Modeling Short-Lived TCP Connections with Open Multiclass Queuing Networks*

M. Garetto, R. Lo Cigno, M. Meo, E. Alessio, and M. Ajmone Marsan

Dipartimento di Elettronica – Politecnico di Torino,
Corso Duca degli Abruzzi, 24 – 10129 Torino, Italy,
{garetto,locigno,michela,ajmone}@polito.it

Abstract. In this paper we develop an open multiclass queuing network model to describe the behavior of short-lived TCP connections sharing a common IP network for the transfer of TCP segments. The queuing network model is paired with a simple model of the IP network, and the two models are solved through an iterative procedure. The combined model needs as inputs only the primitive network parameters, and produces estimates of the packet loss probability, the round trip time, the TCP connection throughput, and of the average TCP connection completion time (that is, of the average time necessary to transfer a file with given size over a TCP connection). The model presentation is centered on TCP-Tahoe, but the model of TCP-Reno is also available and results are presented. The analytical performance predictions are validated against detailed simulation experiments in a realistic networking scenario, proving that the proposed modeling approach is accurate.

1 Introduction

Modeling the TCP behavior is receiving great attention both from the academic and the industrial community. The reason for such enormous interest is essentially one: The behavior of TCP drives the performance of the Internet. Over 90% of the Internet connections use TCP, and over 95% of the bytes that travel over the Internet use TCP; hence, the availability of an accurate model of TCP is a necessity for the design and planning of Internet segments and corporate Intranets.

Models appeared so far in the literature can be grouped in two classes:

1. Models that assume that the RTT and the loss characteristics of the IP network are known, and try to derive from them the throughput (and the delay) of TCP connections; to this class belong works as [1,2,3];
2. Models that assume that only the primitive network parameters (topology, number of users, data rates, propagation delays, buffer sizes, etc.) are known, and try to derive from them the throughput and the delay of TCP connections, as well as the RTT and the loss characteristics of the IP network; to this second class belong the works in references [4,5,6,7,8] and many others.

* This work was supported by the Italian Ministry for University and Scientific Research through the PLANET-IP Project.

G. Carle and M. Zitterbart (Eds.): PfHSN 2002, LNCS 2334, pp. 100–116, 2002.

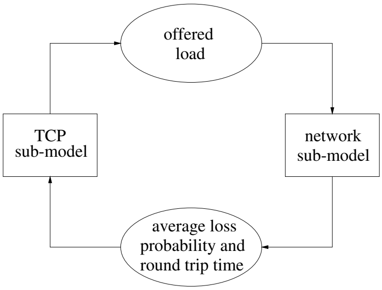

Fig. 1. High-level description of the model solution

Often, models in the second class incorporate a model similar to those of the first class, together with a simplified model of the network that carries the TCP segments and the two models are jointly solved through a fixed point iterative procedure. This situation is illustrated by Fig. 1, where the 'TCP sub-model,' describes the behavior of TCP connections, and the 'network sub-model,' focuses on the underlying IP network. The TCP sub-model computes the offered load to the system, given the average RTT and loss probability. The network sub-model estimates the RTT and the loss probability, given the offered load to the system.

With reference to our previous work, the contribution of this paper lies in the detailed presentation of the Open Multiclass Queueing Network method applied to TCP. The work presented in [7] assumed greedy connections using a closed queueing model. The work presented in [9] was concerned with a performance comparison between TCP-Tahoe and TCP-NewReno; results were obtained with OMQN models; however, the models themselves were not discussed. In [10], finally, we presented an overview of the modeling technique presented here, but a detailed application of the OMQN technique to a specific protocol, describing service times derivation and queues transition probabilities was not discussed in detail.

In this paper we use open multiclass queuing networks to develop a model that describes the behavior of an arbitrarily large number of short-lived TCP connections sharing a common infrastructure for the transfer of TCP segments. The model needs as inputs only primitive network parameters, and produces estimates of the packet loss probability, the RTT, the TCP connection throughput, and of the average TCP connection completion time. The TCP model presented here allows an unprecedented modeling accuracy for interacting short-lived TCP connections, yet maintaining a computationally simple solution. The level of insight in TCP dynamics offered by the model is also very high.

We use simulation experiments to validate our modeling approach, showing that the performance estimates that our models can generate are extremely accurate.

2 Queuing Network Models of TCP and Customer Classes

The TCP model in this paper is based on a description of TCP with an open multiclass queuing network (OMQN), in which all queues are of type $M/G/\infty$. Each queue represents a state of the TCP protocol, and each customer represents an active TCP connection. Classes are used to identify TCP connections according to the number of remaining segment transmissions before completion. Connections are opened when new customers enter the OMQN, then are used to transfer a file composed of N_P packets, and finally, after the file has been successfully delivered, the connection is closed as the customer leaves the OMQN.

Since all TCP connections are opened in the same state, only the queue corresponding to such state receives external arrivals. Let this queue be q^*.

Let $\lambda_{q,c}$ and $\lambda^e_{q,c}$ respectively denote the total and the external arrival rates at queue q in class c, and let $N_P^{\max}$ denote the maximum allowed file size in packets.

The external arrival rate of the OMQN is defined by the vector $[\lambda^e_{q^*,c}] = \lambda^e[\gamma_c]$ $1 \leq c \leq N_P^{\max}$, where λ^e is a traffic scaling factor, and $[\gamma_c]$ is a probability distribution vector describing the TCP connection file size.

Every time a new TCP segment is successfully transmitted, the class c of the customer is decreased by one; when the last packet is successfully transmitted (and the corresponding ACK is received), the customer leaves the OMQN, and the TCP connection closes.

The service times of the $M/G/\infty$ queues are independent from customer classes. All queues have an infinite number of servers, since there are no limitations to the number of TCP connections in any given state within the system.

