

# Multilevel Approach for Modeling Short TCP Sessions

U. Ayesta<sup>ab</sup>, K.E. Avrachenkov<sup>b</sup>, E. Altman<sup>b</sup>, C. Barakat<sup>b</sup> and P. Dube<sup>c</sup>

<sup>a</sup>France Telecom R&D, 905 rue A. Einstein, 06921 Sophia Antipolis Cedex, France.

<sup>b</sup>INRIA, 2004 route des Lucioles, B.P.93, 06902, Sophia Antipolis Cedex, France.

<sup>c</sup>IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA

## Abstract

We model the TCP/IP network with non persistent sessions on three levels: Packet level, Session level and System level. On the packet level we characterize the packet arrival process at the bottleneck queue and we calculate the packet loss probability using the fixed point approach. In particular, we study by simulation under which conditions the multiplexed traffic is close to Poisson. On the session level, using the fluid model approach and conditioning on the number of loss instants, we calculate the average transfer time of the TCP flows as a function of the packet loss probability and the parameters of the TCP protocol. We show that the expected latency conditioned on the fact that a packet has been lost is not monotone with respect to the file size. Finally, on the system level we apply the  $M/G/\infty$  model to obtain the distribution of the number of active sessions. All analytical results are confirmed by NS simulations.

## 1. Introduction

Up to the present, most significant research efforts have been devoted to the analysis of persistent TCP connections. However, the measurements on real IP networks [9] show that the TCP traffic is mainly composed of short TCP transfers (with the average size of 10Kbytes). The principal source of this type of traffic is Web HTTP-based applications. Still only few papers are available on modeling short TCP traffic [10,18]. In [10,18], the expected latency of the given document size is calculated as a function of the packet loss probability and the average round trip time. For design and dimensioning purposes, however, we cannot assume that the packet loss probability is known. Thus it is necessary to provide a mathematical model for its estimation.

In our work we follow the general idea of multilevel approach outlined in [12]. Namely we consider the TCP/IP network on three levels:

- (1) On the packet level we give bounds for the packet loss probability. Furthermore, using Fixed Point Approach (FPA), in the particular case of slow access links (in comparison to the bottleneck link) and high multiplexing we are able to calculate the packet loss probability and the load at the bottleneck queue with good accuracy. The main idea behind FPA for the TCP/IP network is to combine a model for the IP network at the packet level with a model for the TCP connection performance based on some given packet loss process [8,13,14].
- (2) Once the loss probability is obtained, we calculate the expected latency of a file transfer. This corresponds to the session level.
- (3) Finally, on the system level we calculate the distribution of the number of active TCP

sessions. For that purpose we use an  $M/G/\infty$  model for the TCP sessions, where the expected service time is obtained at the previous step.

We would like to note that already [10,18] treat the session level, that determines expected file latency as a function of loss probability. These references contain some rough approximations which we are able to avoid in the present work. Furthermore, with our approach we were able to model the new TCP modifications such as Increasing the Initial Window [2] and the Limited Transmit Algorithm [1].

Another interesting performance measure that we obtain is the expected latency of a file conditioned on the number of losses it suffers. We identify a paradoxical behavior in which this conditional (as opposed to the unconditional) expected latency is not monotone in the file size (Fig. 8), and we provide an explanation to this phenomenon.

## 2. Benchmark network model

Throughout this paper we demonstrate the application of the multilevel approach on the benchmark example of TCP/IP network with a single bottleneck. This topology may for instance represent an access network. The capacity of the bottleneck link is denoted by  $C$  and its propagation delay is denoted by  $d$ . The capacities of  $N$  links leading to the bottleneck link are supposed to be large enough (or the load on each access link is small enough) so that they do not hinder the traffic. Each of these  $N$  links has a propagation delay  $d_i$  (the difference in propagation delays also improves multiplexing) and we assume that new TCP connections arrive on link  $i$  according to a Poisson process with rate  $\lambda_i$ . Thus, the nominal load can be calculated by  $\rho_0 = E[doc.size] \sum_{i=1}^N \lambda_i / C$ , where  $E[doc.size]$  is the average document size to be transferred. We use the exponential and Pareto distributions for the document size (with  $E[doc.size] = 10Kbytes$  and Pareto with infinite variance ) [9,12].

All theoretical results presented in the paper are confirmed by NS<sup>1</sup> simulations. In the NS simulations we use the following values for the network parameters: bottleneck capacity is 100Mbps, the bottleneck buffer size 50 packets, bottleneck link propagation delay 40ms, the number of access links 100, the access link capacity is the same for all accesses links and is varied between 200Kbps and 2000Mbps , the propagation delays of access links are uniformly distributed between 20 and 60ms, and the maximum segment size (MSS) is 500bytes. We use the New Reno TCP version. As for the buffer management, we consider Drop-Tail policy. It is still the most commonly used buffer management policy in the Internet. The buffer sizes of the access links are chosen large enough so that losses occur only in the bottleneck queue.

