Multilevel Approach for Modeling Short TCP Sessions

Urtzi Ayesta, Konstantin Avrachenkov, Eitan Altman, Chadi Barakat, Parijat Dube

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Multilevel Approach for Modeling Short TCP Sessions

Urtzi Ayesta, Konstantin Avrachenkov,Eitan Altman,Chadi Barakat,Parijat Dube

N° 4705

Janvier 2003

THÈME 1

ISSN 0249-6399 ISRN INRIA/RR--4705--FR+ENG

INRIA
SOPHIA ANTIPOLIS

Multilevel Approach for Modeling Short TCP Sessions

Urtzi Ayesta,* Konstantin Avrachenkov,Eitan Altman,Chadi Barakat,Parijat Dube

Thème 1 — Réseaux et systèmes
Projets MISTRAL

Rapport de recherche n° 4705 — Janvier 2003 — 23 pages

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.

Key-words: TCP/IP modeling, short file transfers, $M/G/\infty$, Fixed Point Approach

* Urtzi.Ayesta@sophia.inria.fr

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Modélisation de connexions TCP de courte duree sur plusieurs niveaux

Résumé : Nous modélisons le réseau TCP/IP avec des sessions de courtes durées sur trois niveaux : niveau paquet, niveau session et niveau syste'me. Au niveau paquet, nous caractérisons le processus d'arrivée au goulot d'étranglement et calculons la probabilité de perte de paquet en utilisant la méthode du point fixe. Au niveau session, nous calculons le temps de transfert moyen d'un flux TCP en utilisant l'approche fluide et en conditionnant sur le nombre de pertes durant la transmission du flux. En particulier, nous montrons que le temps de transfert moyen conditionne sur la perte d'un paquet n'est pas monotone relativement à la taille des fichiers transmis. Finalement, au niveau système, nous appliquons le mode'le M/G/oo pour obtenir la distribution du nombre de sessions actives. Tous les résultats analytiques sont vérifies par des simulations sur NS.

Mots-clés : TCP/IP, connexions de duree courte,$M/G/\infty$,Approche Point Fix

1 Introduction

Up to the present, most significant research efforts have been devoted to the analysis of persistent TCP connections, see for instance [4, 20, 23] and references therein. However, the measurements on real IP networks [11, 22] 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 [12, 25]. In [12, 25], 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 [16]. 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 [10, 13, 17, 18].
(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 [12, 25] 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 [3] 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. 17), and we provide an explanation to this phenomenon.

The structure of the paper is as follows. The benchmark network model is introduced in Section 2. The packet level is analyzed in Section 3. The expected conditional latency is derived in Section 4, and finally the distribution of the number of active sessions as well as the overall average latency are derived in Section 5. We conclude with a section discussing the obtained results and future research directions.

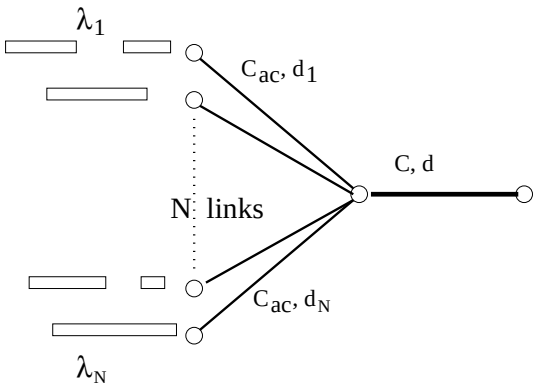

Figure 1: IP network with a single bottleneck link

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 (see Figure 1). 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 λ_i. Thus, the *nominal* load can be calculated as follows:

$$\rho_0 = \frac{E[doc.size] \sum_{i=1}^{N} \lambda_i}{C}, \tag{1}$$

where $E[doc.size]$ is the average document size to be transfered. We use the exponential and Pareto distributions for the document size (with $E[doc.size] = 10Kbytes$ and Pareto with infinite variance) [11, 22, 16].

All theoretical results presented in the paper are confirmed by NS[1] simulations. In the NS simulations we use the following values for the network parameters: bottleneck capacity – {100,50,10,5,1}Mbps, bottleneck buffer size – 50 packets, bottleneck link propagation delay – 40ms, the number of access links – {10,50,100}, access link capacity is the same for all accesses links and is varied between 200Kbps and 2000Mbps , propagation delays of access links are uniformly distributed between 20 and 60ms, and the maximum segment size (MSS) is 500bytes. As for the buffer management, we consider Drop-Tail policy. It is still the most