From the specification of external arrival rates and service times, it is possible to derive the distribution of customers in the different queues and classes. From it we can derive the average number of customers of each class in every queue, $E[N_{q,c}]$, and the average TCP connection completion time.

The solution of the OMQN model is obtained solving the system of flow balance equations:

$$\lambda_{q,c} = \lambda^e_{q,c} + \sum_{i \in S} \sum_{j=c}^{N_P^{\max}} \lambda_{i,j} P(i,j;q,c) \qquad \forall (q,c) \tag{1}$$

where S is the set of queues in the OMQN, and $P(i,j;q,c)$ is the transition probability from queue i in class j to queue q in class c. Note that $\lambda^e_{q,c} = 0$ for any $q \neq q^*$.

The complete model solution is obtained by iterating the solutions of the network submodel and the TCP sub-model (as shown in Fig. 1) with a fixed point algorithm (FPA), until convergence is reached, according to a specified threshold. All the performance figures of interest depend on the average distribution of customers in the queues, which is obtained by solving the flow balance problem defined by equation (1). The complexity of this step is $O([M_q \cdot N_P^{\max}]^3)$. Moreover, exploiting the triangular structure of the system of linear equations, deriving from the fact that customers can only decrease their class (or keep it constant) on transitions, the complexity reduces to $O([M_q \cdot N_P^{\max}]^2)$. Since M_q is of the order of few hundreds, the CPU time required for each step of the fixed point algorithm is extremely small.

The number of iterations before convergence depends on the accuracy required of the FPA, but a relative accuracy of 10^{-6} is generally reached in a few tens of iterations.

3 The Model of TCP-Tahoe

Let W be the maximum TCP window size expressed in segments and C the maximum number of retransmissions before TCP closes the connection. Fig. 2 reports the OMQN model of short-lived TCP-Tahoe connections, in the case $W = 10$, which is the smallest maximum window size that allows a complete description of all the states of the protocol. New TCP connections open in the state described by queue FE_1 (thus, $q^* = FE_1$). The shaded queues describe the steady-state behavior of greedy TCP connections, while the other queues describe the transient behavior of the protocol. Non shaded queues are needed to describe the first slow start phase, which is different from the others, since *ssthresh*, the threshold separating the slow start phase from the congestion avoidance phase, is not set, and the congestion window can grow exponentially to W. When transferring short files, the initial transient has a major impact on performance. Customers leave the system whenever they reach class 0, regardless from the queue they are in; these transitions are not shown in Fig. 2 to avoid cluttering the graph.

The queues in Fig. 2 are arranged in a matrix pattern: all queues in the same row correspond to similar protocol states, and all queues in the same column correspond to equal window size. It is clear that the OMQN in Fig. 2 is highly structured, since, in spite of the large number of queues present in the model, all queues can be grouped in only 13 classes, described in some detail below, assuming that the reader is familiar with TCP-Tahoe (see [12] and [13]). Table 1 reports the service rates of the queues.

$FE_i\,(1 \leq i \leq W)$ **and** $E_i\,(1 \leq i \leq W/2)$ model the exponential window growth during slow start; the index i indicates the transmission window size. Queues FE_i are visited only during the first slow start phase after the connection is opened, while queues E_i model all other slow start phases; during the first slow start phase the growth is limited only by losses, since *ssthresh* is not yet assigned. As can be seen from Table 1, the average service time at these queues depends on the round-trip time through a factor σ which is an apportioning coefficient that takes into account the fact that during slow start the TCP transmission window grows geometrically with base 2 (see [7] for details); we use $\sigma = 2/3$.

$ET_i\,(1 \leq i \leq W)$ model the TCP transmitter state after a loss occurred during slow start: the transmission window has not yet been reduced, but the transmitter is blocked, because its window is full; the combination of window size and loss pattern forbids a fast retransmit (i.e., less than 3 duplicate ACKs are received). The TCP source is waiting for a timeout to expire. Queues $ET_i\,(W/2 \leq i \leq W)$ can be reached only from queues FE_i, i.e., during the first slow start. The service time is a function of the term T_0 which in the model is approximated with $\max(3\,\text{tic}, 4\,\overline{\text{RTT}})$, since the estimation of RTT variance is not available.

$FF_i\,(4 \leq i \leq W)$ **and** $EF_i\,(4 \leq i \leq W/2)$ model a situation similar to that of queues ET_i, where, instead of waiting for the timeout expiration, the TCP source is waiting for the duplicated ACKs that trigger the fast retransmit. Different queues have been used to distinguish the first loss event during a connection, i.e., fast retransmit events triggered from queues FE_i, from all others cases.

$L_i\,(2 \leq i \leq W)$ model the linear growth during congestion avoidance (notice that queue L_1 does not exist).

