3-30-2006

Modeling the Throughput Performance of the SF-SACK Protocol

Laura M. Voicu
University of South Florida

Modeling the Throughput Performance of the SF-SACK Protocol

by

Laura M. Voicu

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Science
Department of Computer Science and Engineering
College of Engineering
University of South Florida

Major Professor: Miguel A. Labrador, Ph.D.
Ken Christensen, Ph.D.
Adriana Iamnitchi, Ph.D.

Date of Approval:
March 30, 2006

Keywords: performance evaluation, TCP modeling, congestion window-based protocols, send rate, Markov regenerative processes.

Table of Contents

List of Tables ii

List of Figures iii

Abstract v

Chapter 1 Introduction 1
1.A. Contributions of the Thesis 2
1.B. Organization of the Thesis 3

Chapter 2 Related Work 4
2.A. TCP and TCP Modeling 5
2.B. Markov Regenerative Processes 8

Chapter 3 The SF-SACK Protocol 11

Chapter 4 The Analytical Model of the SF-SACK Protocol 14
4.A. Losses are Detected Exclusively by Triple-Duplicate Acknowledgments and There is One Loss in Each Scheduler's Interval 18
4.B. Losses are Detected Exclusively by Triple-Duplicate Acknowledgments 26
4.C. Losses are Detected Exclusively by Triple-Duplicate Acknowledgments or by Time-Outs 35
4.D. Calculation of the Throughput of a Bulk Transfer SF-SACK Flow 46

Chapter 5 Conclusions and Future Work 51

References 52

List of Tables

Table 1. Definitions and notations. 16

List of Figures

Figure 1. Evolution of the congestion window of TCP Tahoe and TCP Reno 6

Figure 2. Evolution of the congestion window of a SF-SACK flow 13

Figure 3. Evolution of the window size over time in the case of losses detected only by triple-duplicate acknowledgments, with a loss detected each scheduler interval. 19

Figure 4. Packets sent during a TDP. 20

Figure 5. Evolution of the window size when loss indications are only triple-duplicate acknowledgments. 27

Figure 6. Evolution of the window size in a sequence of consecutive scheduler's intervals that do not contain losses 29

Figure 7. Packet and acknowledgment transmissions towards the end of a TDP 31

Figure 8. Comparison with TCP Reno, model with no time-outs, RTT = 0.016 sec, τ = 7 * RTT 34

Figure 9. Comparison with TCP Reno, model with no time-outs, RTT = 0.024 sec, τ = 7 * RTT 34

Figure 10. Comparison with TCP Reno, model with no time-outs, RTT = 0.24 sec, τ = 7 * RTT 35

Figure 11. Evolution of the window size when loss indications are triple-duplicate acknowledgments or time-outs 36

Figure 12. Comparison with TCP Reno, RTT = 0.016 sec, T_0 = 3*RTT, τ =7*RTT 44

Figure 13. Comparison with TCP Reno, RTT = 0.024 sec, T_0 = 3*RTT, τ =7*RTT 44

Figure 14. Comparison with TCP Reno, RTT = 0.16 sec, T_0 = 3*RTT, τ =7*RTT 45

Figure 15. Influence of the value RTT on the send rate, p = 0.2, τ = 7 * RTT 45

Figure 16. Influence of τ on the send rate 46

Figure 17. Comparison between the throughput of SF-SACK and TCP Reno, RTT = 0.016 sec, T_0 = 3*RTT 49

Figure 18. Comparison between the throughput of SF-SACK and TCP Reno, RTT = 0.024 sec, T_0 = 3*RTT 49

Figure 19. Comparison between the throughput of SF-SACK and TCP Reno, RTT = 0.24 sec, T_0 = 3*RTT 50

Modeling the Throughput Performance of the SF-SACK Protocol

Laura M. Voicu

ABSTRACT

Besides the two classical techniques used to evaluate the performance of a protocol, computer simulation and experimental measurements, mathematical modeling has been used to study the performance of the TCP protocol. This technique gives an elegant way to gain insights when studying the behavior of a protocol, while providing useful information about its performance.

This thesis presents an analytical model for the SF-SACK protocol, a TCP SACK based protocol conceived to be appropriate for data and streaming applications. SF-Sack modifies the multiplicative part of the Additive Increase Multiplicative Decrease of TCP to provide good performance for data and streaming applications, while avoiding the TCP-friendliness problem of the Internet. The modeling of the SF-SACK protocol raises new challenges compared to the classical TCP modeling in two ways: first, the model needs to be adapted to a more complex dynamism of the congestion window, and second, the model needs to incorporate the scheduler that SF-SACK makes use of in order to maintain a periodically updated value of the congestion window. Presented here is a model that is progressively built in order to consider these challenges. The first step is to consider only losses detected by triple-duplicate

acknowledgments, with the restriction that one such loss happens each scheduler interval. The second step is to consider losses detected via triple-duplicate acknowledgments, while eliminating the above restriction. Finally, the third step is to include losses detected via time-outs. The result is an analytical characterization of the steady-state send rate and throughput of a SF-SACK flow as a function of the loss probability, the round-trip time (RTT), the time-out interval, and the scheduler interval.

The send rate and the throughput of SF-SACK were compared against available results for TCP Reno. The obtained graphs showed that SF-SACK presents a better performance than TCP. The analytical model of the SF-SACK follows the trends of the results that are presently available, using both the ns-2 simulator and experimental measurements.

Chapter 1

Introduction

The amount of streaming traffic over the Internet continues to grow. Many applications not available a few years ago are considered main stream today. This is the case of videoconferencing and voice, which were traditionally transported over circuit-switched networks.

The Internet, dedicated to carry traffic from data-oriented applications, has now to handle both types of traffic, while providing good performance. Unfortunately, the transport layer protocols meant to carry data and the ones designed for streaming applications do not work together very well. The cause of the unfairness that results from TCP flows competing for bandwidth with unresponsive UDP flows is the absence of an end-to-end flow and congestion control mechanism in UDP [5]. Many proposals to solve this problem have been brought into attention. One of these proposals is SF-SACK, a TCP-SACK based protocol meant to be appropriate for data and real-time applications and, at the same time, to provide flow and congestion control [1]. In this thesis, the SF-SACK protocol is evaluated.

Three important techniques are used to evaluate the performance of TCP: experimental measurements, computer simulation, and mathematical modeling.

The third method came into sight for numerous reasons. The first and maybe the most forcible reason is the extent that the use of TCP has today. Thus, any protocol that is intended not only for data, but for streaming applications also, will double this extent. This kind of magnitude has to rely on mathematical support in order to find theoretical bounds, especially when some aspects cannot be really measured or even anticipated, like the number of existent connections or the way in which the protocol responds to other transport protocols used over the Internet [7]. Mathematical models are required in order to design an optimal transport protocol, such that the performance metrics and control strategies are chosen.

The theoretical model of SF-SACK, as the one for any TCP version, needs to include two processes: the dynamics of the congestion window and the packet loss process. These two processes, observed at the sender side of the end-to-end TCP connection, are indicative of the actual traffic loads and congestion within the network [7]. The two processes are included in the model progressively. First, the model considers only losses detected by triple-duplicate acknowledgments, with the restriction that one such loss happens during each scheduler's interval. The model then incorporates losses detected via triple-duplicate acknowledgments, while eliminating the above restriction. Finally, the model includes losses detected via time-outs.

1.A. Contributions of the Thesis

Several mathematical models have been developed to analyze the original TCP and other versions of transport layer protocols. Modeling SF-SACK is different and challenging because of the complexity of processes that need to be

included in the model. First, the analysis of the congestion window dynamic is complicated by a recurrent formula that also depends on time and on the type of event that triggered the update. Second, the calculation of the congestion window does not occur only when a loss is detected, as in the case of TCP, but also, in the absence of such losses, it is done at periodic intervals, dictated by a scheduler.

The most important contribution of this thesis is to build a mathematical model that addresses these challenges while still maintaining the approach of a classical TCP model. This model constitutes an addition to the two methods that are already available for the evaluation of the SF-SACK protocol, computer simulation and experimental measuring, in order to provide a complete and powerful mean of performance evaluation of the protocol.

1.B. Organization of the Thesis

The rest of the thesis is organized as follows. Chapter Two provides basic information in the area of TCP modeling and background knowledge on Markov regenerative processes, which are used to model the send rate of SF-SACK. Chapter Three gives a presentation of the SF-SACK protocol, while Chapter Four presents the analytical characterization of the SF-SACK steady-state send rate and throughput as a function of loss rate, round trip time, and the duration of the scheduler's interval. Here are also included graphs to illustrate a comparison between the behavior of SF-SACK and versions of TCP. Chapter Five presents a brief conclusion of the thesis and grounds for future work.

Chapter 2

Related Work

The use of TCP as the prevalent transport protocol over the Internet and the continuous expansion of the Internet caused the increasing interest in modeling the TCP protocol. As a result, analytical models and performance evaluations of the most important TCP versions are available. Since SF-SACK is a TCP-based protocol, studying the classical TCP analytical models is the starting point in understanding the process of construction of the SF-SACK model.

Numerous models for TCP have been proposed so far, concentrating on different aspects of the protocol [3], [7], [10], [11], [14], [15], [16]. Some concentrate on modeling the throughput of infinite TCP connections as a function of round-trip time and packet loss rate [10], [11], [14], [15], [16], others on modeling the latency of finite connections as a function of transfer size, round-trip time, and packet loss rate [3], [15].

This chapter is divided in two sections. The first gives a brief background on TCP and presents a series of TCP models. The second section provides the theoretical background for the analytical model presented in this thesis.

2.A. TCP and TCP Modeling

Early implementations of TCP used a go-back-n model, where losses were detected only via time-outs [15]. The congestion-control mechanism of TCP introduces a variable named the congestion window that controls the rate at which a TCP sender can send data. To be more specific, the amount of unacknowledged data at the sender cannot exceed the congestion window. TCP Tahoe [8] introduces three mechanisms:

1. slow-start. When a TCP connection starts or restarts after a time-out, the congestion window is set to 1. The current window size is divided by 2 and saved as a threshold value. Then for each received acknowledgment, the congestion window is increased by one, leading to an exponential increase. The slow-start phase lasts until the congestion window reaches the threshold value.
2. congestion avoidance. When the congestion window becomes larger than the threshold value, the window size increases linearly
3. fast retransmit. After receiving three duplicate acknowledgments, the sender retransmits the missing segment before the retransmission timer expires.

TCP Reno [9] adds to TCP Tahoe the fast recovery algorithm, by canceling the slow-start phase after a triple duplicate acknowledgment. TCP SACK [4] is an extension of TCP Reno that allows out-of sequence acknowledgments. By requiring selective acknowledgments, it is intended to eliminate time-outs in the case when multiple losses happen within the same window [15].

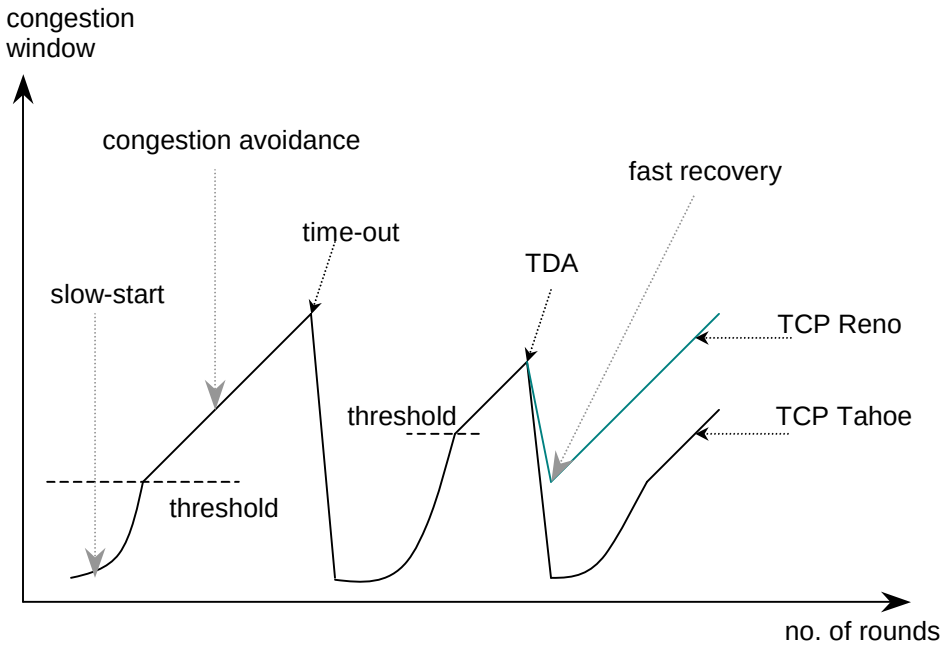

Figure 1. Evolution of the congestion window of TCP Tahoe and TCP Reno.

