Chapter 3

A Generalization of a TCP Model: Multiple Source-Destination Case with Arbitrary LAN as the Access Network

Oleg Gusak and Tuğrul Dayar
Bilkent University

Abstract This paper introduces an analytical model of an access scheme for Wide Area Network resources. The model is based on the TCP-modified Engset model of Heyman, Lakshman, and Neidhardt. The proposed model employs a LAN with arbitrary topology as the access network and is able to take into consideration files of arbitrary size. Through simulations, we show that our model is applicable for the multiple source case where each source corresponds to a different Web server at a possibly different location on the Internet. The average reception time of a file computed analytically is acceptably close to the simulation results.

Keywords: TCP-Reno, TCP-modified Engset model, LAN-WAN access scheme.

3.1 Introduction

Nowadays the Transmission Control Protocol (TCP) is omnipresent. TCP indeed is Internet itself and it makes the network transparent for any application using this protocol as a transport service. A complete TCP utilization requires appropriate models that can describe the behavior of TCP under certain conditions and network patterns. Various studies have been done in this

direction. The new research direction in this area relates to new TCP modifications [4, 8, 9, 10] such as TCP Reno and TCP Sack that perform dynamic flow control.

The majority of previous studies consider the TCP model for a single source-destination pair, where the source is a particular server or a station like a router, which accumulates traffic from different sources. Detailed studies of TCP under the single source-destination assumption have allowed researchers to develop more complex and realistic models of networks. Our work is based on one such model proposed in [6] by Heyman, Lakshman, and Neidhardt. That model considers an access network to Internet resources, such as Web servers, for users working from terminals through Integrated Services Digital Network (ISDN) lines. The model gives an analytical description for the process of Web page retrieval from a remote server residing on a Wide Area Network (WAN) and connected to local terminals by a backbone link. Each terminal is described by average idle (thinking) time and average file size requested from the remote Web server. Such a process in which each station alternates between working and thinking states is called an alternating renewal process. The steady-state probability of finding a station in working state depends only on the average idle time and the average file size requested by that station [6, p.26]. Hence, it is possible to use a memoryless distribution for both idle time and busy time which in turn allows modeling of terminal behavior as an ordinary birth-death process.

The contribution of our paper is the following. First, we extend the TCP-Modified Engset model [6, pp. 30–31] to the general case of an access network, which is a Local Area Network (LAN) with arbitrary topology. The performance measure of interest is the average reception time of a file for a station residing on the LAN. Second, we introduce the description of a slow start phase in the TCP-Modified Engset model, which allows us to take into consideration small file sizes. As recent studies show [2, 3], the average file size of Web traffic tends to be small. However, the model in [6] considers the average file size requested by a station to be sufficiently large so that the effect of the slow start phase of the TCP protocol can be neglected. In this connection, the simulations in [6] use an average file size of 200 Kbytes. In our model, we do not have any constraints on file size. Finally, through simulations we show that the improved model can be applied to the multiple source (i.e., Web servers) case which implies variable Internet delay for packets sharing the same backbone link.

The paper is organized as follows. In Section 3.2, we discuss a general model describing TCP behavior over a bottleneck channel. In Section 3.3, we present an improved model for the LAN-WAN access scheme and the particular case where the access network is Ethernet. We verify the analytical model through simulations in Section 3.4 and conclude in Section 3.5.

3.2 TCP Behavior: Contest for the Pipe

We consider a model for TCP Reno, which is currently one of the most popular protocols on Internet. From this point on, we refer to TCP Reno as TCP. There are two operating phases of TCP called slow start and congestion avoidance (see [12, pp. 285–286, 310–312] for details). When the sender starts transmitting at the beginning of a connection, it is in the slow start phase, and current window size, W_c, is equal to 1. Upon receiving each successful ACK, the sender increments W_c by 1. When W_c becomes large enough to create congestion, packet losses occur. When the sender detects packet losses by duplicate ACKs, it halves W_c, records the value in W_t, and enters the congestion avoidance phase. When losses are detected by timeouts, TCP halves W_c, records the value in W_t, sets W_c to 1, and enters the slow start phase (if it is already not in it). During the congestion avoidance phase, W_c is incremented by 1 each time W_c packets are successfully acknowledged. When TCP is in the slow start phase and W_c reaches W_t, it enters the congestion avoidance phase.