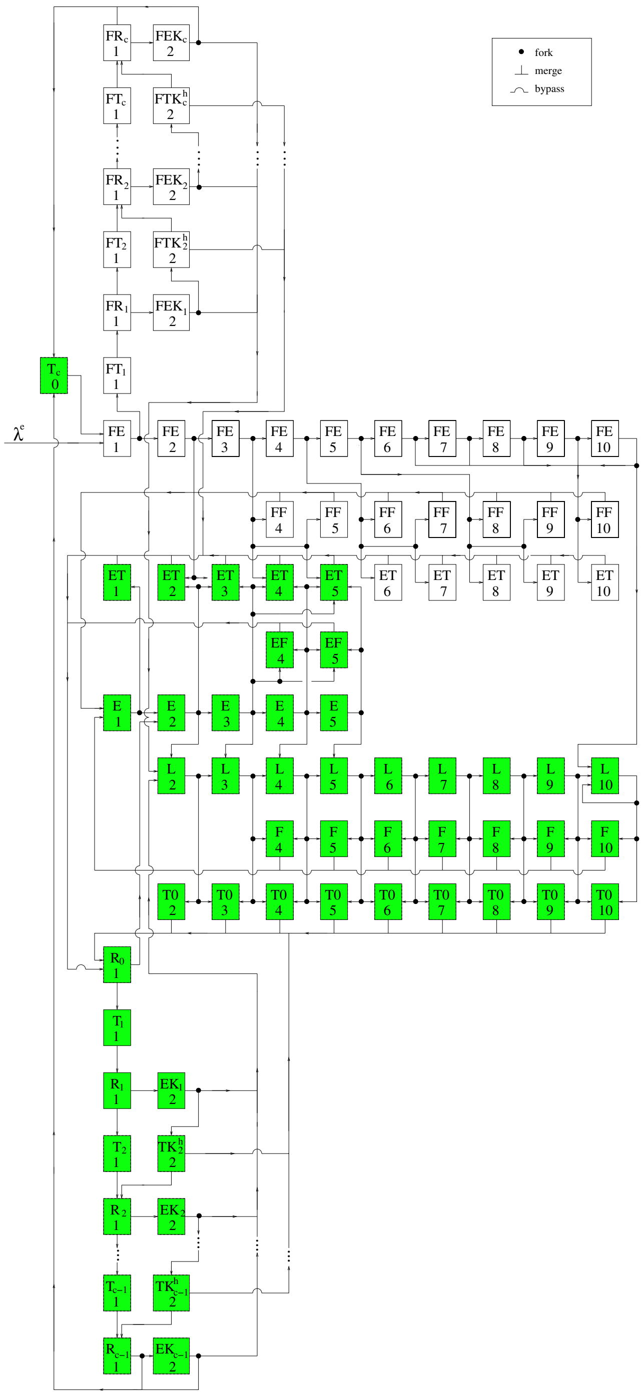

Fig. 2. The OMQN model of TCP-Tahoe

Table 1. Queues service times

Queue	Service time
E_1, FE_1	$\overline{\text{RTT}}$
E_2, FE_2	$\overline{\text{RTT}}$
$E_i, FE_i \quad i = 2^n \quad n \geq 2,$	$\sigma\,\overline{\text{RTT}}$
$E_i, FE_i \quad 2^{n-1}+1 \leq i \leq 2^n - 1$	$\frac{(1-\sigma)\overline{\text{RTT}}}{2^{n-1}-1}$
$ET_i \quad 1 \leq i \leq 3$	$T_0 - \overline{\text{RTT}}$ [1]
$ET_i \quad i \geq 4$	T_0 [1]
EF_i	T_0 [1]
FF_i, F_i	$\frac{\overline{\text{RTT}}}{2} + \frac{4\overline{\text{RTT}}}{i}$
L_i	$\overline{\text{RTT}}$
$T0_2$	$T_0 - \overline{\text{RTT}}$ [1]
$T0_i \quad 3 \leq i \leq W$	$T_0 - \overline{\text{RTT}}/2$ [1]
$T_i, FT_i \quad 1 \leq i \leq 6$	$2^i\,T_0$ [1]
$T_i, FT_i \quad 7 \leq i \leq C-1$	$64\,T_0$ [1]
R_i, FR_i, EK_i, FEK_i	$\overline{\text{RTT}}$
TK_i^1, FTK_i^1	$T_i - \overline{\text{RTT}}$ [1]
TK_i^2, FTK_i^2	$T_0 - \overline{\text{RTT}}$ [1]
T_C	180 s

[1] Remember that T_0 and T_i assume different values whether the first packet of a connection is lost or otherwise

$F_i\,(4 \leq i \leq W)$ model losses that trigger a fast retransmit during congestion avoidance.

$T0_i\,(2 \leq i \leq W)$ model the detection of losses by timeout during congestion avoidance.

$FT_i\,(1 \leq i \leq C)$ **and** $T_j\,(1 \leq j \leq C-1)$ model the time lapse before the expiration of the i-th or $(j+1)-$th timeout for the same segment, i.e., model the backoff timeouts. Queues FT_i model the backoff procedure in case the first segment is lost, when the RTT estimate is not yet available to the protocol and T_0 is set to a default value, typically 12 tics. This event, that may seem highly unlikely, has indeed a deep impact on the TCP connection duration.

Queue T_C models TCP connection that were closed for an excessive number of timeouts. Closed connections should leave the system; however, the model solution we use is correct only if all customers entering the OMQN in any class, leave in class 0 (a 'work conservation' principle), hence from queue T_C connections are supposed to re-open.

$FR_i\,(0 \leq i \leq C)$ **and** $R_j\,(0 \leq j \leq C-1)$ model the retransmission of a packet when the timeout expires.

$FEK_i\,(0 \leq i \leq C)$ **and** $EK_j\,(1 \leq j \leq C-1)$ model the first stage of the slow start phase (i.e., the transmission of the first 2 non-retransmitted packets) after a backoff timeout. During this phase the Karn algorithm ([14] and [12] Ch. 21.3) has a deep impact on the protocol performance under heavy load.

FTK_i^h $(0 \leq h \leq C)$ **and** TK_j^h $(1 \leq j \leq C-1)$**; h=1,2** model the wait for timeout expiration when losses occurred in queues FEK_i (EK_j); the superscript h in Fig. 2 discriminates two queues that are drawn together, but have different service times. The one with $h=1$ is entered when the first packet of the pair transmitted in queues FEK_i (EK_j) is lost, the other when the lost packet is the second. The transition from FTK_i^1 (TK_i^1) is to queue FR_i (R_i), while the transition from FTK_i^2 (TK_i^2) is to queue FR_0 (R_0).

The average service time represents the time a typical connection spends in a given protocol state. The service times derive directly from the protocol properties and are mainly function of the average round-trip time $\overline{\text{RTT}}$. Service times are independent from the customer class, as can be seen in Table 1.

