The University of Kansas

Information and Telecommunication Technology Center

Technical Report

A Study onthe Cause of Long-Range Dependence Observed in Empirical TCP Traffic Traces

Georgios Lazarou and Victor Frost

ITTC-FY2000-TR-10980-28

July 1999

Project Sponsor:
Sprint Corporation
Under prime contract to
Defense Advanced Research Projects Agency
Contrct Number DABT 63-94-C-0068

Contents

Abstract

Several recent studies on a wide variety of networks have empirically observed that aggregate packet flows exhibit long-range dependence, i.e., the correlation between neighboring exclusive blocks of data does not asymptotically vanish when the block size is increased. Thus, actual network traffic is bursty over a broad range of time scales, in sharp contrast to conventional Markovian-type traffic models. Conventional models show traffic burstiness at only short time scales while traffic is smooth at large time scales. Additional studies have demonstrated that long-range dependence in traffic can have serious effects on network performance, but none of the previous work gives a complete answer of what is causing long-range dependence in network traffic. Since the majority of empirical studies were performed with TCP traffic, the initial hypothesis was that the dynamics of TCP, such as flow and congestion control algorithms, are the primary factor contributing to long-range dependence (LRD) in TCP traffic. Preliminary simulation results supported our assumption. However, futher investigation and further results from a large number of simulations contradict our assumption; the dynamics of TCP are not the main cause of the LRD observed in TCP traffic. Our results show that the presence of long-range dependence in network traffic does not necessarily depend on whether or not a reliable and flow- and congestion-controlled protocol is employed at the transport layer. Further, the results of this study show that in a client/server network environment, if traffic exhibits long-range dependence, then the distribution of message sizes is not necessary heavy-tailed.

1 Introduction

Knowledge of the nature of traffic of large internets or high-speed networks such as B-ISDN is essential for engineering, operations, and performance evaluation of these networks. Recent studies on a wide variety of networks have empirically observed that aggregate packet flows are statistically *self-similar* in nature, i.e., the statistical properties of the aggregate network traffic remain the same over an extremely wide range of time scales or over all time scales [24, 20, 4, 13, 28, 36]. Thus, actual network traffic is bursty over a broad range of time scales, in sharp contrast to conventional Markovian-type traffic models, which show traffic burstiness at only short time scales while traffic is smooth at large time scales. Since self-similar traffic has observable burst on all time scales, it exhibits *long-range dependence* (LRD); the correlation between neighboring exclusive blocks of data does not asymptotically vanish when the block size is increased [39]. This correlation is zero or decays exponentially fast towards zero for traditional traffic models.

1.1 Implication of Traffic Long-Range Dependence on Network Performance

The observation of traffic long-range dependence leads to the following question: *what is the impact of long-range dependence on network and protocol design, congestion control, and performance analysis*? The studies by [31, 14, 12, 33, 29, 25, 3] show that the performance of queueing models with long-range dependent arrival traffic can be drastically different from the performance predicted by conventional traffic models, especially by Markovian models. A direct implication of long-range dependence on network performance is that the burstiness of the traffic typically intensifies as the number of active sources increases. This is in contrast to the traditional idea that traffic becomes less bursty as the number of traffic sources increases, which is the typical nature of aggregate traffic of the Poisson-type models. The results show that network performance as captured by throughput, packet loss rate, and packet retransmission rate, degrades gradually as the intensity of long-range dependence increases. In practice, not accounting for the traffic long-range dependence characteristic at the network modeling stage of a system design can lead to overly optimistic performance predictions and thus to quality-of-service (QoS) requirements that are impossible to guarantee in a realistic network scenario [50].

Specifically, the overall packet loss decreases gradually with increasing buffer capacity, in strong contrast to Poisson-based models where losses decrease exponentially fast with increasing buffer size. Moreover, the queueing delay increases drastically as buffer capacity increases, again in sharp contrast to the traditional models where delay does not exceed a fixed limit regardless of buffer size. Further, the large variations in the network traffic on time scales of hours, days, or months complicates careful sizing of network components, since small errors in traffic engineering can incur drastic penalties in loss or delay [15]. Although some of the standard traffic models suggest that congestion problems essentially disappear with sufficient buffer capacity, in long-range dependent environments such behavior can not be expected. Increasing buffer capacity won't prevent congestion from occurring but instead will lead to large queueing delays [15, 33, 34].