Figure 1 illustrates the dynamic of the congestion window for TCP Tahoe and TCP Reno and names the principal phases of a TCP flow.

TCP Vegas [2] is a modification of TCP Reno that introduces three major changes:

1. new retransmission mechanism. The RTT is estimated by measuring the time interval between the moment the segment is sent and the moment the corresponding acknowledgment arrives. If a duplicate acknowledgment is received and the difference between the current time and the timestamp for the relevant segment is larger than the time-out value, then the segment is retransmitted. Also, for the first or the second non-duplicate acknowledgment that is received after a retransmission, if the time interval since the relevant segment was sent is larger than the time-out value, then the segment is retransmitted. In addition, a further modification provides that the congestion

window is decreased only if the retransmitted segment was initially sent after the last decrease.

2. congestion avoidance mechanism. The measured send rate is compared with an expected send rate value with the purpose of controlling the amount of extra data that the connection has in transit. When the difference between the expected send rate and the actual send rate is smaller than a threshold value α, the congestion window is increased linearly during the next RTT. When the difference is larger than a threshold value β, the congestion window is decreased linearly during the next RTT.
3. modified slow-start mechanism. Congestion during slow-start is avoided by allowing exponential growth only every other RTT. In between, the congestion window remains fixed such that the actual and expected send rates are compared. When the actual rate falls below the expected rate, Vegas changes from slow-start to linear increase/decrease mode.

The analytical models for TCP proposed so far, analyze two main characteristics of the protocol: either the throughput of TCP connections with infinite amount of data to send is modeled as a function of round-trip time and packet loss rate [10], [11], [14], [15], [16], or the latency of connections with finite amount of data to send is modeled as a function of transfer size, round-trip time, and packet loss rate [3], [15].

The most well-known model that characterizes the steady-state send rate of a bulk transfer TCP flow is presented in [11]. The model is aimed at capturing both the fast retransmit and the time-out mechanisms of TCP Reno. The model

calculates the send rate and throughput of a TCP flow as functions of the loss rate and round trip time.

Analytical models for TCP Tahoe, Reno, and SACK are included in [15] to estimate both the latency and the steady-state throughput of the protocols. Based on the analytical models, the three versions of TCP are then compared.

The throughput of TCP Vegas is modeled in [14], as a function of the average round trip time, minimum round trip time, and loss rate of the transfer. The model include the slow-start, congestion avoidance and congestion recovery mechanisms.

The latency of a TCP connection is modeled in [3] as a function of the transfer size, round trip time, and packet loss rate. The model includes both the connection establishment and data transfer phases. This approach is intended to predict the performance of both short and long TCP flows under different packet loss conditions. The model extends the results in [11] by deriving new models for the connection establishment phase and the slow-start phase. Depending on the analytical model, it may or may not include certain aspects of a TCP connection, thus, its accuracy may not be complete. But, by including the essential processes, each of the existent models can be considered as a starting point for developing new models.

2.B. Markov Regenerative Processes

The send rate and the throughput of a SF-SACK flow are both computed as the reward rate of a Markov regenerative process. This section provides a

basic background on Markov processes and renewal theory. Unless otherwise specified, the following information is referenced from [6].

A stochastic process *{X(t), $t \in T$}* is a collection of random variables. The index *t* is often referred to as time, while *X(t)* is the state of the process at time *t*. The set *T* is called the index set of the process.

A Markov chain is a stochastic process with the property that, conditional on its present value, the future is independent of the past. The process X is a Markov chain if it satisfies the Markov condition:

$$P\left(X_n = s \mid X_0 = x_0, X_1 = x_1, ..., X_{n-1} = x_{n-1}\right) = P\left(X_n = s \mid X_{n-1} = x_{n-1}\right)$$

for all $n \geq 1$ and all $s, x_1, x_2, ..., x_{n-1} \in S$, where S is a countable set.

A renewal process is a recurrent-event process with independent identically distributed inter-event times. More formally, a renewal process $N = \{N(t) : t \geq 0\}$ is a process such that

$$N(t) = \max\{n : T_n \leq t\}$$

where $T_0 = 0$, $T_n = X_1 + X_2 + ... + X_n$ for $n \geq 1$, and $\{X_i\}$ is a sequence of independent identically distributed non-negative random variables. T_n is called the 'time of the n^{th} arrival' and X_n is referred to as the 'n^{th} inter-arrival time'.

In the case that there are rewards or costs associated with a renewal process, they may be introduced as follows.

Let $\{(X_i, R_i)\}_{i \geq 1}$ be independent and identically distributed pairs of random variables such that $X_i > 0$. For a pair (X, R), the quantity X is to be interpreted as an inter-arrival time of a renewal process, and the quantity R as a reward

associated with that inter-arrival time. It is not assumed that X and R are independent. The renewal process N is constructed by $N(t) = \max\{n : T_n \le t\}$, where $T_n = X_1 + X_2 + ... + X_n$, and the 'cumulative reward process' C by $C(t) = \sum_{i=1}^{N(t)} R_i$.

The reward function is $c(t) = E[C(t)]$.

The Renewal-reward theorem: Suppose that $0 < E[X] < \infty$ and $E[|R|] < \infty$. Then

$\frac{C(t)}{t} \to \frac{E[R]}{E[X]}$ as $t \to \infty$ with probability 1, and

$\frac{c(t)}{t} \to \frac{E[R]}{E[X]}$ as $t \to \infty$.

A regenerative process is a stochastic process $\{X(t), t \ge 0\}$ with state space $\{0,1,2,...\}$ having the property that there exist time points at which the process restarts itself, meaning that there exists a time T_1 such that the continuation of the process beyond T_1 is a probabilistic replica of the whole process starting at 0 [12].

The renewal-reward theorem has a major importance in the evolution of the analytical model for the SF-SACK protocol, since it allows the deduction of the send rate and of the throughput, as it will be seen in Chapter 4.

Chapter 3

The SF-SACK Protocol

The amount of streaming traffic over the Internet continues to grow. As a consequence, the Internet has now to handle both traffic from data-oriented applications and real-time applications, while providing good performance. UDP, the transport layer protocol commonly used to transfer real-time traffic, and TCP, the protocol utilized to transmit data-oriented traffic, do not work together very well. If TCP and UDP share the same congested bottleneck link, UDP may obtain considerably more bandwidth than TCP. The “TCP-unfriendliness” of UDP is a well-known problem of the Internet [5], which is becoming more and more important as the amount of real-time traffic continues to grow [1].

A solution to this problem is proposed in [1], the Smooth Fair TCP SACK-based (SF-SACK) protocol. This protocol is meant to be appropriate for real-time applications while including flow and congestion control. The modification that the protocol brings to TCP SACK is the dynamic of the congestion window, *cwnd,* when packet losses occur. The multiplicative decrease part of TCP is substituted by a smooth decrease strategy that considers the history in the evolution of the congestion window. For this purpose, a discrete time filter is used and the *cwnd* value of the congestion window at time t_k is calculated using the formula:

$$cwnd_{t_k} = \frac{\frac{2\tau}{t_k - t_{k-1}} - 1}{\frac{2\tau}{t_k - t_{k-1}} + 1} cwnd_{t_{k-1}} + \frac{1}{\frac{2\tau}{t_k - t_{k-1}} + 1} (cwnd\_sample_{t_k} + last\_cwnd\_sample_{t_{k-1}}) \quad (1)$$

where $cwnd_{t_k}$ is the filtered value of the congestion window at time t_k, $1/\tau$ is the cut-off frequency of the filter, and $t_k - t_{k-1}$ is the time interval of consecutive packet losses.

The value of $cwnd\_sample_{t_k}$ depends on the type of congestion event. If the packet loss is detected by three duplicate acknowledgements, the value is set to $cwnd/2$ and when the loss is detected via a timeout it is set to 1.

In addition, a scheduler is used to update $cwnd_{t_k}$ every $\tau/2$ seconds, since there is no guarantee that packet losses will occur with enough frequency to have a proper sampling frequency as required by the Nyquist theorem. The cut-off frequency of the filter is $1/\tau$ and according to the Nyquist sampling theorem, the sampling interval should be at most $\tau/2$.

In case the *cwnd* value is calculated as a result of a scheduler update, the value of $cwnd\_sample_{tk}$ is set to the current *cwnd.* Also, since no packets have been lost, the value of *cwnd* is not updated and the algorithm continues the additive increase process.

Figure 2 illustrates the evolution of the congestion window for SF-SACK, represented by the bold line, and provides a comparison with the way TCP works, illustrated by the dashed line. As it can be observed, the *cwnd_sample* values calculated in the case of loss detection (time-outs or triple-duplicate

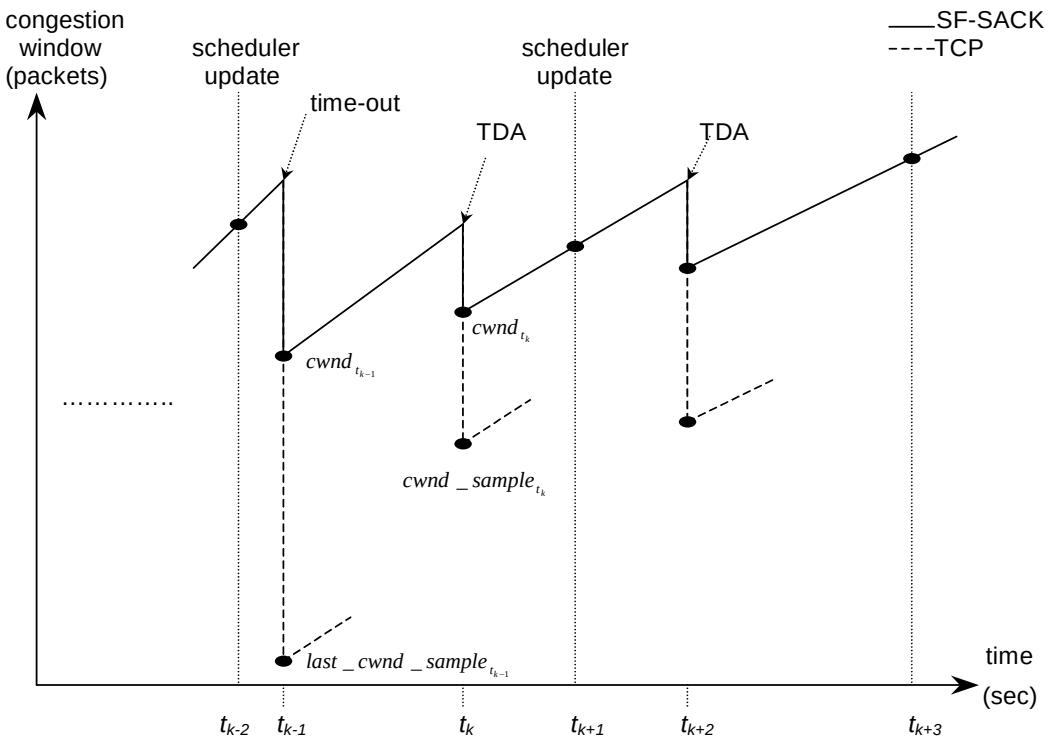

Figure 2. Evolution of the congestion window of a SF-SACK flow.

acknowledgments) are equal to the values at which TCP would reduce the congestion window.

The filter in Equation (1) is appropriate to be used for a smooth decrease congestion control algorithm since it will determine a proper weight of history and of the current samples in the evolution of the congestion window, as a reflection of the changes in the inter-arrival time between packet loss events.

Chapter 4

The Analytical Model of the SF-SACK Protocol

This chapter presents the stochastic model for SF-SACK, specifically of its congestion avoidance mechanism. The model is inspired by the model proposed in [11] and since SF-SACK is TCP based, some parts even follow the same course. The assumptions are also the same, but the skeletons of the models are different, as for the SF-SACK the dynamic of the congestion window is completely different and the processes that need to be included are much more complex. As in [11], we will have a set of assumptions as follows.

Each time an ACK is received, the congestion window, *W*, is increased by *1/W.* Each time a packet loss is detected, *W* is decreased to the value *CW*, which is calculated with formula (1).

The model uses the notion of “rounds”, which is the period of time between the beginning of transmission of the first packet from the packets that fall within the congestion window and the reception of the first ACK. This ACK marks the end of the current round and the beginning of the following round. The duration of a round is equal to the Round Trip Time (RTT) and it is assumed to be independent of the window size. It is also assumed that the time needed to send all the packets in a window is smaller than the RTT.