## 3. Packet level: Calculation of the packet loss probability

From extensive simulations (see [4]), we noticed that, if the ratio  $NC_{ac}/C$  is not large (around 2 for the bottleneck network topology) the distribution of the packet interarrival times is very close to exponential. In [4] we validate this statement for different number of access links, loads and average file sizes and in the presence of persistent TCP connections. However we note that even if  $NC_{ac}/C \approx 2$ , the packet interarrival times are correlated (see Figure 3). In particular, we observe that the number of lags corresponding to the maximum value of the correlation is equal to the ratio between the access link transmission time and the average packet interarrival time at the bottleneck node. The interpretation for this is that the correlation is introduced by packet pairs sent in the Slow-Start phase from the same access link.

In Figures 1, 2 we plot the distribution of the packet interarrival time for the cases of exponential and Pareto distributions and different access link capacities. Let us now explain in more detail the case depicted in Figure 1. In the case of slow access links ( $C_{ac} = 2Mbps$ ) the

<sup>1</sup><http://www.isi.edu/nsnam/ns/index.html>. Release 2.18a

interarrival time distribution practically coincides with the exponential. With the increase of the access link capacity the packet interarrival distribution starts to deviate from the exponential one. In particular, one can see the appearance of steps in the distribution function. In the case of access link capacities smaller than the one in the bottleneck (see  $C_{ac} = 50Mbps$ ) there is only one step corresponding to the transmission time of the access link. We can explain this again by the fact that packets come in pairs during the slow-start phase. When the access link capacity is greater than the bottleneck link capacity (see  $C_{ac} = 200Mbps$  and  $2000Mbps$ ) there are two steps. The first one corresponds to the transmission time of the access link and the second one corresponds to the transmission time of the bottleneck link. This second step can be interpreted as a typical time interval between two pairs of packets coming from the same access link. The above observations prompt us to approximate in the case of high access link the input process at the bottleneck queue as a batch arrival process.

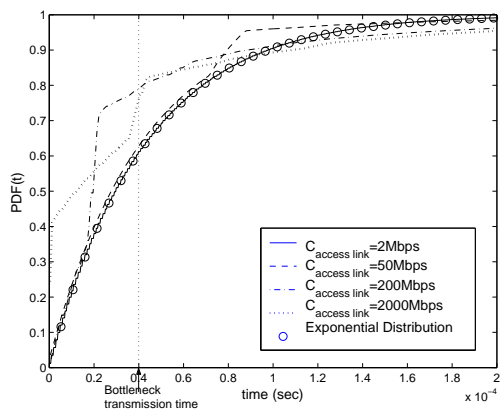


Figure 1. Bottleneck 100Mbps, 100 access links, Exponential file size and load 0.9.

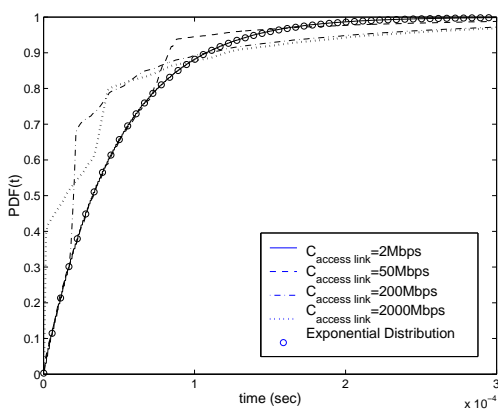


Figure 2. Bottleneck 100Mbps, 100 access links, Pareto file size and load 0.9

In the most extreme case all packets from the same round can be considered as a single batch. Thus in this extreme case the distribution of the batch size is given by the distribution of the congestion window size. We compute this distribution assuming the session does not experience any loss and that it will remain always in Slow-Start phase. This will give us an upper bound on the batch size. Conditioning on the number of rounds the probability of having a window of size  $w_i \in \{1, 2, 4, 8, \dots\}$  is,  $P(W = w_i) = \sum_{j=\lceil \log(W_i)+1 \rceil}^{\infty} \frac{1}{j} (F(2^j) - F(2^{j-1}))$  where  $F(j)$  is the distribution function of the file size in terms of packets. The expression  $\lceil \log(W_i) + 1 \rceil$  corresponds to the number of rounds which are needed in order to reach the congestion window  $w_i$ . Because of the monotonicity of the congestion window evolution during the slow start the probability of having window size  $w_i$  given the session lasts  $j$  rounds is simply equal to  $1/j$ . Next assuming the batches arrive according to a Poisson process and the service time is exponentially distributed we form the transition matrix of the corresponding Markov process and compute the steady state distribution  $\pi_i$ ,  $i = 0, \dots, K$ , where  $K$  is the bottleneck buffer size. Then we compute the packet loss probability  $p$  as follows:  $p = \frac{1}{E[W]} \sum_{i=0}^K \pi_i \sum_{w_j=K-i+1}^{\infty} (w_j - K + i) P(W = w_j)$ . In Figure 4 we plot the above packet loss probability for the batch model as a function of load. In the same figure we also plot the classical  $M/M/1/K$  model

$$p = \rho^K (1 - \rho) / (1 - \rho^{K+1}), \quad (1)$$

and NS simulation results for different access link capacities. All points obtained from NS simulations correspond to the following set of nominal loads  $\rho_0 = \{0.9, 0.925, 0.95, 0.975\}$ .