The transition probabilities between queues derive from the TCP dynamics and characteristics, and they are related mainly to the packet loss probability. It is well known (see for instance [2,7]), that TCP dynamics introduce correlations in packet losses, and these have a deep impact on performance and consequently on any modeling process. Indeed, with interacting TCP connections, there exist both short-term and long-term correlations. Short-term correlations are between packets within the same transmission window. On a temporal axis, this correlation extends for roughly a round trip time. Long-term correlations are much more difficult to identify, because they are generated by the interaction among different connections. Strictly speaking, an OMQN model cannot take into account long-term correlations, but some heuristics allows the approximation of its effect, by introducing the following quantities:

P_L – average packet loss ratio, $P_S = 1 - P_L$;

P_{L_f} – loss ratio for the *first* packet in a window; this is also the probability of loosing a burst of packets, $P_{S_f} = 1 - P_{L_f}$;

P_{L_a} – packet loss ratio *after* the first packet in a window is lost; this is also the packet loss ratio within a burst and takes into account the short term correlation, $P_{S_a} = 1 - P_{L_a}$;

$P_{L_{fc}}$ – loss ratio for the *first* packet in a window under long term *correlation*, $P_{S_{fc}} = 1 - P_{L_{fc}}$;

$P_T(i)$ – probability that the exponential growth threshold *ssthresh* has value i; $P_C(i)$ cumulative distribution of $P_T(i)$;

$Z_{i,j}^T$ – $= \frac{P_T(i)}{1-P_C(j)}$; $Z_{i,j}^C = \frac{1-P_C(i)}{1-P_C(j)}$.

The relationship between P_L, P_{L_f}, P_{L_a} and $P_{L_{fc}}$, as well as their derivation and rationale are discussed in Section 4.2.

Table 2, reports the transition probabilities that model the succesful trnsmission of a whole window. The probability $P(q_i, c; q_j, c-k)$ reported in the table is the tramsition from source queue q_i (first column) to destination queue q_j (second column), decreasing the customer class of k units (third column) under the conditions indicated in the last column; $\wedge$ is the logical AND operator. Deterministic transitions are not reported. These probabilities are rather straightforward to compute. The only modeling problem arises in transitions from exponential to linear growth, where the value of *ssthresh* is crucial. Unfortunately, the value of *ssthresh* does not depend on the current state, but on the state

Table 2. Transition probabilities modeling successful transmission events; $P(\cdot)=P(q_i,c;q_j,c-k)$

q_i	q_j	k	$P(\cdot)$	Condition
FE_1	FE_2	1	P_{S_f}	
FE_i	FE_{i+1}	2	$P_{S_f}^2$	$2 \leq i \leq W-1$
FE_W	L_W	2	$P_{S_f}^2$	
FEK_i	FE_3	2	$P_{S_f}^2$	
E_1	E_2	1	$P_{S_{fc}}$	
E_i	E_{i+1}	2	$P_{S_{fc}}^2 Z_{i+1,i}^C$	$2 \leq i < W/2$
	L_i	2	$P_{S_{fc}}^2 Z_{i,i}^T$	
L_i	L_{i+1}	i+1	$P_{S_f}^{i+1}$	$2 \leq i \leq W-1$
L_W	L_W	W	$P_{S_f}^W$	
R_0	E_2	1	$P_{S_{fc}}$	
EK_i	L_2	2	$P_{S_{fc}}^2 Z_{2,2}^T$	
	E_3	2	$P_{S_{fc}}^2 Z_{3,2}^C$	

when the last packet was lost. Explicitly taking into account the value of *ssthresh*, leads to the explosion of the number of queues needed, that are approximately $W/2 \times M_q$. We resorted to a stochastic description of the distribution of *ssthresh*, that is taken into account by the terms $Z_{i,j}^T$ and $Z_{i,j}^C$ (see [7] for details).

Due to the lack of space, we do not report here all the transition probabilities of the model, that can be found in [15].

3.1 Packet Generation and Network Load

A TCP connection in a given queue q and class c generates a number of packets $\Pi_{q,c}$, which is the minimum between the number of packets allowed by the protocol state, Π_q, and the remaining packets to be transmitted during the TCP session, $\Pi_{q,c} = \min(\Pi_q, c)$. It follows that, from each queue q of the OMQN model, the load offered to the network by a TCP connection is

$$\Lambda_q = \sum_{c=1}^{N_P^{\max}} \lambda_{q,c} \Pi_{q,c} \tag{2}$$

where $\lambda_{q,c}$ is the arrival rate at queue q in class c computed from the OMQN. The total load offered to the network is then

$$\Lambda = \sum_{i \in S} \Lambda_q . \tag{3}$$

To solve (2) and (3) we need to define Π_q for each $q \in S$.

Some queues represent states of the protocol in which no packet is generated: queues T_i and FT_i, which model the backoff timeouts, and queue ET_1, which stands for the timeout expiration when the only packet in the window has been lost.

One packet per service is generated in queues R_i and FR_i, which model the retransmission of a packet when timeout expires, and in queues FE_1 and E_1, where the TCP window size equals 1.

The generation of two packets in queues FE_i and E_i, with $i > 1$, is due to the exponential window size increase in slow-start mode. Two packets per service are generated also in queues EK_i and FEK_i, standing for the transmission of the first 2 non-retransmitted packets after a backoff timeout. Similarly, two packets are generated in queues TK_i^3 and FTK_i^3.

