time and MSS the Maximum Segment Size of the connection . This approximation is valid within the range of error probabilities typical of wireless networks, i.e. for p ranging between 10^{-3} and 10^{-1} as we said before. This result is particularly interesting, expecially if compared to the goodput achieved by TCP Reno connections running over lossy links, which has been shown in literature to be proportional to $\frac{1}{RTT}\frac{1}{\sqrt{p}}$.

To compare the throughput given by the model with that obtained with simulation, we considered a simple scenario with a wireless link which corrupts packets with a packet error rate equal to p.

The numerical results obtained are compared in Figure 9. The simplified model just described shows to predict quite well the throughput achieved by the TCP connections and to be very accurate when the packet error rate is high.

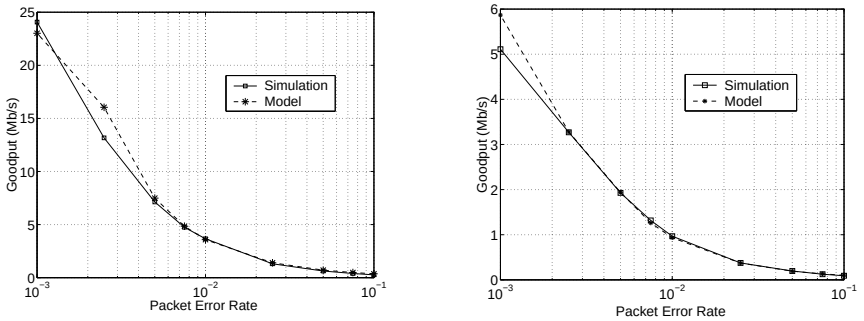


Figure 9. Comparison of Analytical Model and Simulation with (a) RTT = 100 ms (b) RTT = 400ms

4. Conclusions

In this paper we have discussed the issues related to the use of enhanced bandwidth estimation algorithms for TCP congestion control. These algorithms can potentially improve the throughput of TCP over wireless links affected by independent or correlated errors. We have shown that previously proposed schemes do not fairly share resources with TCP Reno over wired links and thus they cannot be smoothly introduced in the Internet. For this reason we have proposed TIBET, a new algorithm, which allows TCP to achieve a higher throughput over wireless links while keeping a good fairness level over wired links. We have shown that the estimation accuracy is a key issue for fairness and we have developed an approximate model which considers an ideal scheme with an error free bandwidth estimation. This model provides an upper bound to the throughput of all schemes based on the new bandwidth estimation approach, and allows to show that TIBET reasonably approaches such a performance.

References

[1] T.V.Lakshman and U.Madhow. The performance of TCP/IP for networks with high bandwidth-delay products and random loss. *IEEE/ACM Transactions on Networking*, 5(3):336–350, 1997.

[2] A.Capone and F.Martignon. Bandwidth Estimates in the TCP Congestion Control Scheme. In *Proceedings of Tyrrhenian International Workshop on Digital Communications (IWDC 2001)*, Taormina, Italy, Sep.2001.

[3] S.Mascolo, C.Casetti, M.Gerla, M.Y.Sanadidi, and R.Wang. TCP Westwood: End-to-End Bandwidth Estimation for Efficient Transport over Wired and Wireless Networks. In *Proceedings of ACM MOBICOM'01*, 2001.

[4] J.C.Hoe. Improving the Start-up Behavior of a Congestion Control Scheme for TCP. *ACM SIGCOMM Computer Communications Review*, 26(4):270–280, October 1996.

[5] M.Allman and V.Paxson. On Estimating End-to-End Network Path Properties. In *Proceedings of ACM SIGCOMM'99*, 1999.

[6] L.S. Brakmo and L.L. Peterson. TCP Vegas: End-to-End Congestion Avoidance on a Global Internet. *IEEE Journal on Selected Areas in Communications*, 13(8):1465–1480, October 1995.

[7] J.C.Mogul. Observing TCP Dynamics in Real Networks. In *Proceedings of ACM SIGCOMM'92 Symposium on Communications Architectures and Protocols*, pages 305–317.

[8] L.Zhang, S.Shenker, and D.Clark. Observations on the Dynamics of a Congestion Control Algorithm: The Effects of Two-Way Traffic. In *Proceedings of SIGCOMM'91 Symposium on Communications Architectures and Protocols*, pages 133–147, Zurich, September 1991.

[9] V.Paxson and M.Allman. Computing TCP's Retrasmission Timer. *RFC 2988*, November 2000.

[10] A.A.Abouzeid, S.Roy, and M.Azizoglu. Stochastic Modeling of TCP over Lossy Links. In *IEEE INFOCOM 2000*, Tel Aviv, Israel, March 2000.

[11] G.Xylomenos, G.C.Polyzos, P.Mahonen, and M.Saaranen. TCP Performance Issues over Wireless Links. *IEEE Communications Magazine*, 39(4):52–58, April 2001.