A Detailed and Accurate Closed Queueing Network Model of Many Interacting TCP Flows

M. Garetto, R. Lo Cigno, M. Meo, M. Ajmone Marsan

Abstract—**This paper presents a new analytical model for the estimation of the performance of TCP connections. The model is based on the description of the behavior of TCP-Tahoe in terms of a closed queueing network, whose solution can be obtained with very low cost, even when the number of TCP connections that interact over the underlying IP network is huge. The protocol model can be very accurate, deriving directly from the finite state machine description of the protocol. The assessment of the accuracy of the analytical model is based on comparisons against detailed simulation experiments developed with the *ns-2* package. Numerical results indicate that the proposed closed queueing network model provides extremely accurate performance estimates, not only for average values, but even for distributions, in the case of the classical single-bottleneck configuration, as well as in more complex networking setups.**

Keywords—**TCP, analytical model, queueing network, model decomposition, fixed point analysis.**

I. Introduction & Paper Contribution

TCP, the Internet transport layer protocol, is by far the most widely used algorithm for the end-to-end control of the information transfer in packet networks. This widespread diffusion is a result of the capability of TCP to adapt to networks with very high bandwidth-delay products and with a huge number of connections per link. Nevertheless, the extreme conditions in which TCP is likely to be used in a few years (terabit per second routers and channels, and millions of connections sharing one physical link) require a careful investigation, at least to support the design and planning of large network segments with high traffic loads.

While no model of TCP, be it analytical or simulative, is probably capable of coping with the full complexity of such extreme conditions, analytical models can help understand the characteristics of the TCP behavior in those environments.

The pace of research in TCP modeling has been almost frenetic in recent years, as indicated by the large number of high-quality papers that appeared in the literature (see Section II for a discussion of some of the main contributions). Why then another model of TCP? The answer is that, in spite of the large number of available analytical models, some fundamental aspects of the behavior of TCP connections still require investigation. Examples are: i) the interaction of TCP with multiple bottleneck links, ii) the sensitivity of TCP connections to correlated losses, iii) the transfer delay of short files, iv) the TCP behavior when losses occur and the round trip time (RTT) estimation is unreliable (see the Karn algorithm [1]), and v) the effect of the interaction of thousands of connections.

Two are the reasons why available models do not allow the investigation of those aspects:

1. important details either of the protocol operations or of the interaction among TCP connections are neglected in the model development, mainly to obtain an acceptable model complexity;
2. the description of the allocation of the resources in the underlying IP network is either too simple, or embedded in the protocol model.

This paper adopts the approach introduced in [2] to develop an analytical model of TCP, whose main contributions are the following.

- The protocol description in the model is quite detailed, but still the model solution remains very simple.
- The performance estimates produced by the model are extremely accurate: even the *window size distribution* at TCP transmitters is accurately estimated.
- The model solution complexity is *independent* of the number of interacting TCP connections.
- The description of the allocation of resources in the underlying IP network is decoupled by the protocol description, so that complex scenarios, such as multiple bottlenecks, can be modeled with the desired accuracy.

II. Background & Related Work

As observed in the introduction, the literature on TCP modeling is vast, so that it is impossible to provide here a comprehensive overview of previous contributions; rather, we only refer to those papers that are most related to our work.

In [2], the authors introduced the principles for the development of closed queueing network models of window protocols, explaining in detail the solution technique, the derivation of performance metrics and the limitations of the method; however, no model of a *real* protocol was developed. The approach we adopt in this paper is a direct application of that presented in [2].

The analysis of TCP behavior proposed in [3] is based on measurements. The work in [4], presents a Markov reward model of several TCP versions on lossy links. The proposed approach consists in modeling the TCP behavior in terms of transmission cycles defined between two loss events; several performance metrics can be derived as rewards during these cycles. The loss process is assumed to be independent from the load offered by the TCP connections and implicitly forbids the study of interacting TCP connections.

The modeling approach adopted in [5] can cope with correlated losses, and is actually based on the idea that packets are lost in bursts that cover a whole congestion window fraction; the loss rate and round trip time are needed as inputs in order to allow the model to compute the TCP throughput and average window size. Results obtained with this model are compared against actual measurements.

A somewhat similar modeling approach can be found in [6], where the performances of TCP-Reno and TCP-Vegas are com-

This work was supported by the Italian Ministry for University and Scientific Research through the MQOS Project.

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

pared, modeling the two versions through a fluid-based, differential equation approach.

In [7] the authors propose a technique based on differential stochastic equations modeling together the macroscopic behavior of TCP, and the network itself. The approach is able to cope with multiple connections and multiple bottlenecks, but does not model the slow start phase and possible correlated losses, assuming that buffers are actively managed with a RED algorithm. This approach defines a closed-loop system that can be studied with powerful control-theoretic approaches.

Another line of research is represented by [8], [9]. The evolution of TCP is represented here in terms of Max-Plus algebra.

The approach followed in [10], [11] is based on Markovian modeling. The work in [10] assumes an underlying ATM network, and is limited by the solution complexity when the number of TCP connections grows. An innovation introduced in [11], that does not assume an underlying ATM network, is the presence of non-greedy connections.

III. Model Description and Solution

Fig. 1 presents a simplified high-level description of the proposed model of the system. The model is split in two parts, the 'TCP sub-model,' which describes, by means of a closed queueing network, the behavior of TCP connections; and the 'network sub-model,' which focuses on the underlying IP network. The two parts interact by exchanging (from the TCP sub-model to the network sub-model) the traffic load Λ collectively offered to the network by TCP connections, and (from the network sub-model to the TCP sub-model) the average packet loss probability P_L and average round trip time ($\overline{\text{RTT}}$). The global model solution is obtained with a fixed point procedure, based on the iterative solution of the two sub-models until equilibrium is reached. The fixed point procedure is stopped when the relative error of the estimates of P_L, $\overline{\text{RTT}}$ and Λ becomes smaller than a predefined threshold ϵ.

The fundamental aim of this model is capturing the closed loop interaction of TCP connections with a congested network, that reacts by dropping packets. The key assumption is that the global process (i.e., the process describing the network and all TCP connections) is a stable process that reaches a steady-state working point, though the individual processes describing each TCP connection do not have a stable working point but oscillate in time, as TCP congestion windows actually do. This key assumption, that proves to be successful, allows the compact description of the system state with the probability vector that any given connection is in any given protocol state, disregarding the exact state of each connection. By doing so, the model complexity remains limited; on the contrary, accounting for the individual connection states would inevitably lead to the explosion of the model complexity.

A. The TCP Sub-Model

We model the TCP-Tahoe protocol, as described in [12], assuming that fixed-size TCP segments are transmitted over the underlying IP network. TCP-Tahoe is coherent with the BSD implementation, as well as with the implementation found in the *ns-2* simulator [13] that we use for validation purposes.

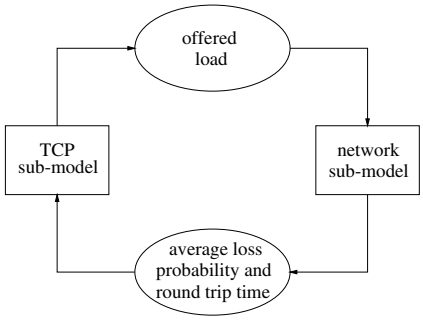

Fig. 1. High-level description of the model.