1.2 What Gives Rise to Long-Range Dependence in Network Traffic?

The importance of long-range dependence in network traffic raises another major question: *what causes long-range dependence in network traffic*? The results in [50] show that the superposition of many independent ON/OFF traffic sources with strictly alternating ON- and OFF-periods and whose ON-periods or OFF-periods have heavy-tailed probability distribution functions results in aggregate packet streams that are consistent with measured local-area network (LAN) traffic and exhibits the same long-range dependence property as can be observed in the data [24]. Although the ON/OFF model gives a plausible explanation of the empirically observed self-similar nature of LAN traffic, it ignores interaction among traffic sources contending for network resources which in real networks can be as complicated as the feedback congestion control algorithm of many transport protocols, i.e., Transmission Control Protocol (TCP). The simulation study in [34] which is motivated by the ON/OFF traffic model shows that **if** the distribution of file sizes being transferred over the network is heavy-tailed[1] **then** the superposition of many file transfers in a client/server network environment induces self-similar traffic and this causal relationship is not significantly affected by changes in network resources[2], network topology, the influence of cross-traffic, or

[1]Meaning that the distribution behaves like a power law thus generating very large file transfers with non-negligible probability.

[2]Bottleneck bandwidth and buffer capacity.

the distribution of interarrival times. It is also shown that the degree to which file sizes are heavy-tailed directly determines the intensity of traffic long-range dependence. In agreement with the above results, the simulation experiments performed in [20] indicate that if the joint distribution of the number of packets per conversation and conversation transmission rates is heavy-tailed then TCP traffic exhibits long-range dependence. In addition, the relationship between file sizes and self-similar traffic was also suggested by the work described in [10] which showed that self-similarity in World Wide Web traffic might arise due to the heavy-tailed distribution of files present in the Web. Evidences that file systems indeed possess heavy-tailed file distributions are noted in [10, 2, 17, 36].

Importantly, the work in [34] shows that the presence of long-range dependence depends on whether reliable and flow-controlled communication is employed at the transport layer. In particular, the reliable transmission and flow control mechanisms of transport protocols, like TCP, serve to maintain the long-range dependence structure induced by heavy-tailed file size distributions. In contrast, if a non-flow-controlled and unreliable transport protocol (such as User Datagram Protocol, UDP) is used, the resulting traffic shows little self-similar characteristics: although still bursty at short time scales, the degree of self-similarity is very small.

1.3 Our Work

Our work is based on our assumption that the long-range dependence in aggregated TCP traffic is induced by the *dynamics*[3] of TCP. This study performed to validate our assumption was motivated by the following considerations:

1. TCP traffic was used in most empirical and simulation studies [24, 10, 20, 34, 36, 38, 49, 50] performed to either detect the presence of long-range dependence (LRD) in network traffic or give a possible explanation of what causes the LRD in network traffic.

2. The results in [50] provide compelling evidence in favor of explaining the self-similar nature of aggregate LAN traffic in terms of the heavy-tailed probability distribution functions of the ON-periods or OFF-periods of the individual ON/OFF source-destination pairs that make up the aggregate packet stream. But, they fail to explain the empirically observed self-similar nature of WAN traffic.

[3]Reliable flow and congestion control, see next section.

3. Although the ON/OFF model gives a plausible explanation of the empirically observed self-similar nature of LAN traffic, it ignores interaction among traffic sources contending for network resources which in real networks can be as complicated as the feedback congestion control algorithm of many transport protocols, i.e., Transmission Control Protocol (TCP).

4. The empirical studies in [24, 20, 36, 38] have shown that the aggregate TCP traffic exhibits long-range dependence in nature.

5. The significant work in [34] is a simulation study of TCP client/server flows which is focused only when the distribution of the file sizes being transferred over the network is heavy-tailed. But, in a large and complex real network environment such as Internet, this might not be the only case.

6. As shown by the work in [34], the presence of long-range dependency in network traffic depends on whether reliable and flow-controlled communication is employed at the transport layer.

7. TCP through its reliable, flow and congestion mechanisms controls the rate at which data packets are transmitted. Importantly, the state of the network governs TCP's behavior and transmission rate.

8. TCP traffic is very bursty in nature. Since TCP is a sliding window protocol, it transmits packets within a window as fast as it can and then waits for acknowledgment[4].

9. TCP is the most widely used transport protocol in the Internet, a global collection of networks connecting millions of computers and users, and incorporating a large variety of different network technologies. Asynchronous Transfer Mode (ATM) technology is the emerging standard adopted by telecommunications and computer vendors for high speed backbone networks. Most of the existing ATM backbone networks employ TCP over ATM technology. Thus, understanding the nature of TCP traffic is critical in order to properly design and implement future networks.

10. Since long-range dependence in traffic can have serious effects on network performance, it is very important to be able to control its intesity. But to do that, it is important to identify and evaluate all cases that cause TCP's traffic to be long-range dependent.

[4]See next section for details.

The above considerations led to the hypothesis that the primary factor contributing to long-range dependence in TCP traffic is when the dynamics of TCP are in effect. Further, we assumed that, in many cases, some application-level characteristics and the number of active connections have also a major contribution in the long-range dependency of TCP traffic, but they are not the main factors. Preliminary simulation results [22] supported our hypothesis. However, futher investigation and further results from a large number of simulations contradict our initial hypothesis; the dynamics of TCP are not the main factor of the presence of LRD observed in TCP traffic.

