

TCP performance in case of bi-directional packet loss *

Ran Yang ^a, R.E. Kooij ^b and R.D. van der Mei ^{a,c}

^a Vrije Universiteit, Faculty of Sciences, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands

^b TNO Information and Communication Technology, P.O. Box 5050, 2600 GB Delft, The Netherlands

^c Centre for Mathematics and Computer Science, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands

Abstract

Performance modeling of the Transport Control Protocol (TCP) has received a lot of attention over the past few years. The most commonly quoted results are approximate formulas for TCP throughput [1] and document download times [2] which are used for dimensioning of IP networks. However, the existing modeling approaches unanimously assume that packet loss only occurs for data packets (i.e. for packets from the server to the client), whereas in reality the packets in the direction from the client to the server (e.g., ACKs) may also be dropped. Our simulations with ns-2 show that this bi-directional packet loss may have a strong impact on TCP performance. Motivated by this, we refine the models in [1, 2] by including bi-directional packet loss, also including correlations between packet loss occurrences. Simulations show that the proposed model leads to significant improvements of the accuracy of the TCP performance predictions.

Index terms Quality of Service, TCP, download times, response times, correlated packet loss

1. Introduction

Motivated by performance problems that occur on the Internet, many studies have been investigating performance models for TCP. Pioneering work in this field was done by Mathis et al. [3] and Ott et al. [4], who derive simple square-root formulas for the throughput of infinitely long TCP traffic flows under idealized periodic behaviour of the TCP congestion window, including the impact of the packet loss rate (p) and the round trip time (RTT). Padhye et al. [1] propose a refinement of the models in [3, 4] by taking into account detailed packet-level dynamics of the TCP window mechanism, and show that this refined model is able to more accurately predict TCP throughput and is accurate over a wider range of loss rates. However, the Padhye model does not take into account the behaviour of the TCP slow start and fast recovery mechanism. Based on the model [1], Cardwell et al. [2] propose a model for the mean total data download time (TDT) to transfer limited amounts of data, taking into account TCP slow start and fast recovery. This model is quite accurate in predicting the mean TDT under the assumption that losses happen only in the direction from the server to the client and losses in the successive rounds are independent; here, a round is the period of time between the departure of the first segment of the current window and the arrival of its acknowledgement (ACK). A limiting factor of the model in [2] is that it assumes that packet loss only occurs for data packets (i.e. for packets from the server to the client), whereas in reality the packets in the direction from the client to the server (e.g., ACKs) may also be dropped, which may have a significant impact on the TDTs experienced by the end users.

Motivated by these observations, the contribution of this paper is twofold. First, simulations with the ns-2 [5] demonstrate that the bi-directional packet loss may have a strong impact on the performance of TCP. Second, motivated by this we extend the Cardwell model [2] by considering packet losses occurring in both directions, and correlations between successive packet loss occurrences. The accuracy of the models is validated extensively by simulations

* This work is partially funded by the Dutch Ministry of Economic Affairs under the programme 'TS ICT Doorbraakprojecten', project TSIT 2031 EQUANET.

with the ns-2 simulator. The results show that the models are highly accurate for a wide range of parameter settings, and as such are a significant improvement of the model in [2].

2. QoS metrics that determine the end-user's perception

In [6] extensive research has been done to investigate the main factors that determine the end user's perception of Quality of Service (QoS) for Web browsing applications. The results of this research shows that the following two factors dominate the user perception of web browsing quality:

1. **Response Time (RT):** time from clicking on a link until first packet arrives and something appears on the screen, and
2. **Total Download Time (TDT):** time between clicking on a link and the arrival of the last packet.

The RT and TDT are depicted in Figure 1. Here, the parameters T_i have the following meaning: T_1 = time when user clicks, T_2 = user receives first response, and T_3 = total data transfer complete.

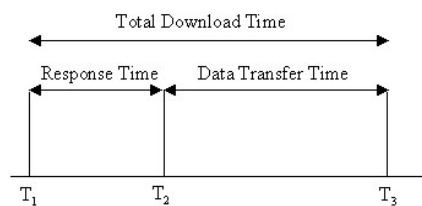


Figure 1: QoS metrics for TCP downloads

Thus, we can write

$$\text{TDT} = \text{RT} + \text{DTT},$$

where DTT stands for the Data Transfer Time (DTT). Note that in general, not only the mean of the RT and TDT are relevant, but also the variability of these performance metrics is relevant. However, within in this paper we will only focus on mean values for the RT and TDT.