The send rate is defined as the number of packets sent by the sender in the time period and is measured in terms of packets per unit time instead of bytes per unit time.

Since one packet is acknowledged by an ACK, if *W* packets are sent during the first round and all are received and acknowledged correctly, then W acknowledgments will be received corresponding to this round. As the window size is increased by *1/W* after each acknowledgment, then in the absence of a loss the window size increases linearly, with a slope of *1.*

A packet loss is detected either by time-outs (TO), or by the reception of triple-duplicate acknowledgments (TD).

It is assumed that a packet loss in a round is independent of any packet loss in other rounds. Further, if a packet is lost, it is assumed that all the remaining packets in that round are also lost.

In addition to this, for SF-SACK, it is safely assumed that the scheduler's interval ($\tau / 2$) is small enough such that within an interval there will be at most one loss, detected either by a triple-duplicate acknowledgment, or by a time-out occurrence.

The mathematical model is built progressively. First, the model considers only losses detected by triple-duplicate acknowledgments, with the restriction that one such loss happens during each scheduler's interval. The model then incorporates losses detected via triple-duplicate acknowledgments, while eliminating the above restriction. Finally, the model includes losses detected via time-outs. Table 1 presents the definitions and notations used within this chapter.

Table 1. Definitions and notations.

A_i	the duration of the period between two consecutive loss indications
A_{ij}	the duration of the j[th] period of the interval Z_i^{TD}
$A(w,k)$	the probability that the first k packets are acknowledged in a round of w packets, assuming that there is a loss in the round
B	the long-term steady rate of a connection
CS_i	the value calculated for the congestion window size after the i[th] time the algorithm is run by the scheduler
CW_i	the value of the congestion window immediately after the packet loss is detected
H_i	$Z_i^{TDS} + Z_i^{TO}$
m_i	the number of rounds in Z_i^{SC} (sequence of consecutive scheduler's intervals that do not contain any loss detection)
M_i	the number of packets sent during TS_i
M_{ij}	the number of packets sent in the j[th] TDP of interval Z_i^{TDS} (the time interval between two consecutive time-out sequences)
n_i	the number of TDP intervals in Z_i^{TD}
N_i	the number of packets sent during H_i
nr_i	the number of TS_{ij} periods in the interval Z_i^{TDS} (the time interval between two consecutive time-out sequences)
NR_i	the total number of packets sent in Z_i^{TO} (sequence of time-outs)
nto_i	the number of time-out intervals in Z_i^{TO} (sequence of time-outs)
NW_{ij}	the window size at the end of the j[th] TDP of interval Z_i^{TDS} (the time interval between two consecutive time-out sequences)
NR_i'	the number of packets that reach the destination during Z_i^{TO} (sequence of time-outs)
p	the probability that a packet is lost
q_i	the round number within TDP_i that corresponds to the scheduler update
Q	the probability that a window size update at the end of a TDP is a scheduler update
Q'	the probability that there is a loss detected by a triple-duplicate acknowledgment
$\hat{Q}(w)$	the probability that there is a loss in the penultimate round and that this loss is detected by a triple-duplicate acknowledgment
r_{ij}	the length of the j[th] round in TDP_i

Table 1. Continued.

r_{ij}^{SC}	the duration of a round in Z_i^{SC} (sequence of consecutive scheduler's intervals that do not contain any loss detection)
RTT	round trip time = the duration of a round
s_i	the number of scheduler intervals in Z_i^{SC} (sequence of consecutive scheduler's intervals that do not contain any loss detection)
S_i	the updated value of the window size after the i^{th} time the algorithm is run by the scheduler
S_{ij}	the duration of the j^{th} TDP of interval Z_i^{TDS} (the time interval between two consecutive time-out sequences)
SC_i	the number of packets sent during the period Z_i^{SC} (sequence of consecutive scheduler's intervals that do not contain any loss detection)
SC_i'	the number of packets received by the receiver during Z_i^{SC} (sequence of consecutive scheduler's intervals that do not contain any loss detection)
T_0	the time-out interval
T_i	the period between a time that the scheduler is run and the first loss that occurs after that
$T(n,m)$	the probability that only the first *m* packets from the total number of *n* are acknowledged in the last round
TDP_i	the period between two consecutive losses detected by triple-duplicate acknowledgments
TS_i	$Z_i^{TD} + Z_i^{SC}$
TS_{ik}	the duration of a sequence of consecutive scheduler's intervals that do not contain any loss detection plus the time interval between two such consecutive sequences
TW_{ij}	the number of packets sent during a time-out interval
W_i	the congestion window size at the end of a TDP
W_{ij}	congestion window at the end of the j^{th} period of the interval Z_i^{TD}
W_{i0}	congestion window at the beginning of the period Z_i^{SC} (sequence of consecutive scheduler's intervals that do not contain any loss detection)
X	the probability that a loss indication at the end of a TDP is a time-out
X_i	the round where the first loss occurs in TDP_i
$\hat{X}(w)$	the probability that a loss in a window of *w* is determined via a time-out
Y_i	the number of packets sent in TDP_i

Table 1. Continued.

Y_{ij}	the number of packets sent in the j[th] period of the interval Z_i^{TD}
Y_i'	the number of packets that reach the destination during a TDP period
Z_i^{SC}	the duration of a sequence of consecutive scheduler's intervals that do not contain any loss detection
Z_i^{TD}	the time interval between two consecutive sequences Z_i^{SC}
Z_i^{TDS}	the time interval between two consecutive time-out sequences
Z_i^{TO}	the duration of a sequence of time-outs
α_i	the first packet lost in TDP_i
β_i	the number of packets sent in the last round of TDP_i
$\tau / 2$	the duration of a scheduler's interval

4.A. Losses are Detected Exclusively by Triple-Duplicate Acknowledgments and There is One Loss in Each Scheduler's Interval

In this section it is assumed that loss indications are exclusively due to triple-duplicate acknowledgments and that the window size is not limited by the receiver's advertised window. It is also assumed that one and only one loss is detected within a scheduler's interval, that is $0 < A_i < \tau$, where A_i is the duration of the period between two loss indications. Figure 3 shows the evolution of the window size in this case. After the i[th] time the algorithm is run by the scheduler, the updated value of the window size using Equation (1) is noted by S_i and the value obtained for the congestion window size is noted by CS_i. In Equation (1) $cwnd_{t_k}$ (and also $cwnd_{t_{k-1}}$) is an element from the set $\{CW_i\}_i$ if it corresponds to an update triggered by a loss detection or from the set $\{CS_i\}_i$ if it corresponds to a calculation initiated by the scheduler. $cwnd\_sample_{t_k}$ (and also

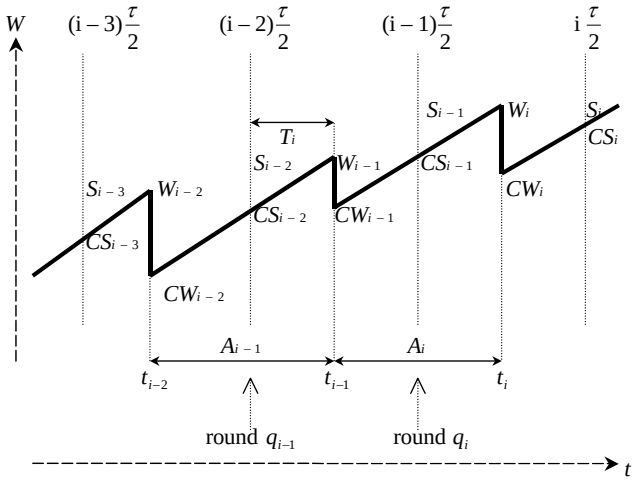

Figure 3. Evolution of the window size over time in the case of losses detected only by triple-duplicate acknowledgments, with a loss detected each scheduler interval.

$last\_cwnd\_sample_{t_{k-1}}$), depending on the kind of update for the congestion window size, is set to a value from $\{S_i\}_i$ if the update is initiated by the scheduler, or from $\left\{\frac{W_i}{2}\right\}_i$ if the update is due to a lost detected by a triple-duplicate acknowledgment, or is set to 1 if the update is due to a lost detected by a time-out. Let us also denote with q_i the round number within TDP_i that corresponds to the scheduler update.

When defining the send rate, the considerations found in [11] can also be applied to SF-SACK, as follows. For a given time $t > 0$, let N_t be the number of packets transmitted in the interval [0, t]. The send rate in that interval is then $B_t = N_t/t$. Thus, the long-term steady rate of a connection is $B = \lim_{t\to\infty} B_t = \lim_{t\to\infty} \frac{N_t}{t}$.

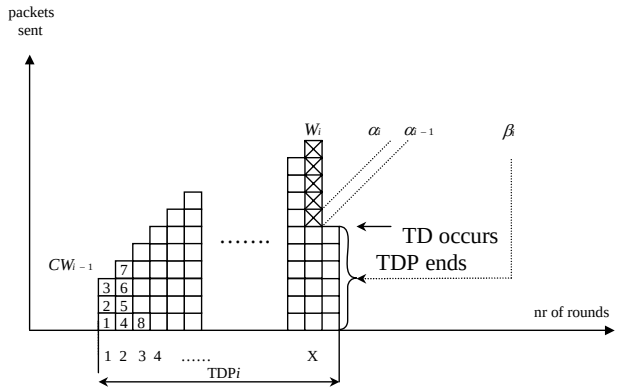

Figure 4. Packets sent during a TDP.

Let TDP (TD period) be a period between two TD loss indications. Then the duration of a TDP is A_i and the window size at the end of the period is W_i. Let Y_i be the number of packets sent in the period. Considering $\{W_i\}_i$ to be a Markov regenerative process with rewards $\{Y_i\}_i$, then, as a consequence of the Renewal-reward theorem, it follows that

$$B = \frac{E[Y]}{E[A]} \tag{2}$$

Derivation of the means of *Y* and *A* will also follow closely the model found in [11]. Therefore, let us consider a TDP, as illustrated in Figure 4. The initial window size is CW_{i-1}. The number of packets sent per round is incremented by one every round. Let α_i be the first packet lost in TDP$_i$, and X_i be the round where this loss occurs. Let β_i be the number of packets sent in the last round in TDP$_i$. Before the TD indication occurs, W_i - 1 more packets are sent after packet α_i. Thus, $Y_i = \alpha_i + W_i$ - 1 and from this it follows that

$$E[Y] = E[\alpha] + E[W] - 1. \tag{3}$$

Since packet losses in a round do not depend on losses in other rounds, $\{\alpha_i\}_i$ is a sequence of independent and identically distributed random variables. The probability that $\alpha_i = k$ is equal to the probability that k-1 packets are acknowledged before a loss occurs and is equal to

$$P[\alpha_i = k] = (1-p)^{k-1} p \qquad k=1, 2,\ldots \tag{4}$$

where p is the probability that a packet is lost. It follows that the mean of α is

$$E[\alpha] = \sum_{k=1}^{\infty} k(1-p)^{k-1} p = \frac{1}{p} \tag{5}$$

From (3) and (5) it follows that

$$E[Y] = \frac{1-p}{p} + E[W] \tag{6}$$

Let r_{ij} be the duration of the j^{th} round in TDP_i. Thus,

$$A_i = \sum_{j=1}^{X_i+1} r_{ij}$$

and since r_{ij} are independent and identically distributed random variables, it follows that

$$E[A] = (E[X]+1) \cdot E[r]$$

where $E[r]$ is the average value of the round-trip time, and will be noted by RTT. Thus,

$$E[A] = RTT \cdot (E[X]+1) \tag{7}$$

There are still needed the calculations of $E[X]$ and $E[W]$. From Figure 4 it can be seen that, since the increase of the window size is linear with slope 1,

from CW_{i-1} to W_i, it follows that

$$W_i = CW_{i-1} + X_i \tag{8}$$

During TDP$_i$ the number of packets sent, Y_i, is

$$Y_i = \sum_{k=0}^{X_i-1}(CW_{i-1}+k)+\beta_i = X_i \cdot CW_{i-1} + \frac{X_i}{2}\cdot(X_i-1)+\beta_i ,$$

where β_i is the number of packets sent in the last round. By using (8) it can be obtained that:

$$Y_i = X_i \cdot CW_{i-1} + \frac{X_i}{2}(W_i - CW_{i-1} - 1)+\beta_i = \frac{X_i}{2}(CW_{i-1}+W_i-1)+\beta_i$$

and it follows that

$$E[Y] = \frac{E[X]}{2}\cdot(E[CW]+E[W]-1)+E[\beta] \tag{9}$$

For simplicity it is assumed that β_i is uniformly distributed between 1 and $W_i - 1$,

thus $E[\beta] = \frac{E[W]}{2}$ (10)