[1] http://www.isi.edu/nsnam/ns/index.html. Release 2.18a

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

Let us study the input process at the bottleneck queue. In particular, we are interested under which conditions the aggregated traffic arriving to the bottleneck queue is close to Poisson. We choose the nominal system loads $\rho_0 = \{0.3, 0.6, 0.9\}$. In Figures 2, 3, 4 , 5, 6, 7, 8 , 9, 10, 11 we plot the distribution of the packet interarrival time for different parameter settings. In general, we observe that when the capacity of the access link is much smaller than the one of the bottleneck, the interarrival time distribution practically coincides with the exponential. For example, one can see that with 50 access links (Figures 3, 6), a capacity of 2Mbps and 200Kbps respectively induce the interarrival time to be very close to exponential. Comparing Figures 2, 5 versus Figures 4, 7 we observe that the distribution of the packet interarrival times is closer to exponential. In Figures 8, 9, 10 we consider Pareto file size distribution for different nominal loads and in Figure 11 we consider Pareto with a larger average file size ($30KBytes$). All the above observations still hold. Thus, we conclude that with enough multiplexing, there exists such a "small enough" capacity for the access link so that the interarrival time distribution becomes very close to exponential. Namely, we noticed that, if the ratio NC_{ac}/C it is not large (around 2 in these particular settings) the distribution of the packet interarrival times is very close to exponential. However we would like to note that even in the case of slow access links the packet interarrival times are correlated (see Figure 12). 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. Let us now explain in more detail the case depicted in Figure 2. In the case of slow access links ($C_{ac} = 2Mbps$) the 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.

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 $w_i \in \{1, 2, 4, 8...\}$ is, $P(W = w_i) =$

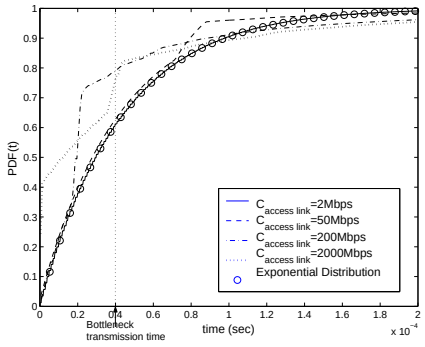

Figure 2: Bottleneck 100Mbps, 100 access links, Exponential file size and load 0.9.

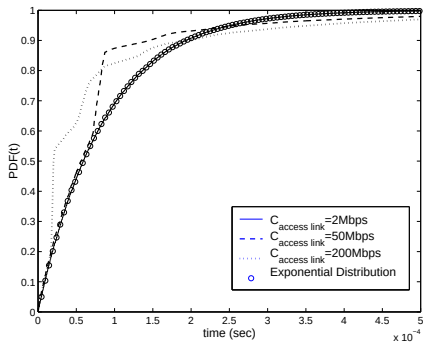

Figure 3: Bottleneck 50Mbps, 50 access links, Exponential file size and load 0.9

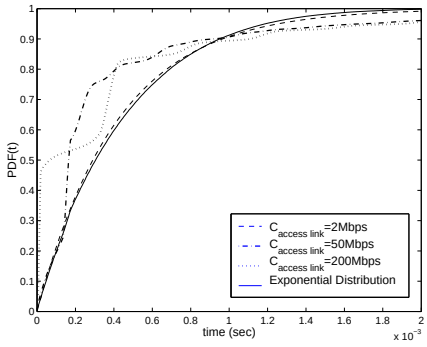

Figure 4: Bottleneck 10Mbps, 10 access links, Exponential file size and load 0.9

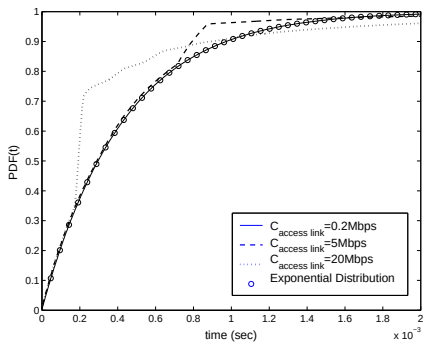

Figure 5: Bottleneck 10Mbps, 100 access links, Exponential file size and load 0.9.

$\sum_{j=\lfloor \log(W_i)+1 \rfloor}^{\infty} \frac{1}{j} \left(F(2^j - 1) - F(2^{j-1} - 1) \right)$ where $F(j)$ is the distribution function of the file size in terms of packets. The expression $\lfloor \log(W_i) + 1 \rfloor$ 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 π_i, $i = 0, ..., K$, where K is the bottleneck buffer size. Then we compute the packet loss probability p as follows:

$$p = \frac{1}{E[W]} \sum_{i=0}^{K} \pi_i \sum_{j=K-i+1}^{\infty} (j - K + i) P(W = w_i).$$