More complex is the derivation of the number of packets that the protocol allows to generate for each service in queues EF_i, ET_i and FF_i, for $i \geq 3$, since it depends on which packet was lost in the previously visited queue. It results: $\Pi_{EF_i} = \Pi_{ET_i} = \Pi_{FF_i} = \Pi_i$

$$\Pi_i = \left[(i-2)\, \frac{P_{ll}}{1 - P_{ss}^2} + (i-1)\, \frac{P_{ss}\, P_{ll}}{1 - P_{ss}^2} \right] \tag{4}$$

where P_{ll} (P_{ss}) assumes the values P_{L_f} or $P_{L_{fc}}$ (P_{S_f} or $P_{S_{fc}}$), depending on the previously visited queue (see Section 4.2). Instead, the load offered by the connections in queues L_i is:

$$\Pi_{L_i} = \left[P_{L_f}\, i + (1 - P_{L_f})\,(i+1) \right] \quad i \leq W - 1 \tag{5}$$

$$\Pi_{L_W} = W \tag{6}$$

The computation of the load offered to the underlying IP network by connection at individual queues of type F_i and $T0_i$ is cumbersome, since for each queue it depends on the loss pattern. On the contrary, if we consider the aggregate load collectively offered by the set of queues

$$\Phi_{q_{ll}} = \bigcup_i [F_i \cup T0_i] \tag{7}$$

then its computation is much easier

$$\Lambda_{\Phi_{q_{ll}}} = \sum_{i=2}^{W} \left[\sum_{j=1}^{i} j\, P_{S_f}^j P_{L_f} \right] \lambda_{L_i} \tag{8}$$

Finally, for queues TK_i^2, FTK_i^2 and ET_2, we can write:

$$\Pi_{TK_i^2} = \Pi_{FTK_i^2} = \Pi_{ET_2} = \frac{P_{ss}\, P_{ll}}{P_{ll} + P_{ss}\, P_{ll}} \,. \tag{9}$$

4 Modeling the Underlying IP Network

Since the modeling focus in this paper is on TCP, the network model is kept as simple as possible. Nevertheless, it must allow the correct estimation of the key parameters that drive TCP performance. We assume that in its overall performance the network has a "drop tail" behavior.

A single server queue is the simplest possible model for the underlying IP network. The network is modeled as an $M^{[D]}/M/1/B$ queue, where packets arrive in batches whose size varies between 1 and W with distribution $[D]$. The variable size batches model the burstiness of the TCP transmission within the $\overline{\text{RTT}}$. The Markovian arrival of the batches finds its reason mainly in the Poisson assumption for the TCP connections arrival process, as well as in the fairly large number of connections present in the model. The Markovian assumption about service times is more difficult to justify; indeed, the transmission time of fixed length packets is constant, but the influence on results of using an $M^{[D]}/D/1/B$ queue was observed not to be worth the increased complexity.

4.1 Evaluation of $\overline{\text{RTT}}$ and P_L

The average packet loss ratio P_L, and the average queueing time t_B, are obtained directly from the solution of the $M^{[D]}/M/1/B$ queue; $\overline{\text{RTT}}$ is then estimated as the sum of t_B, the average two-way propagation delay of connections, and the packet transmission time in routers. The key point of the modeling process is the determination of the batch size distribution $[D]$. We compute the batch sizes starting from the number of segments N_R sent by a TCP transmitter during $\overline{\text{RTT}}$. This is clearly a gross approximation and a pessimistic assumption, since TCP segments are not transmitted together, and they do not arrive together at the router buffers. To balance this pessimistic approximation, the batch size is reduced by a factor $\mu < 1$. Actually, since the TCP burstiness is much higher during slow start than during congestion avoidance, we use two different factors: μ^e during the exponential window growth, and μ^l during the linear window growth. Unfortunately, it was not possible to find a direct method for the computation of these two factors, but a simple heuristic optimization led to the choice $\mu^e = 2/3$, $\mu^l = 1/3$, that yields satisfactory results in every tested scenario.

The computation of N_R is straightforward, starting from the solution of the OMQN of Fig.2. For every queue whose service time is $\overline{\text{RTT}}$, $N_R = \Pi_q$. In the other cases, queues must be grouped, based on the protocol dynamics. For instance, during exponential growth the grouping is based on the window doubling.

Besides the computation of batch sizes, it is also necessary to evaluate the generation rates of batches. For reasons that will be clear in Section 4.2, we separate the generation of bursts during the first slow start phase and during congestion avoidance, the relative rate being $\lambda_{bf}(i)$, from the generation of bursts during steady-state slow start phases, whose rate is $\lambda_{bc}(i)$. $\lambda_{bf}(i)$ and $\lambda_{bc}(i)$ are different for each session length, since, specially for short sessions, the number of packets to be transferred influences the probability that a connection visits a given queue (e.g., a connection that must transfers 10 packets will never transmit a packet with window size larger than 6).

4.2 Evaluation of Short- and Long-Term Correlations

Describing the TCP model in Section 3, we introduced the loss probabilities P_{L_f}, P_{L_a} and $P_{L_{fc}}$, that take into account short and long term correlations in packet losses. The short term correlation due to loss detection latency is well known, and it is easily accounted for in our model, since it extends only over a single window.

Long term correlation is due to a completely different phenomenon, rooted in the elastic nature of TCP traffic, that extends congestion over time.

In order to approximate this phenomenon, we assume that long term correlations extend in the slow start phase that follows a loss event: During this phase the probability of experiencing another loss event is higher, and it is represented by $P_{L_{fc}}$. Newly opened connections and connections in congestion avoidance phases experience instead a lower probability (P_{L_f}) of incurring in a loss event.