Let us consider the behavior of TCP for an arbitrary source and destination pair connected by a link where the source is persistent. The link is characterized by the transmission speed S, the round trip delay D (which includes channel transmission and propagation delays for a packet and an acknowledgement), and the transmitter buffer capacity B. There are two communication patterns that are possible under the given assumptions. The first one takes place when the transmission speed of the source is less than or equal to the link bandwidth. In this case, congestion in the buffer cannot occur, and data is transferred at a rate equal to the source transmission speed. This is the trivial case and it is not interesting for modeling purposes. The second one happens when the link bandwidth is less than the transmission speed of the source and the link is the bottleneck. In this case, the TCP algorithm increments W_c, which after some time causes congestion in the buffer (i.e., some packets will be dropped). The value of W_c at the point of congestion is called the congestion window size and is denoted by $W_{highest}$. Hence, for the case of always dropping a single packet upon congestion, W_c oscillates between $W_{highest}$ and $W_{highest}/2$. When packets are lost and consequently W_c is reduced, the transmission proceeds with a rate less than the link bandwidth. The goal is to estimate this reduction in transmission rate.

A detailed description of the TCP model construction can be found in [4, 6]. Therein, the discussion is based on estimating the channel capacity, which is the number of packets that can reside in the channel between source and destination. Right before congestion occurs, the channel contains $D/P_t + B$ packets [6, p. 28], where $P_t = P_{tcp}/S$ is the time required to transmit one TCP packet of size P_{tcp} by the bottleneck link since the buffer is full before congestion.

Assuming that loss detection happens only through duplicate ACKs, after

TCP discovers a packet loss, it halves W_c and enters the congestion avoidance phase. During this phase, W_c increases by 1 starting from $W_{highest}/2$ up to $W_{highest}$ each D time units. We remark that in the last part of this growing phase when the population of packets in the channel has reached its maximum value, packets start accumulating in the buffer, and W_c will be increasing by 1 in larger time intervals than D due to the queuing of packets in the buffer. When buffer size is small, we can neglect the nonlinearity of the growth during this last phase and assume that W_c grows linearly. When the channel is full, the source transmits with a rate equal to the link bandwidth. Due to the linear growth assumption of W_c, we can estimate the TCP degradation factor using the normalized lowest and largest window sizes of the connection. Normalization has to be done with respect to the link capacity excluding the buffer size, since packets accumulated in the buffer do not contribute to a service rate increase. In other words, packets are served with maximum possible speed equal to S when the channel is full, and the situation does not improve with additional packets in the buffer. The maximum number of packets that can fit to the channel is therefore $W_{highest} = D/P_t + B$. The lowest packet population size of the connection is the highest packet population size divided by 2 for each packet discarded due to buffer overflow. In other words, $W_{lowest} = 2^{-l} \cdot (D/P_t + B)$, where l is the number of packets lost in a congestion. In [6], various cases for different queuing service disciplines and lower layer protocols such as ATM are considered. There it is shown that when each TCP packet is divided into more than one lower layer cell depending on the service policy of the router, it is possible to lose up to 3 TCP packets in a congestion [6, p. 29]. For the sake of simplicity, in our model we consider the case in which TCP packets are not segmented, that is, each TCP packet can fit into one cell of the lower layer protocol. Under this assumption, only one packet is lost per congestion at the router's buffer.

As stated before, normalization has to be done with respect to channel capacity excluding buffer size, i.e., D/P_t. Hence, the corresponding normalized values of $W_{highest}$ and W_{lowest} can be written as [6, p. 29] $w_{highest} = W_{highest}/(D/P_t) = 1 + b$, $w_{lowest} = W_{lowest}/(D/P_t) = 2^{-l} \cdot (1 + b)$, where $b = B/(D/P_t)$ is the normalized buffer size.