The aim of this paper is to construct approximate analytical expressions for RT and TDT that will be based upon network performance characteristics, such as packet loss and RTT, as well as upon parameters determined by the TCP client and server such as maximum window size and the packet size. In Section 3 models for the RT and the DTT will be derived, and in Section 4 the accuracy of these models will be validated.

3. Modelling

In Section 3.1 we will develop an analytical expression for the RT, and in Section 3.2 we will develop a model for the DTT. The models are then combined to obtain an expression for the TDT.

3.1 Response Time

In this section we develop a model for the RT, i.e., the time it takes to establish a TCP connection and the additional time it takes to send the first packet containing data. From a user's point of view, the RT is simply the time from clicking on a URL link until the first packet

arrives and something appears on the screen. Each TCP connection starts with a “three-way handshake”, in which the client and server exchange initial sequence numbers. The RT is determined by the time it takes to send four packets successfully; here, the first three packets are related to the three-way handshake while the fourth packet contains the first data. Figure 2 shows an example where none of the sent packets are lost.

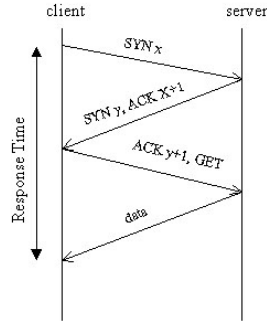


Figure 2: TCP connection establishment and first data packet

We assume that the packet loss probabilities in forward and reverse direction are the same and will be denoted by p . Suppose there are n failures (i.e., packet losses) before the first packet containing data is received. Failures can be classified as two kinds. The first kind is due to a packet loss from the client to the server. The second kind is caused by a packet loss from the server to the client. For both kinds of failures, packets will be retransmitted from the client side. We define the following parameters in our discussion:

- P_n probability of receiving the first data packet after exactly n packet losses;
- RT_n conditional mean response time when the first data packet is received after exactly n packet losses;
- RT the mean response time.

Lemma 1

For $n=0,1,\dots$,

$$P_n = (n + 1)(1 - p)^4 p^n (2 - p)^n.$$

Proof: It is convenient to define the concept of a cycle. It is simply the RTT if both a packet sent by the client and its ACK are sent successfully. We call this a successful cycle. The probability that this occurs is $p_s = (1-p)^2$. If either the packet sent by the client is lost or its ACK then the cycle is defined as the time between sending the packet and the time it is retransmitted. This will be denoted as an unsuccessful cycle. The probability that this occurs is $p_u = p + (1-p)p = p(2-p)$. If exactly n losses occur before the first data is received successfully then this implies that exactly $n+2$ cycles have passed of which n are unsuccessful and 2 are successful. Obviously the last cycle needs to be successful. Therefore for the other successful cycle there are $(n+1)$ possible locations. Therefore P_n satisfies $P_n = (n + 1)p_u^n p_s^2$. Substitution of the values of p_u and p_s yields the lemma.

For the situation described in this section, loss is detected through a Retransmission Time Out (RTO). If a packet sent by the client is not acknowledged before the Retransmission Timer expires then the packet is retransmitted. With each retransmission the Retransmission Timer is doubled, up to a maximum value 64 times its original value. Denote by T_0 the initial value of the

Retransmission Timer. The simulation package ns-2, see [6], uses $T_0 = 6$ seconds, although RFC2988 [7] recommends $T_0 = 3$ seconds. Note that TCP uses sample values of the RTT to adjust the Retransmission Timer. However, according to Karn's algorithm, this only occurs for packets that are not being retransmitted. Hence, in our situation, such an adjustment can only occur if the first two packets are sent successfully. According to [7], upon its first update the Retransmission Timer becomes

$$T_u = \max\{1, RTT + \max\{G, 2RTT\}\},$$

where G denotes the TCP timer granularity. In many TCP implementations G is set to 500ms. Based on [1], it can be shown that

$$RT_n = \begin{cases} (2^n - 1)T + 2RTT; & n \leq 6, \\ [63 + 64(n - 6)]T + 2RTT; & n \geq 7, \end{cases} \quad (2)$$

where

$$T = \frac{1}{n+1}T_u + \frac{n}{n+1}T_0.$$

Lemma 2

The mean response time RT is given by the following expression:

$$RT = \frac{A(b_0 + b_1A + b_2A^2 + b_3A^3 + b_4A^4 + b_5A^5 + b_6A^6 + b_7A^7)}{1 - A} + 2RTT \quad (3)$$

where

$$A = p(2 - p), b_0 = T_0 + T_u, b_1 = 3T_0, b_2 = 6T_0 + T_u, b_3 = 14T_0 + 2T_u, \\ b_4 = 32T_0 + 4T_u, b_5 = 72T_0 + 8T_u, b_6 = 160T_0 + 16T_u, b_7 = -160T_0 - 32T_u.$$