1.4 Lessons Learned

From this study several important lessons have been learned:

- In a greedy-source[5] network environment, the activation of TCP's dynamics by packet losses is a possible source of long-range dependence in TCP traffic.
- In a client/server network environment, if traffic exhibits long-range dependence, then a) it is not caused by the dynamics of TCP, b) the dynamics of TCP most likely had no effect on its intensity, c) the distribution of file sizes is not necessary to be heavy-tailed.
- In a client/server network environment, if the distribution of file sizes is exponential or uniform with very high means (or a combination of low and high means), the aggregation of many flows can result in traffic that exhibits long-range dependence[6]. The aggregation of network connections whose files sizes are exponentially distributed with low mean (4 KB) with connections whose file sizes are also exponentially distributed but with high mean (5 MB or higher) can generate traffic with similar burstiness with traffic created by a combination of connections whose files sizes are heavy-tailed distributed.
- The presence of long-range dependence in network traffic does not necessarily depend on whether or not a reliable and flow- and congestion-controlled protocol is employed at the transport layer.

[5] A greedy source has always data to send.

[6] This statement is based on our simulation results. It is not yet proved by rigorous mathematical analysis.

- In a client/server network environment, if periodicity is observed in TCP traffic by the autocorrelation function, then most likely it is caused by the dynamics of TCP.

- To obtain accurate estimates of the intensity of long-range dependence of an empirical trace either measured or simulated, we recommend several hours (10 or more) of traffic be collected. In our study we collected traffic in the range from about 3 hours to 229 hours, while most published results are based on at most three hours of traffic traces. The definition of long-range dependence applies only to infinite time series. In detecting the presence of LRD in finite traffic traces, there is no rule of how many samples are required to get an accurate estimate of the self-similar parameter H. The number of samples required for an accurate estimate of H depends very much on the traffic process. In most cases, incorrect estimates of H are obtained when traffic traces are short. We also recommend that the intensity of the long-range dependence be estimated at different time scales, i.e., at 0.01, 0.1, and 1.0 second aggregation intervals. It is possible for self-similar parameter H estimators to show evidence of LRD in network traffic only at short time scales.

- The self-similar parameter H does not give a complete characterization of a traffic process. Two traffic traces with the same value of H can have totally different traffic patterns. For different network conditions, application-level characteristics, and transport protocol parameters the characteristics of network traffic can be very different. Hence, constructing a general traffic model from traffic traces may not be achievable.

The rest of this technical report is organized as follows: Section 2 provides an overview of transport protocols, specifically it presents a brief description of the Transmission Control Protocol (TCP); Section 3 provides the definitions of long-range dependent and self-similar stochastic processes, and discusses some of their properties; Section 4 defines the network model used in all simulations; Section 5 presents the simulation results; and Section 6 presents a discussion of results and the goal of future work.

2 Transport Protocols

The International Standards Organization (ISO) developed an architectural model to describe the structure and function of data communications protocols. This model is called the *Open Systems Interconnect* (OSI) *Reference Model*, and it provides a common frame of reference for data communications models. The OSI Reference Model (Figure 1) contains seven layers that define the functions of data communications protocols. Each layer repre-

7	**Application Layer**
6	**Presentation Layer**
5	**Session Layer**
4	**Transport Layer**
3	**Network Layer**
2	**Data Link Layer**
1	**Physical Layer**

Figure 1: The OSI reference model

sents a function performed when data is transferred between cooperating application across a network. A layer does not define a single protocol but a data communications function that may be performed by any number of protocols. Hence, each layer may contain multiple protocols, each providing a service suitable to the function of that layer.

For this study, only the *transport layer* (the fourth layer in Figure 1) is considered. The transport layer has a number of functions, not all of which are necessarily required in any given network. The most common transport layer functions are [32]:

- addressing,
- connection establishment and termination,

- flow and congestion control,
- buffering,
- multiplexing,
- segmentation and reassembly
- handling duplicated packets,
- error recovery and control.

There are many transport protocols exist in standard commercial networks[7] but the most popular are TCP and UDP. Since the focus in this simulation study is on TCP and UDP, a brief overview of these protocols is given next.

2.1 Transmission Control Protocol

Transmission Control Protocol (TCP) provides reliable data connection services to applications and contains the mechanisms that guarantee that data is delivered error-free, without omissions, and in sequence. It is a reliable byte stream, connection-oriented, and end-to-end flow and congestion control protocol in the transport layer of the TCP/IP[8] protocol suite (see Fig. 2). A detailed analysis on the functionality of each layer in the TCP/IP protocol suite can be found in [42]. TCP is the most widely used protocol in the Internet, and it was designed to work independently of the lower layer implementation for transferring data. TCP operates on top of Internet protocol (IP) layer, which provides a best effort service. That is, there are no guarantees that an IP packet successfully gets to its destination. Applications that require the transport protocol to provide reliable data delivery use TCP because it verifies that data is delivered across the network accurately and in proper sequence.