The TCP degradation factor, ρ, is the average of the normalized packet population over the increase of W_c from $W_{highest}$ to W_{lowest} [6, p. 30]:

$$\rho = 1 - \frac{((1 - w_{lowest})^+)^2}{2 \cdot (w_{highest} - w_{lowest})} . \tag{3.1}$$

Note that when $w_{highest}$ exceeds 1, ρ remains equal to 1. The positive operator enables handling the case where W_c is always greater or equal to D/P_t (i.e., channel is always full).

3.3 Multiple Source-Destination Model

Let us consider a typical model of access for local users to Internet resources. Local users are sharing a common LAN (e.g., campus network) and this network is connected by a dedicated link to the Internet. Usually the bandwidth of the dedicated link is smaller than the transmission rate of the LAN as well as the throughput of the Internet server. In this case, the Internet link can be considered as a bottleneck along the way from a local user to an Internet resource. The communication pattern for such access scheme is the following. Local users request files from servers residing on the Internet, which may have different network distances to the Internet router (router through which the link is connected to the Internet) and hence different Internet delays. We assume that this delay is a random variable with mean D_I. We also assume that the number of bytes (e.g., Web page size) requested by each station has the same distribution with mean f_s for all stations. Let t_f denote the average reception time of a file for a station. Upon retrieval of a file, a station enters the idle phase (i.e., thinking state) which has mean t. Such a process in which local stations oscillate between idle and working states is called an alternating renewal process [11, p. 66]. Previous studies of such a process show that the long run probability of a station being in the working state does not depend on the distribution functions of these on and off periods but it depends only on their means: $P(\text{station is active}) = t_f/(t_f + t)$ [11, p. 67]. This fact allows us to use exponential distribution for the on and off periods. Hence, one can model the access network behavior as an ordinary birth-death process with, respectively, the following birth and death rates:

$$\begin{aligned} \beta_j &= (n-j)\cdot\lambda, \quad j = 0, 1, \ldots, n-1, \\ \delta_j &= j\cdot\mu, \qquad\quad j = 1, 2, \ldots, n\ , \end{aligned} \tag{3.2}$$

where $\lambda = 1/t$ is the rate of exit from the idle state, n is the number of stations on the LAN, and $\mu = 1/t_f$ is the rate of file retrieval from the server.

Under the given assumptions, when the Internet link is the bottleneck, each station competes to acquire the resource, which in turn leads to congestion. So, the situation in the channel will be the same as that of the single source communication scheme we considered before. That is, channel efficiency will be reduced by the factor ρ. We remark that for the multiple station case, $W_{highest}$ and W_{lowest} are, respectively, the highest and lowest total number of packets that can reside on the channel for all sessions. Let us also note that in order to be able to determine the measure ρ for the multiple source-destination model, the round trip delay of section 2 has to be extended by the delays packets experience in the LAN (D_L) and the Internet (D_I). In this case, the file transfer (i.e., death) rate can be expressed as $\delta_j = \rho\cdot S/f_s$, $j = 1, 2, \ldots, n$. Such a

system has the trivial product-form solution [7, p. 92]

$$P_j = P_0 \cdot \frac{\beta_0 \cdot \beta_1 \cdot \ldots \cdot \beta_{j-1}}{\delta_1 \cdot \delta_2 \cdot \ldots \cdot \delta_j}, \quad j = 1, 2, \ldots, n \ , \tag{3.3}$$

where P_j is the probability of having j active stations and P_0 is the normalization coefficient. Now, let $r_j = \rho \cdot S/(f_s \cdot j)$ denote the file transmission rate of each active station when there are j active stations. Then the average file transmission rate can be expressed as $r = \sum_{j=1}^{n} P_j \cdot r_j$.

In the next subsection we extend our model to the case of small file size, and in the second subsection we consider the particular case of Ethernet as the access network.