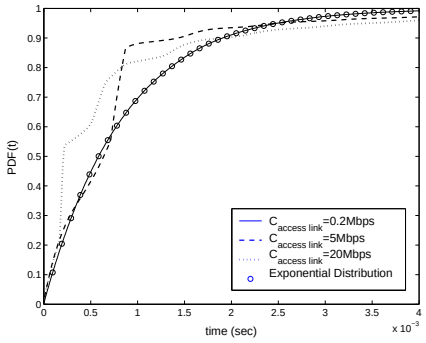

Figure 6: Bottleneck 5Mbps, 50 access links, Exponential file size and load 0.9

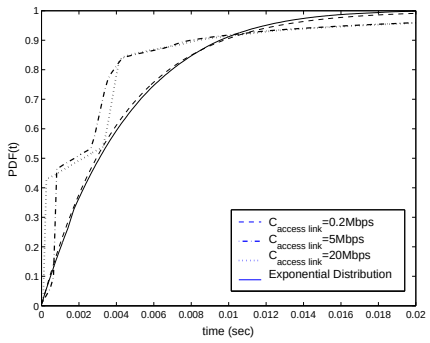

Figure 7: Bottleneck 1Mbps, 10 access links, Exponential file size and load 0.9.

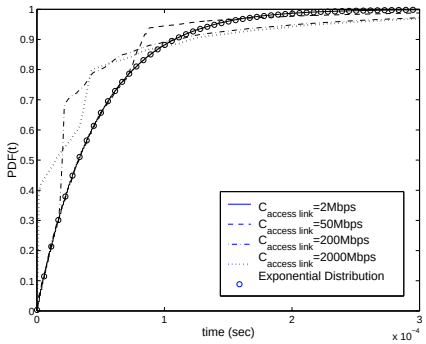

Figure 8: Bottleneck 100Mbps, 100 access links, Pareto file size and load 0.9

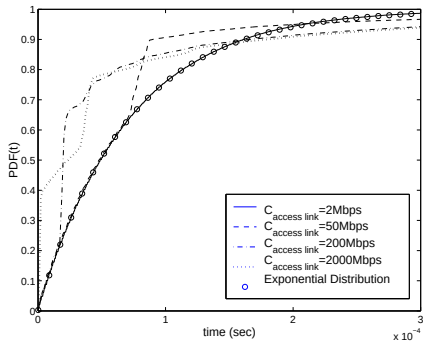

Figure 9: Bottleneck 100Mbps, 100 access links, Pareto file size and load 0.6

In Figure 13 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

$$p = \frac{\rho^K(1-\rho)}{(1-\rho^{K+1})}, \tag{2}$$

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 would like to note that for small values of ratio between the access link capacity and the bottleneck link capacity (about $1/50$ in our benchmark example) the value of the loss probability is very sensitive to this ratio. For instance for $C_{ac} = 2Mbps$ the points lie close to the curve of $M/M/1/K$ model, but a slight change

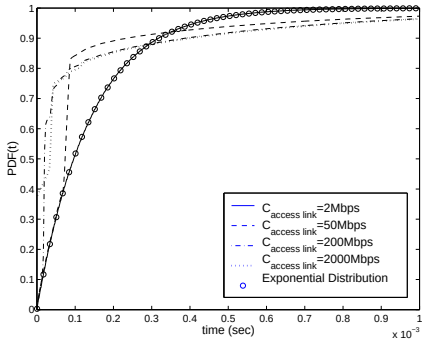

Figure 10: Bottleneck 100Mbps, 100 access links, Pareto file size and load 0.3

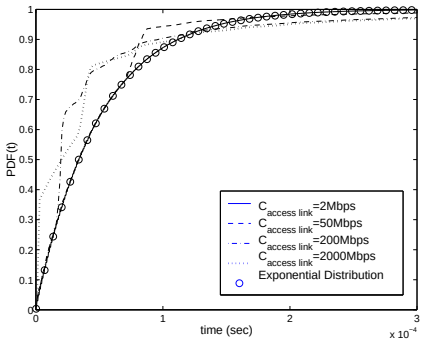

Figure 11: Bottleneck 100Mbps, 100 access links, Pareto file size, load 0.9, average file size 30KBytes

in the access link capacity (see points for $C_{ac} = 3Mbps$) results in a significant increase of the loss probability. On the other hand, when the ratio C_{ac}/C is greater than $1/2$ (in our model), the value of the packet loss probability stabilizes with the increase of C_{ac}.