TCP is connection-oriented. Two end hosts using TCP must establish a logical end-to-end connection with each other before they can exchange data. Control information, called a *handshake*, is exchanged between the two end hosts to establish the logical connection. As soon as the connection is established, data can be transferred. The established connection

[7] A list of most widely used transport protocol is given in [32].

[8] Officially it is known as the Internet Protocol (IP) suite, but because TCP and IP are the most known protocols from the IP suite, it has become common to use the term TCP/IP to refer to the whole protocol suite.

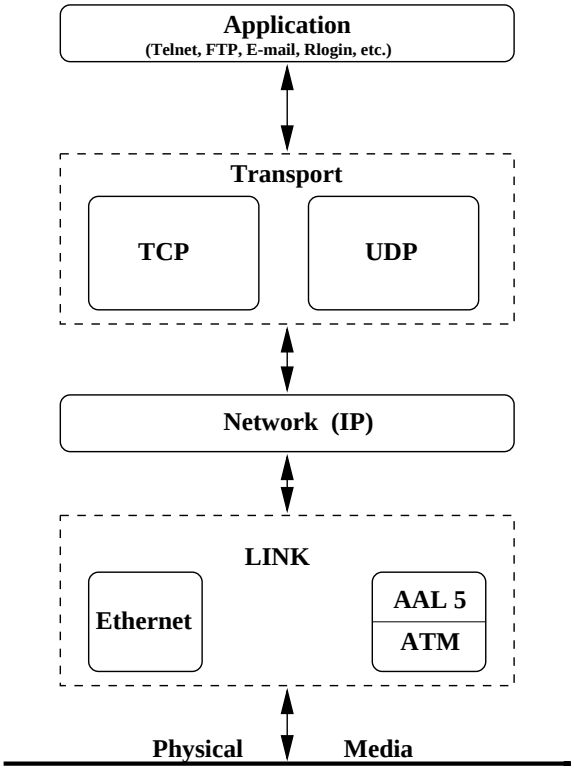

Figure 2: TCP/IP protocol suite reference layer model

between the end hosts is *full duplex*. This means that data can be flowing in each direction, independent of the other direction. When the data transfer between the two end hosts is finished the connection is terminated.

When an application passes data to TCP for delivery, TCP breakes the data stream into smaller chunks and adds a protocol information header to form a *segment*. This is the unit of data that TCP passes to IP. IP appends its own protocol information header and it forms a *datagram*. The largest chunk of data that TCP can include in each segment is called the *maximum segment size* (MSS). During the connection setup phase, each end host announces its MSS and TCP choose the smallest one.

To transfer data reliably, TCP uses a technique called *positive acknowledgement with retransmission*. TCP views the data it sends as a continuous stream of bytes, not as independent packets. So, to maintain the sequence in which bytes are sent and received, a TCP segment header contains a *sequence number* and an *acknowledgement number*. Each byte of data sent is numbered sequentially from an initial sequence number. The sequence number identifies the sequential position in the data stream of the first data byte in the segment.

The receiver is expected to acknowledge the received data. The acknowledgement (ACK) tells the sender how much data has been received in order and the acknowledgement number specifies the sequence number of the next byte (octet) that the receiver expects to receive. But data segments and ACKs can get lost. TCP handles this by setting a timeout every time it sends a segment. If the segment isn't acknowledged when the timeout expires, TCP assumes that the segment was lost or corrupted and retransmits it.

The amount of time a sender waits for an ACK before retransmission is called *retransmission timeout (RTO)*. A TCP segment may traverse a single low-delay network (i.e., high-speed LAN), or it may travel across multiple intermediate networks through multiple routers. The delay that each segment may experience through each intermediate network and at each router depends on the network traffic. Hence, the total time, called the *round-trip time (RTT)*, required for a segment to travel to the destination and an ACK to return to the sender changes significantly over time as traffic load varies. To adopt in the delay variations encountered in networks, TCP uses an *adaptive retransmission algorithm* that tracks delay changes on each connection and adjusts its RTO accordingly. The exact details of calculating the RTO are given in [18, 8, 42].

TCP segments are passed to IP which routes them as IP datagrams to the destination. Since IP is a *connectionless* service, datagrams can arrive to the destination from different routes. Therefore, IP datagrams and hence TCP segments can be received out of order. TCP resequences the out-of-order segments before it passes them to the application. Further, TCP detects transmission errors in the received segment by maintaining checksum. The sending TCP computes a checksum over the entire segment and then stores it in the checksum field in the segment's header. The receiving TCP computes again the checksum of the receiving segment and compares it to the content of the checksum field in the segment's header. If the values don't much, TCP discards it and doesn't acknowledge receiving it. Also, since segments or ACKs can get lost or arrive late due to excessive delay in the network, duplicate segments may arrive at the receiver. TCP detects the duplicate segments by comparing their sequence numbers with the sequence number of the data byte it expects to receive next from the sender. Duplicate segments are discarded.