Proof: Obviously RT satisfies

$$RT = \sum_{n=0}^{\infty} P_n RT_n = \sum_{n=0}^6 P_n [(2^n - 1)T + 2RTT] + \sum_{n=7}^{\infty} P_n [(63 + 64(n - 6))T + 2RTT].$$

After a tedious calculation, which was performed with the help of the Computer Algebra software Maple, this expression can be simplified to equation (3).

3.2. Data Transfer Time

The Data Transfer Time (DTT) is the time between sending the first data packet and receiving the last data packet. From the previous section we know the TCP connection establishment time is the time taken to send three packets successfully. Therefore, we approximate it by 3/4 times the RT , discussed in Section 3.1. In [2], a model is proposed under the assumption that packet loss happens only in the direction from sender to receiver. This model directly depends on this

one-way packet loss, whereas in reality not only data segments can be lost during a TCP data transmission but also the ACKs of data packets can be dropped in the direction from the receiver to the sender. Therefore, we extend the Cardwell model by including the impact of the loss of ACKs. First, we define the new variables introduced in this section:

- p_f : forward packet loss rate (data packet loss rate);
- p_b : backward packet loss rate (ACK loss rate);
- w : current window size.

Packet loss may occur in one of the following two cases:

Case 1: Packet is lost during transmission from the sender to the receiver

As a first step, we discuss in which situation a data packet is considered lost by TCP. We focus on a single data packet. We assign to this data packet an index k indicating the position of the data packet within the current window. Therefore k satisfies the condition $1 \leq k \leq w$. Obviously, if k is lost during transmission from the sender to the receiver, then this packet is considered lost by the sender TCP. The probability of the occurrence of Case 1 is equal to p_f .

Case 2: Packet is sent successfully, but the ACK is lost

Case 2 occurs when packet k is sent successfully, but the ACK for packet k is lost. The probability of the occurrence of Case 2 is $(1 - p_f)p_b$. For Case 2, we cannot determine immediately whether data packet k is considered lost by the sender TCP or not. In fact, if packet $k+1$ and its ACK are sent successfully then the sender does not notice that the ACK of packet k was lost, hence packet k is not considered lost. If either packet $k+1$ or its ACK is lost then we have to take into account packet $k+2$ and its ACK. We have to repeat this analysis until we have considered all packets in the current window. In conclusion, in Case 2, if $\exists m \in [1, w - k]$, the ACK of packet $k+m$ is received by the sender, then the sender is ensured that packet k is sent successfully. For this situation, Figure 3 (left) illustrates an example. If for $\forall m \in [1, w - k]$, the ACK of packet $k+m$ is not received by the sender, packet k is considered lost by the sender TCP, see Figure 3.

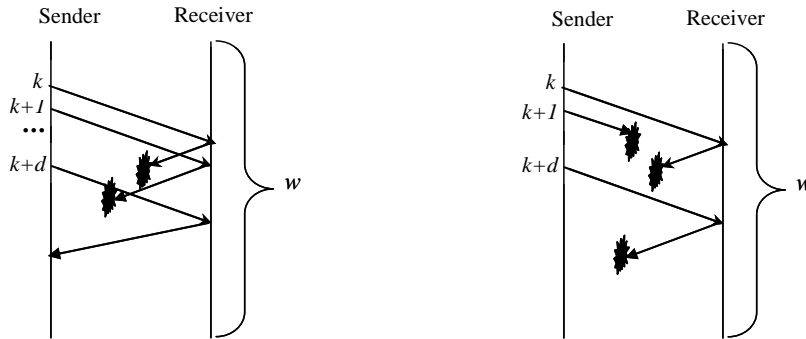


Figure 3: Packet k considered sent (left) and lost (right)