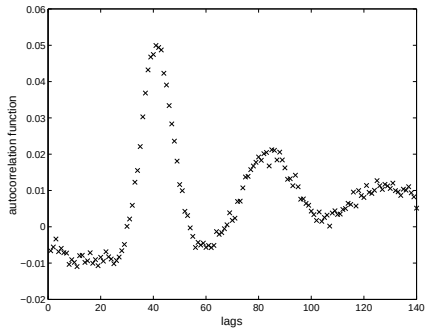

Figure 12: Correlation function of the packet arrival process:Bottleneck 100Mbps, 100 access links and $C_{ac} = 2Mbps$

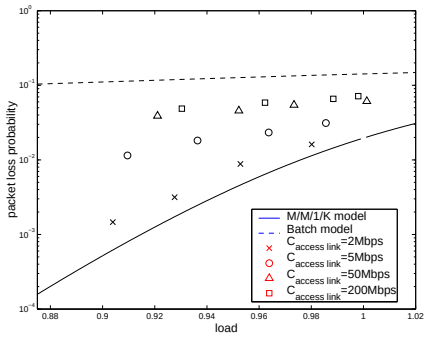

Figure 13: Bounds on packet loss probabilities

Next let us consider the particular case of small ratio C_{ac}/C when the $M/M/1/K$ model provides a good approximation (from now on, we use the following parameters in NS simulations: 100Mbps for the bottleneck, 100 access links and 2Mbps for the access link). 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]

$$\rho = \frac{\rho_0}{1-p}. \tag{3}$$

where ρ_0 is given by (1). 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 *If $\rho_0 < 1$, the system of equations* (2) *and* (3) *has a unique solution. This solution can be found by fixed point iterations*

$$\rho^{(n+1)} = \rho_0 \frac{1-(\rho^{(n)})^{K+1}}{1-(\rho^{(n)})^K} \tag{4}$$

which converges for any initial value $\rho^{(0)} \in [0,\infty)$.

PROOF: We substitute the expression for the packet loss probability (2) into (3) to get $\rho = \frac{\rho_0(1-\rho^{K+1})}{1-\rho^K}$ Then we rewrite it as follows:

$$\rho_0 = \frac{\rho(1-\rho^K)}{1-\rho^{K+1}} = \frac{\rho+\rho^2+...+\rho^K}{1+\rho+\rho^2+...+\rho^K} = 1 - \frac{1}{1+\rho+\rho^2+...+\rho^K}$$

Now let us consider the left hand side of the above equation. Clearly this function of ρ is strictly increasing on the interval $[0,\infty)$. Moreover, it has a horizontal asymptote $y=1$. Hence, if $\rho_0 \geq 1$, there is no solution and if $\rho_0 < 1$ there is a unique solution.

Next we show that the fixed point iterations (4) converge to the solution of (3) and (2). Since the above considerations demonstrate that there is a unique fixed point (of course, we are now interested only in the case $\rho_0 < 1$), we only need to prove that the fixed point iterations converge. To prove this, it is enough to show that the derivative of the right hand side of (4) is less than one.

$$\frac{\mathrm{d}}{\mathrm{d}\rho}\left[\rho_0\frac{1-\rho^{K+1}}{1-\rho^K}\right] = \frac{\mathrm{d}}{\mathrm{d}\rho}\left[\rho_0\left(1+\frac{\rho^K}{1+\rho+...+\rho^{K-1}}\right)\right] =$$
$$\rho_0\frac{K\rho^{K-1}+(K-1)\rho^K+...+2\rho^{2K-3}+\rho^{2K-2}}{(1+\rho+...+\rho^{K-1})^2} =$$
$$\rho_0\frac{K\rho^{K-1}+...+2\rho^{2K-3}+\rho^{2K-2}}{1+2\rho+...+K\rho^{K-1}+...+2\rho^{2K-3}+\rho^{2K-2}} < \rho_0 < 1$$

This completes the proof.

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 transfered $\bar{y}$, 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 [12, 25, 23] here we assume that packets are lost independently with probability p. The fluid model [4] 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 transfered 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 λ of the exponential distribution can be determined from the following relation

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

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

$$L(\bar{y}, W_0, W_{SS}) = \sum_{k=0}^{\infty} p(k|n) L_k(\bar{y}, W_0, W_{SS}), \tag{5}$$

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 Appendix A). For $k = 1, 2, ...$ 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 [12].

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

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

PROOF: In the case of an ideal reliable protocol, every lost packet is retransmitted until it is received by the end host. The transfer of n packets is considered to be completed when all n packets reached successfully the destination. Clearly the last packet of each completed session is successfully transmitted. Thus the probability of having k retransmissions in a session of n packets is given by the negative binomial distribution $p^k(1-p)^n C_k^{n+k-1}$. (For detail discussion on negative binomial distribution see for example [21]).

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[2] 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[3] 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

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

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

[2] dup_{rt} is commonly set to three.
[3] dup_t is equal to 1 if the sender use LT and dup_{rt} otherwise.