2.1.1 Flow Control

Flow control is concerned with the regulation of the rate at which the sender transmits packets to match the rate at which the destination station receives data, so that it is not

overwhelmed. That is, without control, the sender may transmit packets at a rate too fast for the receiver. This may cause queue overflow at the receiver, leading to packet losses, retransmissions, and degraded performance. Thus, a flow control technique protects the receiver from being overflowed by the sender.

Having a mechanism for flow control is essential in an internet environment, where machines of various speeds and sizes communicate through networks and routers of various speeds and capacities. TCP attains end-to-end flow control by using the *sliding window* technique [6, 8, 42]. It allows the sender to transmit multiple segments before it stops and waits for an ACK, which results in high network utilization and throughput. But, the amount of unacknowledged data that the sender can have in transit to the receiver depends on the window size which is controlled by the receiver. The receiver has a finite amount of data buffer space for each TCP connection. The received data are stored in the buffers until read by the corresponding application. The window indicates how much buffer space the receiver has available for the incoming data. A zero window tells the sender to cease transmission of data until it receives a non-zero window value. Thus, the purpose of this window is to allow the receiver to control the rate at which it receives data and to prevent a fast transmitting host from overflowing the data buffers on a slower host.

2.1.2 Congestion Control and Avoidance

Flow control between the source and the destination does not help much toward reducing the possibility of congestion within the network. Congestion is a condition of severe delay caused by an overload of packets at one or more switching nodes within the network. It can occur whenever the offered load to the network exceeds its capacity. Even in a well-designed network, statistical variations in traffic flows may lead to congestion. As the arrival packet rate at a network node becomes greater than its transmission packet rate, its queue length grows dramatically. As the node becomes congested, queue starts to overflow, (i.e., packets that arrive at a time the queue is full are discarded), and delays increase beyond acceptable levels. Dropped packets are eventually retransmitted by the source causing the traffic load to further increase. As the number of retransmissions increases, more nodes become congested and more packets are dropped. A typical pattern of network performance as a function of the offered load is shown in Figure 3. As the offered load increases beyond the knee point, throughput increases slowly but the delay increases dramatically. After the load reaches the network capacity, throughput stops increasing. When load exceeds the *congestion collapse*

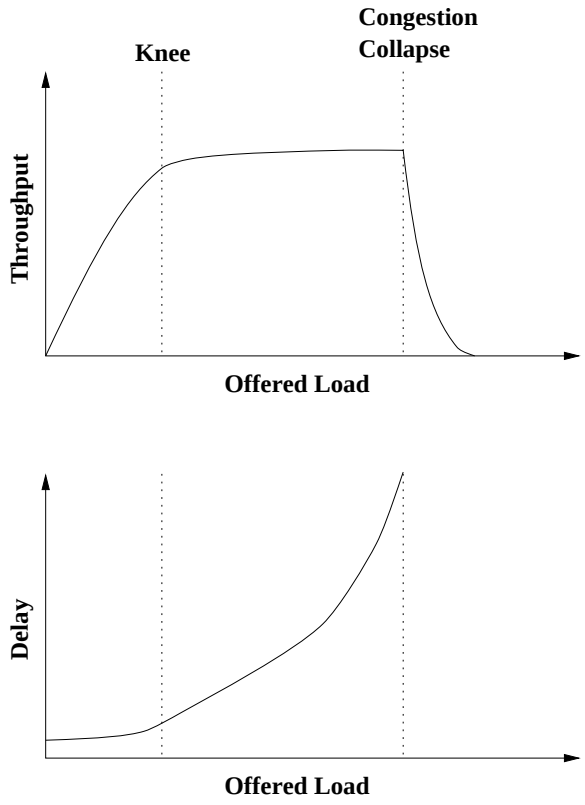

Figure 3: Network performance as a function of the offered load

point, throughput falls off rapidly approaching zero.

There are two broad classes of congestion control mechanisms: *preventive control* and *reactive control*. Preventive congestion control techniques attempt to prevent congestion by taking appropriate actions, such as regulating the traffic from the source, before it actually occurs. A preventive congestion control technique that allows the network to operate at the knee point as shown in Figure 3, (i.e., the region of low delay and high throughput), is called a *congestion avoidance* technique. However, reactive congestion control techniques are still required to protect the network should it reach the congestion collapse state due to transient changes in the network. The purpose of a reactive congestion control technique is to detect the fact that the network is congested and help it to recover by reducing the traffic flow into the network.