We model the behavior of N concurrent TCP connections using a closed network of $M/M/\infty$ queues, where each queue describes a *state* of the TCP protocol; the number of customers in a generic queue Q, N_Q, is the number of TCP connections that are in that specific state.

The chosen modeling approach is extremely flexible, and allows the adoption of an almost arbitrary level of abstraction. We chose to model the protocol in full detail, neglecting only marginal aspects of the protocol operations, that are expected not to have a significant impact on performance. The queueing network in Fig. 2 shows the TCP dynamics when the maximum window size is $W = 10$ segments. Each queue is characterized by the congestion window (*cwnd*) size expressed in number of segments. In addition, the queue describes whether the transmitter is in slow start, congestion avoidance, waiting for a fast retransmit or a timeout. Note that the queues in Fig. 2 are arranged in a matrix pattern, since all queues in the same row correspond to similar protocol states, and all queues in the same column correspond to equal window size. Queues below R_0 model backoff timeouts, retransmissions, and the Karn algorithm.

For the specification of the queueing network model, it is necessary to define for each queue Q:
1. the average service time, τ_Q, which models the average time spent by TCP connections in the state described by queue Q;
2. the transition probabilities $P(Q_i, Q_j)$, which are the probabilities that TCP connections enter the state described by queue Q_j after leaving the state described by queue Q_i.

The queueing network comprises 11 different types of queues, described in some detail below. In Table I, queue types are listed together with their average service time, which, as already mentioned, represents the time a connection spends in a given protocol state. Appendix A reports the derivation of the least intuitive service times; the others derive directly from the protocol properties and are mainly function of the average round-trip time $\overline{\text{RTT}}$.

Queues $E_i \quad (1 \leq i \leq W/2)$**,** model the exponential window growth during slow start; the index i indicates the congestion window size. As can be seen from Table I, the average service time depends on the round-trip time by means of a factor σ which is an apportioning coefficient that takes into account the fact that during slow start the TCP congestion window grows geometrically with base 2 (see Appendix A for details).

Queues $ET_i \quad (1 \leq i \leq W/2)$**,** model the TCP transmitter state after a loss occurred during slow start: the congestion window has not yet been reduced, but the transmitter is blocked

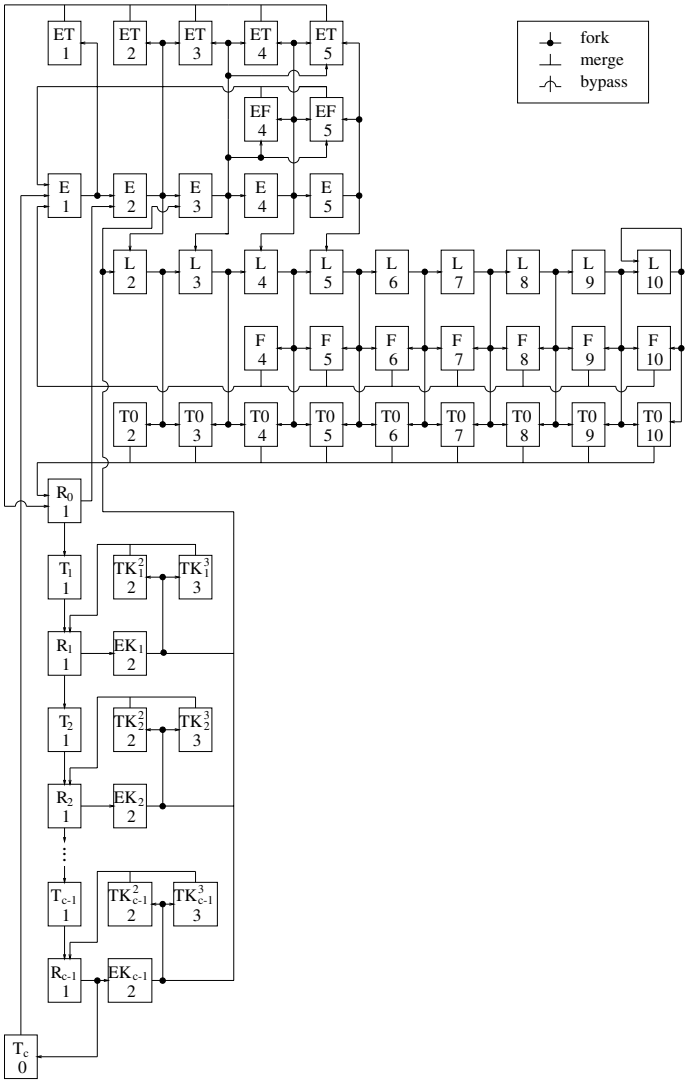

Fig. 2. Queueing network model of TCP-tahoe.

because its window is full; the combination of window size and loss pattern forbids a fast retransmit (i.e., less than 3 duplicated ACKs are received), so that the TCP source is waiting for a timeout to expire. The service time is function of the term T_0 which equals $\max(3\,\text{tic}, 4\,\overline{\text{RTT}})$[1] (again, see Appendix A for details).

Queues $EF_i \quad (4 \leq i \leq W/2)$**,** 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.

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

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

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

Queues $T_i \quad (1 \leq i \leq C)$**,** model the time lapse before the expiration of the $(i+1)-$th timeout for the same segment, i.e., model the backoff timeouts. C is the maximum number of consecutive retransmissions allowed before closing the connection. Queue T_C, actually models TCP connection that were closed for an excessive number of timeouts. In TCP-Tahoe this happens after 16 retransmissions, i.e., $C = 16$. Closed connections are supposed to re-open after a random time; we chose $\tau_{T_C} = 180\,\text{s}$, however this value has a marginal impact on results.

Queues $R_i \quad (0 \leq i \leq C-1)$**,** model the retransmission of a packet when timeout expires.

Queues $EK_i \quad (1 \leq i \leq C-1)$**,** 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 ([1] and [12] Ch. 21.3) has a deep impact on the protocol performance under heavy load.

Queues $TK_i^2 \quad (1 \leq i \leq C-1)$**,** model the wait for timeout expiration when losses occurred in queues EK_i leaving the congestion window at 2.

Queues $TK_i^3 \quad (1 \leq i \leq C-1)$**,** model the wait for timeout expiration when losses occurred in queues EK_i leaving the congestion window at 3.

The total number of queues needed to model TCP-Tahoe with our approach is

$$M_Q = 2(W-1) + (W-3) + 2\frac{W}{2} + (\frac{W}{2} - 3) + 5(C-1) + 2$$

For instance, if the TCP maximum segment size (MSS) is 1024 bytes, the advertised receiver window size is 64 kbytes and $C = 16$ (the case we use for the model validation), then $W = 64$, and we need $M_Q = 357$ queues.

The probabilities $P(Q_i, Q_j)$ that customers completing their service at queue Q_i move to queue Q_j can be computed from the dynamics of TCP. These dynamics depend on the working conditions of the network, i.e., they depend on the round-trip time as well as on the packet loss probability. In order to cope with the correlation among losses of packets within the same congestion window[2], we introduce two different packet loss probabilities:

TABLE I

QUEUES AND SERVICE TIMES.