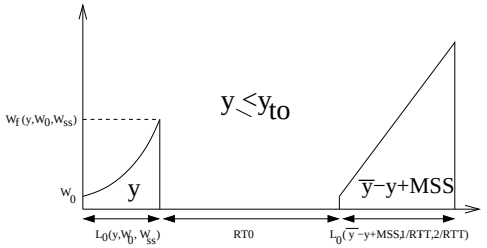

Figure 14: Sending rate evolution when a packet is lost before y_{to} bytes are sent

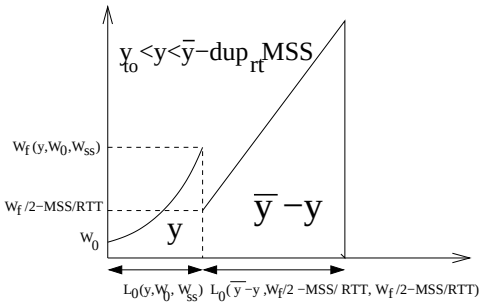

Figure 15: Sending rate evolution when a packet is lost when y ($y_{to} < y < \bar{y} - dup_{rt}MSS$) bytes are sent

in [24] 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 [15].

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 14, 15 and 16). In Figure 14 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 15, 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 16, 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).

After a timeout, the value of W_0 and W_{SS} are changed according to [2]. In particular, the value of W_0, regardless the value of the sending rate just before the timeout W_f, must be set to no more than 1 packet and the new value for the slow-start threshold is $W_{SS} \approx \max(W_f/2, 2MSS/RTT)$.

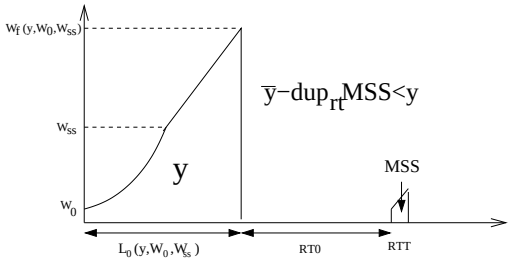

Figure 16: Sending rate evolution when a packet is lost when y ($\bar{y} - dup_{rt}MSS < y$) bytes are sent

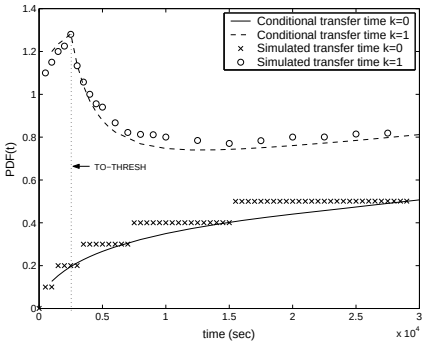

Figure 17: Expected conditional transfer time obtained by our fluid model approach

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 [14]. 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

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

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 (5).

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 3500bytes and 7000bytes by the formula (5). We consider two access links, with the shortest and the longest propagation

delays. The average RTT is computed using the standard $M/M/1/K$ model

$$RTT_i = 2(d + d_i) + \frac{MSS}{C}\left[\frac{1}{1-\rho} - K\frac{\rho^K}{1-\rho^K}\right]. \quad (6)$$

In addition we calculate the expected TCP latencies using the formulas derived in [12, 25] and we also obtain the expected TCP latencies from the measurements of the NS simulations[4] (see Figure 18).

All considered models perform similarly well. In particular, it is surprising how well the simple experimental formula of [25] performs. Even though our approach looks more complicated than the models of [12, 25], it has the next advantages:

- First of all, we have succeeded to avoid the introduction of several rough approximations and assumptions that were imposed in [12] and [25]. For example, the model of [12] uses the steady-state formula of [23] right after the first lost 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, especially on the links of high load, see the data for $\rho_0 = 0.975$ in Figure 18. The model of [25] 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 [12, 25].

- Our model has several input parameters that can easily be controlled. In the models of [12, 25] the slow-start threshold is always set to the maximum receiver window size and in the model of [25] 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 17 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) [3] and Limited Transmit (LT) [1] algorithm have become standards. The flexibility of the present model permits to analyze the IW and LT proposals [5].

[4] Two models are in fact proposed [25]. We use the empirical one which has been shown in [25] to be "a very close approximation" of the analytical model.