The  $M/M/1/K$  and the Batch model provide indications for lower and upper bounds for the packet loss probability. We note the loss probability is very sensitive to the value of the ratio  $NC_{ac}/C$ . For instance for  $C_{ac} = 2Mbps$  ( $NC_{ac}/C = 2$ ) the points lie close to the curve of  $M/M/1/K$  model, but a slight change in the access link capacity (see points for  $C_{ac} = 5Mbps$ ,  $NC_{ac}/C = 5$ ) results in a significant increase of the loss probability.

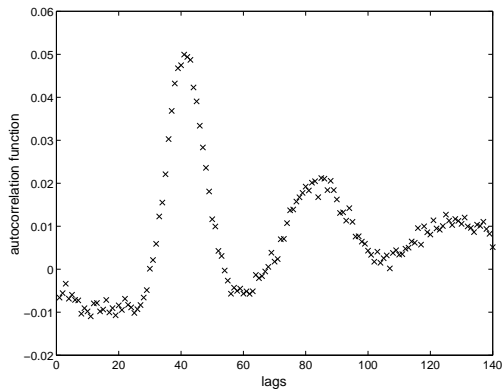


Figure 3. Correlation function of the packet arrival process: Bottleneck 100Mbps, 100 access links and  $C_{ac} = 2Mbps$

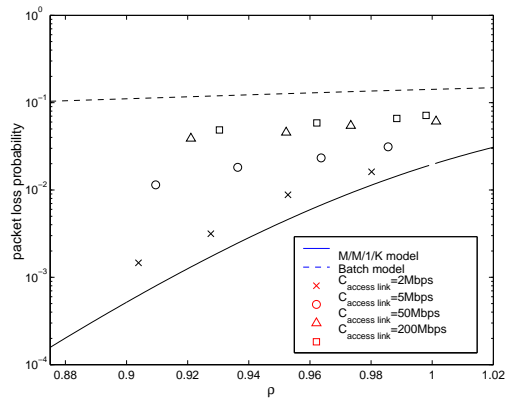


Figure 4. Bounds on packet loss probabilities

Next let us consider the particular case of small ratio  $NC_{ac}/C \approx 2$  when the  $M/M/1/K$  model provides a good approximation. In this case we are able not only to give bounds but also to compute actually the packet loss probability with good accuracy. We shall use the fixed point approach.

We recall that TCP is a protocol for the reliable data transfer. In particular, this means that lost packets have to be retransmitted again. Taking into account retransmissions, the actual load on the bottleneck link is given by the following formula [7]

$$\rho = \rho_0 / (1 - p), \quad (2)$$

where  $\rho_0 = E[doc.size] \sum_{i=1}^N \lambda_i / C$ . In the above formula one takes into account that only one packet is retransmitted per packet loss. This equation models the ideal behavior of TCP but in reality TCP senders may retransmit more packets than actually lost, e.g., in the case of timeout.

**Proposition 1** [3] *If  $\rho_0 < 1$ , the system of equations (1) and (2) has a unique solution. This solution can be found by fixed point iterations  $\rho^{(n+1)} = \rho_0(1 - (\rho^{(n)})^{K+1}) / (1 - (\rho^{(n)})^K)$ , which converges for any initial value  $\rho^{(0)} \in [0, \infty)$ .*

#### 4. Session Level: The expected latency of TCP transfers

In this section we present the model we use to calculate the expected time of a TCP file transfer. The expected latency  $L$  is computed conditioning on the number of losses. The input parameters for the model are the packet loss probability  $p$ , the average round trip time  $RTT$ , the document size to be transferred  $\bar{y}$ , the initial window size  $W_0$ , the initial slow start threshold  $W_{SS}$ , the maximum receiver's advertised window  $M$  and the number of packets the receiver acknowledges by one ACK  $b$  ( $b = 2$  in the DelAck option). As in all related previous works [10,18] here we assume that packets are lost independently with probability  $p$ . The fluid model

approach is used to represent the evolution of the congestion window. In particular, this means that instead of a discrete number of packets, a continuous volume of data (of course, with the same size) is transferred over the network. It turns out that the fluid model approach is more analytically tractable than the discrete one. Given that packets are lost in an independent fashion the number of packets successfully sent between two consecutive losses has a geometric distribution. To adapt this assumption to the fluid model, the standard approximation of the geometric distribution by an exponential distribution is used. The parameter  $\lambda$  of the exponential distribution can be determined from the following relation

$$1/\lambda = E[\text{transmitted bytes between two consecutive losses}] = MSS/p.$$