Assume that the loss correlation is positive, and is represented by multiplicative coefficients: $K_s > 1$ for the short term and $K_l > 1$ for the long term. The following equations allow the computation of P_{L_f}, $P_{L_a}$and $P_{L_{fc}}$starting from the average loss probability P_L

$$\begin{aligned}
P_{L_a} &= K_s P_{L_f} \;;\; P_{L_{fc}} = K_l P_{L_f} \\
\Lambda P_L &= \Lambda P_{L_a} + (1-K_s)\sum_{i=1}^{W} \lambda_{bf}(i)[1-(1-P_{L_f})^i] \\
&\quad +(1-\frac{K_s}{K_l})\sum_{i=1}^{W} \lambda_{bc}(i)[1-(1-P_{L_{fc}})^i]\,.
\end{aligned} \tag{10}$$

Unfortunately, the coefficients K_s and K_l depend on the network congestion, and are far from easy to compute, hence we need some simplifying assumptions. First of all, let's assume for simplicity $P_{L_{fc}} = P_L$, i.e., the probability of losing a burst of packets during a steady-state slow start phase is equal to the average packet loss ratio. If short term correlations are stronger than long term ones, as it is reasonable, we also have $P_{L_f} < P_{L_{fc}} < P_{L_a}$, so that our assumption that $P_{L_{fc}} = P_L$ is also equivalent to assuming that the average loss ratio is equal to the median value of the approximating discrete distribution.

The works in [2] and [7], hints to the possibility that the loss probability within the window of the first lost packet is almost constant. The value that seems to best fit a general scenario is $P_{L_a} = 0.2$. Once P_{L_a} and $P_{L_{fc}}$ are determined, P_{L_f} is computed from Eqs. (10).

5 Computation of the Session Duration

In order to compute the average session duration, let's assume that all customers enter the OMQN with the same class N_P. The average time Θ spent in the OMQN is derived from Little's theorem $\Theta = \frac{N}{\lambda^e}$, where N is the total average number of customers in the OMQN, and λ^e is the external arrival rate of customers.

When the initial class of the arriving customers is not constant, Little's result computes the average over all connections, regardless of the initial class. Class N_P customers comprise connections entering the OMQN with class N_P as well as all those connections whose initial class is larger than N_P, but still have N_P packets to transmit. Yet, the traffic mix influences the overall network performance, hence we devised a solution based on a two-step solution of the OMQN model.

Step 1 – The model is solved with the chosen external arrival distribution $\lambda^e \;\; [\gamma_c]$. During this step, parameters such as $\overline{\text{RTT}}$, P_L, K_s and K_l are computed.

Step 2 – Using the parameters computed in step 1, the OMQN model is solved again for each external arrival class $c = 1, \dots N_P^{\max}$. During this step the *ssthresh* distribution is computed again, since the one obtained in step 1 is not representative of each input class (e.g., a connection with 10 packets to transmit cannot have *ssthresh* $= 20$), but only of their average. For the same reason $\lambda_{bf}(c)$, $\lambda_{bc}(c)$, P_{L_f} and P_{L_a} are also recomputed. Finally, Little's result is used to compute each input class latency.

6 Validation and Results

In order to validate our analytical model of TCP, we compare the performance predictions obtained from the OMQN model solution against point estimates and 95% confidence intervals obtained from very detailed simulation experiments. The tool used for simulation experiments is *ns version 2* [11]; confidence intervals were obtained with the "batch means" technique, using 30 batches. Simulations last for 1,000–2,000 s when the bottleneck is a 45 Mbit/s link, and are four times longer when the bottleneck is a 10 Mbit/s link.

6.1 Network Topology and Traffic Load

We chose a networking environment which closely resembles the actual path followed by Internet connections from our University LAN to Web sites in Europe and the USA, where two clearly distinct traffic patterns can be identified.

The topology of the network we consider is shown in Fig. 3; at the far left we can see a set of terminals connected to the internal LAN of Politecnico di Torino. These terminals are the clients of the TCP connections we are interested in (white circles in the figure represent TCP clients; grey circles represent TCP servers). The distance of these clients from the Politecnico router is assumed to be uniformly distributed between 1 and 10 km. The LAN of Politecnico is connected to the Italian research network, named GARR-B/TEN-155, through a 10 Mb/s link whose length is roughly 50 km (this link will be called POLI-TO). Internally, the GARR-B/TEN-155 network comprises a number of routers and 155 Mb/s links. One of those connects the router in Torino

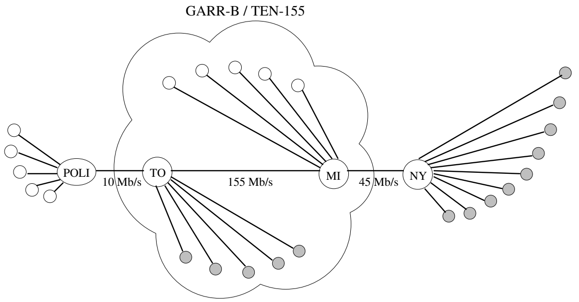

Fig. 3. Abstract view of the Internet from the Politecnico di Torino LAN; servers are shaded, clients are white

with the router in Milano; its length is set to 100 km. Through the GARR-B/TEN-155 network, clients at Politecnico can access a number of servers, whose distance from Politecnico di Torino is assumed to be uniformly distributed between 100 and 9,900 km. From Milano, a 45 Mb/s undersea channel whose length is about 5,000 km reaches New York, and connects GARR-B to North-American Internet backbones (this link will be called MI-NY). Many other clients use the router in Milano to reach servers in the US. The distance of those clients from Milano is assumed to be uniformly distributed between 200 and 2,800 km. The distance of servers in the US from the router in NY is assumed to be uniformly distributed between 200 and 3,800 km.

At present, the GARR-B/TEN-155 network is over-dimensioned, thus it is never congested. Both the POLI-TO and the MI-NY link, instead, are bottleneck channels that often incur heavy and persistent congestion. From the abstract network represented in Fig. 3 we carve out the following two different traffic patterns that give rise to the two scenarios we name the US Brow and the Loc Acc.

US Brow: This scenario becomes relevant when the traffic pattern makes the MI-NY link the bottleneck of the system. When this is the case, we have connections with lengths between 5,400 and 11,600 km that compete for the 45 Mbit/s link.

Loc Acc: This scenario corresponds to moments when the access link of our University is the bottleneck of the system. Connections compete for the 10 Mbit/s on the POLI-TO link, and their lengths are distributed between 151 and 9,960 km.