From (6), (9), and (10) it follows that

$$\frac{1-p}{p} + E[W] = \frac{E[X]}{2}(E[CW]+E[W]-1)+\frac{E[W]}{2} \tag{11}$$

But, from (8),

$$E[X] = E[W] - E[CW] \tag{12}$$

and it follows that

$$\frac{1-p}{p} + E[W] = \frac{1}{2}(E[W]-E[CW])(E[CW]+E[W]-1)+\frac{E[W]}{2} \tag{13}$$

The difference between [11] and the SF-SACK model will intervene in the way *E[W]* and *E[CW]* are deduced. From the evolution of the window size shown in Figure 3 and by using Equation (1) (Chapter 3), it can be deduced that

$$CW_{i-1} = \frac{\frac{2\tau}{t_{i-1} - (i-2)\frac{\tau}{2}} - 1}{\frac{2\tau}{t_{i-1} - (i-2)\frac{\tau}{2}} + 1} \cdot CS_{i-2} + \frac{1}{\frac{2\tau}{t_{i-1} - (i-2)\frac{\tau}{2}} + 1} \cdot \left(\frac{W_{i-1}}{2} + S_{i-2} \right) \quad (14)$$

where

$$S_{i-2} = CW_{i-2} + q_{i-1} \quad (15)$$

and

$$CS_{i-2} = \frac{\frac{2\tau}{(i-2)\cdot\frac{\tau}{2} - t_{i-2}} - 1}{\frac{2\tau}{(i-2)\cdot\frac{\tau}{2} - t_{i-2}} + 1} \cdot CW_{i-2} + \frac{1}{\frac{2\tau}{(i-2)\cdot\frac{\tau}{2} - t_{i-2}} + 1} \cdot \left(S_{i-2} + \frac{W_{i-2}}{2} \right) \quad (16)$$

Let us consider the notation $T_i = t_{i-1} - (i-2)\frac{\tau}{2}$.

It follows that $(i-2)\frac{\tau}{2} - t_{i-2} = A_{i-1} - T_i$. Equations (14) and (16) will then become

$$CW_{i-1} = \frac{\frac{2\tau}{T_i} - 1}{\frac{2\tau}{T_i} + 1} \cdot CS_{i-2} + \frac{1}{\frac{2\tau}{T_i} + 1} \cdot \left(\frac{W_{i-1}}{2} + S_{i-2} \right) \quad (17)$$

$$CS_{i-2} = \frac{\frac{2\tau}{A_{i-1} - T_i} - 1}{\frac{2\tau}{A_{i-1} - T_i} + 1} \cdot CW_{i-2} + \frac{1}{\frac{2\tau}{A_{i-1} - T_i} + 1} \cdot \left(S_{i-2} + \frac{W_{i-2}}{2} \right) \quad (18)$$

T_i denotes the duration of the time interval between the moment the scheduler is run and the packet loss that follows after this update (it is assumed that there will be one loss before the following scheduler update). It can be assumed that $\{T_i\}_i$ is uniformly distributed between 0 and $\frac{\tau}{2}$, thus its probability density function is given by $\frac{1}{\frac{\tau}{2}}$, and since T_i is a continuous random variable it follows that

$$E[T] = \int_0^{\frac{\tau}{2}} x \cdot \frac{1}{\frac{\tau}{2}} dx = \frac{2}{\tau} \cdot \int_0^{\frac{\tau}{2}} x dx = \frac{2}{\tau} \cdot \frac{x^2}{2} \Bigg|_0^{\frac{\tau}{2}} = \frac{2}{\tau} \cdot \frac{\tau^2}{8} = \frac{\tau}{4}$$

Thus $E[T] = \frac{\tau}{4}$ (19)

With the assumption that A_i is uniformly distributed between 1 and τ -1, it results

$$E[A] = \int_0^{\tau} x \cdot \frac{1}{\tau} dx = \frac{1}{\tau} \cdot \int_0^{\tau} x dx = \frac{1}{\tau} \cdot \frac{x^2}{2} \Bigg|_0^{\tau} = \frac{\tau}{2} \quad (20)$$

From (19), (20), and (17) it can be deduced that

$$E[CW] = \frac{\frac{8\tau}{\tau} - 1}{\frac{8\tau}{\tau} + 1} \cdot E[CS] + \frac{1}{\frac{8\tau}{\tau} + 1} \cdot \left(\frac{E[W]}{2} + E[S] \right),$$

which after simplifications becomes

$$E[CW] = \frac{7}{9} \cdot E[CS] + \frac{1}{9} \cdot \left(\frac{E[W]}{2} + E[S] \right) \quad (21)$$

Since $E[q] = \frac{E[A] - E[T]}{RTT}$, it follows from (15), using (19) and (20) that

$$E[S] = E[CW] + \frac{\tau}{4RTT} \tag{22}$$

From (18), (19), and (20) it results that

$$E[CS] = \frac{7}{9} \cdot E[CW] + \frac{1}{9} \cdot \left(E[S] + \frac{E[W]}{2} \right) \tag{23}$$

By using the Equations (21), (22), and (23) the following equation for $E[CW]$ is obtained:

$$\begin{aligned} E[CW] &= \frac{7}{9} \cdot \left[\frac{7}{9} \cdot E[CW] + \frac{1}{9} \cdot \left(E[CW] + \frac{\tau}{4RTT} + \frac{E[W]}{2} \right) \right] \\ &+ \frac{1}{9} \cdot \left(\frac{E[W]}{2} + E[CW] + \frac{\tau}{4RTT} \right) \end{aligned} \tag{24}$$

From (7), (11), (12), and (20) it follows that

$$E[W] = \frac{p \cdot (\tau^2 - 2 \cdot \tau \cdot RTT - 8 \cdot RTT^2) + 8 \cdot RTT^2}{2 \cdot p \cdot RTT \cdot (\tau - 3 \cdot RTT)} \tag{25}$$

and $$E[CW] = \frac{p \cdot (3 \cdot \tau - 14 \cdot RTT) + 8 \cdot RTT}{2 \cdot p \cdot (\tau - 3 \cdot RTT)} \tag{26}$$

From (24), (25), and (26) it can deduced

$$p = \frac{128 \cdot RTT^2}{23 \cdot \tau^2 - 167 \cdot \tau \cdot RTT + 320 \cdot RTT^2} \tag{27}$$

In order to have satisfied the condition $p \leq 1$, it is assumed that $\tau \geq 6RTT$.

From (25) and (27) it results that

$$E[W] = \frac{39 \cdot \tau^2 - 199 \cdot \tau \cdot RTT + 192 \cdot RTT^2}{32 \cdot RTT \cdot (\tau - 3 \cdot RTT)} \tag{28}$$

From (6), (25), and (27) the expression of $E[Y]$ can be deduced:

$$E[Y] = \frac{23 \cdot \tau^3 - 80 \cdot \tau^2 \cdot RTT - 103 \cdot \tau \cdot RTT^2 + 192 \cdot RTT^3}{128 \cdot RTT^2 (\tau - 3 \cdot RTT)} \quad (29)$$

From (2), (20), and (28) it follows that

$$B = \frac{23 \cdot \tau^3 - 80 \cdot \tau^2 \cdot RTT - 103 \cdot \tau \cdot RTT^2 + 192 \cdot RTT^3}{64 \cdot \tau \cdot RTT^2 \cdot (\tau - 3 \cdot RTT)} \quad (30)$$

4.B. Losses are Detected Exclusively by Triple-Duplicate Acknowledgments

In this section it is also assumed that loss indications are exclusively due to triple-duplicate acknowledgments, but it extends section 4.A. by relaxing the supposition that a loss is detected during each scheduler's interval. Figure 5 presents the evolution of the window size in this case.

Let Z_i^{SC} denote the duration of a sequence of consecutive scheduler's intervals that do not contain any loss detection and let Z_i^{TD} be the time interval between two consecutive sequences Z_i^{SC}. Then, TS_i is defined as the sum of these two intervals, or $TS_i = Z_i^{TD} + Z_i^{SC}$. Also, let M_i be the number of packets sent during TS_i.

Then $\{(TS_i, M_i)\}_i$ is an i.i.d. sequence of random variables and

$$B = \frac{E[M]}{E[TS]}$$

The definition of a TDP given in the previous section is extended to denote either the period between two consecutive losses detected by triple-duplicate acknowledgments, or a period that begins or ends in an update initiated by the

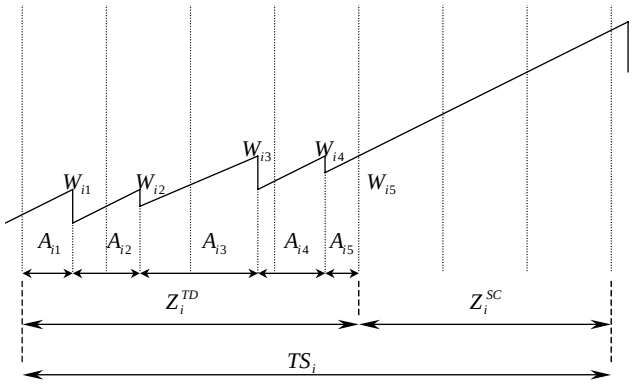

Figure 5. Evolution of the window size when loss indications are only triple-duplicate acknowledgments.

scheduler, update that is not preceded, respectively followed by a loss within the same Z_i^{TD} period.

Let n_i be the number of TDP intervals in Z_i^{TD}. For the j^{th} TDP of interval Z_i^{TD}. Y_{ij} is defined as being the number of packets sent in the period, A_{ij} to be the duration of the period, and W_{ij} to be the window size at the end of the period. Let SC_i be the number of packets sent during the period Z_i^{SC}. Then,

$$E[M] = E\left[\sum_{j=1}^{n_i} Y_{ij}\right] + E[SC]$$

$$E[TS] = E\left[\sum_{j=1}^{n_i} A_{ij}\right] + E\left[Z^{SC}\right]$$

If $\{n_i\}_i$ is assumed to be an i.i.d. sequence of random variables, independent of $\{Y_{ij}\}$ and of $\{A_{ij}\}$, then

$$E\left[\sum_{j=1}^{n_i} Y_{ij}\right] = E[n] \cdot E[Y]$$

$E\left[\sum_{j=1}^{n_i} A_{ij}\right] = E[n] \cdot E[A]$ and, thus

$$E[M] = E[n] \cdot E[Y] + E[SC]$$

$$E[TS] = E[n] \cdot E[A] + E[Z^{SC}]$$

which leads to the equation

$$B = \frac{E[n] \cdot E[Y] + E[SC]}{E[n] \cdot E[A] + E\left[Z^{SC}\right]}$$

To derive *E*[*n*], note that during Z_i^{TD} there are n_i TDP's, where each of the first $n_i - 1$ ends in a triple-duplicate acknowledgments and the last one ends in a scheduler update. This means that in Z_i^{TD}, out of n_i window updates at the end of a TDP, there is one initiated by the scheduler. If Q is the probability that a window size update at the end of a TDP is a scheduler update, then $Q = \frac{1}{E[n]}$ and it follows that

$$E[M] = E[Y] \cdot \frac{1}{Q} + E[SC] \quad (31)$$

$$E[TS] = E[A] \cdot \frac{1}{Q} + E[Z^{SC}] \quad (32)$$

As a consequence, the send rate can be expressed as

$$B = \frac{E[Y] + Q \cdot E[SC]}{E[A] + Q \cdot E\left[Z^{SC}\right]} \quad (33)$$

$E[A]$ and $E[Y]$ are the same as those obtained in Section 4.A., given by Equations (20) and (29).

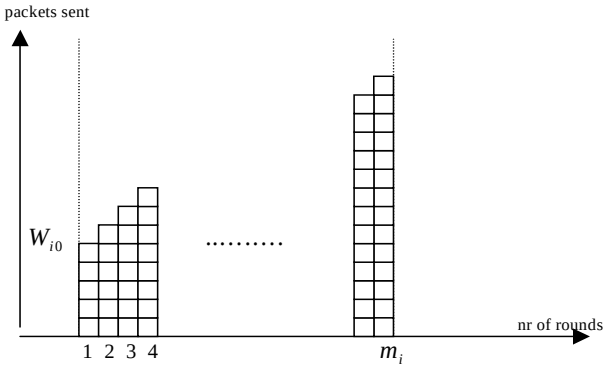

Figure 6. Evolution of the window size in a sequence of consecutive scheduler's intervals that do not contain losses.