Since we condition on the number of losses, the expected latency for a file size of  $\bar{y}$  bytes is given by:

$$L(\bar{y}, W_0, W_{SS}) = \sum_{k=0}^{\infty} p(k|n) L_k(\bar{y}, W_0, W_{SS}), \quad (3)$$

where  $L_k(\bar{y}, W_0, W_{SS})$  is the conditional expected latency given that exactly  $k$  losses happened during the file transfer  $L_k(\bar{y}, W_0, W_{SS}) = E[\text{latency}|p, RTT, \bar{y}, W_0, W_{SS}, \text{loss no.} = k]$ . and  $n = \lceil \bar{y}/MSS \rceil$ . In the case of no losses, the file transfer time  $L_0$  can be calculated from the analysis of the deterministic evolution of the TCP congestion window (see the derivation in [3]). For  $k = 1, 2, \dots$ , we calculate  $L_k(\bar{y}, W_0, W_{SS})$  using the recursive approach as outlined below. It is worth noting that in this paper we do not take into account the three way handshake mechanism of TCP. One can easily take this additional delay into account as in [10].

First we calculate  $p(k|n)$  which is the probability of having  $k$  retransmission when  $n$  packets are transmitted.

**Proposition 2** [3] *Let the file size of  $n$  packets be transferred over a lossy link (with packet loss probability  $p$ ) by an ideal reliable protocol. Then the probability of having  $k$  retransmissions is given by  $p(k|n) = p^k(1-p)^n C_k^{n+k-1}$ .*

The model is based on the TCP NewReno and SACK versions. Under this assumptions both flavors of TCP would behave in a similar way. In the model we take into account the Limited Transmit Algorithm (LT)[1], an important modification to the TCP protocol. With LT, the TCP sender sends a new data segment in response to each of the first two duplicate ACKs. Eventually it will receive a third duplicate which will trigger off fast retransmission and fast recovery phases. Let us define as  $dup_{rt}$  the number of duplicate ACKs the sender must receive in order to infer a loss has occurred<sup>2</sup> and start running fast recovery and fast retransmission algorithms. Similarly, let us define  $dup_t$  as the number of duplicate ACKs the sender must receive to avoid a timeout<sup>3</sup>. It is important to remark that neither  $dup_t$  nor  $dup_{rt}$  depend on  $b$ , since the receiver acknowledges every packet received out of order.

In general, when a loss occurs, the congestion window of the sender will continue sliding forward until the lost packet gets to the left most position. If the value of the congestion window is less than  $1 + dup_{rt}$  the TCP session will timeout. Given the initial settings of a TCP session we define the parameter  $y_{to}$  as the amount of bytes sent until this value is reached. Therefore the value for  $y_{to}$  is simply given by the amount of bytes sent  $y_f$  up to the final sending rate  $W_f$  reaches the value  $(1 + dup_{rt})MSS/RTT$ . However, if the initial window size is greater or equal to  $1 + dup_{rt}$  the sender will not timeout. Thus, we have

$$y_{to} = y_f((1 + dup_{rt})MSS/RTT, W_0, W_{SS})1\{W_0 < (1 + dup_{rt})MSS/RTT\}$$

<sup>2</sup> $dup_{rt}$  is commonly set to three.

<sup>3</sup> $dup_t$  is equal to 1 if the sender use LT and  $dup_{rt}$  otherwise.

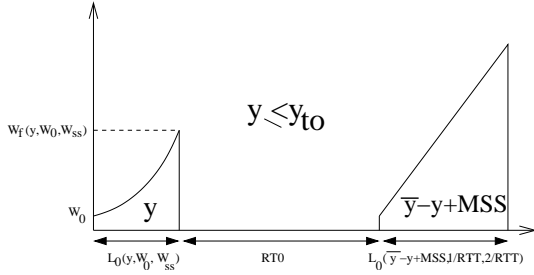


Figure 5. Sending rate evolution when a packet is lost before  $y_{to}$  bytes are sent

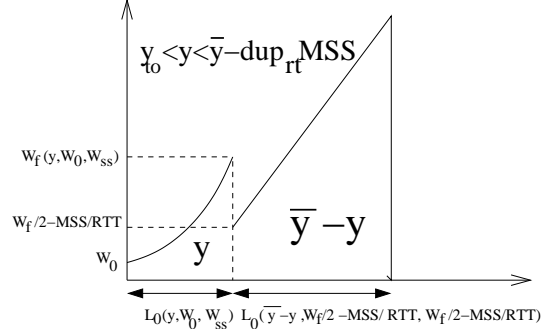


Figure 6. Sending rate evolution when a packet is lost when  $y$  ( $y_{to} < y < \bar{y} - dup_{rt}MSS$ ) bytes are sent

There is yet another situation when the sender will inevitably timeout. Namely, if a loss occurs when the remaining amount of data is less than  $dup_{rt}MSS$ , no matter what the actual value of the sending rate is, the sender will not receive three duplicate ACKs and will have to rely on a timeout to detect the loss.