TCP standards define two congestion control mechanisms, one preventing and one reactive. Although the two techniques are different, in practice they are implemented together. The combine scheme is known as *Slow Start and Congestion Avoidance Scheme*. With

this scheme, the sender maintains three variables for each connection: a congestion window ($cwnd$), a send window (snd_wnd), and a slow start threshold ($ssthresh$)[9]. The sender never transmits more than the minimum of ($cwnd$) and the receiver's advertised window (rcv_wnd). That is, always the transmission window size is given by

$$snd\_wnd = MIN(cwnd,\ rcv\_wnd).$$

The congestion window is for a flow control imposed by the sender based on its assessment of perceived network congestion, while the receiver's advertised window is for a flow control imposed by the receiver related to its amount of available buffer space. The slow start threshold determines when the transmission state of the sender shifts from the slow start phase into the congestion avoidance phase.

Current implementations of TCP can not distinguish segment losses due to transmission errors from segment losses due to congestion. So it makes the conservative assumption that all losses are due to congestion. When a timeout occurs due to a segment loss, TCP first doubles the current RTO for all unacknowledged[10] segments and then enters the slow start phase to recover from congestion. Slow start begins at the sender by first setting $cwnd$ to one segment and

$$ssthresh \leftarrow \frac{snd\_wnd}{2}.$$

It then sends one segment and waits for an ACK. If there aren't any unacknowledged segments, the retransmission timer for this segment is also set to $2 * current\ RTO$. When the ACK is received, a new RTO value is estimated and $cwnd$ is incremented by one segment, and hence two segments can be transmitted. For each ACK the sender receives, $cwnd$ is incremented by one segment which leads to an exponential window increase. That is, $cwnd$ doubles for every RTT. Slow start ends when $cwnd$ reaches $ssthresh$ or rcv_wnd, or segment loss is detected. Since slow start gradually increases the transmission rate, whenever TCP establishes a new connection, it begins transmitting data from the slow start phase in order to avoid flooding the network with additional traffic, leading to network congestion. In this case, $ssthresh$ is set equal to rcv_wnd, and the total time required for slow start to achieve the bandwidth allowed by the receiver is calculated as follow:

$$T_{ss}(W = ssthresh) = RTT * log_2 W. \quad (1)$$

[9] In practice, *cwnd*, *snd_wnd*, and *ssthresh* are expressed in terms of bytes, but here they are expressed in terms of segments for simplifying the discussion.

[10] A segment that has been transmitted but no ACK has been received.

When *cwnd* reaches *ssthresh*, TCP shifts to the congestion avoidance phase and slows down the rate of window increment. In the congestion avoidance phase, the network is probed for available bandwidth by transmitting one additional segment for each RTT, until the receiver's advertised window is reached or a segment loss is detected. The total time required for congestion avoidance to reach the bandwidth allowed by the receiver given no segment loss is calculated as follow:

$$T_{ca} = RTT * (rcv\_wnd - ssthresh). \tag{2}$$

Figure 4 shows the slow start and congestion avoidance phases for three different cases. In

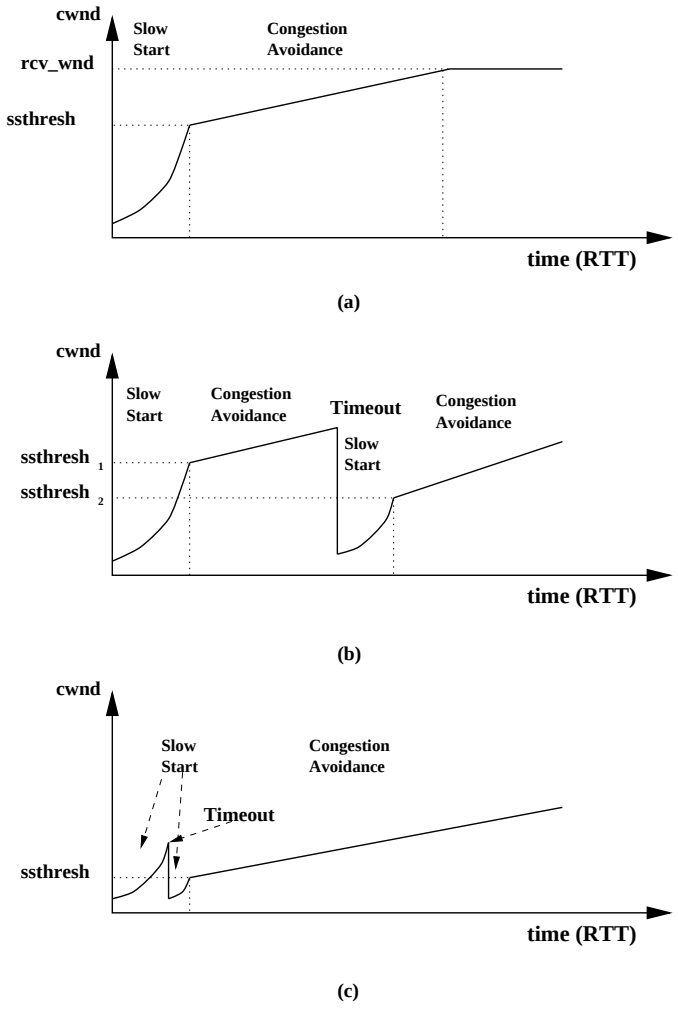

Figure 4: Visual description of slow start and congestion avoidance

all three cases, TCP enters the slow start phase after a segment loss is detected by a timeout.