Denote p_k as the probability that packet k is considered lost by the sender. Then we have,

$$p_k = [p_f + (1 - p_f)p_b]^{w-k}.$$

Therefore, in Case 2 the expected probability that a packet is considered lost by the sender, denoted by $E[p_{case2}]$, can be expressed as follows:

$$E[p_{case2}] = \sum_{k=1}^w \frac{p_k}{w} = \sum_{k=1}^w \frac{[p_f + (1-p_f)p_b]^{w-k}}{w}.$$

Combining the results for Case 1 and Case 2, the complete expression for p , denoting the probability that a packet is considered lost by the sender, can be deduced:

$$p = p_f + (1-p_f)p_b \sum_{k=1}^w \frac{[p_f + (1-p_f)p_b]^{w-k}}{w} = p_f + \frac{p_b}{w(1-p_b)} [1 - (p_f + p_b - p_f p_b)^w]. \quad (4)$$

Expression (4) includes the unknown variable w , the current window size. But in fact, we should not just focus on one window size or on one TCP round. If the ACKs for the packets sent before packet k are received by the sender, then that triggers sending new data packets in the new round. The ACKs of the newly sent data packets have also influence on determining whether packet k is considered lost by the sender or not. Therefore, we redefine w as the expected number of packets sent up to and including the first lost packet. Because $p_3 = p_f + (1-p_f)p_b$, we approximate w as:

$$w \approx \sum_{k=1}^l k(1-p_3)^{k-1} p_3 + l(1-p_3)^l = \frac{1 - (1-p_3)^{l+1}}{p_3}, \quad (5)$$

where l represents the number of remaining data packets to send when a packet loss occurs. Note that l reduces during the data transmission. Since the value of l is between 0 and d , where d stands for the file size, for simplification, we approximate l as $d/2$. Our model for the mean TDT is now complete:

$$TDT = \frac{3}{4} RT + DTT(p),$$

where RT denotes the Response Time derived in Section 3 and DTT is the Cardwell formula for the DTT where we use the packet loss probability given in (4) where w satisfies (5) with $l = d/2$.

5. Validation

To assess the accuracy of the models developed in Section 4, we have performed extensive ns-2 simulations. The results are outlined below. The topology of our simulation is depicted in Figure 4.

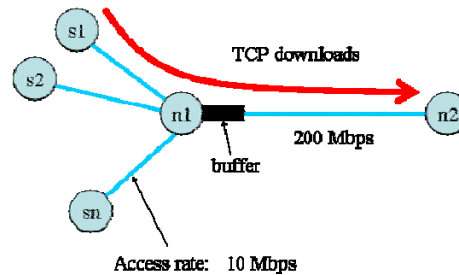


Figure 4: Simulation topology

In Section 5.1 we assess the accuracy of the model for the RT , developed in Section 3.1. In Section 5.2 we validate the model for the TDT developed in Section 3.2.

5.1 Response Times

For the validation of the model for the RT, the packet loss p at the aggregation link n1-n2 has been varied between 0-10%. It is assumed that the packet loss is random, i.e. without correlation between consecutive lost packets. In our simulation packets in both up- and downlink directions suffer from this packet loss. The (minimum) RTT is set to be 600 ms. The NS-2 script has recorded actual values of the packet loss, RTT and the mean Response Time. Figure 5 below shows the mean RT as a function of the loss probability, for the simulations and for the model.

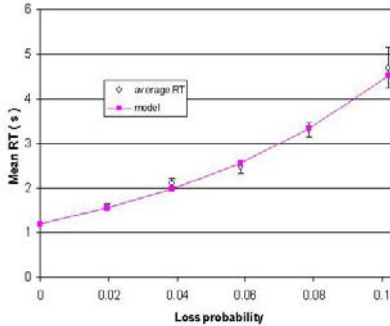


Figure 5: Comparison between model and simulation for mean Response Time

We conclude from Figure 5, and various additional simulations, that our model for the mean RT is highly accurate. In fact, analytical result for the mean RT always falls within the 95%-confidence interval of the simulated values.

5.2. Total Download Times

In this section we report ns-2 simulation experiments that have been run as a validation for our model for the mean Total Download Time. The simulation topology is given by Figure 4. We ran several simulations where we varied the access link rate, the RTT, the maximum window size W_{\max} and the file size. For all simulations we have set the number of data packets acknowledged by one ACK equal to 1 while the initial slow-start window size is 1 packet. In Figure 6 and Figure 7 we have MSS=1640 Bytes and the link capacity equal to 200 Mbps. The forward loss probability is set equal to the backward loss probability.