As a consequence, one can identify three situations where TCP sessions are prone to timeouts and where a single packet loss induces a timeout. The first case corresponds to the beginning of the session when the congestion window is below 4 (2 with LT) segments. The second case corresponds to the middle part of the transfer when the congestion window is small. For example the limit imposed by the receiver advertised window is small, the link has a small bandwidth-delay product or after the loss recovery phase. The third case corresponds to the very end of the transmission. Namely if any of the last three segments are lost the sender will not receive three duplicate ACKs and it will inevitably timeout.

A timeout is very harmful from the performance point of view. The TCP retransmission time  $RTO$  is based on measured round-trip times between sender and receiver as specified in [17] and therefore to avoid retransmissions of packets that are only delayed and not lost the minimum  $RTO$  is conservatively chosen to be 1 second. For computational purposes we substitute the value of the timeout  $RTO$  with the expression  $Max(1.0, 4 * RTT)$  as suggested in [11].

Let us analyze what can happen during the transmission of a file when a packet is lost. We consider three scenarios depending on the amount of bytes  $y$  sent before the loss occurs, see (Figures 5, 6 and 7). In Figure 5 we observe that when  $y \leq y_{to}$  bytes are sent, the total transfer time will be equal to the sum of the time required to send  $y$  bytes, the retransmission time  $RTO$  and the time required to send  $\bar{y} - y + MSS$  bytes.

In the second scenario, Figure 6, the TCP sender will detect the loss event upon the reception of three duplicate ACKs. In this case the TCP sender will halve its sending rate  $W_f$  and it will transmit  $W_f RTT/2 - MSS$  new bytes in addition to the lost  $MSS$ . In the next round upon the reception of all the ACKs it will transmit  $W_f RTT/2$  new bytes. For the sake of simplicity, in the fluid model we consider that the remaining data is  $\bar{y} - y$  and the new sending rate is set to  $W_f/2 - MSS/RTT$ . This approximation on the remaining data allows us to keep exact track of the evolution of the sending rate. In this scenario the total transfer time is equal to the sum of the time required to send  $y$  bytes with the initial parameters and the time required to send  $\bar{y} - y$  after the loss event with the new TCP settings ( $W_0$  and  $W_{SS}$ ).

The third scenario, Figure 7, corresponds to the situation when a loss occurs and the amount of data remaining is less than  $dup_{rt}MSS$ . In this case the transfer time is equal to the sum of the time required to send  $y$  bytes, the retransmission time  $RTO$  and the time required to retransmit the lost packet (around one  $RTT$ ).

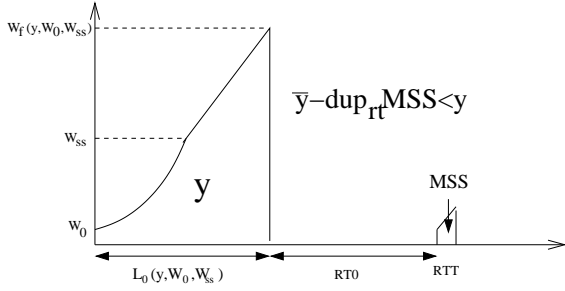


Figure 7. Sending rate evolution when a packet is lost when  $y$  ( $\bar{y} - \text{dup}_{rt} MSS < y$ ) bytes are sent

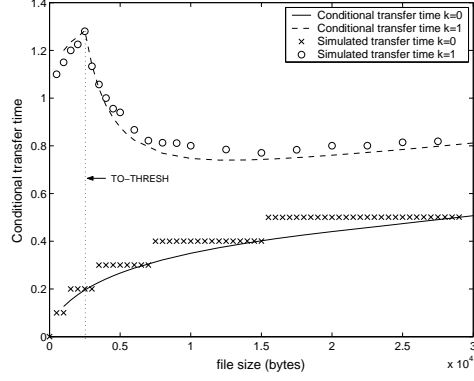


Figure 8. Expected conditional transfer time obtained by our fluid model approach

After a timeout, the value of  $W_0$ , regardless the value of the sending rate just before the timeout  $W_f$ , is 1 packet and the new value for the slow-start threshold is  $W_{SS} \approx \max(W_f/2, 2MSS/RTT)$ .