2.1.3 Fast Retransmit and Fast Recovery

TCP is required to generate an immediate *duplicate ACK*[11] when an out-of-order segment is received. The fast retransmit scheme detects the loss of a single segment without having to wait for the retransmission timer to expire. When sender receives a predefined number (usually 3) of duplicate ACKs, it deduces that a segment loss occurred. It then sets *ssthresh* to one-half the current *cwnd* and retransmits the lost segment as indicated by the duplicated ACKs. Next, it sets *cwnd* to *ssthresh* plus three segments. Each time another duplicate ACK is received, fast retransmit increments *cwnd* by one segment and transmits a new segment if it is allowed by the new value of *cwnd*. Upon receiving a non-duplicate ACK, the scheme shifts to fast recovery. It sets *cwnd* to *ssthresh* and begins congestion avoidance, without falling back to slow start.

2.1.4 Delayed Acknowledgements

TCP standards recommend delaying ACKs at the receiver. Since ACKs are cumulative, one ACK can acknowledge multiple segments. Sending one ACK for more than one segment reduces the return path bandwidth used by the ACKs. In most implementations, TCP does not send an ACK the instant it receives a segment. Instead, it delays the ACK in case there data to send to the sender, and so the ACK can be sent along with the data. Usually, TCP delays ACKs up to 200 ms to see if there is data to send with the ACK. An exception is when out-of-order or two segments are received. In such cases, an immediate ACK is sent. Equation (1) is not valid if delayed ACK is implemented.

2.2 User Datagram Protocol (UDP)

UDP is a non-flow-controlled, unreliable, and connectionless transport protocol. It allows applications to exchange data (messages, files, etc.) over the network with a minimum of protocol overhead. There are no techniques in UDP for verifying that data reached the destination correctly, therefore, it does not guarantee that data is delivered error-free. Also, UDP does not use any flow and congestion control mechanism like TCP. Two end hosts using UDP don't establish any logical end-to-end connection with each other before they can exchange data.

[11] It's called duplicate ACK because it again acknowledges the last in sequence received and acknowledged segment.

3 Long-Range Dependence and Self-Similarity

This section gives the mathematical definition of long-range dependent and self-similar processes, and a brief discussion on self-similar stochastic modeling.

3.1 Wide-Sense Stationary Stochastic Processes

A discrete-time real-valued stochastic process $X = \{X_t, t = 0, 1, 2, \ldots\}$ is *strictly stationary* process if all of the distribution functions describing the process are invariant under a translation of time. X is said to be *stationary in the wide sense,* or *weakly stationary,* if its mean $\mu = E[X_t]$ is a constant, its variance $\sigma^2 = E[(X_t - \mu)^2] < \infty$, and its autocovariance function

$$C_k = Cov(X_t, X_{t+k}) = E[(X_t - \mu)(X_{t+k} - \mu)] \qquad k = 0, 1, 2, \ldots \tag{3}$$

depends only on the time difference k. The autocorrelation function of X is then given by:

$$r(k) = \frac{C_k}{C_{k=0}} = \frac{E[(X_t - \mu)(X_{t+k} - \mu)]}{\sigma^2} \qquad k = 0, 1, 2, \ldots \tag{4}$$

which depends also only on k. Hence, for each k, $r(k)$ measures the correlation between elements of X separated by k units of time.

3.2 Long-Range Dependence

Let X be a wide-sense stationary stochastic process with an autocorrelation function of the form

$$r(k) \sim k^{-\beta} L(k) \qquad as \quad k \to \infty \tag{5}$$

where $0 < \beta < 1$ and L is slowly varying at infinity, that is,

$$\lim_{k \to \infty} \frac{L(kx)}{L(k)} = 1 \qquad \forall x > 0.$$

Such functions are $L(t) = const$, $L(t) = log(t)$, or any function having a nonzero horizontal asymptote at ∞.