Queue		Service time
E_1		$\overline{\text{RTT}}$
E_2		$\overline{\text{RTT}}$
E_i	$i = 2^n \quad n \geq 2,$	$\sigma\,\overline{\text{RTT}}$
E_i	$2^{n-1}+1 \leq i \leq 2^n - 1$	$\frac{(1-\sigma)\overline{\text{RTT}}}{2^{n-1}-1}$
ET_i	$1 \leq i \leq 3$	$T_0 - \overline{\text{RTT}}$
ET_i	$i \geq 4$	T_0
EF_i		$\frac{\overline{\text{RTT}}}{2} + \frac{4\,\overline{\text{RTT}}}{i}$
L_i		$\overline{\text{RTT}}$
F_i		$\frac{\overline{\text{RTT}}}{2} + \frac{4\,\overline{\text{RTT}}}{i}$
$T0_2$		$T_0 - \overline{\text{RTT}}$
$T0_i$	$3 \leq i \leq W$	$T_0 - \overline{\text{RTT}}/2$
T_i	$1 \leq i \leq 6$	$2^i\,T_0$
T_i	$7 \leq i \leq C-1$	$64\,T_0$
R_i		$\overline{\text{RTT}}$
EK_i		$\overline{\text{RTT}}$
TK_i^2		$T_i - \overline{\text{RTT}}$
TK_i^3		$T_i - \overline{\text{RTT}}$

[1]The 'tic' is the time granularity of the TCP protocol, i.e., the minimum amount of time that separates non-self clocked protocol operations.

[2]For the characteristics of TCP, the correlation in packet losses is limited to one congestion window, independently from the window size.

TABLE II

TRANSITION PROBABILITIES.

Q_i	Q_j	$P(Q_i, Q_j)$	Condition
E_1	E_2	P_{S_f}	
	ET_1	P_{L_f}	
E_2	ET_2	$P_{L_f} + P_{S_f} P_{L_f} Z^T_{2,2}$	
	ET_3	$P_{S_f} P_{L_f} Z^C_{3,2}$	
E_i	E_{i+1}	$P^2_{S_f} Z^C_{i+1,i}$	$2 \le i < W/2$
	L_i	$P^2_{S_f} Z^T_{i,i}$	$2 \le i \le W/2$
	ET_j	$P_{L_f} Z^T_{j,i} P_{to}(j-1)$	$i \le j < 2(i-1)$ $\wedge$
	EF_j	$P_{L_f} Z^T_{j,i} P_{ft}(j-1)$	$3 \le i \le W/4+1$
	ET_j	$P_{L_f} Z^C_{j,i} P_{to}(j-1)$	$j = 2(i-1)$ $\wedge$
	EF_j	$P_{L_f} Z^C_{j,i} P_{ft}(j-1)$	$3 \le i \le W/4+1$
	ET_j	$P_{L_f} Z^T_{j,i} P_{to}(j-1)$	$i \le j \le W/2$ $\wedge$
	EF_j	$P_{L_f} Z^T_{j,i} P_{ft}(j-1)$	$i > W/4+1$
	ET_j	$P_{S_f} P_{L_f} Z^T_{j,i} P_{to}(j-1)$	$i \le j < 2i-1$ $\wedge$
	EF_j	$P_{S_f} P_{L_f} Z^T_{j,i} P_{ft}(j-1)$	$3 \le i \le \frac{W/2+1}{2}$
	ET_j	$P_{S_f} P_{L_f} Z^C_{j,i} P_{to}(j-1)$	$j = 2i-1$ $\wedge$
	EF_j	$P_{S_f} P_{L_f} Z^C_{j,i} P_{ft}(j-1)$	$3 \le i \le \frac{W/2+1}{2}$
	ET_j	$P_{S_f} P_{L_f} Z^T_{j,i} P_{to}(j-1)$	$i \le j \le W/2$ $\wedge$
	EF_j	$P_{S_f} P_{L_f} Z^T_{j,i} P_{ft}(j-1)$	$i > \frac{W/2+1}{2}$
L_2	$T0_2$	P_{L_f}	
	$T0_3$	$P_{S_f} P_{L_f} + P^2_{S_f} P_{L_f}$	
L_3	$T0_3$	P_{L_f}	
L_i	$T0_i$	$P_{L_f} P_{to}(i-1)$	$4 \le i \le W-1$
	F_i	$P_{L_f} P_{ft}(i-1)$	
	$T0_{i+1}$	$P_{S_f}(1-P^i_{S_f}) P_{to}(i)$	$3 \le i \le W-1$
	F_{i+1}	$P_{S_f}(1-P^i_{S_f}) P_{ft}(i)$	
	L_{i+1}	$P^{i+1}_{S_f}$	$2 \le i \le W-1$
L_W	L_W	$P^W_{S_f}$	
	$T0_W$	$(1-P^W_{S_f}) P_{to}(W-1)$	
	F_W	$(1-P^W_{S_f}) P_{ft}(W-1)$	
EK_i	L_2	$P^2_{S_f} Z^T_{2,2}$	$1 \le i \le C-1$
	E_3	$P^2_{S_f} Z^C_{3,2}$	
	TK^2_i	$P_{L_f} + P_{S_f} P_{L_f} Z^T_{2,2}$	
	TK^3_i	$P_{S_f} P_{L_f} Z^C_{3,2}$	
R_0	E_2	P_{S_f}	
	T_1	P_{L_f}	
R_i	EK_i	P_{S_f}	$1 \le i \le C-1$
	T_{i+1}	P_{L_f}	

the probability of loss of the first segment of the active sliding window, P_{L_f} (where f stands for 'first'), and the loss probability for any other segment of the same window P_{L_a} (where a stands for 'after' the first segment loss). Notice that P_{L_f} is also the probability of losing a burst of packets, since P_{L_a} is much larger than P_{L_f}. The derivation of the expressions for P_L, P_{L_f} and P_{L_a} is reported in Section III-B; assume, for the moment, that they are known.

Introduce the notation $P_S = 1 - P_L$, $P_{S_f} = 1 - P_{L_f}$ and $P_{S_a} = 1 - P_{L_a}$ for the successful packet delivery probabilities. Finally, define the probability $P_T(i)$ that the window growth threshold *ssthresh* has value i, the corresponding cumulative distribution $P_C(i)$, and the quantities $Z^T_{i,j} = \dfrac{P_T(i)}{1-P_C(j)}$ and $Z^C_{i,j} = \dfrac{1-P_C(i)}{1-P_C(j)}$, that are useful in the definition of the transition probabilities.

Table II reports the transition probabilities $P(Q_i, Q_j)$ from source queue Q_i, listed in the first column, to destination queue Q_j, in the second column, under the conditions indicated in the last column; $\wedge$ is the logical AND operator.

Transition probabilities between queues modeling the exponential and the linear congestion window growth are rather straightforward, since they follow from the correct delivery of all the packets offered to the network (see transitions $P(E_i, L_i)$ and $P(E_i, E_{i+1})$ in Table II).

Transitions to and from queues associated with timeouts and retransmissions are also relatively easy to compute, though their definition requires a careful account of Karn's algorithm; refer to transitions $P(R_i, \cdot)$ in the Table.

Successful and failed transmissions of the first packets after the retransmission, i.e., while the RTT estimates and the timeout value are not updated, cause transitions out of queues EK_i.