3.3.1 Adaptation to Small File Size

In the previous discussion, we did not take into consideration the slow start phase of TCP. We assumed that W_c always oscillates between W_{lowest} and $W_{highest}$. When $W_{highest}$ is large enough, a connection has to spend a relatively significant amount of time to reach this threshold. Moreover, if we consider small file sizes, the time spent to reach $W_{highest}$ will be significant compared to the total time required for file transmission.

For the case of small files, we make the following observation. The file size can be so small that W_c does not exceed $W_{highest}$. That is, TCP does not enter the congestion avoidance phase and the results obtained in Section 3.2 and earlier in this section are not applicable. Before we start discussing this extreme case, let us first consider the process of window growth in the slow start phase.

By definition (see TCP description in Section 3.2), W_c of a single connection starts growing from 1. After D time units (recall that D for the multiple source-destination case is extended by delays D_I and D_L), the sender is acknowledged upon the successful transmission of the first packet. In response to this event, the sender increments W_c by 1 and emits two new packets. The first new packet is transmitted because the very first packet has already been acknowledged. The second new packet is transmitted because $W_c = 2$ and there is only one unacknowledged packet. So, it is easy to see that each D time units, the number of packets in the channel is doubled. Based on this observation, we can determine the duration of the slow start phase for the multiple station case as follows. By using the total population of packets residing in the channel and the number of active stations, we can find the largest window size for each active connection. Then the duration of the slow start phase is given by

$$D_{sl\_st} = D \cdot \left(\log_2 \left(W^j_{highest} \right) + 1 \right) \ , \tag{3.4}$$

where $W^j_{highest} = W_{highest}/j$ is the largest window size per active connection when there are j active stations.

To determine the average retrieval time of a file taking into account the slow start phase, we have to divide the total number of packets required to transmit a file of average size f_s into two groups: packets transmitted during the slow start phase,

$$N_s = 2 \cdot W^j_{highest} - 1 \ , \tag{3.5}$$

and packets transmitted during the following congestion avoidance phase(s),

$$N_{ca} = f_s / P_{tcp} - N_s \ . \tag{3.6}$$

Thus, the total average retrieval time of a file for an active connection when j stations are active can be expressed as $t^j_f = N_{ca} \cdot P_{tcp}/(\rho \cdot S/j) + D_{sl\_st}$.

Thereafter, the modified death rates for the system are given by the following:

$$\begin{aligned} 1/\delta_j &= t^j_f / j \\ &= \left[\frac{(f_s/P_{tcp} - N_s) \cdot P_{tcp}}{\rho \cdot S/j} + D \cdot \log_2(W_{highest}/j + 1) \right] / j \ . \end{aligned} \tag{3.7}$$

Let us now return to the extreme case in which the file size is smaller than the number of packets transmitted during the slow start phase. In this case, since $N_s = f_s/P_{tcp}$, the largest window size is obtained from Eq. (3.6) as $W_{sfs} = (f_s/P_{tcp} + 1)/2$. Hence, the average retrieval time of a file using Eq. (3.4) is given by $t^j_{sfs} = D \cdot (\log_2(W_{sfs}) + 1)$.

Finally, the death rates for the system in general can be written as

$$\delta_j = \begin{cases} j/t^j_{sfs}, & [2 \cdot (W_{highest}/j) - 1] \cdot P_{tcp} \geq f_s \\ j/t^j_f, & \text{otherwise} \end{cases} \tag{3.8}$$

3.3.2 A Case Study: Ethernet as the LAN

We consider an application of our model where the access network is Ethernet. The communication scenario remains the same. We assume that the LAN stations do not communicate with each other except for the communication between each station and the router, which is also a station on the LAN. Stations request files of average size f_s from a remote server on the Internet. Between successive requests, each station spends an arbitrary amount of time with mean t in the idle state. In order to adapt our model to this communication scheme, we have to estimate the round trip delay experienced by a packet of a connection. The LAN delay D_L is equal to $D_{Ethernet}$, which is the average delay packets experience on the Ethernet. For $D_{Ethernet}$, we use the result derived in [5, p. 230, Eq. (8.30)].