Next, let us define  $f(y, k, \bar{y})$  as the density function of the amount of data  $y$  that is sent before the first loss occurs given that  $k$  loss events happened during the transmission of  $\bar{y}$  data. This density is related to the uniform distribution, namely, the distribution of the amount of bytes sent before the first loss  $y$  is the minimum of  $k$  independent uniformly distributed random variables. Thus the corresponding density function is given by  $f(y, k, \bar{y}) = f(y|N_{\bar{y}} = k) = \frac{k}{\bar{y}} \left(1 - \frac{y}{\bar{y}}\right)^{k-1}$ . Putting together all above results and using the recursive approach the expected conditional latency of a TCP file transfer is calculated by

$$\begin{aligned}
L_k(\bar{y}, W_0, W_{SS}) &= \\
&= \int_0^{\max(0, \min(y_{to}, \bar{y} - \text{dup}_{rt} MSS))} \left[ L_0(y, W_0, W_{SS}) + RTO + L_{k-1}(\bar{y} - y + MSS, \frac{MSS}{RTT}, \frac{W_f}{2}) \right] f(y, k, \bar{y}) dy + \\
&+ \int_{\max(0, \min(y_{to}, \bar{y} - \text{dup}_{rt} MSS))}^{\bar{y}} \left[ L_0(y, W_0, W_{SS}) + L_{k-1}(\bar{y} - y, \frac{W_f}{2} - \frac{MSS}{RTT}, \frac{W_f}{2} - \frac{MSS}{RTT}) \right] f(y, k, \bar{y}) dy \\
&+ \int_{\max(0, \bar{y} - \text{dup}_{rt} MSS)}^{\bar{y}} \left[ L_{k-1}(y, W_0, W_{SS}) + RTO + L_{k-1}(MSS, \frac{MSS}{RTT}, \frac{W_f}{2}) \right] f(y, k, \bar{y}) dy.
\end{aligned}$$

Now knowing the expected conditional latency  $L_k(\bar{y}, W_0, W_{SS})$  and the probabilities  $p(k|\bar{y})$  one can calculate the expected latency by formula (3).

#### 4.1. Model validation and comparison with related works

We have already validated the derivation of the loss probability in Section 3. We proceed here to validate the second step: determining the expected conditional latency of a transfer for given packet loss probabilities.

Using the packet loss probabilities for different loads obtained by the fixed point approach, we compute the expected latency for document sizes of 7000bytes (see [3] for more results) by the formula (3) for the links with the shortest and the longest propagation delays. The average  $RTT$  is computed using the standard  $M/M/1/K$  model  $RTT_i = 2(d + d_i) + \frac{MSS}{C} \left[ \frac{1}{1-\rho} - K \frac{\rho^K}{1-\rho^K} \right]$ . In addition we calculate the expected TCP latencies using the formulas derived in [10,18]<sup>4</sup> and we also obtain the expected TCP latencies from the measurements of the NS simulations(see Figures 9 10.

<sup>4</sup>Two models are in fact proposed [18]. We use the empirical one which has been shown in [18] to be a very close approximation of the analytical model.

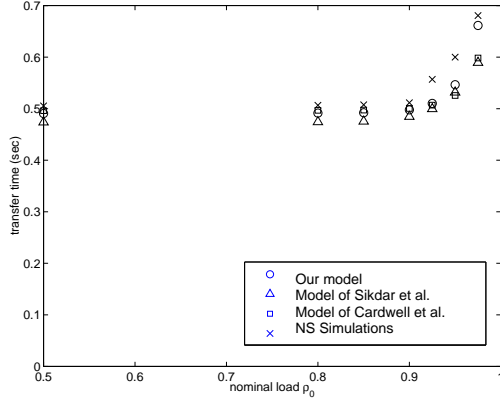


Figure 9. Transfer time for 7000bytes document size over the link with the shortest propagation delay

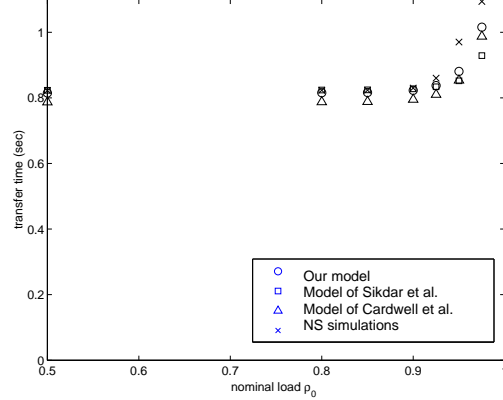


Figure 10. Transfer time for 7000bytes document size over the link with the longest propagation delay

All considered models provide good estimation for the average transfer time. In particular, it is surprising how well the simple experimental formula of [18] performs. Even though our approach looks more complicated than the models of [10,18], it has the following advantages:

- First of all, we have succeeded to avoid the introduction of several rough approximations and assumptions that were imposed in [10] and [18]. For example, the model of [10] uses the steady-state formula of [16] right after the first loss occurs and it neglects the possibility of timeouts in the case of small congestion window. The effect of timeouts is of crucial importance in the modeling of short TCP transfers, specially on the links of high load, see the data for  $\rho_0 = 0.975$  in Figures 9 10. The model of [18] uses empirical approximations in order to derive a model for more than one loss (Section 3.5 there). Furthermore, the possibility of time out that inevitably occurs when one of the last  $dup_{rt}$  packets is lost is not modeled in [10,18].
- Our model has several input parameters that can easily be controlled. In the models of [10,18] the slow-start threshold is always set to the maximum receiver window size and in the model of [18] the initial window is fixed to 2 packets. These parameters could be changed or even could be variable in the future TCP modifications.
- Our approach provides the expected conditional TCP latency given the number of losses. In Figure 8 we plot the expected conditional latencies as functions of the document size. Our model captures a very interesting phenomena – the non monotonicity of the expected conditional latencies. This behavior is due to the conservative value of the retransmission timer, typically 1s. In fact, one can define a threshold on the file size ( $TO - THRESH$ ), such that if the file size is less than this threshold a loss will inevitably lead to a timeout. The value of  $TO - THRESH$  is given by the sum of  $y_{to}$  and  $dup_{rt}MSS$ . This explains the high variability in the transfer time we have observed between sessions depending whether or not they experience a loss. The value of  $TO - THRESH$  may be reduced deploying LT and retransmitting early packets at the end of the transmission. For more detailed discussion on reducing the value of  $TO - THRESH$  we refer the reader to [5,6].
- Recently several modifications to TCP such as Increasing the Initial Window (IW) [2] and Limited Transmit (LT) [1] algorithm have become standards. The flexibility of the present model permits to analyze the IW and LT proposals [5].
- Taking into consideration small number of losses one can obtain various explicit approximating formulas. This can be an interesting issue for future research.



## 5. System Level: An $M/G/\infty$ model for the number of active sessions

In this section we use  $M/G/\infty$  model to calculate the distribution of the number of active TCP sessions. Let us first obtain the distribution of the number of active TCP session going through access link  $i$ ,  $i = 1, \dots, N$ . The expected transfer time of a document going through link  $i$  is given by  $\bar{L}^{(i)} = \int L(p, RTT_i, y, W_0, W_{SS}) dP_y$ , where  $P_y$  is the distribution of the file size and  $p$  is the packet loss probability.

Then, according to the  $M/G/\infty$  model, the number of active TCP sessions on the access link  $i$  is distributed according to the Poisson distribution  $Pr\{k \text{ TCP sessions are on}\} = \frac{1}{k!} (\lambda^{(i)} \bar{L}^{(i)})^k e^{-\lambda^{(i)} \bar{L}^{(i)}}$ . For the bottleneck link, we can also apply the  $M/G/\infty$  model but with the following parameters  $\lambda = \sum_{i=1}^N \lambda_i$ ,  $\bar{L}^{bn} = \sum_{i=1}^N \frac{\lambda_i}{\lambda} \bar{L}^{(i)}$ . Namely, the distribution of active TCP sessions sharing the bottleneck link is given by  $Pr\{k \text{ TCP sessions are on}\} = \frac{1}{k!} (\lambda \bar{L}^{bn})^k e^{-\lambda \bar{L}^{bn}}$ .

As in [15,12], we suggest to use  $M/G/\infty$  up to loads at which the behaviors of TCP flows are independent. The interval of such loads can be detected from Figure 11. Namely this is the interval of loads at which the latency stays approximately constant. For example in Figure 12 we present the  $M/G/\infty$  model and the measurements from the NS simulation for the nominal load 0.8 respectively. One can see that  $M/G/\infty$  models well the behavior of the TCP sessions on the system level even up to significant loads.

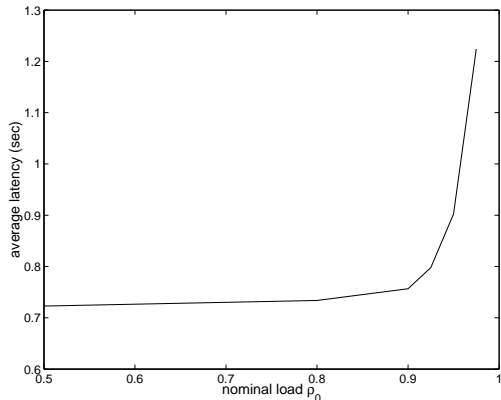


Figure 11. Average latency for different nominal loads

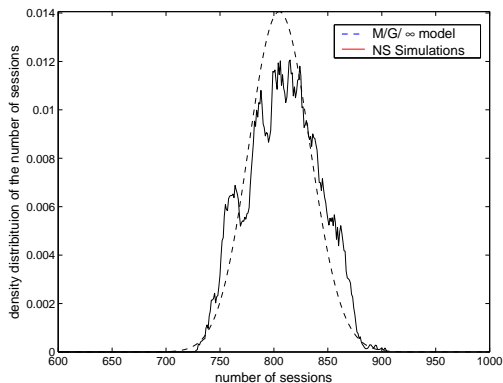


Figure 12.  $M/G/\infty$  model for load 0.8

## 6. Discussion

The methods proposed in this paper model TCP/IP network on several levels, from the packet level up to the high system level. On the packet level we are able to compute the packet loss probability and the load at the bottleneck queue by using the fixed point approach in the case of high multiplexing and slow access links. In particular we provide condition under which the aggregated traffic arriving at the bottleneck is close to Poisson. Then, for the session level we obtain the mathematically rigorous model for the expected TCP file transfer latency. In particular, using fluid model we obtain the conditional expected latency given the number of losses. We show that the expected latency conditioned on the fact that a packet has been lost is not monotone with respect to the file size. Finally for the system level we use  $M/G/\infty$  model to compute the distribution of the number of active flows.