The packet size is 1,024 bytes; the maximum window size is 64 packets. We consider the cases of buffer sizes equal to either 128 or 64 packets, and TCP tic equal to 500 ms. The amount of data to be transferred by each connection (i.e., the file size) is expressed in number of segments: 50% of the connections are 10 segments long 40% are 20 and 10% are 100.

6.2 Numerical Results

Fig. 4 reports the average packet loss probability computed with the model, as well as point estimates and confidence intervals obtained via simulation in both the US Brow and Loc Acc scenarios. Results are plotted versus the normalized external load, which is the load the network would have if no packets were lost, so that no retransmissions are necessary. The upper curve refers to the case of 64 packet buffers, while the lower one refers to the case of 128 packet buffers. Markers correspond to simulations and report the confidence intervals.

As predicted by the model, the different conditions of the US Brow and Loc Acc scenarios have a minor impact on the average packet loss probability, that is predicted very accurately by the model over a large range of network loads.

Fig. 5 reports the average completion time for 10 segments files (left plot) and 100 segments files (right plot). The file size has a major impact on results (notice the different y-scales of the plots), as expected. The TCP performance is still dominated by the buffer size (that drives the loss ratio), while the scenario has a minor impact.

The presence of instability asymptotes around load 0.97 is evident in both figures. The vertical asymptotes correspond to points where the actual load of the network (considering retransmissions) approaches 1.

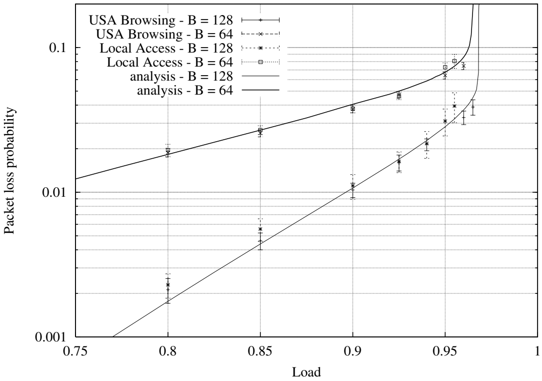

Fig. 4. Packet loss probability as a function of the external load

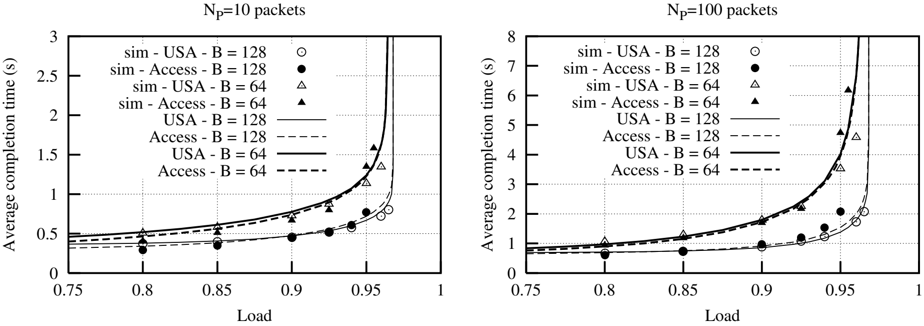

Fig. 5. Average session completion time as a function of the external load; 10 packets sessions in the left plot and 100 packets sessions in the right plot

6.3 Extension to Other TCP Versions

Up to this point we have only described or modeling approach in the case of TCP-Tahoe, but our method can be extended to other TCP versions as well. So far we have fully modeled TCP-NewReno, whose model is also briefly discussed in [8], though that paper focuses more on modeling a multibottlenek IP network.

The main difference between Tahoe and NewReno lies in the presence of the fast recovery procedure [13], which avoids the slow start phase if losses are detected via duplicated ACKs, entering congestion avoidance after halving the congestion window. The number of queues and queue types needed for the description of TCP-NewReno, is approximately the same as for TCP-Tahoe, though some queues change their meaning, for instance to model the fast recovery procedure where the actual transmission window keeps growing to allow the transmission of new packets while the lost ones are recovered.

Lack of space forbids a detailed description of this model, giving all the queues service times and transition probabilities. Instead, we present numerical results for the same cases we considered for TCP-Tahoe, again with validation results obtained with *ns-2* .

Fig. 6 reports the average packet loss probability in the same conditions as those of Fig. 4. The loss probability is similar to TCP-Tahoe, but the instability asymptotes are now closer to 1 (namely, a little beyond load 0.98).

For what concerns the session transfer time, we present the results in a different form, so as to give additional insight. Fig. 7 reports the results for buffer 128 only. the case with buffer 64 yields similar results, not reported for lack of space. The left plot refers to the US BROW scenario, while the right one refers to the LOC ACC scenario. Both plots report three curves, one for each type of connection mix in the network. For light and medium loads, the session latency is dominated by the transmission delay, since loss events are rare; hence longer sessions have longer transfer times and the increase is roughly linear. The interesting fact is that the asymptote is the same for all connection lengths, meaning that the protocol is reasonably fair towards connections of different lengths.