There are needed now the derivations of $E[SC]$, $E[Z^{SC}]$, and *Q*. Figure 6 presents the evolution of the window size during Z_i^{SC}. Let us denote by W_{i0} the window size at the beginning of the period and by m_i the number of rounds in Z_i^{SC}.

It can be deduced then that $SC_i = \sum_{k=0}^{m_i-1}(W_{i0}+k) = m_i \cdot W_{i0} + \frac{m_i \cdot (m_i - 1)}{2}$. Thus

$$E[SC] = E[m] \cdot \left(E[W] + \frac{E[m]-1}{2} \right) \tag{34}$$

$E[W]$ was already calculated in Section 4.A and its expression is given by Equation (28). If r_{ij}^{SC} is the duration of a round in Z_i^{SC}, then

$E[Z^{SC}] = E[r^{SC}] \cdot E[m] = RTT \cdot E[m]$, thus

$$E[m] = \frac{E[Z^{SC}]}{RTT} \tag{35}$$

Let s_i be the number of scheduler intervals in Z_i^{SC}. Then

$$Z_i^{SC} = s_i \cdot \frac{\tau}{2} \Rightarrow E[Z^{SC}] = E[s] \cdot \frac{\tau}{2}$$

To derive *E*[*s*], note that during Z_i^{SC} there are s_i scheduler intervals, where each of the last $s_i - 1$ follow an update of the congestion window initiated by the scheduler and the first one follow an update caused by a triple-duplicate acknowledgment. Since Q is the probability that a window size update at the end of a TDP is a scheduler update, then $1 - Q = \frac{1}{E[s]}$ and it follows that

$E[s] = \frac{1}{1-Q}$, thus

$$E[Z^{SC}] = \frac{\tau}{2(1-Q)} \tag{36}$$

From (28), (34), (36), and (36) it results that

$$E[SC] = \frac{\tau}{2 \cdot RTT \cdot (1-Q)} \cdot \left[E[W] + \frac{\tau - 2 \cdot RTT \cdot (1-Q)}{4 \cdot RTT \cdot (1-Q)} \right] \tag{37}$$

where

$$E[W] = \frac{39 \cdot \tau^2 - 199 \cdot \tau \cdot RTT + 192 \cdot RTT^2}{32 \cdot RTT \cdot (\tau - 3 \cdot RTT)}$$

It will be considered now for derivation *Q*, the probability that a window size update at the end of a TDP is a scheduler update. This is similar to the correspondent derivation found in [11]. In Figure 7, the “penultimate round” is the round where a loss indication occurs. If the current congestion window size is *w*, then packets $p_1, p_2, ..., p_w$ are sent in the penultimate round.

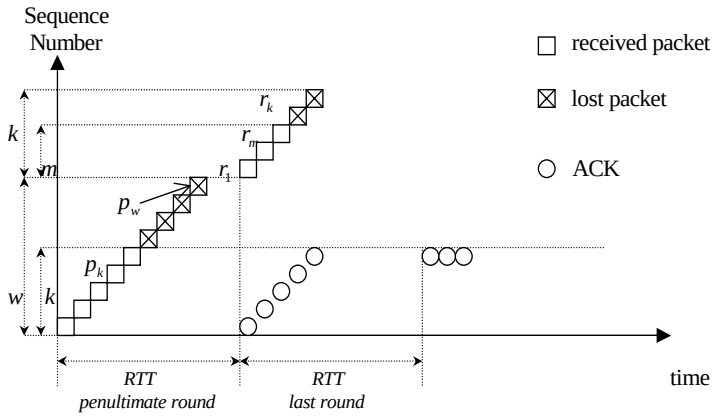

Figure 7. Packet and acknowledgment transmissions towards the end of a TDP.

Packets $p_1,...,p_k$ are acknowledged, and packet p_{k+1} is the first one that is lost or not acknowledged, followed by $p_{k+2},...,p_w$, considering the assumption that if a packet is lost, all the remaining packets in that round are also lost. Since packets $p_1,...,p_k$ are acknowledged, other *k* packets, $r_1, r_2,..., r_k$ are sent in the "last round". This last round may have another lost, packet r_{m+1}, followed by packets $r_{m+2},...,r_k$. Each of the *m* packets that are successfully sent in the last round will cause an ACK for the first packet that was lost, p_{k+1}, thus will cause duplicate acknowledgments. As a consequence, the number of duplicate acknowledgments is equal to the number of successfully received packets in the last round. Since our model does not include yet time-outs, if the number of such duplicate ACKs is less than 3, then a TD was not detected yet, thus the update of the congestion window is initiated by the scheduler.

Let $\hat{Q}(w)$ be the probability that there is a loss in the penultimate round and that this loss is detected by a triple-duplicate acknowledgment. Let $A(w,k)$ denote the probability that the first *k* packets are acknowledged in a round of *w* packets, assuming that there is a loss in the round. Then $A(w,k)$ is expressed as a conditioned probability; as the probability that the first k packets are acknowledged, divided by the probability that there is at least one loss in the round.

$A(w,k) = \frac{(1-p)^k p}{1-(1-p)^w}$ and

$$\hat{Q}(w) = \begin{cases} 0, & if \quad w \le 3 \\ \sum_{k=3}^{w-1} A(w,k) \cdot (1-p)^3, & otherwise \end{cases}$$

But

$$\sum_{k=3}^{w-1} A(w,k) \cdot (1-p)^3 = \sum_{k=3}^{w-1} \frac{p(1-p)^3 (1-p)^k}{1-(1-p)^w} = \frac{p(1-p)^3}{1-(1-p)^w} \sum_{k=3}^{w-1} (1-p)^k$$
$$= \frac{p(1-p)^3}{1-(1-p)^w} \cdot (1-p)^3 \cdot \frac{1-(1-p)^{w-3}}{1-(1-p)} = \frac{(1-p)^6 \cdot \left[1-(1-p)^{w-3}\right]}{1-(1-p)^w}$$

And it follows that

$$\hat{Q}(w) = \max\left(0, \frac{(1-p)^6 \cdot \left[1-(1-p)^{w-3}\right]}{1-(1-p)^w}\right)$$

Since $\lim_{p \to 0} \frac{(1-p)^6 \cdot \left[1-(1-p)^{w-3}\right]}{1-(1-p)^w} = \frac{w-3}{w}$, the following approximation can be used:

$$\hat{Q}(w) = \max\left(0, \frac{w-3}{w}\right) = \max\left(0, 1-\frac{3}{w}\right)$$

Q', the probability that there is a loss detected by a TD is then approximated with $Q' = \hat{Q}(E[W])$. It follows that

$$Q = 1 - Q' = 1 - \max\left(0{,}1 - \frac{3}{E[W]}\right) = \min\left(1, \frac{3}{E[W]}\right),$$

where $E[W]$ is given by Equation (26). Equation (34) can be written now as

$$B = \frac{2(1-Q)}{\tau} \cdot \left\{ \frac{1-p}{p} + E[W] + \frac{Q\tau}{2RTT(1-Q)} \left[E[W] + \frac{\tau - 2RTT(1-Q)}{4RTT(1-Q)} \right] \right\} \quad (38)$$

where $E[W] = \dfrac{39 \cdot \tau^2 - 199 \cdot \tau \cdot RTT + 192 \cdot RTT^2}{32 \cdot RTT \cdot (\tau - 3 \cdot RTT)}$

Figures 8, 9, and 10 present comparisons between the analytical characterization of the send rate of SF-SACK and of TCP Reno, using the results in [11], results given by the formula

$$B_{\text{Re}\,no} = \frac{\frac{1}{p} + \sqrt{\frac{8}{3p} - \frac{5}{3}}}{RTT\left(\frac{3}{2} + \sqrt{\frac{2}{3p} - \frac{5}{12}}\right)} .$$

The graphs only consider partial results, of the models that do not include yet the case of losses detected by time-outs. A first observation would be that both models suffer from inaccuracies, since time-outs have a significant influence over the performance of the protocols. This can be observed from the trend that the send rate has when the loss rate approaches 1. A second observation is that, as expected, the performance of SF-SACK is superior to the one of TCP.

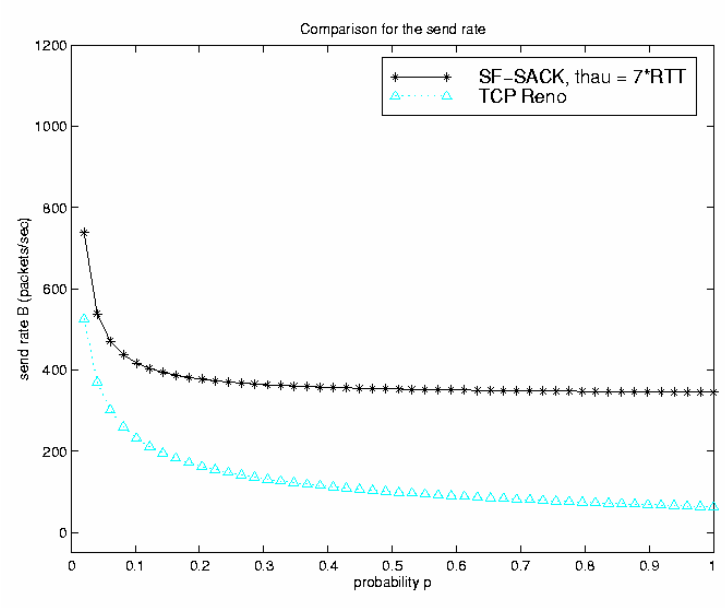

Figure 8. Comparison with TCP Reno, model with no time-outs, RTT = 0.016 sec, τ = 7 * RTT.

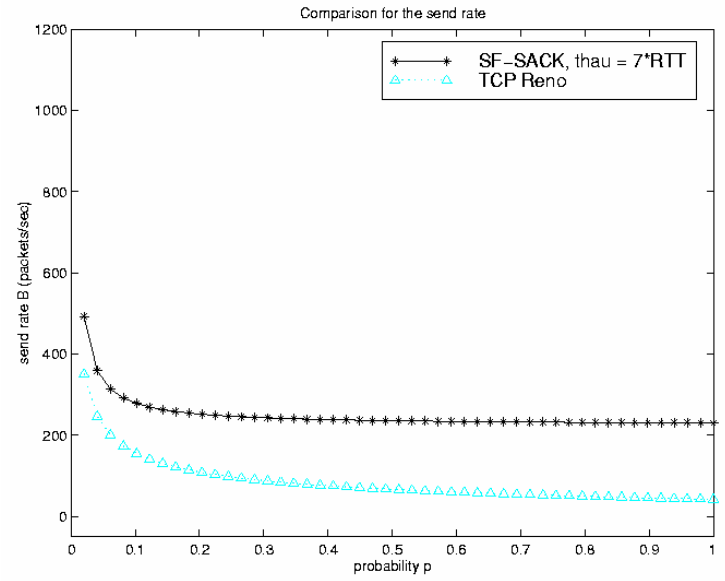

Figure 9. Comparison with TCP Reno, model with no time-outs, RTT = 0.024 sec, τ = 7 * RTT.

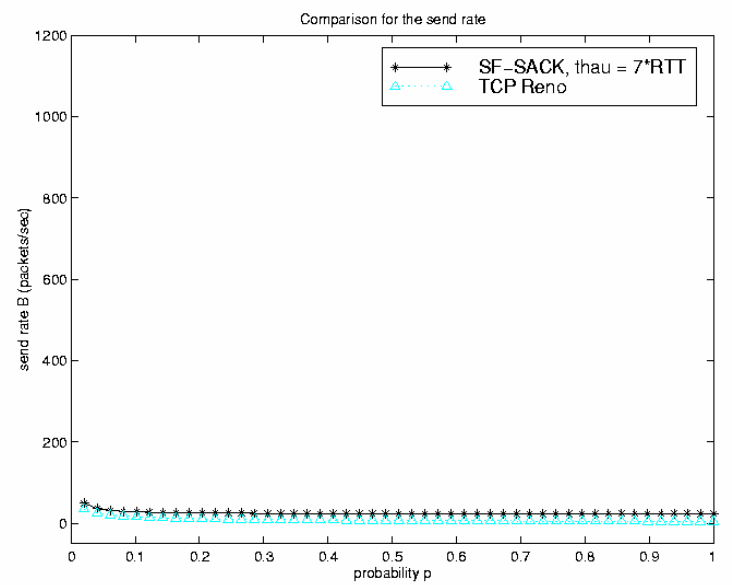

Figure 10. Comparison with TCP Reno, model with no time-outs, RTT = 0.24 sec, τ = 7 * RTT.