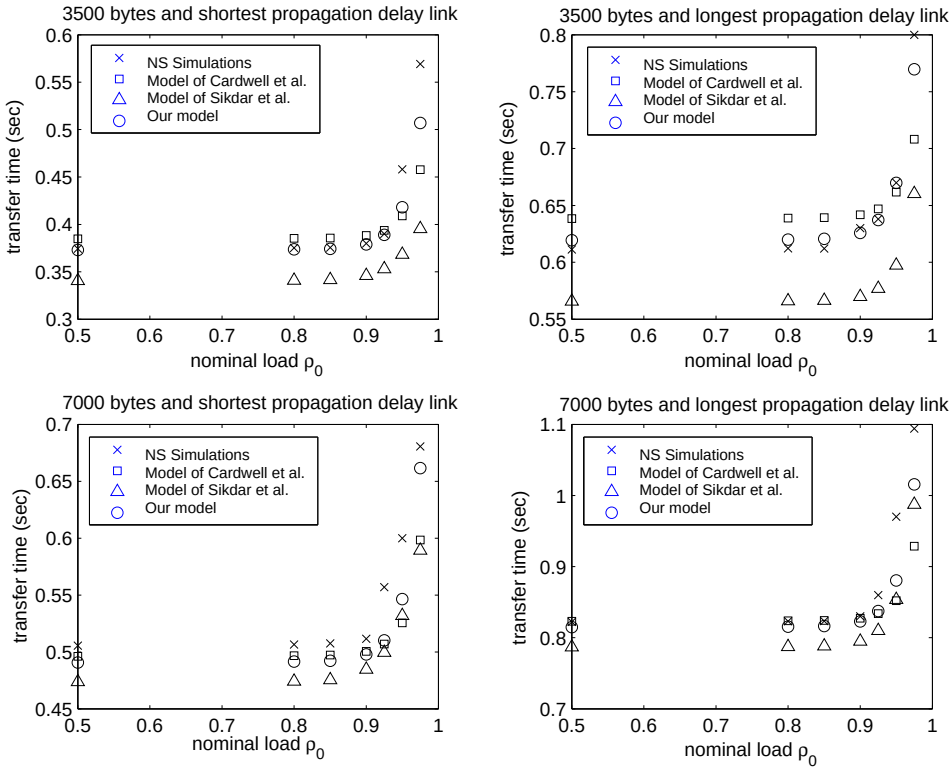

Figure 18: Transfer time for document sizes of 3500 and 7000 bytes over the links with the shortest and the longest propagation delays

- Taking into consideration small number of losses one can obtained 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, $i = 1, ..., N$. 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 (e.g., exponential with mean 10Kbytes), p is the packet loss probability that can be calculated as in Section 2 and the average Round Trip Time is given by the formula (6).

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 $Pr\{\text{k TCP sessions are on}\} = \frac{1}{k!}(\lambda^{(i)}\bar{L}^{(i)})^k \mathrm{e}^{-\lambda_i \bar{L}^{(i)}}$. 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

$$Pr\{\text{k TCP sessions are on}\} = \frac{1}{k!}(\lambda \bar{L}^{bn})^k \mathrm{e}^{-\lambda \bar{L}^{bn}}.$$

As in [19, 16], 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 19. Namely this is the interval of loads at which the latency stays approximately constant. For example in Figure 20 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.

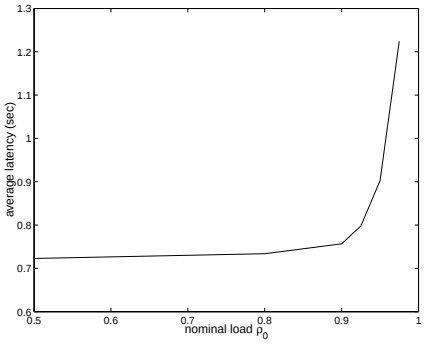

Figure 19: Average latency for different nominal loads

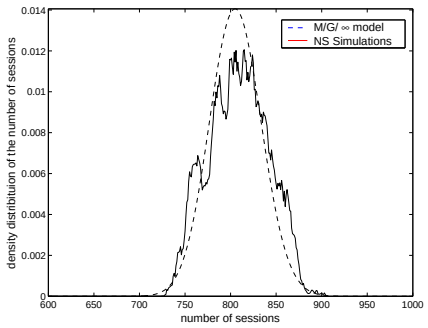

Figure 20: $M/G/\infty$ model for load 0.8

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.

All three levels of our model have been validated by NS simulations, and a part of our session level model has been compared to other existing models. Our session level model also yields the expected latency conditioned on the number of losses, and an interesting paradox has been discovered when the conditional latency is not monotone in the file size.

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 (see e.g. [8, 9]), 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.

A *Derivation of* $L_0(y, W_0, W_{SS})$

If a TCP session does not experience any loss, the TCP congestion window evolves according three different patterns, first in the *Slow-Start*(SS) phase it increases exponentially over the time, then in the *Congestion Avoidance*(CA) phase it increases linearly and finally when the maximum window M is achieved its value stays constant. In the next subsections we analyze these evolution phases. In particular we are interested in the calculation of the following quantities: the time $L_0(y, W_0, W_{SS})$ required to send a certain amount of bytes y in the non lossy environment, the final value $W_f(y, W_0, W_{SS})$ of the sending rate and the amount of data sent $y_f(W_f, W_0, W_{SS})$ as a function of the final sending rate W_f.