For transitions that are consequences of losses, we need to introduce the probability that a loss is detected by timeout given that i segments were transmitted after the lost one:

$$P_{to}(i) = \sum_{j=0}^{2} \binom{i}{j} P^{i-j}_{L_a} P^j_{S_a}, \quad 3 \le i \le W,$$

and the corresponding probability that the loss is detected by fast retransmit $P_{ft}(i) = 1 - P_{to}(i)$. P_{to} and P_{ft} appear in the transition probabilities $P(L_i, \cdot)$, which correspond to losses during the congestion avoidance phase. The possibility of losing more than one packet within the congestion window complicates the description, since the destination queue depends on the loss pattern.

Transitions due to losses during slow start are instead quite cumbersome to consider, because the loss pattern and the *ssthresh* distribution (see Section III-A.2) heavily influence how much the congestion window is opened before the transmitter detects the loss, either by fast retransmit or by timeout expiration. Refer to transitions $P(E_1, ET_1)$, $P(E_2, ET_2)$ and $P(E_2, ET_3)$ in the Table.

We distinguish between transitions happening when the first packet transmitted with window size i is lost, and transitions happening when the second packet transmitted with window size i is lost; in the first case the term P_{L_f} appears in the probabilities $P(E_i, ET_j)$ and $P(E_i, EF_j)$, while in the second case the expression of the probability contains $P_{S_f} P_{L_f}$.

A.1 Load offered to the network

In order to compute the total load Λ offered to the network by all TCP connections, we need to derive the load Λ_Q offered by connections in the state described by each queue Q. Each connection in queue Q offers to the underlying IP network a number of packets equal to $\mathcal{P}_Q$, so that the actual load offered to the IP network by each queue is $\Lambda_Q = \dfrac{E[N_Q]\,\mathcal{P}_Q}{\tau_Q} = \lambda_Q \mathcal{P}_Q$, where λ_Q is the arrival rate at queue Q. Terms λ_Q's are computed from the queueing network solution assuming a total of N customers (connections).

Some queues represent states of the protocol in which no packet is generated: queues T_i which model the backoff time-

outs, and queue ET_1 which stands for the timeout expiration time when the only packet in the window has been lost.

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

The generation of two packets in queues E_i, with $i > 1$, is due to the exponential window size increase in slow-start mode. Two packets are generated per service also in queues EK_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.

More complex is the derivation of the number of packets generated per service ($\mathcal{P}_Q$) in queues EF_i and ET_i, for $i > 2$, since it depends on which packet was lost in the previously visited queue. It results:

$$\mathcal{P}_{EF_i} = \mathcal{P}_{ET_i} = \left[(i-2)\,\frac{P_{L_f}}{1-P_{S_f}^2} + (i-1)\,\frac{P_{S_f}\,P_{L_f}}{1-P_{S_f}^2} \right]$$

Instead, the load offered by connections in queues L_i is:

$$\mathcal{P}_{L_i} = \left[P_{L_f}\,i + (1-P_{L_f})\,(i+1) \right] \qquad i \le W-1$$
$$\mathcal{P}_{L_W} = W$$

The computation of the load offered to the underlying IP network by 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

$$\mathcal{S}_{Q_{ll}} = \bigcup_i [F_i \cup T0_i]$$

then its computation is much easier

$$\Lambda_{\mathcal{S}_{Q_{ll}}} = \sum_{i=2}^{W} \left[\sum_{j=1}^{i} j\,P_{S_f}^j P_{L_f} \right] \lambda_{L_i}$$

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

$$\mathcal{P}_{TK_i^2} = \mathcal{P}_{ET_2} = \frac{P_{S_f}\,P_{L_f}\,Z_{2,2}^T}{P_{L_f} + P_{S_f}\,P_{L_f}\,Z_{2,2}^T}$$

A.2 Distribution of *ssthresh*

The threshold *ssthresh* that discriminates between the slow start and congestion avoidance window growth modes introduces a memory in the TCP behavior. Indeed, the protocol evolution does not depend only on the present window size, but also on the value of the window at the previous loss event. This implies that a complete description of the protocol in steady-state conditions would require an overall number of queues around $W/2 \times M_Q$. Rather than doing this, we resort to a stochastic approach based on the consideration that the steady-state distribution of the window size, which can be computed from the steady-state distribution of the number of customers in queues, allows the estimation of the distribution of *ssthresh*.

Let us introduce the set of queues visited only after a loss event:

$$\mathcal{S}_{Q_a} = \bigcup_i [EF_i \cup ET_i \cup T0_i \cup F_i]$$

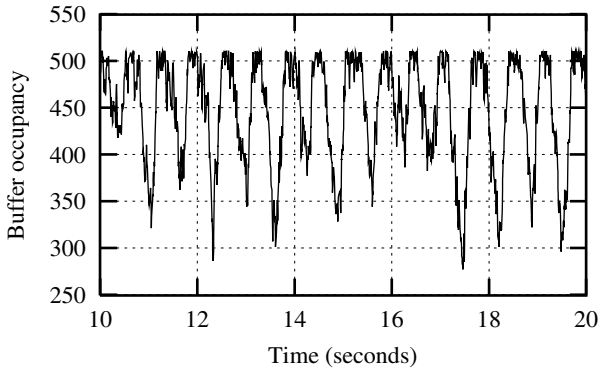

Fig. 3. Example buffer behavior with 200 TCP connections, measured with *ns-2* in a WAN scenario.

and let $\lambda_a(i)$ be the aggregate arrival rate at queues in $\mathcal{S}_{Q_a}$ that have the same window size i. Then the probability $P_T(i) = P\{ssthresh = i\}$ is

$$P_T(2) = \frac{\sum_{k=1}^{5} \lambda_a(k)}{\sum_{k=1}^{W/2} \lambda_a(k)}$$

$$P_T(i) = \frac{\lambda_a(2i) + \lambda_a(2i+1)}{\sum_{k=1}^{W/2} \lambda_a(k)} \quad 3 \le i \le W/2.$$

B. Network Sub-Model

The TCP model we described so far requires as input the loss probabilities P_{L_f} and P_{L_a}, as well as the average round trip time $\overline{\text{RTT}}$ of TCP connections. If these parameters are available, e.g., from network measurements, the TCP model can accurately predict parameters such as the TCP offered load, throughput, time spent in timeout, and window size or *ssthresh* distributions. However, the objective of our model is to predict the performance of concurrent TCP connections given a physical description of the network (i.e., knowing only network span, buffer size, etc.), obtaining as results also the loss probability, the average buffer occupancy and the value of $\overline{\text{RTT}}$ as estimated by TCP transmitters. In order to obtain these results, we need a model of the interaction of TCP connections with the underlying IP network.

The first step is the definition of the basic network element, i.e., a single router interface with its transmission buffer of B packets and its link of capacity C_P packets/s.

Losses in IP networks do not follow Bernoulli patterns [5], [16] due to the presence of FIFO droptail buffers and synchronization among TCP sources. Fig. 3 reports a typical behavior of a droptail buffer loaded by 200 TCP WAN connections as obtained with the *ns-2* simulator. The behavior is pseudo-cyclic, alternating periods without losses to loss bursts.