4.C. Losses are Detected by Triple-Duplicate Acknowledgments or by Time-Outs

This section extends the analytical model, by including the case when the loss indication is a time-out. This case occurs when packets or acknowledgments are lost and less than three-duplicate acknowledgments are received. Since in this case the scheduler only calculates the congestion window without updating it, for the simplicity of the model, the scheduler will not be included. By doing this, the interval between two consecutive calculations of the congestion window will be actually larger than in reality. This increase in the inter-arrival time between two calculations will cause the filter to weight the current samples more than the history, leading to a slightly smaller send rate. Since the main purpose of this model is to prove a better send rate / throughput for SF-SACK than for TCP, this

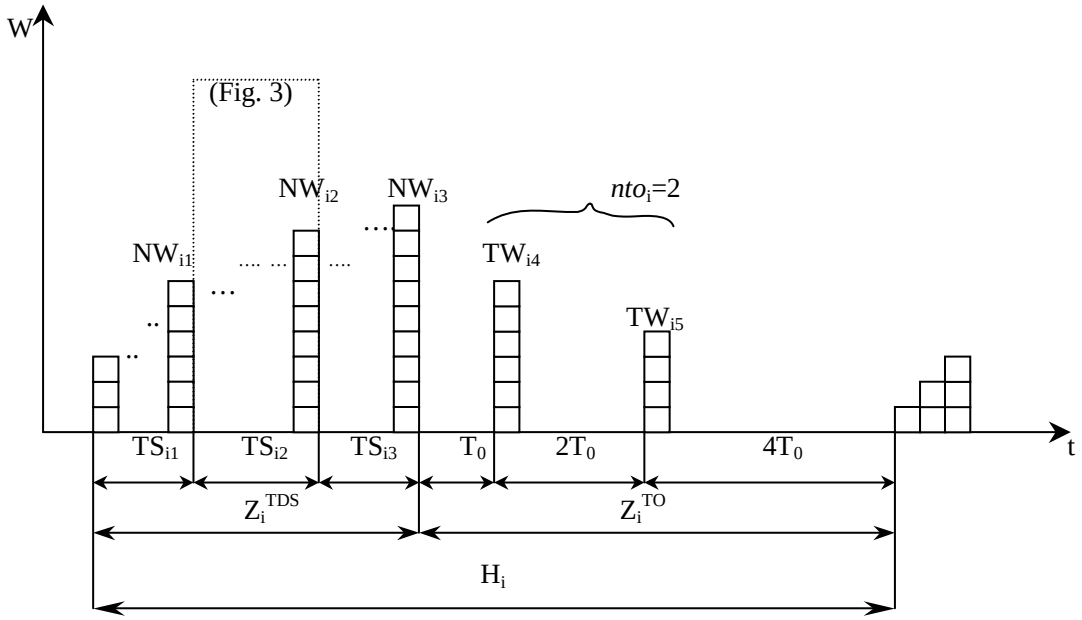

Figure 11. Evolution of the window size when loss indications are triple-duplicate acknowledgments or time-outs.

effect will be accepted, as even with this limitation, the scope of the model will be achieved.

Figure 11 presents an example of the evolution of the congestion window size in this case. The derivations in this section are fairly close to the ones found in [11]. The sender waits for a period of time T_0, then retransmits packets, beginning with the first unacknowledged packet. In the case that another time-out occurs before retransmitting the packets lost during the first time-out, the time-out interval doubles to $2T_0$. This doubling of the time-out interval repeats for each unsuccessful retransmission until it reaches $64T_0$. After this, the time-out interval remains constant, at $64T_0$. TS_{ik} was defined in section 4.B. and is illustrated in Figure 5. It represents the duration of a sequence of consecutive scheduler's

intervals that do not contain any loss detection plus the time interval between two such consecutive sequences. Let Z_i^{TO} be the duration of a sequence of time-outs and Z_i^{TDS} the time interval between two consecutive time-out sequences. Then let H_i be the sum of these two intervals, $H_i = Z_i^{TDS} + Z_i^{TO}$. If N_i is the number of packets sent during H_i, then $\{(H_i, N_i)\}_i$ is an i.i.d. sequence of random variables and

$$B = \frac{E[N]}{E[H]}$$

The definition of a TDP given in the previous section is again extended, to also include periods starting after or ending in a time-out loss indication. Let nr_i be the number of TS_{ij} periods in the interval Z_i^{TDS}. For TS_{ij}, let M_{ij} be the number of packets sent in the period, S_{ij} be the duration of the period, and NW_{ij} the window size at the end of the period. Let nto_i denote the number of time-out intervals in Z_i^{TO}, TW_{ij} be the number of packets sent during a time-out interval, and NR_i the total number of packets sent in Z_i^{TO}.

$$N_i = \sum_{j=1}^{nr_i} M_{ij} + NR_i \quad \Rightarrow E[N] = E\left[\sum_{j=1}^{nr_i} M_{ij}\right] + E[NR]$$

$$H_i = \sum_{j=1}^{nr_i} TS_{ij} + Z_i^{TO} \quad \Rightarrow E[H] = E\left[\sum_{j=1}^{nr_i} TS_{ij}\right] + E[Z^{TO}]$$

With the assumption that $\{nr_i\}_i$ is an i.i.d. sequence of random variables, independent of M_{ij} and TS_{ij}, it follows that

$$E\left[\sum_{j=1}^{nr_i} M_{ij}\right] = E[nr]\cdot E[M]$$

$E\left[\sum_{j=1}^{nr_i} TS_{ij}\right] = E[nr]\cdot E[TS]$ and, thus

$$B = \frac{E[nr]\cdot E[M]+E[NR]}{E[nr]\cdot E[TS]+E[Z^{TO}]}$$

where $E[M]$ and $E[TS]$ are the same as those derived in Section 4. B. To derive $E[nr]$, note that during Z_i^{TDS} there are nr_i TS_{ij} subintervals, where each of the first $nr_i - 1$ ends in a triple-duplicate acknowledgment and the last one ends in a time-out. It follows that in Z_i^{TDS} there is one time-out out of nr_i loss indications. If X is the probability that a loss indication at the end of a TDP is a time-out, then $X = \frac{1}{E[nr]}$ and it follows that

$$B = \frac{E[M]+X\cdot E[NR]}{E[TS]+X\cdot E[Z^{TO}]} \qquad (39)$$

The following entities need now to be calculated: $E[NR]$, $E[Z^{TO}]$, and X.

$$E[NR] = \sum_{k=1}^{\infty} k\cdot P[NR=k]$$

For *NR=k* to be true, it is needed that there are *k-1* consecutive losses followed by a successfully transmitted packet. This means that

$P[NR=k] = p^{k-1}(1-p)$, thus

$$E[NR] = \sum_{k=1}^{\infty} k\cdot p^{k-1}(1-p) = \sum_{k=1}^{\infty} k\cdot p^{k-1} - \sum_{k=1}^{\infty} k\cdot p^{k} = 1+\sum_{k=1}^{\infty}(k+1)\cdot p^{k} - \sum_{k=1}^{\infty} k\cdot p^{k}$$

$$E[NR] = 1 + \sum_{k=1}^{\infty} p^k = \sum_{k=0}^{\infty} p^k = \frac{1}{1-p} \tag{40}$$

The difference between SF-SACK and TCP will determine a different approach in the calculation of *E[TW].* By considering the evolution of the window size during a time-out period, as shown in Figure 11, the following recursive formula is obtained:

$TW_{ik} = \dfrac{\dfrac{2\tau}{2^{k-1} \cdot T_0} - 1}{\dfrac{2\tau}{2^{k-1} \cdot T_0} + 1} \cdot TW_{ik-1} + \dfrac{1}{\dfrac{2\tau}{2^{k-1} \cdot T_0} + 1} \cdot 2$, which is equivalent to

$$TW_{ik} = \frac{2\tau - 2^{k-1} \cdot T_0}{2\tau + 2^{k-1} \cdot T_0} \cdot TW_{ik-1} + \frac{2^k \cdot T_0}{2\tau + 2^{k-1} \cdot T_0} \tag{41}$$

With the notation $\varepsilon = E[k] - 1$, it follows from (41):

$(2\tau + 2^{\varepsilon} \cdot T_0) \cdot E[TW] = (2\tau - 2^{\varepsilon} \cdot T_0) \cdot E[TW] + 2^{\varepsilon+1} \cdot T_0$, which leads to

$E[TW] \cdot 2^{\varepsilon+1} \cdot T_0 = 2^{\varepsilon+1} \cdot T_0$, meaning that

$$E[TW] = 1 \tag{42}$$

Since

$NR_i = \sum_{k=1}^{nto_i} TW_{ik}$, if $\{nto_i\}_i$ is assumed to be an i.i.d. sequence of random variables, independent of $\{NR_i\}$ and of $\{CW_{ik}\}$, then

$E[NR] = E\left[\sum_{k=1}^{nto_i} TW_{ik}\right] = E[nto] \cdot E[TW] = E[nto]$, thus

$$E[nto] = E[NR] = \frac{1}{1-p} \tag{43}$$

As a consequence of Equation (42), it can be considered that only one packet is sent every time-out interval. By making this assumption, the rest of the calculations will follow the approach of a classical TCP model, as found in [11]. The first six time-outs intervals in a sequence have the length $L_i = 2^{i-1} \cdot T_0$, then all the following intervals have the length $64T_0$. Then the duration of a sequence of k time-outs can be written as

$$L_{ik} = \begin{cases} (2^k - 1) \cdot T_0 & for \quad k \le 6 \\ (63 + 64(k-6)) \cdot T_0 & for \quad k \ge 7 \end{cases}, \text{ thus}$$

$$E[Z^{TO}] = \sum_{k=1}^{\infty} L_{ik} \cdot P[NR = k] = \sum_{k=1}^{\infty} L_{ik} \cdot p^{k-1} \cdot (1-p) = \sum_{k=1}^{\infty} L_{ik} \cdot p^{k-1} - \sum_{k=1}^{\infty} L_{ik} \cdot p^k =$$

$$= L_1 + \sum_{k=1}^{\infty} L_{k+1} \cdot p^k - \sum_{k=1}^{\infty} L_k \cdot p^k = L_1 + \sum_{k=1}^{\infty} (L_{k+1} - L_k) \cdot p^k =$$

$$= L_1 + (L_2 - L_1)p + (L_3 - L_2)p^2 + ... + (L_7 - L_6)p^6 + \sum_{k=7}^{\infty} 64 \cdot T_0 \cdot p^k =$$

$$= T_0 \cdot (1 + 2p + 4p^2 + 8p^3 + 16p^4 + 32p^5 + 64p^6) + \frac{64T_0 p^7}{1-p} =$$

$$= T_0 \cdot \frac{1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6}{1-p}$$

Thus, $E[Z^{TO}] = T_0 \cdot \frac{1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6}{1-p}$ (44)

There is left only the derivation of *X*, the probability that a loss indication at the end of a TDP is a time-out. For this, Figure 5 will be reconsidered in the case when the analytical model includes now the time-outs. It is known that the number of duplicate acknowledgments is equal to the number of packets received successfully in the last round. If the number of acknowledgments is

higher than three, then the packet loss is indicated by a TD, otherwise, the congestion window update is either initiated by the scheduler, either caused by a time-out occurrence.

Following the notations used in Section 4.B., let $A(w,k)$ denote the probability that the first *k* packets are acknowledged in a round of *w* packets, assuming that there is a lost in the round. Then

$$A(w,k) = \frac{(1-p)^k p}{1-(1-p)^w}$$

Let $T(n,m)$ be the probability that only the first *m* packets from the total number of *n* are acknowledged in the last round. Then

$$T(n,m) = \begin{cases} (1-p)^m \cdot p & if \quad m \le n \\ (1-p)^n & if \quad m = n \end{cases}$$

Let $\hat{X}(w)$ be the probability that a loss in a window of *w* is determined via a time-out. With the observation that the number of duplicate acknowledgments is equal to the number of successful received packets in the last round, a time-out occurs if either the number k of successfully transmitted packets in the penultimate round is at most 2, or is at least 3 and the number m of successfully received packets in the ultimate round is at most 2.