In calculations we do not take into account the dynamics of RTT and we only use its average value. This is justified in the case of a large number of connections sharing the same network resources, so that a single TCP session does not have significant influence on the queuing delay. We note that in sequel W_0, W_{SS}, M stand for sending rate rather than for the congestion window. The sending rate is equal to the congestion window divided by the average RTT.

A.1 The analysis of the Slow-Start phase

In the SS phase, the value of the congestion window at the next round is equal to the value of the congestion window at the current round multiplied by the constant $\gamma' = 1 + \frac{1}{b}$ [12], where b is the number of packets the receiver acknowledges by one ACK. In fluid model the evolution of the sending rate is given by, $W(t) = W_0 e^{\gamma t}$ where $\gamma = \log(\gamma')/RTT$.

Let a_1 be the time at which the sending rate reaches the Slow-Start sending rate threshold W_{SS}. $a_1 = \frac{1}{\gamma}\log(\frac{W_{SS}}{W_0})$ The amount of data sent in SS phase during time t is calculated by integrating the value of the instantaneous sending rate over the time, $A(t) = \int_0^t W_0 e^{\gamma s} ds = \frac{W_0}{\gamma}[e^{\gamma t} - 1]$ In particular for $t = a_1$ we denote $l_1 := A(a_1) = \frac{W_{SS} - W_0}{\gamma}$. The time t_{ss} required to send a certain amount of data y smaller than l_1 is given by $t_{ss}(y, W_0) = \frac{1}{\gamma}\log(\frac{y\gamma}{W_0} + 1)$ Then, given the amount of data y, the final value of the sending rate W_f is given by $W_f(y, W_0, W_{SS}) = y\gamma + W_0$, and on opposite knowing the final value of the congestion window W_f, the amount of data sent is $y_f(W_f, W_0, W_{SS}) = \frac{W_f - W_0}{\gamma}$.

A.2 The analysis of the Congestion Avoidance phase

In the CA phase the sending rate evolves according to $W(t) = W_{SS} + \beta(t - a_1)$, where $\beta = b/RTT^2$. Let a_2 be the time when the sending rate reaches the maximum M, namely $a_2 = \frac{(M - W_{SS})}{\beta} + a_1$ At time t, $a_1 < t < a_2$, the amount of data sent is given by $A(t) = l_1 + (t - a_1)\left[W_{SS} + \frac{(t-a_1)\beta}{2}\right]$ and in particular at time a_2 $l_2 := A(a_2) = l_1 + \frac{(M^2 - W_{SS}^2)}{2\beta}$. Given an amount of data greater y, $l_1 < y < l_2$, the time t_{ca} required to send this amount of data is given by the solution of the following quadratic equation, $y = l_1 + (t_{ca} - a_1)\left[W_{SS} + \frac{(t_{ca}-a_1)\beta}{2}\right]$. Namely we have $t_{ca}(y, W_0, W_{SS}) = \frac{-2}{\beta}W_{SS} + 2a_1 + \sqrt{\frac{4W_{SS}^2}{\beta^2} + \frac{8}{\beta}(y - l_1)}$

Then for an amount of data y, $l_1 < y < l_2$, the final value of the sending rate is given by,

$$W_f(y, W_0, W_{SS}) = (W_{SS} + \beta(t_{ca}(y, W_0, W_{SS})y - a_1))$$

and on opposite, the amount of bytes sent given the final value of the sending rate is given by

$$y_f(W_f, W_0, W_{SS}) = \left(l_1 + \frac{(W_f - W_T)}{\beta}(W_{SS} + \frac{W_f - W_{SS}}{2})\right).$$

A.3 The analysis of the Maximum sending rate phase

Finally we analyze the case when the maximum window is reached. The amount of data sent in time t, $t > a_2$, is given by $A(t) = l_2 + M(t - a_2)$ and respectively the time required to sent the data y is given by $t_{mw}(y, W_0, W_{SS}) = (y - l_2)\frac{1}{M} + a_2$.

Under maximum window regime, given $y > l_2$ we know that the final value of the sending rate is always M.

A.4 General expressions in the case of no losses

Collecting together all the above results, we have:

- The time required to send a certain amount of data y is

$$\begin{aligned} L_0(y, W_0, W_{SS}) &= t_{ss}(y, W_0, W_{SS})\ 1\{y \le l_1\} + t_{ca}(y, W_0, W_{SS})\ 1\{l_1 < y \le l_2\} + \\ &\quad t_{mw}(y, W_0, W_{SS})\ 1\{y > l_2\} + \frac{bRTT}{2} \end{aligned}$$

  We add $bRTT/2$ (being b the delayed parameter) to the final transfer time in order to adapt the fluid model to the packet level. In Figure 21 the area below the curves up to time t corresponds to the amount of bytes sent. For instance we observe that the area below our fluid model is greater than the area under the TCP packet level behavior. Hence, adding $bRTT/2$ to the final result obtained from our fluid model, we observe that the corresponding curve to this correction captures well the TCP dynamics at packet level.