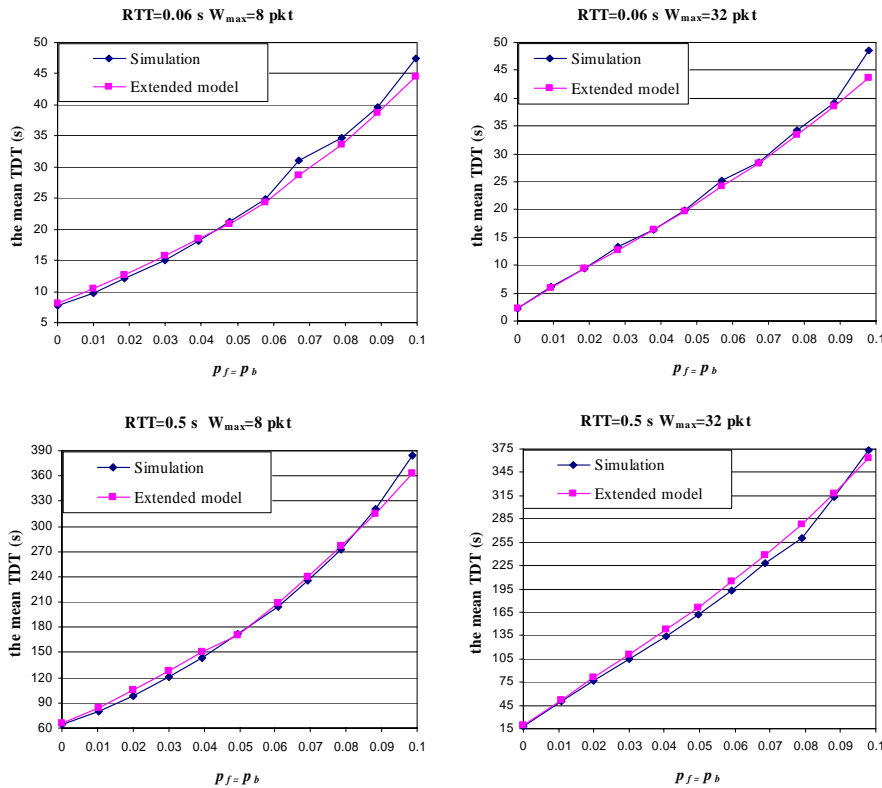


Figure 6: Simulation vs. model results for bi-directional packet loss: access link = 30 Mbps

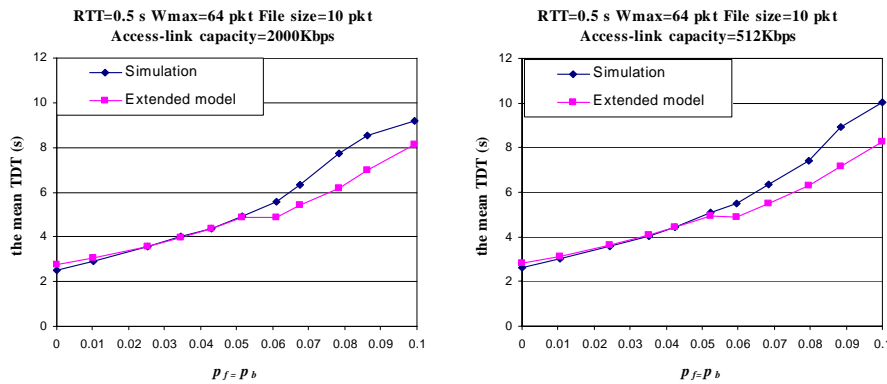


Figure 7: Simulation vs. model results for bi-directional packet loss: access link = 2 Mbps (left), access link = 512 kbps (right)

Figure 6 and Figure 7 and additional simulations show that our model works very well for predicting the mean TDT for large files. The relative errors under these new situations are very low; in most cases they are within $\pm 6\%$. For small files our model is very accurate under the condition that the forward and backward loss probabilities are at most 5%.

The experiments mentioned above are all under the assumption that the forward loss probability is equal to the backward loss probability. Obviously in a realistic network environment this is not necessarily the case. Therefore we simulated another scenario in which the loss probabilities in both directions are not necessarily equal. For this experiment we set MSS=1000 Bytes with a

link capacity of 100 Mbps. Furthermore, we assume that $RTT = 0.3$ seconds, access-link capacity = 2 Mbps, $W_{\max} = 32$ packets, while the backward loss probability is fixed at 2%. The forward loss probability is varied between 0%-10%.