All input parameters for $D_{Ethernet}$ (such as Ethernet transmission speed, propagation delay, etc.) are well defined. The only parameter we have to

determine is the input rate imposed on the LAN. According to our model, D is calculated for the case when packet population reaches it highest value, $W_{highest}$. In this case, the channel is full and packets are transmitted with the maximum possible speed (i.e., the channel rate S). Based on this observation and the fact that each packet is acknowledged by the receiver, the total input rate to the LAN in packets per unit time can be expressed as $\gamma = 2 \cdot S/P_{tcp}$.

In the following section, we present simulation and analytical results for the multiple source-destination LAN-WAN access scheme with Ethernet as the LAN.

3.4 Simulations and Analytical Results

In our simulations, we consider Ethernet (10 Mbps, 300 m cable segment) as the LAN. Internet link transmission speed is 512 Kbps, and end-to-end propagation delay is 0.1 sec. Internet delay with mean D_I modeled as Gamma distribution (see [1, p. 290] for a justification) with shape parameter b taking the values 1, 2, or 4, and D_I taking the values 0.1 or 0.2 sec. Note that when $b = 1$ we have the exponential distribution. We also run simulations for the single source case, i.e., when Internet delay is constant. Capacity of the Internet link router buffer is 20 packets. TCP packets have a fixed size of 512 bytes. For a particular simulation, 1,000 events (i.e., file retrievals) per station are generated. Confidence intervals of 90% with 9 degrees of freedom are computed for the average retrieval time of a file in seconds. Analytical and simulation results with confidence intervals for the model in Section 3 are presented in Tables 3.1 and 3.2 for 20 stations, and in Table 3.3 for 30 stations on the LAN.

According to the analytical and simulation results, the average retrieval time of a file increases almost linearly with average file size. Analytical results for $D_I = 0.1$ sec are within 5% and for $D_I = 0.2$ sec are within 8% of simulation results. The difference between analytical and simulations results is also relatively larger for small average file sizes. Our explanation is the following. When the average file size is small, we are more likely to have a smaller average number of active stations, and a larger average number of files can be transmitted within the slow start phase. In this case, connections cannot fill the channel to create congestion; that is, there are no collisions and TCP never leaves the slow start phase. However, traffic is bursty since upon receiving each ACK, the sender increments the current window size by one and therefore emits two packets, doubling the number of packets in the channel each round trip delay. Therefore, packets are not distributed uniformly across the channel. See each table for increasing average file size. For instance, in Table 3.1 the const column for average file size of 20 kB has a value which is within 10% of the analytical result.

Table 3.1 $D_I = 0.1$, 20 Stations

f_s	analytical	const	b=1	b=2	b=4
20	4.718	4.267±0.025	4.307±0.013	4.290±0.025	4.277±0.026
60	17.628	16.721±0.075	16.706±0.067	16.756±0.077	16.761±0.088
100	30.671	29.144±0.163	29.322±0.120	29.347±0.119	29.424±0.149
140	43.724	41.633±0.116	41.956±0.116	41.949±0.164	41.887±0.190
180	56.781	54.095±0.166	54.837±0.252	54.733±0.091	54.428±0.193

Table 3.2 D_I=0.2, 20 Stations

f_s	analytical	const	b=1	b=2	b=4
20	5.157	4.378±0.024	4.480±0.014	4.456±0.022	4.442±0.020
60	18.386	16.727±0.075	16.801±0.069	16.807±0.109	16.822±0.100
100	31.81	29.151±0.164	29.518±0.126	29.500±0.127	29.547±0.151
140	45.253	41.635±0.115	42.328±0.125	42.290±0.174	42.162±0.115
180	58.701	54.099±0.167	55.374±0.251	55.142±0.232	54.777±0.380

Table 3.3 D_I=0.1, 30 stations

f_s	analytical	const	b=1	b=2	b=4
20	7.737	7.262±0.024	7.254±0.025	7.244±0.037	7.240±0.030
60	27.29	25.950±0.126	25.951±0.088	26.003±0.081	26.054±0.097
100	46.876	44.617±0.123	44.814±0.134	44.756±0.097	44.658±0.104
140	66.465	63.642±0.136	63.571±0.209	63.482±0.146	63.297±0.235
180	86.055	82.311±0.131	82.448±0.241	82.214±0.405	82.366±0.144