- The final value of the sending rate given y bytes are sent.

$$W_f(y, W_0, W_{SS}) = (y\gamma + W_0)\ 1\{y \le l_1\} + (W_{SS} + \beta(t_{ca}(y, W_0, W_{SS})y - a_1))\ 1\{l_1 < y \le l_2\} + M\ 1\{y > l_2\}$$

- For the SS and CA phases the amount of bytes sent as a function of the final sending rate is

$$y_f(W_f, W_0, W_{SS}) = \frac{W_f - W_0}{\gamma} 1\{W_f \le W_{SS}\} + \left(l_1 + \frac{(W_f - W_T)}{\beta}(W_{SS} + \frac{W_f - W_{SS}}{2})\right) 1\{W_f > W_{SS}\}$$

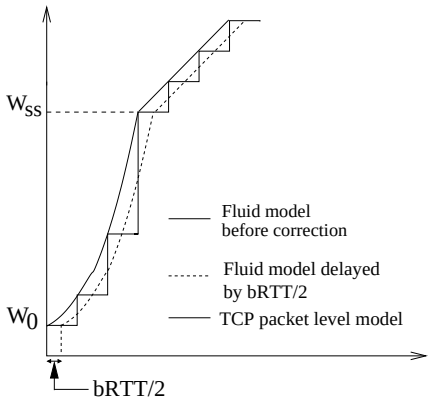

Figure 21: Correction of the fluid model

References

[1] M. Allman, H. Balakrishnan, S. Floyd, "RFC3042: Enhancing TCP's Loss Recovery Using Limited Transmit", January 2001.

[2] M. Allman, V. Paxon, W. Stevens, "RFC2581: TCP Congestion Control", April 1999.

[3] M. Allman, S. Floyd, C. Partridge, "RFC 3390: Increasing TCP's Initial Window", October 2002.

[4] E. Altman, K.E. Avrachenkov and C. Barakat, "A stochastic model of TCP/IP with stationary random losses", *ACM SIGCOMM 2000*, Stockholm, Sweden, also in *Computer Communication Review*, v.30, no.4, pp.231-242, 2000.

[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] U. Ayesta, K.E. Avrachenkov "On reducing the number of TimeOuts for short-lived TCP connections" Internet-Draft, October 2002, Work in progress. Available online at: http://www.ietf.org/internet-drafts/draft-ayesta-to-short-tcp-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] J-C. Bolot, "End-to-end packet delay and loss behavior in the Internet", *Proc. ACM Sigcomm '93, pp. 289-298*, San Francisco, CA, Sept. 1993.

[9] O. J. Boxma, "Sojourn times in cyclic queues – the influence of the slowest server", *Computer Performance and Reliability*, G. Iazeolla, P. J. Courtois and O. J. Boxma (Editors), Elsevier Science Publishers B.V. (North-Holland), pp. 13-24, 1988.

[10] T. Bu and D. Towsley, "Fixed point approximation for TCP behaviour in an AQM network", in Proceedings of SIGMETRICS'2001 conference.

[11] The web site of the Cooperative Association for Internet Data Analysis (CAIDA): http://www.caida.org/.

[12] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency", *IEEE INFOCOM*, pp.1742-1751, Tel Aviv, Israel, March 2000.

[13] C. Casetti, M. Meo, "A New Approach to Model the Stationary Behavior of TCP Connections", *IEEE INFOCOM*, Tel Aviv, Israel, March 2000.

[14] E. Cinlar, "Introduction to stochastic processes", Prentice-Hall, 1975.

[15] S. Dawkins, G. Montenegro, M. Kojo, V. Magret, N. Vaidya, "End-to-end Performance Implications of Links with Errors", August 2001.

[16] S. Ben Fredj, T. Bonald, A. Proutiere, G. Regnie, J. Roberts, "Statistical Bandwidth Sharing: A Study of Congestion at Flow Level", SIGCOMM 2001.

[17] 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.

[18] 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.

[19] A.A.Kherani, A. Kumar, "Stochastic Models for Throughput Analysis of Randomly Arriving Elastic Flows in the Internet", *IEEE INFOCOM 2002*, New York.

[20] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", *ACM Computer Communication Review*, Jul 1997.

[21] R. Nelson, "Probability, Stochastic Processes, and Queueing Theory", Springer-Verlag, 1995.

[22] The web site of the National Laboratory for Applied Network Research (NLANR): http://www.nlanr.net/.

[23] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation", *ACM SIGCOMM*, Sep 1998.

[24] V.Paxson, M. Allman, "RFC2988: Computing TCP's Retransmission Timer", November 2000.

[25] 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.

Contents

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399