In [5] the authors assume that the intra-connection correlation among losses is 1, i.e., after a loss all the remaining packets in a congestion window are also lost, though the first lost segment is not necessarily the first segment in the window. In our case, considering that the TCP behavior is such that the first lost packet in a burst of losses is the one in the lowest position in a window (ACKs make the window slide until this is true), it seems more

realistic to assume that the loss rate within a burst is obtained by scaling the initial loss rate:

$$P_{L_a} = \alpha P_{L_f} \qquad 1 \leq \alpha \leq \frac{1}{P_{L_f}} \tag{1}$$

with the constraint that the average loss rate is respected. Empirical observations show that the loss correlation becomes stronger as the window size grows; to catch this behavior we set $\alpha = \overline{w}$, subject to the above constraint, where $\overline{w}$ is the average congestion window size computed by the TCP model.

Though using empirical observations and heuristics, (1) provides a correlation function that allows the use of a very simple model for the basic network element. Indeed, while several different approaches with variable complexity were tested to model the basic network element, a simple $M/M/1/B$ queue was observed to provide sufficiently accurate estimates of P_L and $\overline{\text{RTT}}$ with very limited cost; other approaches with significantly greater complexity provided only marginal improvements in the estimate accuracy.

C. Solution Complexity

The model solution is obtained by iterating between the network sub-model and the TCP sub-model with a fixed point algorithm, until convergence is reached.

Since the network sub-model solution simply consists in computing the closed-form formulas of the M/M/1/B queue, the complexity of each step of the iterative procedure is dominated by the solution of the TCP sub-model. For the solution of the closed queueing network describing the TCP behavior, we adopt the mean value analysis (MVA) approach. A property of MVA is that the solution of a closed network with infinite-server queues is independent on the number of customers; thus, basically, the queueing network solution reduces to the computation of the average arrival rates λ_Q's; in order to derive those, a system of linear equations has to be solved; employing standard techniques, the complexity of this step is $O(M_Q^3)$, where M_Q is the total number of queues. Moreover, exploiting the almost triangular structure of the system of linear equations, the complexity reduces to $O(M_Q^2)$. 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 ϵ and on the initial setting of the parameters computed iteratively. For the first point of a run (a 'run' corresponds to the solution of the model for a given setup varying the number of concurrent connections), we initially set $P_L = 0.01$ and $P_T(i) = 2/W$; for all other points of the run, we use the values obtained for the point just solved. With accuracy $\epsilon = 10^{-6}$, the number of iterations ranges form 20–30 to 200–300. The comparison with *ns-2* simulations is striking. For the results presented in this paper, the CPU time (on a 350 MHz Pentium II PC running Linux) for each simulation point ranges from 5-6 minutes up to several hours. The speedup obtained with the model is at least 2 orders of magnitude in the case of very few connections up to nearly 4 orders of magnitude in the case of hundreds of connections. Results for the case of thousands of connections can be obtained by analysis only.

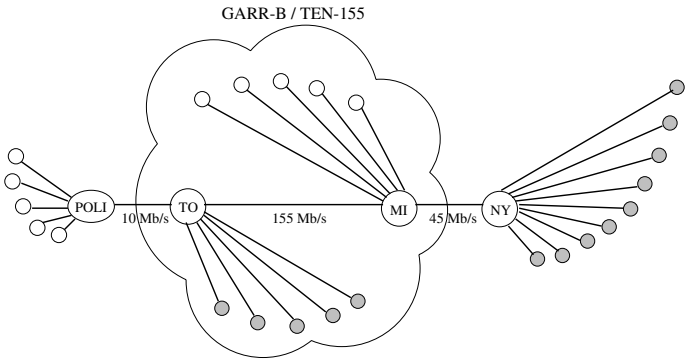

Fig. 4. Abstract representation of the Internet as seen from hosts connected to the LAN of Politecnico di Torino; white circles represent TCP clients; grey circles represent TCP servers.

IV. Model Validation and Numerical Results

In order to validate our analytical model of TCP, we compare the performance predictions obtained from the queueing network solution against point estimates and 95% confidence intervals obtained from very detailed simulation experiments. The tool used for simulation experiments is *ns version 2* [13]; confidence intervals were obtained with the "batch means" technique, using 30 batches.

A. The network topology

Most of the analytical studies of TCP consider the single bottleneck topology as a significant setup for the assessment of the accuracy of approximate models. We chose a somewhat more complex networking environment for the validation of the proposed queueing network model of TCP, which closely resembles the actual path followed by Internet connections from our University LAN to web sites in Europe and the USA.

The topology of the network that we shall consider is shown in Fig. 4; 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 IP network for universities and research institutions, named GARR-B (a portion of the European TEN-155 academic network), 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 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 is assumed to be uniformly distributed between 100 and 6,800 km. From Milano, a 45 Mb/s undersea channel whose length is about 5,000 km reaches New York, and connects the GARR-B network to the north-American Internet backbone (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.

Finally, the distance of servers in the US from the router in NY is assumed to be uniformly distributed between 200 and 3,800 km.

For simplicity, in our performance study we shall restrict our interest to three types of TCP connections:
1. TCP connections for the transfer of web pages from US web sites (servers) to users at Politecnico di Torino (clients); with our previous assumptions, the total length of these connections ranges from 5,351 to 8,960 km.
2. TCP connections for the transfer of web pages from Italian and European servers to clients at Politecnico di Torino; connections lengths range between 151 and 6,860 km.
3. TCP connections for the transfer of web pages from US web sites to users connected to the GARR-B/TEN-155 network through the MI-NY link; connections lengths range between 5400 and 11,600 km.

We shall assume that congestion in the network can be due to the overload of either the 10 Mb/s POLI-TO channel (which is crossed by connections of types 1 and 2) or the 45 Mb/s MI-NY channel (which is crossed by connections of types 1 and 3). We do not consider the possibility of congestion of the 155 Mb/s GARR-B channels (which in practice never happens, according to measured data).

We assume that 25% of the connections traversing the POLI-TO channel also use the MI-NY link. Note that the higher this parameter is, the more correlation exists in the congestion of the POLI-TO and MI-NY channels.

The packet size is assumed constant, equal to 1,024 bytes; the maximum window size is assumed to be 64 packets. We consider the cases of buffer sizes equal to either 128 or 512 packets, and TCP tic values equal to either 100 or 500 ms.

From the estimation of the loss probabilities over the POLI-TO and MI-NY channels, we can estimate the total loss probability of TCP connections for the transfer of web pages from US web sites to users at Politecnico di Torino as $P_L = P_{L_1} + P_{L_2} - P_{L_1} P_{L_2}$, where P_{L_1} is the loss probability on link POLI-TO and P_{L_2} is the loss probability on link MI-NY.

Instead, the average window size value for the same TCP connections can be calculated from the estimation of the total loss probability and total round trip time, by using the queueing network model of TCP with fixed values of loss probability and round trip time.

B. Results for the MI-NY *channel*

Let us first consider the MI-NY channel. Fig. 5 contains two separate plots, that show: i) the analytical and simulation estimates for the packet loss probability, as a function of the number of connections traversing the link (top), for the different considered buffer sizes (either 128 or 512 packets) and TCP tic values (either 100 or 500 ms); ii) the average TCP window value and the corresponding average threshold value for TCP tic equal to 100 ms, as a function of the number of connections traversing the link (lower plot), for the different considered buffer sizes (either 128 or 512 packets). The analytical estimates can be observed to predict with very good accuracy the results produced by simulation experiments.