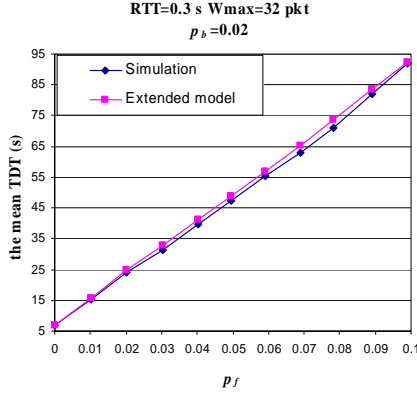


Figure 8: Simulation vs. model results for bi-directional packet loss with fixed ACKs loss rate

From Figure 8, and similar simulation results not presented here, we conclude that our model is also accurate for predicting the mean TDT under the situation that the forward packet loss probability is unequal to the backward packet loss probability.

5. Correlated packet loss

According to [8], the correlation structure of the packet loss process can be modeled with low order Markov chains. In particular, the two-state Gilbert model was found to be an accurate model in many studies, see for instance [9, 10]. Therefore we use the Gilbert model to simulate packet loss patterns over links. For simplification, we only consider packet losses in the forward direction, thus it is assumed to ACK's are not lost i.e. $p_b = 0$.

In the Gilbert model, one state represents a lost packet which is called state 'B', and the other state represents the situation when a packet is successfully delivered to the destination which is referred to as state 'G'. Let m denote the probability of going from state 'G' to state 'B', and n is that of staying in state 'B'. On the link between the sender and the receiver, we denote π_B as the average packet drop rate which satisfies

$$\pi_B = \frac{m}{1 + m - n}.$$

Substituting this packet loss probability defined in our model for TDT (replace p_f by π_B), we get the model for predicting the mean TDT in the situation when packet losses are correlated according to the two-state Gilbert loss model.

To validate the accuracy of the model for the case of correlated packet loss, we ran a variety of additional NS-2 simulations for the simulation topology depicted in Figure 4. For this experiment, $MSS = 1000$ Bytes while the link capacity was set to 200 Mbps. Furthermore, we assume that $RTT = 0.2$ seconds, access-link capacity = 10 Mbps, $W_{\max} = 32$ packets. In the forward direction the probability of going from state 'G' to state 'B' (m): $\{0, 0.01, \dots, 0.1\}$ while the probability of going from state 'B' to state 'G' (n): $\{m/5, 5m\}$. Figure 9 shows the mean TDT as a function of m .

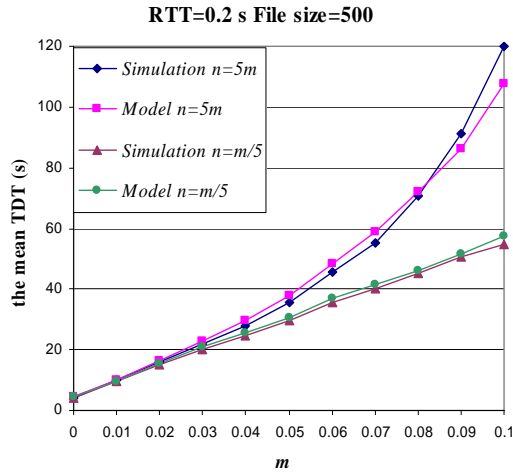


Figure 9: Simulation vs. model results for correlated packet loss

Figure 9 clearly demonstrates that when packet loss on links is correlated and the correlation of packet loss are known, we can apply our model to predict the mean TDT by substituting the packet loss probability p by π_B .

6. Comparison

To justify the relevance of the inclusion of bi-directional packet loss in the model, we again have performed various simulations. Figure 10 below shows the mean TDT as a function of the packet loss ratio for the following two scenarios. In Scenario 1 (left-hand side), the RTT = 60 ms and the maximum TCP window size 8 packets, while in Scenario 2 (right-hand side) we have the RTT = 500 ms and the maximum window size 32 packets.