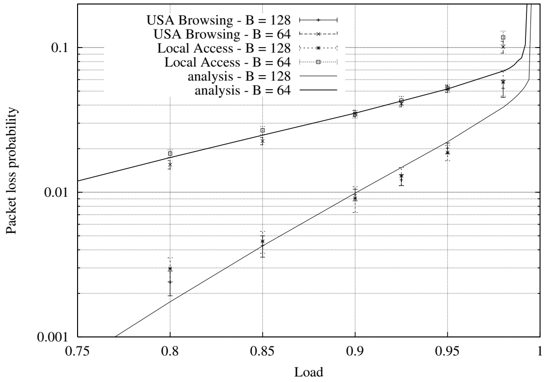

Fig. 6. Packet loss probability as a function of the external load for NewReno connections

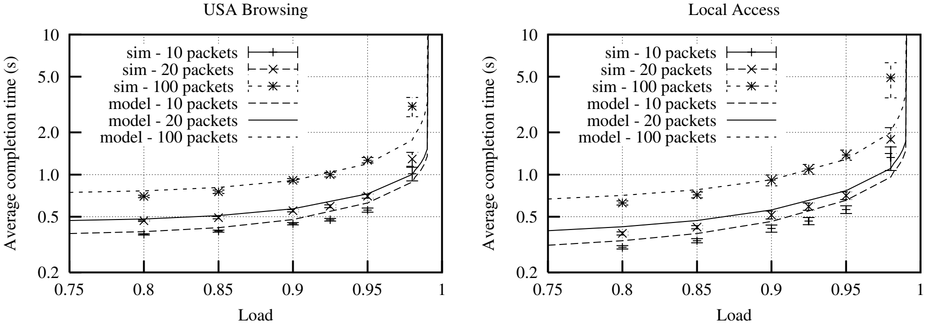

Fig. 7. Average session completion time as a function of the external load for the US BROW scenario (left plot) and for the LOC ACC scenario in the case of 128 packets buffer

The same is true also for TCP-Tahoe, but it is less evident from the presentation format of Fig. 5, which, on the other hand, better highlights the role of the buffer and, by contrast, the small role of the bottleneck capacity on the transfer delay.

7 Conclusions

In this paper we have developed an OMQN model to very accurately describe the behavior of short-lived TCP connections that exploit a common IP network for the transfer of finite-size files.

Starting from the primitive network parameters, the queuing network model is solved together with a simple model of the IP network, through an iterative procedure, and the average time for the completion of the file transfer is estimated, along with the TCP connection throughput and goodput.

The presented modeling approach was applied to TCP-Tahoe, and TCP-Reno (which is also shortly discussed in [8]), and it is currently being extended also to TCP-SACK.

The analytical performance predictions generated by the model for realistic networking scenarios have been validated against detailed simulation experiments, proving that the proposed modeling approach is extremely accurate and flexible, besides providing an excellent insight into TCP dynamics and behavior.

Results validating the model discussed in the paper were obtained for relatively slow networks (10–45 Mbit/s bottlenecks) due to the impossibility of obtaining reliable simulation results with higher speeds. It must be noted, however, that the model complexity is *independent* from the network speed. Hence OMQN models are a good means to ferecast the bahavior of protocols like TCP in very high speed netorks.

References

1. V. Paxson, "Empirically-Derived Analytic Models of Wide-Area TCP Connections," *IEEE/ACM Transactions on Networking*, 2(4):316–336, August 1994.
2. J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation," *Proc. ACM SIGCOMM'98*, Sept., 1998.
3. N. Cardwell, S. Savage, T. Anderson, "Modeling TCP Latency," *Infocom 2000*, Tel Aviv, Israel, March 2000.
4. V. Mishra, W. B. Gong, D. Towsley, "Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with and application to RED", *in Proc. SIGCOMM'2000*, Aug. 28–Sept. 1 2000, Stockholm, Sweden.
5. C. V. Hollot, V. Mishra, W. B. Gong, D. Towsley, "A Control Theoretic Analysis of RED," *in Proc. IEEE Infocom 2001*, Anchorage, Alaska, USA, April 22–26, 2001.
6. A. Kumar, "Comparative Performance Analysis of Versions of TCP in a Local Network with a Lossy Link," *IEEE/ACM Transactions on Networking*, 6(4):485–498, August 1998.
7. M. Garetto, R. Lo Cigno, M. Meo, M. Ajmone Marsan, "A Detailed and Accurate Closed Queueing Network Model of Many Interacting TCP Flows," *in Proc. IEEE Infocom 2001*, Anchorage, Alaska, USA, April 22–26, 2001.
8. M. Garetto, R. Lo Cigno, M. Meo, M. Ajmone Marsan, "Queuing Network Models for the Performance Analysis of Multibottleneck IP Networks Loaded by TCP Short Lived Connections," Politecnico di Torino Tech. Rep. DE/RLC/2001-5, Politecnico di Torino, June 2001. Available at http://www.tlc-networks.polito.it/locigno/papers/de-rlc-01-5.ps

9. E. Alessio, M. Garetto, R. Lo Cigno, M. Meo, M. Ajmone Marsan, “Analytical Estimation of the Completion Time of Mixed NewReno and Tahoe TCP Traffic over Single and Multiple Bottleneck Networks,” *Proc. IEEE Globecom 2001*, San Antonio, Tx, USA, Nov. 25–29, 2001.
10. M. Garetto, R. Lo Cigno, M. Meo, M. Ajmone Marsan, “On the Use of Queueing Network Models to Predict the Performance of TCP Connections,” *Proc. 2001 Tyrrhenian International Workshop on Digital Communications*, Taormina (CT), Italy, Sept. 17–20, 2001.
11. ns-2, network simulator (ver.2). LBL, http://www-mash.cs.berkeley.edu/ns.
12. W. R. Stevens. *TCP/IP Illustrated, vol. 1.* Addison Wesley, Reading, MA, USA, 1994.
13. W. Stevens, “TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms,” RFC 2001, IETF, Jan. 1997
14. P. Karn, C. Partridge, “Improving Round-Trip Time Estimates in Reliable Transport Protocols,” *Computer Communication Review*, Vol. 17, No. 5, pp. 2–7, Aug. 1987.
15. M. Garetto, R. Lo Cigno, M. Meo, M. Ajmone Marsan, “Modeling Short-Lived TCP Connections with Open Multiclass Queuing Networks – Extended Version,” Politecnico di Torino ech. Rep. DE/RLC/2001-4, Politecnico di Torino, June 2001. Available at http://www.tlc-networks.polito.it/locigno/papers/de-rlc-01-4.ps