A similar accuracy of the analytical estimates can be observed also for other performance parameters. For example, in Fig. 6

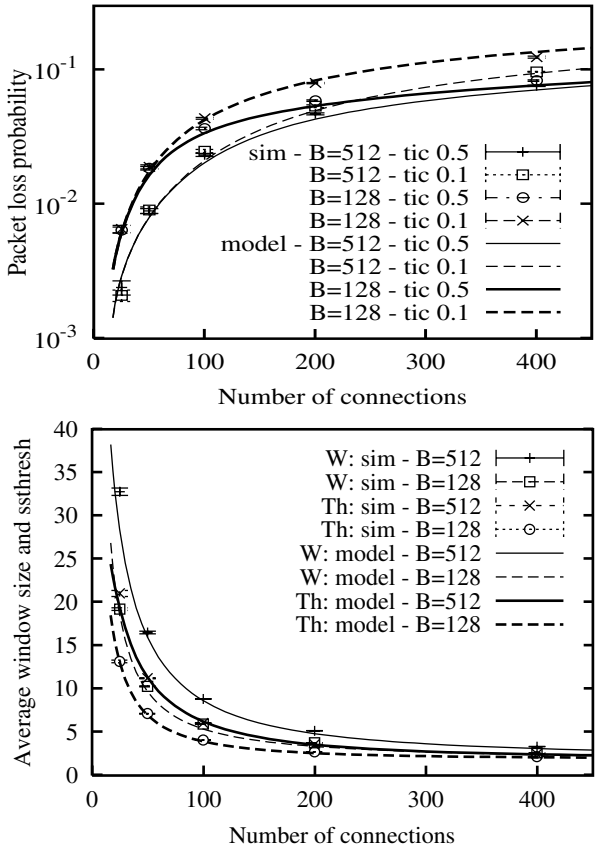

Fig. 5. Packet loss probability versus number of connections (upper plot); average window and *ssthresh* for 100 ms tic (lower plot) when the bottleneck is the MI-NY link.

we show the curves of the distribution of the TCP window size and of the distribution of the corresponding threshold value, obtained from the model and the simulator in the case of 512 packets buffers and TCP tic equal to 500 ms, and 25 connections crossing the MI-NY channel. The fact that the queueing network model is capable of providing accurate estimates not only for average values, but also for distributions is a remarkable indication of the fact that it actually captures most of the internal dynamics of the TCP protocol. Indeed, the spikes shown by the window size distribution in Fig. 6 for values 4, 8, 16 and 32 correspond to the congestion window dimensions that are most likely during the slow start phase, when the window is doubled every RTT. Since TCP tends to cluster packet transmission at the beginning of a cycle when RTT is larger than the window transmission time, the result is that the protocol spends much more time in states corresponding to window sizes that are powers of 2, while, during slow start, quickly skips the states with a window size that is not a power of two.

The model also allows the evaluation of performance indices whose computation may be difficult within a *ns-2* simulation. As an example, we show in Fig. 7 the curves of the probability that TCP connections experience a timeout versus the number of connections traversing the link, for the different considered buffer sizes (either 128 or 512 packets) and TCP tic values (either 100 or 500 ms). These results show interesting behaviors, indicating that the probability of experiencing a timeout is quite sensitive to the number of connections and their parameters.

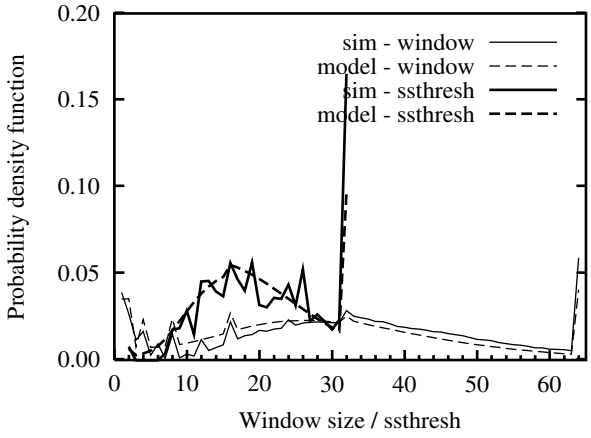

Fig. 6. Window and *ssthresh* distribution when the bottleneck is the MI-NY link with B=512 with 25 competing connections.

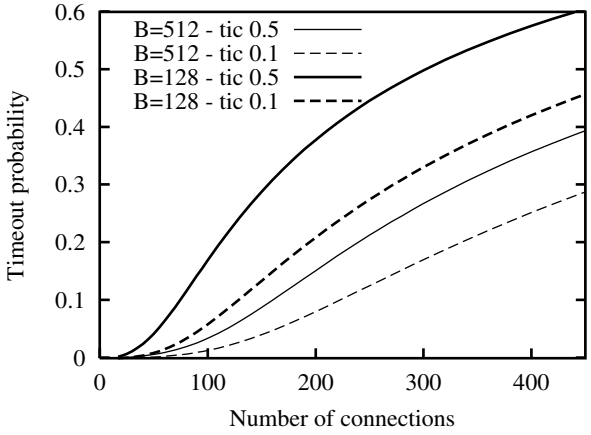

Fig. 7. Fraction of time spent waiting a timeout versus the number of connections.

C. Results for the POLI-TO channel

By considering the POLI-TO channel, it is possible to derive results equivalent to those shown for the MI-NY channel. For the sake of brevity we only report here in Fig. 8 the results equivalent to those previously shown in the top plot of Fig. 5.

Also in this case we can observe an excellent match between simulation results and analytical predictions. Loss probability values are now higher than for the MI-NY channel because the link has smaller bandwidth, and the shorter round trip time allows TCP sources to behave more aggressively.

D. Results for connections from US web sites to Politecnico

After results have been obtained for both channels POLI-TO and MI-NY, they can be combined to obtain performance estimates referring to TCP connections for the transfer of web pages from US web sites to users at Politecnico di Torino, as we mentioned.

In Fig. 9 we show results for the packet loss probability of those connections; the upper plot presents analytical results and simulation point estimates versus N_1 and N_2 (the numbers of connections on the POLI-TO and MI-NY channels, respectively); the lower plot presents analytical results together with simulation point estimates and confidence intervals versus N_1 for different values of N_2. In Fig. 10 we show results for the average

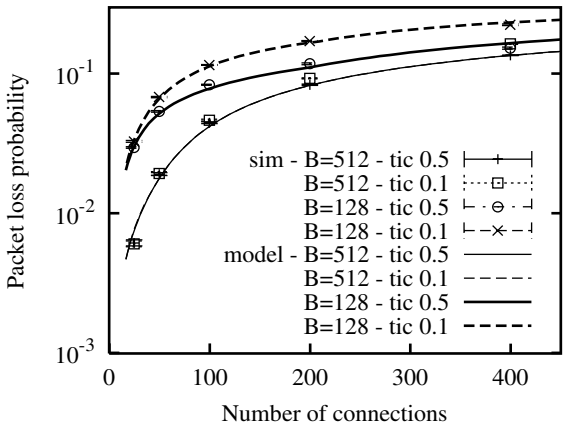

Fig. 8. Packet loss probability versus the number of connections when the bottleneck is the POLI-TO link.

window size of TCP connections crossing both the POLI-TO and MI-NY channels. Even in this more complex scenario the accuracy of analytical performance predictions is excellent.