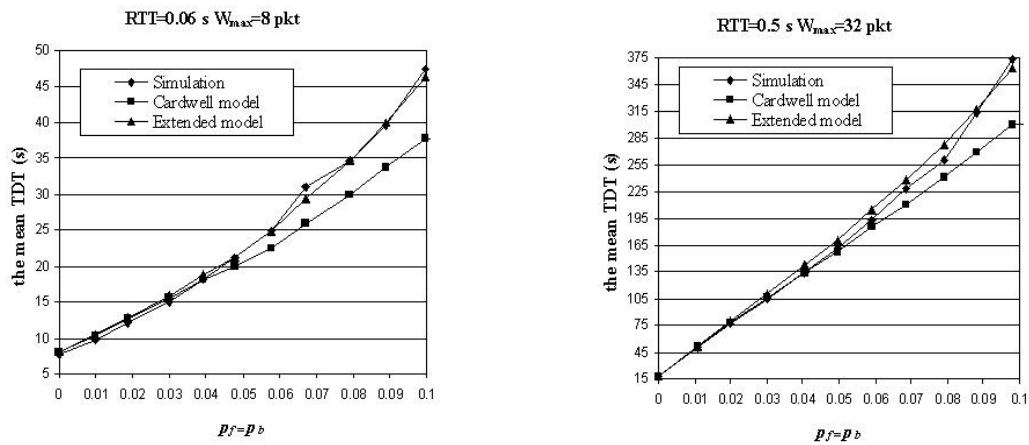


Figure 10: Simulation vs. model results for correlated packet loss

The results shown in Figure 10 demonstrate that our extended model outperforms the Cardwell model [2], especially when the packet loss becomes significant. Moreover, it shows that the inclusion of bi-directional packet loss is indeed justified, and leads to more accurate performance predictions. As such the model extension discussed in the paper leads to more accurate TCP performance predictions.

8. Conclusions and topics for further research

In this paper we have extended the commonly used TCP performance model for the download times by Cardwell [2] by including the impact of bi-directional packet loss. Extensive simulations with NS-2 show that this bi-directional packet loss may have a strong impact on TCP performance, and that the proposed model refinement accurately captures the impact of bi-directional packet loss, and correlations between the loss occurrences. As such the model refinement is highly valuable for IP link dimensioning purposes.

The results lead to a number of challenges for further research. First, the model considered in this paper is focused on TCP performance over a single network domain. However, next-generation communication services (e.g., online consumer services, E-commerce applications) will typically have highly distributed architecture, crossing multiple administrative domains. Extension of the model towards the inclusion of multiple domains is a challenging topic for further research. Second, in many communication networks TCP-based applications and UDP-based applications will be integrated. Inclusion of the inclusion of UDP streams on the performance of TCP is also a challenging area for further research.

9. References

- [1] J.Padhye, V.Firoiu, D. Towsley, J.Kurose, Modeling TCP throughput: A Simple Model and its Empirical Validation, *IEEE/ACM Transactions on Networking*, 8(2): 133-145 (April), 2000.
- [2] N. Cardwell, S. Savage, T. Anderson, Modeling TCP latency, *Proceedings of INFOCOM 2000*, March 2000.
- [3] Matthew Mathis, Jeffrey Semke, Jamshid Mahdavi, *The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm*, *Computer Communication Review*, 27(3), July 1997.
- [4] T. Ott, J. Kemperman, and M. Mathis, *The stationary behavior of ideal TCP congestion avoidance*. Preprint.
- [5] Network Simulator 2 (software), <http://www.isi.edu/nsnam/ns>.
- [6] TNO Telecom (The Netherlands) and Lucent Technologies Network Systems GmbH (Germany): Web browse quality modelling. ITU-T COM 12 – C3 – E, 2004.
- [7] V.Paxson, M.Allman, Computing TCP's Retransmission Timer, *Internet Archives RFC 2988*, November 2000.
- [8] C. Boutremans, J.Y. Le Boudec, Adaptive Joint Playout Buffer and FEC Adjustment for Internet Telephony, *IEEE Infocom.*, 2003
- [9] E. Altman, K.E. Avrachenkov, C. Bakarat, TCP in presence of Bursty Losses", *Proc. Of ACM Sigmetrics*, Santa Clara, California, June 2000.
- [10] E. Altman, K.E. Avrachenkov, C. Bakarat, P. Dube, TCP over a multi-state Markovian path, in *Perf. and QoS in Next Generation Networking*, K. Goto, T. Hasegawa, H. Takagi and Y. Takahashi (Eds.), Springer, 2000, pp. 103-122.