Thus, $\hat{X}(w)$ is given by

$$\hat{X}(w) = \begin{cases} 1 & if \quad w \le 3 \\ \sum_{k=0}^{2} A(w,k) + \sum_{k=3}^{w-1} A(w,k) \cdot h(k) & Otherwise \end{cases} \qquad \text{where } h(k) = \sum_{m=0}^{2} T(k,m).$$

In the case of $w > 3$, the expression of $\hat{X}(w)$ can be simplified by

$$\hat{X}(w) = \sum_{k=0}^{2} A(w,k) + \sum_{k=3}^{w-1} A(w,k) \cdot h(k) =$$

$$= \frac{p}{1-(1-p)^w} + \frac{(1-p)p}{1-(1-p)^w} + \frac{(1-p)^2 p}{1-(1-p)^w} + \sum_{k=3}^{w-1} A(w,k) \cdot (\sum_{m=0}^{2} T(k,m)) =$$

$$= p \cdot \frac{1+(1-p)+(1-p)^2}{1-(1-p)^w} + \sum_{k=3}^{w-1} \frac{(1-p)^k \cdot p}{1-(1-p)^w} \cdot \left[p + (1-p)p + (1-p)^2 p\right] =$$

$$= p \cdot \frac{1+(1-p)+(1-p)^2}{1-(1-p)^w} \cdot \left[1 + p \cdot \sum_{k=3}^{w-1} (1-p)^k\right] = \frac{1-(1-p)^3}{1-(1-p)^w} \cdot \left[1 + (1-p)^3 (1-(1-p)^{w-3})\right]$$

It follows that $\hat{X}(w)$ can be expressed as

$$\hat{X}(w) = \min\left(1, \frac{(1-(1-p)^3)(1+(1-p)^3(1-(1-p)^{w-3}))}{1-(1-p)^w}\right) \quad (45)$$

But since $\lim_{p \to 0} \frac{(1-(1-p)^3)(1+(1-p)^3(1-(1-p)^{w-3}))}{1-(1-p)^w} = \frac{3}{w}$, $\hat{X}(w)$ is approximated by

$$\hat{X}(w) = \min\left(1, \frac{3}{w}\right) \quad (46)$$

As a consequence, *X* is approximated by

$$X \approx \hat{X}(E[NW]) = \hat{X}(E[W] + E[m] - 1) = \hat{X}(E[W] + \frac{\tau}{2RTT(1-Q)} - 1)$$

$$X = \min\left(1, \frac{3}{E[W] + \frac{\tau}{2RTT(1-Q)} - 1}\right) \quad (47)$$

where $E[W]$ is given by Equation (25).

From the Equations (39), (31), and (32), it follows that

$$B = \frac{E[Y] + Q \cdot E[SC] + Q \cdot X \cdot E[NR]}{E[A] + Q \cdot E[Z^{SC}] + Q \cdot X \cdot E[Z^{TO}]} \tag{48}$$

and by considering Equations (20), (29), (36), (37), (40), and (44), it can be obtained that

$$B = \frac{\frac{23\tau^3 - 80\tau^2 RTT - 103\tau RTT^2 + 192RTT^3}{128RTT^2(\tau - 3RTT)} + \frac{Q\tau}{2RTT(1-Q)}\left[E[W] + \frac{\tau - 2RTT(1-Q)}{4RTT(1-Q)}\right] + \frac{QX}{1-p}}{\frac{\tau}{2} + \frac{Q\tau}{2(1-Q)} + QXT_0 \cdot \frac{1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6}{1-p}} \tag{49}$$

where

$$Q = \min\left(1, \frac{3}{E[W]}\right)$$

$$X = \min\left(1, \frac{3}{E[W] + \frac{\tau}{2RTT(1-Q)} - 1}\right)$$

$$E[W] = \frac{39 \cdot \tau^2 - 199 \cdot \tau \cdot RTT + 192 \cdot RTT^2}{32 \cdot RTT \cdot (\tau - 3 \cdot RTT)}$$

Figures 12, 13, and 14 present a comparison of the presented model of SF-SACK with the model of TCP Reno introduced in [11]. The send rate of TCP Reno is given by the equation

$$B_{\mathrm{Re}no} = \frac{\frac{1-p}{p} + E[W] + \frac{1}{1-p} \cdot \min\left(1, \frac{3}{E[W]}\right)}{RTT\left(E[X] + 1\right) + T_0 \frac{f(p)}{1-p} \cdot \min\left(1, \frac{3}{E[W]}\right)}$$, where

$E[W] = 1 + \sqrt{\frac{8(1-p)}{3p} + 1}$ and $f(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6$.

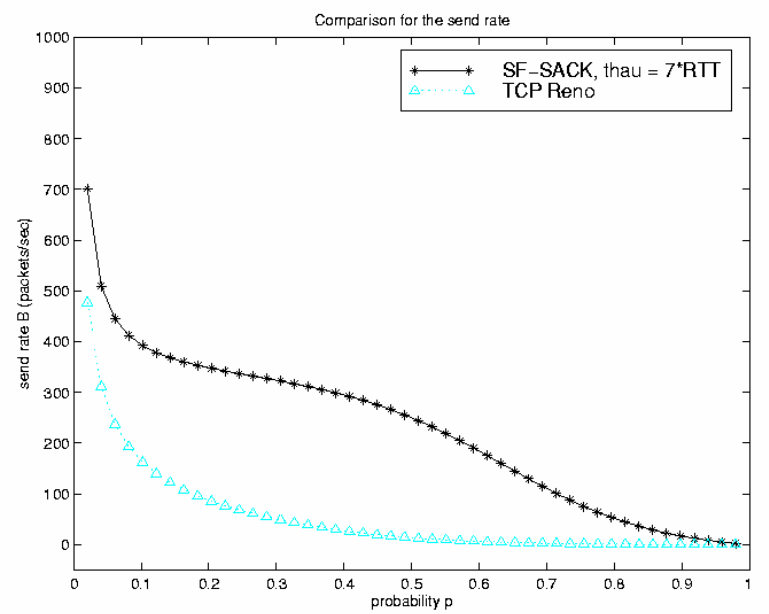

Figure 12. Comparison with TCP Reno, RTT = 0.016 sec, T_0 = 3*RTT, τ =7*RTT.

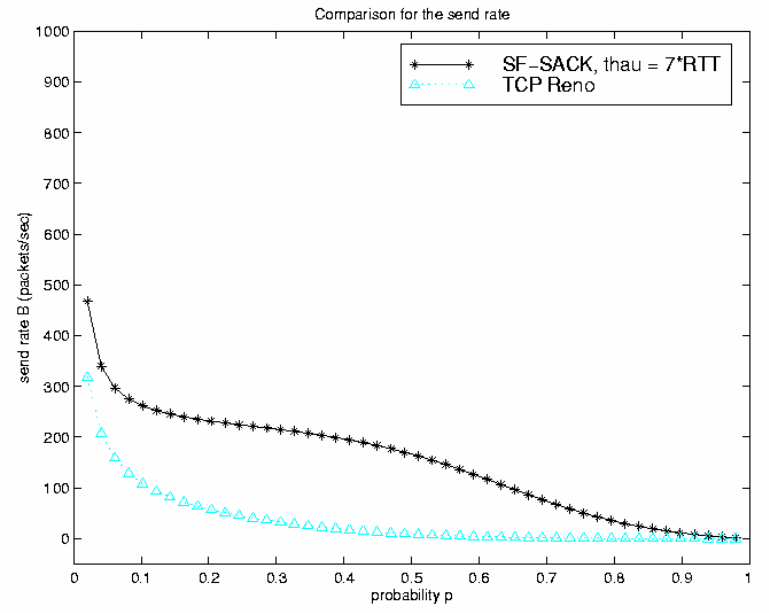

Figure 13. Comparison with TCP Reno, RTT = 0.024 sec, T_0 = 3*RTT, τ =7*RTT.

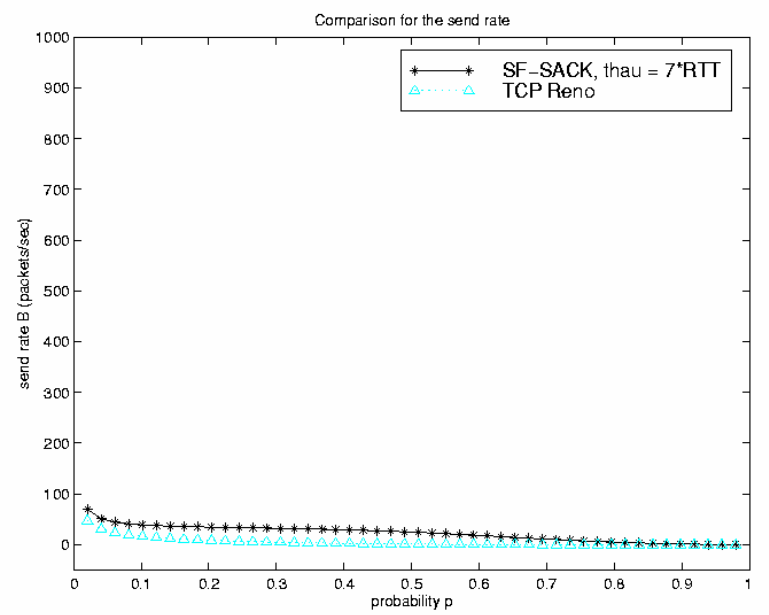

Figure 14. Comparison with TCP Reno, RTT = 0.16 sec, T_0 = 3*RTT, τ =7*RTT.

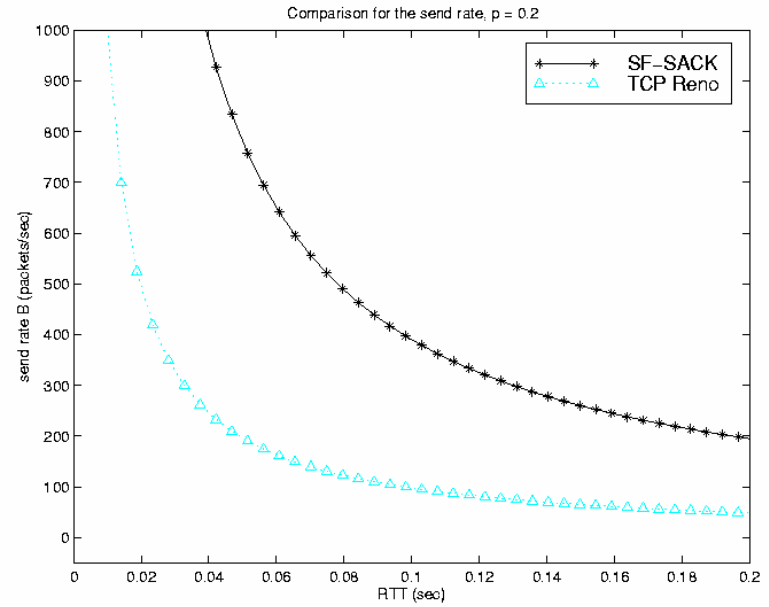

Figure 15. Influence of the value RTT on the send rate, p = 0.2, τ = 7 * RTT.

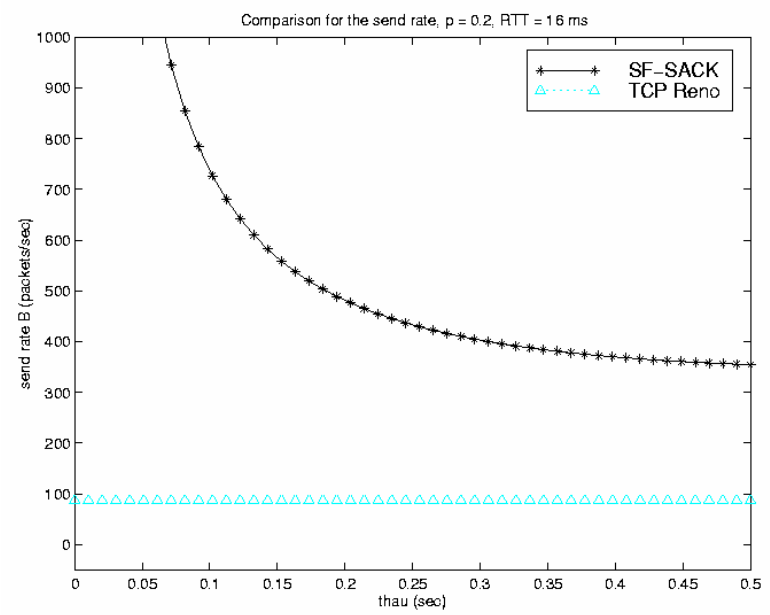

Figure 16. Influence of τ on the send rate.

As it can be observed from the graphs, the SF-SACK protocol has a performance that is superior to TCP Reno. As in the previous case, the influence of RTT and T_0 is more noticeable in the case of SF-SACK than for TCP.

The dependence of the send rate on the value of RTT is also illustrated in Figure 15. This dependence is more obvious for SF-SACK than for TCP, explained by the fact that the window size for SF-SACK depends on the time interval between two consecutive losses, thus, will receive a stronger influence from the value of RTT.