Similar excellent matches between simulation and analytical results were observed for different setups; results are not reported in this paper due to space limitations.

V. Discussion and Ongoing Work

In this paper we have presented a closed queueing network model for the estimation of the performance of TCP connections. The queueing network provides a detailed description of the behavior of TCP-Tahoe connections and of their interaction with the underlying IP network. Numerical results have shown that the performance estimates provided by the closed queueing network model are extremely close to the performance predictions obtained from simulation experiments run with the *ns-2* package, in spite of the very limited computational cost for the solution of the analytical model.

The low complexity of the solution of the closed queueing network model yields a number of advantages; in particular, it allows: i) the study of more realistic networking setups with respect to the traditional single bottleneck network; ii) the investigation of networks with thousands interacting TCP connections (results were not reported here because such setups are extremely difficult to simulate); iii) the description of protocol details that cannot be normally incorporated into analytical models; iv) the extension of the model to other more recent TCP versions (the model of TCP-Reno is being completed).

In addition, thanks to the low complexity of the closed queueing network model that we have described in this paper, we are currently developing an extension of the model that will allow the derivation of accurate predictions for the delay experienced for the transfer of variable length files over TCP connections traversing an IP network. This will provide an extremely useful indication of the performance experienced by end users while accessing web browsing services.

References

[1] 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.

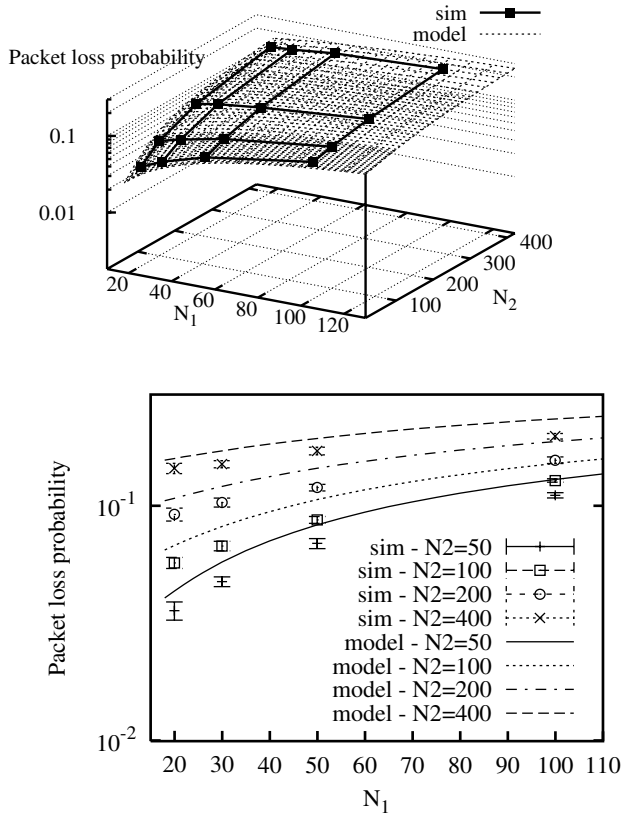

Fig. 9. Packet loss probability when connections cross both bottlenecks versus the number of connections N_1 and N_2 (upper plot), and versus N_1 with constant N_2 (lower plot).

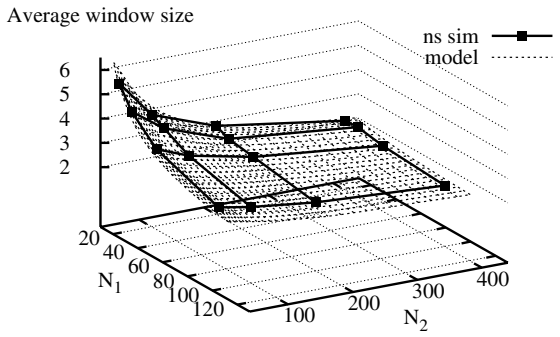

Fig. 10. Average window size when connections cross both bottlenecks versus the number of connections N_1 and N_2

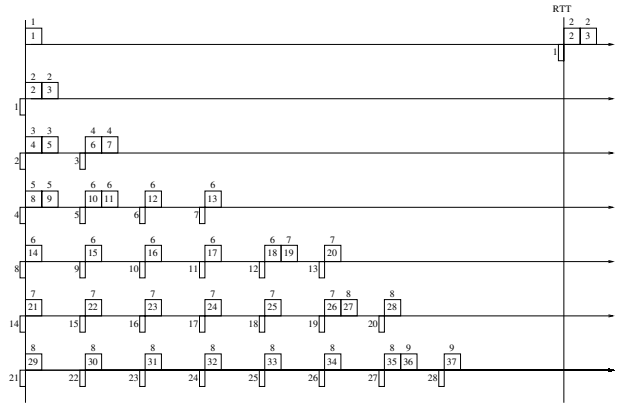

Fig. 11. Idealized TCP cycle with *ssthresh* = 6.

[2] R. Lo Cigno, M. Gerla, "Modeling Window Based Congestion Control Protocols with Many Flows", *Performance Evaluation*, No. 36–37, pp. 289–306, Elsevier Science, Aug. 1999.

[3] V. Paxson. Empirically-Derived Analytic Models of Wide-Area TCP Connections. *IEEE/ACM Transactions on Networking*, 2(4):316–336, Aug. 1994.

[4] 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.

[5] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP Throughput: A Simple Model and its Empirical Validation. *Proceedings of the ACM SIGCOMM'98 - ACM Computer Communication Review*, 28(4):303–314, September 1998.

[6] T. Bonald, "Comparison of TCP Reno and TCP Vegas: Efficiency and Fairness", *Performance Evaluation*, No. 36–37, pp. 307–332, Elsevier Science, Aug. 1999.

[7] V. Mishra, W. B. Gong, D. Towsley, "Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with and application to RED", to appear *in Proc. SIGCOMM'2000*, Aug. 28–Sept. 1 2000, Stckholm, Sweden.

[8] E. Altman, F. Boccara, J. C. Bolot, F. Nain, P. Brown, D. Collange, and C. Fenzy. Analysis of the TCP/IP Flow Control Mechanism in High-Speed Wide-Area Networks. In *34th IEEE Conference on Decision and Control*, pages 368–373, New Orleans, Louisiana, USA, December 1995.

[9] F. Bacelli, D. Hong, "TCP is Max-Plus Linear and what it tells us on its throughput", to appear *in Proc. SIGCOMM'2000*, Aug. 28–Sept. 1 2000, Stckholm, Sweden.

[10] M. Ajmone Marsan, R. Lo Cigno, M. Meo, E. de Souza e Silva, "A Markovian Model for TCP over ATM", *Telecommunication Systems*, Vol. 12, No. 4 (1999), pp. 341-368, Baltzer.

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

[12] W. R. Stevens. *TCP/IP Illustrated, vol. 1*. Addison Wesley, Reading, MA, USA, 1994.

[13] ns-2, network simulator (ver.2). LBL, `URL: http://www-mash.cs.berkeley.edu/ns`.

[14] B. Braden, ed., "Requirements for Internet Hosts – Communication Layers," RFC 1122, Oct. 1989.