Several extensions of our model may be carried out in the future to analyze new TCP modifications. Although a single bottleneck has many experimental and theoretical validation, one can propose a fixed point approach to handle a network with a general topology, in which one may not know a-priori where the bottlenecks are located. In the case of highly bursty TCP traffic one can use the fixed point approach in combination with the Batch model for the IP part. Another future research direction is to consider the dependence between per-flow losses, thus avoiding the assumption of i.i.d. packet losses. The research in this direction would be especially useful in the case when only few flows share the bottleneck.

## REFERENCES

1. M. Allman, H. Balakrishnan, S. Floyd, "RFC3042: Enhancing TCP's Loss Recovery Using Limited Transmit", January 2001.
2. M. Allman, S. Floyd, C. Partridge, "RFC 3390: Increasing TCP's Initial Window", October 2002.
3. U. Ayesta, K. Avrachenkov, E. Altman, C. Barakat, P. Dube, Multilevel Approach for Modeling Short TCP Sessions INRIA Technical Report, <http://www.inria.fr/rrrt/rr-4705.html>
4. U. Ayesta, K. Avrachenkov, E. Altman, C. Barakat, P. Dube, Simulation Analysis and Fixed Point Approach for Multiplexed TCP flows INRIA Technical Report, <http://www.inria.fr/rrrt/rr-4749.html>
5. U. Ayesta, K.E. Avrachenkov "The Effect of the Initial Window Size and Limited Transmit Algorithm on the Transient Behavior of TCP Transfers", 15th ITC Specialist Seminar on Internet Traffic Engineering and Traffic Management Wurzburg, Germany, July 2002.
6. M. Allman, K.E. Avrachenkov, U. Ayesta, J. Blanton "Early Retransmit for TCP" Internet-Draft, February 2003, Work in progress. Available online at: <http://www.ietf.org/internet-drafts/draft-allman-tcp-early-rexmt-00.txt>
7. C. Barakat, P. Thiran, G. Iannaccone, C. Diot and P. Owezarski, "A flow-based model for Internet backbone traffic", Technical Report EPFL/DSC/01/44, Jul. 2001. Also a shorter version appears in *ACM SIGMETRICS 2002*.
8. T. Bu and D. Towsley, "Fixed point approximation for TCP behaviour in an AQM network", in Proceedings of SIGMETRICS'2001 conference.
9. The web site of the Cooperative Association for Internet Data Analysis (CAIDA): <http://www.caida.org/>.
10. N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency", *IEEE INFOCOM*, pp.1742-1751, Tel Aviv, Israel, March 2000.
11. S. Dawkins, G. Montenegro, M. Kojo, V. Magret, N. Vaidya, "RFC3155: End-to-end Performance Implications of Links with Errors", August 2001.
12. S. Ben Fredj, T. Bonald, A. Proutiere, G. Regnie, J. Roberts, "Statistical Bandwidth Sharing: A Study of Congestion at Flow Level", SIGCOMM 2001.
13. M. Garetto, R. Lo Cigno, M. Meo, M. Ajmone Marsan, "A Detailed and Accurate Closed Queueing Network Model of Many Interacting TCP Flows," IEEE Infocom 2001, Anchorage, Alaska, USA, April 22-26, 2001.
14. R.J. Gibbens, S.K. Sargood, C. Van Eijl, F.P. Kelly, H. Azmoodeh, R.N. Macfadyen, N.W. Macfadyen, "Fixed-point models for the end-to-end performance analysis of IP networks," 13th ITC Specialist Seminar: IP Traffic Measurement, Modeling and Management, Sept 2000, Monterey, California.
15. A.A.Kherani, A. Kumar, "Stochastic Models for Throughput Analysis of Randomly Arriving Elastic Flows in the Internet", *IEEE INFOCOM 2002*, New York.
16. J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation", *ACM SIGCOMM*, Sep 1998.
17. V. Paxson, M. Allman, "RFC2988: Computing TCP's Retransmission Timer", November 2000.
18. B. Sikdar, S. Kalyanaraman and K. S. Vastola, "An Integrated Model for the Latency and Steady-State Throughput of TCP Connections", *Performance Evaluation*, v.46, no.2-3, pp.139-154, September 2001.