3.5 Conclusion

In this paper, we have extended the TCP-Modified Engset model to the general case of an access network for local users that are on a LAN with arbitrary topology and delay function. We have also introduced modifications that enable us to consider files of arbitrary size. We have shown through simulations that our model can be applied to the multiple source case. The simulation results show that the analytical model provides a good estimation for the average retrieval time of a file in this LAN-WAN access scheme. The error in the analytical model is within 8% of the simulation results for the chosen parameters.

References

[1] J.-C. Bolot, End-to-End Packet Delay and Loss Behavior in the Internet, *Proc. of SIGCOMM '93,* San Francisco, pp. 289–298, August 1993.

[2] H.-W. Braun, K.C. Claffy, Web Traffic Characterization: An Assessment of the Impact of Caching Documents from NCSA's Web Server, *Computer Networks and ISDN Systems,* Vol. 28, pp. 37–51, 1995.

[3] M.E. Crovella, A. Bestravos, Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes, *IEEE/ACM Trans. on Networking,* Vol. 5, pp. 835–846, 1997.

[4] J.-L. Dorel, M. Gerla, Performance Analysis of TCP-Reno and TCP-Sack: The Single Source Case, Computer Science Department, University of California, Technical Report 97003, 1997.

[5] J.F. Hayes, *Modeling and Analysis of Computer Communication Networks,* Plenum, New York, 1984.

[6] D.P. Heyman, T.V. Lakshman, A.L. Neidhardt, A New Method for Analysing Feedback-Based Protocols with Applications to Engineering Web Traffic Over the Internet, *Performance Evaluation Review,* pp. 24–38, 1997.

[7] L. Kleinrock, *Queuing Systems,* Vol. 1, The Theory, Wiley, New York, 1975.

[8] T.V. Lakshman, U. Madhow, The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss, *IFIP Trans. C-26, High Performance Networking V,* North-Holland, pp. 135–150, 1994.

[9] T.J. Ott, J.H. Kemperman, M. Mathis, *The Stationary Behavior of Ideal TCP Congestion Avoidance,* ftp://ftp.bellcore.com/pub/tjo/TCPwindow.ps, 1996.

[10] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, Modeling TCP Throughput: A Simple Model and Its Empirical Validation, Department of Computer Science, University of Massachusetts, CMPSCI Technical Report TR 98-008, 1998.

[11] S.M. Ross, *Stochastic Processes,* Wiley, New York, 1983.

[12] W.R. Stevens, *TCP/IP Illustrated,* Vol. 1. The Protocols, Addison Wesley, 1994.

Vitae

Oleg Gusak received his B.S. degree in computer engineering and his M.S. degree in computer science from Kharkov State Technical University of Radio Electronics, Kharkov, Ukraine, in 1994 and 1995, respectively.

In 1995–1997 he was working as a software engineer of Scientific-Research Institute of Automated Control Systems of Gas Pipelines (Concern UkrGasProm).

Since September 1997, he has been a Ph.D. student at the Department of Computer Engineering and Information Science of Bilkent University, Ankara, Turkey. His present research interests include performance modeling and analysis, mathematical models of Markov chains, and computer and communication networks.

Tugrul Dayar received his B.S. degree in computer engineering from Middle East Technical University, Ankara, Turkey, in 1989, and his M.S. and Ph.D. degrees in computer science from North Carolina State University, Raleigh, NC, USA, in 1991 and 1994, respectively.

Since 1995, he has been an Assistant Professor in the Department of Computer Engineering and Information Science of Bilkent University, Ankara, Turkey. His research interests are in the areas of performance modeling and analysis, numerical linear algebra for stochastic matrices, scientific computing, and computer networks.

Dr. Dayar is a member of Upsilon Pi Epsilon, IEEE Computer Society, ACM Special Interest Group on Measurement and Evaluation, SIAM Activity Group on Linear Algebra, and AMS.