Very important for SF-SACK is a good choice of the length of the scheduler's interval. The larger this value is, the less the value of the congestion window is updated, leading to a smaller send rate. This is illustrated in Figure 16.

4.D. Calculation of the Throughput of a Bulk Transfer SF-SACK Flow

The throughput of a SF-SACK flow is the amount of data received by the receiver per unit time. The same considerations applied in the analysis of the

send rate can be applied for the calculation of the throughput, as in [11]. The modifications in the final formula would be

$$T = \frac{E[Y'] + Q \cdot E[SC'] + Q \cdot X \cdot E[NR']}{E[A] + Q \cdot E[Z^{SC}] + Q \cdot X \cdot E[Z^{TO}]} \quad (50)$$

where Y_i' is the number of packets that reach the destination during a TDP period, SC_i' is the number of packets received by the receiver during Z_i^{SC} (the duration of a sequence of consecutive scheduler's intervals that do not contain any loss detection, Figure 5), and NR_i' is the number of packets that reach the destination during Z_i^{TO} (the duration of a sequence of time-outs, Figure 8).

The derivation of E[Y'] can be followed from Figure 4. In a TDP period, the first packet that is lost is α_i. Thus, the first α_i-1 packets reach the destination. As it is supposed that if one packet is lost in a round all the following packets in that round are also lost, the packets that follow α_i in the penultimate round will not reach the destination. Some of the packets in the last round can also be lost, but for simplicity it will be assumed that they are not. As a consequence, E[Y'] can be expressed as

$$E[Y'] = E[\alpha] + E[\beta] - 1 \quad (51)$$

From Equations (5), (10), (27), and (51) it follows that

$$E[Y'] = \frac{23\tau^2 - 167 \cdot \tau \cdot RTT + 192RTT^2}{128RTT^2} + \frac{E[W]}{2} \quad (52)$$

Since during Z_i^{SC} there are no losses, the number of packets that reach the destination is equal to the number of packets that are sent during that period.

$$E[SC'] = E[SC] = \frac{\tau}{2 \cdot RTT \cdot (1-Q)} \cdot \left[E[W] + \frac{\tau - 2 \cdot RTT \cdot (1-Q)}{4 \cdot RTT \cdot (1-Q)} \right] \quad (53)$$

NR_i' represents the number of packets that reach the destination during a sequence of time-outs. During such a period, only the last packet will be actually received by the receiver, thus

$$E[NR'] = 1 \quad (54)$$

By combining the Equations (20), (36), (44), (49), (52), (53), and (54) the following formula is obtained:

$$T = \frac{\frac{23\tau^2 - 167\tau \cdot RTT + 192RTT^2}{128RTT^2} + \frac{E[W]}{2} + \frac{Q\tau}{2RTT(1-Q)} \left[E[W] + \frac{\tau - 2RTT(1-Q)}{4RTT(1-Q)} \right] + QX}{\frac{\tau}{2} + \frac{Q\tau}{2(1-Q)} + QXT_0 \cdot \frac{1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6}{1-p}}$$

where

$$Q = \min\left(1, \frac{3}{E[W]}\right), \; X = \min\left(1, \frac{3}{E[W] + \frac{\tau}{2RTT(1-Q)} - 1}\right), \text{ and}$$

$$E[W] = \frac{39 \cdot \tau^2 - 199 \cdot \tau \cdot RTT + 192 \cdot RTT^2}{32 \cdot RTT \cdot (\tau - 3 \cdot RTT)}$$

Figures 17, 18, and 19 illustrate comparisons between the throughput of SF-SACK and of TCP Reno. As expected, SF-SACK achieves a better throughput. It is to be noted that as the connection length increases, the performance of SF-SACK approaches the one of TCP, maintaining, though, the realization of a better throughput.

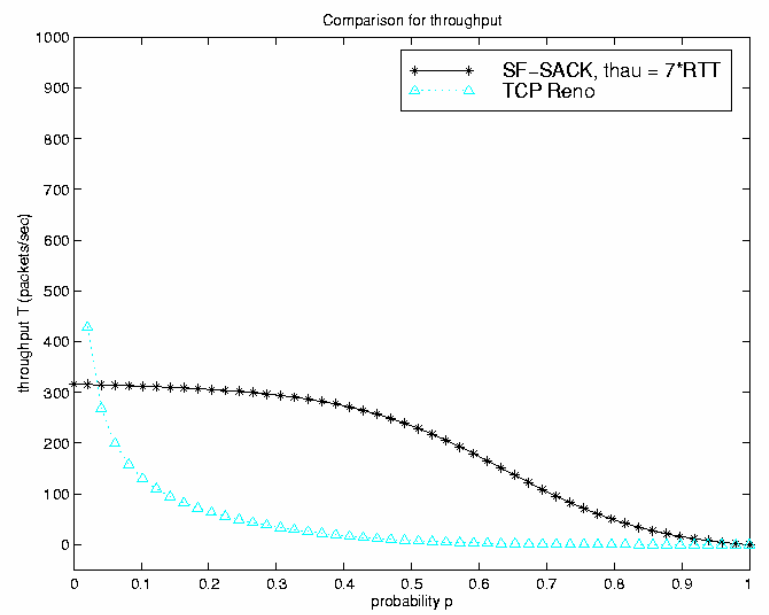

Figure 17. Comparison between the throughput of SF-SACK and TCP Reno, RTT = 0.016 sec, T_0 = 3*RTT.

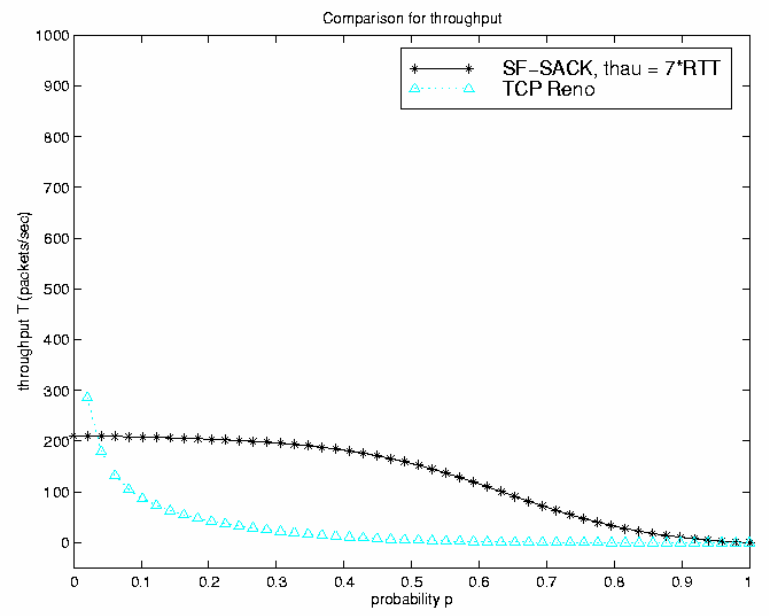

Figure 18. Comparison between the throughput of SF-SACK and TCP Reno, RTT = 0.024 sec, T_0 = 3*RTT.

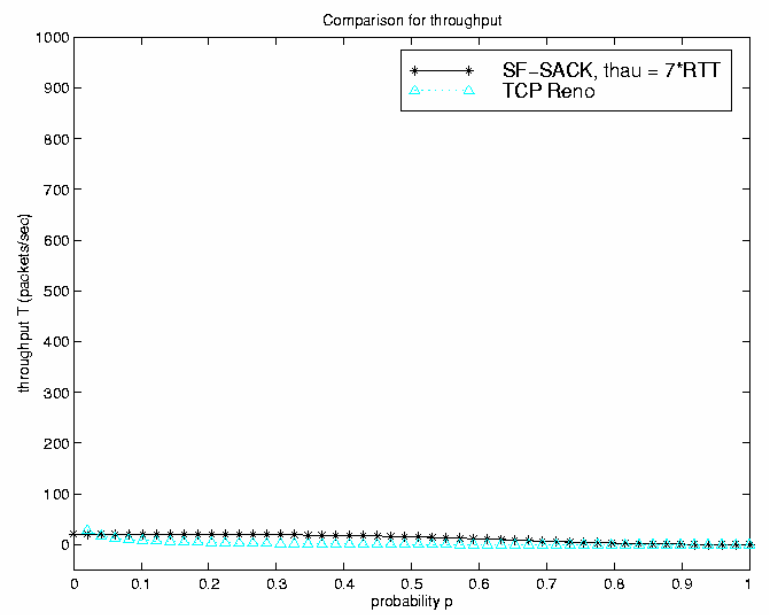

Figure 19. Comparison between the throughput of SF-SACK and TCP Reno, RTT = 0.24 sec, T_0 = 3*RTT.

Chapter 5

Conclusions and Future Work

This thesis presents a model to study the send rate and the throughput performance of a SF-SACK flow as a function of the loss probability, the round-trip time (RTT), the time-out interval, and the scheduler interval. This work provides insights to modeling more dynamic TCP versions and protocols. The model was built progressively and all the new mechanisms that characterize the SF-SACK protocol were included. The model provides theoretical bounds for the performance metrics of the protocol, and also provides means for an optimal choice of the performance metrics of SF-SACK. A performance comparison between SF-SACK and TCP is provided, utilizing the presented model and models available in the current literature. The performance results indicate that the SF-SACK protocol always achieve a better send rate and larger throughput than TCP Reno, which is expected given the less responsive nature of the decrease strategy of the congestion window.

Future research include the empirical validation of the protocol by either computer simulation or experimental measurements, along with a more complete analysis of the SF-SACK protocol, by providing means for a more comprehensive comparison, to a wider choice of TCP versions.

References

[1] S. Bakthavachalu, “SF-SACK: A Smooth Friendly TCP Protocol for Streaming Multimedia Applications”, *Master Thesis, Department of Computer Science and Engineering, College of Engineering, University of South Florida,* 2004.

[2] L. S. Brakmo, L. L. Peterson, “TCP Vegas: End to End Congestion Avoidance on a Global Internet”, *IEEE Journal on Selected Areas in Communication”, volume 13, issue 8,* October 1995.

[3] N. Cardwell, S. Savage, T. Anderson, “Modeling TCP Latency”, *Infocom*, 2000.

[4] K. Fall and S. Floyd, “Simulation-based Comparisons of Tahoe, Reno, and SACK TCP”, *ACM SIGCOMM Computer Communication Review, volume 26, issue 3,* July 1996.

[5] S. Floyd and K. Fall, “Promoting the Use of End-to-End Congestion Control in the Internet”, *IEEE/ACM Transactions on Networking, volume 7, issue 4,* August 1999.

[6] G. Grimmett and D. Stirzaker, “Probability and Random Processes”, *Oxford University Press,* 2001.

[7] M. Hassan and R. Jan, “High Performance TCP/IP Networking”, *Prentice Hall,* 2004.

[8] V. Jacobson, “Congestion Avoidance and Control”, *ACM SIGCOMM Computer Communication Review, Symposium proceedings on Communications architectures and protocols SIGCOMM ’88, volume 18, issue 4,* August 1988.

[9] V. Jacobson, “Modified TCP Congestion Control and Avoidance Algorithms”, *end2end-interest mailing list,* April 1990.

[10] A. Kumar, “Comparative Performance Analysis of Versions of TCP in a Local Network with a Lossy Link”, *IEEE/ACM Transactions on Networking, volume 6, issue 4,* August 1998.

[11] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, “Modeling TCP Reno Performance: A Simple Model and Its Empirical Validation”, *IEEE/ACM Transactions on Networking, volume 8, issue 2,* April 2000.

[12] S. M. Ross, “Applied Probability Models with Optimization Applications”, *Holden-Day*, 1970.

[13] S. M. Ross, “Introduction to Probability Models”, *Academic Press,* 1989.

[14] C. Samios and M. K. Vernon, “Modeling the Throughput of TCP Vegas”, *ACM SIGMETRICS Performance Evaluation Review, Proceedings of the 2003 ACM SIGMETRICS international conference on measurement and modeling of computer systems, volume 1, issue 1,* June 2003.

[15] B. Sikdar, S. Kalyanaraman, and K. S. Vastola, “Analytic Models for the Latency and Steady-State Throughput of TCP Tahoe, Reno, and SACK”, *IEEE/ACM Transactions on Networking, volume 11, issue 6,* December 2003.

[16] I. Yeom and A. L. Narasimha Reddy, “Modeling TCP Behavior in a Differentiated Services Network”, *IEEE/ACM Transactions on Networking, volume 9, issue 1,* February 2001.