[15] V. Jacobson, "Congestion Avoidance and Control," *Computer Communication Review*, vol. 18, no. 4, pp. 314–329, Aug. 1988.

[16] S. Floyd, V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance", *IEEE/ACM Transactions on Networking*, 1(4):397-413, Aug. 1993.

APPENDIX

I. SOME DETAILS OF TCP-TAHOE

The general aspects of the TCP dynamics are well-known, and the specific characteristics of TCP-Tahoe are discussed in [12], [14], [15]; however, since several details of the dynamics of TCP-Tahoe have a significant impact on the proposed queueing network model, and in particular on the definitions of the offered load, of the average service time at queues, and of the transition probabilities between queues, we comment on such aspects in this appendix.

Consider an idealized TCP transmission cycle, beginning with slow start, and continuing with congestion avoidance, up to the first packet loss. Fig. 11 illustrates such cycle in the case *ssthresh* = 6. Time evolves along the horizontal axis from left to right, folding over when the line ends; each line represents one RTT. Larger rectangles above the lines represent transmitted

TCP segments with their sequence number[3]; the number above the segments is the congestion window size w at the time of transmission of that segment; small rectangles below the time line represent ACKs *received* by the transmitter.

During the first RTT (slow start) the TCP connection offers 1 packet to the network, and when the first ACK arrives, 2 packets are transmitted back-to-back; silence then follows for the remaining part of the RTT. ACK #1 and packets #2 and #3 are duplicated on line 1 and 2 to show how time wraps in the plot. The definition of Λ_{E_1}, Λ_{E_2}, τ_{E_1}, τ_{E_2} follows immediately.

At this point in the protocol evolution, several models (see for instance [10] simply assume that transmission times are negligible, and ACKs are received back-to-back, so that w doubles, and 4 TCP segments are transmitted back-to-back. This assumption is a gross approximation for several reasons: i) transmission times cannot be assumed negligible, otherwise a TCP source would need its window size growing to ∞ to saturate the pipe; ii) TCP windows increase in a continuous way, hence, before growing to 4, the window must assume the value 3, and keep such value for at least 2 transmission times, and so forth for the other values; iii) if the network is heavily loaded, then packets and ACKs are interleaved in the network, and ACKs cannot arrive back-to-back at the source. Exactly how far ACKs are spaced is probably impossible to predict and model; hence we simply assume that ACKs are uniformly spaced in the interval between two ACKs that cause the window size to be a power of 2 ($w = 2^i$). In fact, during slow start, the window size grows geometrically with base 2, which makes the window size $w = 2^i$ somewhat more likely than other values, as can be seen from Fig. 11, and confirmed by the window size distributions shown in Fig. 6, that clearly shows peaks in the pdf when $w = 2^i$. These considerations explain why, during slow start, TCP is modeled with a linear increase of the window size, offering to the network 2 packets each time the window size is increased ($\Lambda_{E_i} = 2$), and why the time spent with a given window size is inversely proportional to w for $w \neq 2^i$, while it is set to a constant fraction (σ) of RTT when $w = 2^i$ (see the definition of τ_{E_i}). Interestingly, quantitative results, excluding the distribution of w, are almost independent from the value chosen for σ in $[1/w, 1]$; hence we simply chose $\sigma = 2/3$ to reduce the need for parameter tuning; this choice gives very satisfactory results also for the distributions of w and *ssthresh*, as shown in Fig. 6. Clearly, the peaks in the distribution of w flatten out by choosing $\sigma = 1/w$, and are maximized for $\sigma = 1$.

When *ssthresh* is reached (see lines 4 and 5 in Fig. 11), TCP enters congestion avoidance: w is increased by 1 every w ACKs. In practice, w is increased by 1 every RTT, as can be seen in Fig. 11. A number of observations are in order: i) the linear growth mode is entered *without increasing the window size*, i.e., the transition is $E_i \Rightarrow L_i$; ii) the ACK of the first TCP segment transmitted in congestion avoidance with window size w makes the window size grow to $w+1$; iii) the number of TCP segments transmitted during congestion avoidance with window size w, if no loss occurs, is $w + 1$. Considering points i)–iii), and the possibility of losing the first packet transmitted in linear growth with window i, the derivation of Λ_{L_i} and τ_{L_i} follows.

Let us now consider the dynamics of losses. Whenever a packet transmitted during slow start, say with window size w, is lost, before the transmitter detects the loss, the window keeps growing. The value w' of the window size when the loss is detected depends both on *ssthresh* and on the number of received ACKs, i.e., on w and the number of packets lost. As a general rule, if *ssthresh* $> 2\,w$, then, if the lost packet is the first one transmitted with window w, then $w' = 2\,w - 2$, while if the lost packet is the second one transmitted with window w, then $w' = 2\,w - 1$. This rule can be easily verified in Fig. 11, checking what happens if either packet 6 is lost (this is the first packet transmitted with $w = 4$; all ACKs up to #5 are correctly received), or packet 5 is lost (this is the second packet transmitted with $w = 3$; all ACKs up to #4 are correctly received). The transition probabilities defined in Table II stem from this simple property, associated with the *ssthresh* distribution.

The number of packets transmitted by a TCP source in slow start from the loss instant to the loss detection depends only on the window size when the loss is detected, w', and on the fact that the lost segment was the first or second transmitted with window size w, as stated by Λ_{EF_i} and Λ_{ET_i}. Instead, service rates depend only on the estimation of T_0 if a timeout occurs, or on the rate of received ACKs (i.e., transmitted packets) in the event of a fast retransmit.

When losses occur during congestion avoidance, the model visits queues F_i or $T0_i$. The load offered by these queues depends on which packet was lost among the ones transmitted in the queue L_j which was visited before. Fortunately, if we consider the arrival rate at queues L_i, then the load offered to the network by queues F_i and $T0_i$ can be defined as a function of the arrival rate at queues L_i. This observation led to the definition of $\Lambda_{\mathcal{S}_{Q_{ll}}}$.

Let us now comment briefly about queues modeling the recovery from timeouts. If a lost packet is successfully retransmitted at the first attempt, then the normal evolution of the protocol is retrieved. Karn's algorithm avoids the update of the timeout value RTO, but since this was correctly set there are no drawbacks. On the other hand, if there are multiple retransmissions, the RTO value is consequently backed off, say to the j-th timeout. After the successful retransmission, the window is open ($w = 2$) and the first 2 new packets are transmitted, but Karn's algorithm still prevents the update of both RTO and the backoff value. If one of these two packets is lost, the retransmission occurs using the j-th timeout. Queues EK_i and $TK_i^{2,3}$ model exactly this phenomenon, where the letter K in the queue name recalls that the queues model Karn's algorithm. The impact on performance of this behavior when loss rates exceed 10–15% can be striking, since connections spend much more time in timeout than one would expect. We are aware that the phenomenon we just described is due to *implementation choices*, and is not embedded in the TCP definition found in RFC's; however, this behavior is found in several BSD-based TCP implementations (for example, both in the Tahoe and Reno versions). Hence, we decided to model it with our closed queueing network, with the specific aim of showing the power and the flexibility of the proposed modeling approach.

[3]For the sake of simplicity we number segments, and not bytes, starting from 1 and we number ACKs after the packet they acknowledge, so that segment #1 is acknowledged by ACK #1, not ACK #2, as in the standard TCP numbering scheme.