Definition 1 *A wide-sense stationary stochastic process X is called a stationary process with long memory or long-range dependent if $r(k)$ satisfies relation (5). That is, the autocorrelation function decays hyperbolically as k increases, implying that*

$$\sum_{k=1}^{\infty} r(k) = \infty. \tag{6}$$

Equivalently ([39]), let $S_n = X_1 + \cdots + X_n$, *then*

$$Corr(S_n, S_{2n} - S_n) \to c > 0 \qquad as \;\; n \to \infty,$$

i.e., the correlation between neighboring non-overlapping blocks does not asymptotically vanish when block size is increased. Otherwise X *is called short-range dependent.*

This means that for stationary processes with long memory, the correlations between observations that are apart (in time) decay to zero at a slower rate than one would expect from independent data or data following classic ARMA- or Markov-type models. Note that the definition of long-range dependence applies only to infinite time series. (See [5] for an elaborate discussion on stationary processes with long memory.)

Although for a long-range dependent process the values of $r(k)$ for k large enough are very small, their cumulative effect is of importance because it gives rise to features which are drastically different from the short-range dependent processes considered in the conventional packet traffic models. A short-range dependent process can be characterized by an autocorrelation function that decreases exponentially fast (i.e., $r(k) \sim \alpha^k, \quad 0 < \alpha < 1$), implying a summable autocorrelation function (i.e., $\sum_K^\infty < \infty$).

The simplest models with long-range dependence are self-similar processes. Self-similar processes are particular attractive models because the long-range dependence can be characterized by a single parameter, the Hurst (H) parameter.

3.3 Self-Similar Stochastic Processes

There are several definitions of self-similarity; not all of the definitions are equivalent. The most well known definition (known as the standard, see [5, 45]) states that the time series $Y = \{Y_t, t \in T\}$ is self-similar with self-similarity parameter H if the following property is satisfied:

$$Y_t =^d \alpha^{-H} Y_{\alpha t}, \qquad \forall \alpha > 0, \qquad 0 \leq H < 1, \tag{7}$$

where $=^d$ is equality in the sense of finite-dimensional distributions. An example of such a process is Fractional Brownian Motion (Brownian Motion if H = 1/2). While a process Y satisfying (7) can never be stationary, it is typically assumed to have stationary increments [45].

More appropriate definitions of self-similarity in the context of network data traffic and standard time series theory are as follow [45, 5]: Let $X = \{X_t, t = 0, 1, 2, \ldots\}$ be a wide-sense stationary sequence with an autocovariance function C_k and autocorrelation function $r(k)$ as defined by (3) and (4), respectively. For each $m = 1, 2, 3, \ldots$, let $X^{(m)} = \{X_k^{(m)}, k = 1, 2, 3, \ldots\}$ denote a new aggregated time series obtained by averaging the original series X over non-overlapping blocks of size m, replacing each block by its sample mean. That is, for each $m = 1, 2, 3, \ldots$, $X^{(m)}$ is given by

$$X_k^{(m)} = \frac{X_{km-m+1} + \cdots + X_{km}}{m} \qquad k \geq 1. \tag{8}$$

The new aggregated time series is also wide-sense stationary with autocovariance function $C_k^{(m)}$, variance $\sigma_{(m)}^2 = C_0^{(m)}$, and autocorrelation function $r_k^{(m)}$.

Definition 2 *If X is the increment process of a self-similar process Y defined in (7) (i.e., $X_t = Y_{t+1} - Y_t$), then is called exactly self-similar with self-similarity parameter H if* $\forall m = 1, 2, 3, \ldots$,

$$X =^d \frac{X^{(m)}}{m^{H-1}} \tag{9}$$

Note that this definition of self-similarity is related to the first but they are not equivalent.

Definition 3 *X is called an asymptotically self-similar process with self-similarity parameter H if (9) holds as $m \to \infty$.*

Definition 4 *X is called exactly second-order self-similar with self-similarity parameter H if $\forall m = 1, 2, 3, \ldots$, $\frac{X^{(m)}}{m^{H-1}}$ has the same variance and autocorrelation as X.*

Definition 5 *X is called asymptotically second-order self-similar with self-similarity parameter H if $\frac{X^{(m)}}{m^{H-1}}$ has the same variance and autocorrelation as X as $m \to \infty$. That is, $\forall k$ large enough [24],*

$$r^{(m)}(k) \to r(k) \qquad as \quad m \to \infty$$

That is, X is exactly or asymptotically second-order self-similar if the corresponding aggregated processes $X^{(m)}$ are the same as X or become indistinguishable from X at least with respect to their autocorrelation functions.

The parameter H is called the *Hurst parameter*. It measures the degree of self-similarity of a time series. Specifically, it expresses the speed of decay of time series' autocorrelation function. From [5] (Sect. 2.3), the asymptotic behavior of $r(k)$ is shown to be:

$$\frac{r(k)}{H(2H-1)k^{2H-2}} \to 1 \qquad as \quad k \to \infty \tag{10}$$

For $\frac{1}{2} < H < 1$, the correlations decay to zero so slowly that $\sum_{k=1}^{\infty} r(k) = \infty$. Thus, the process X has long-range dependence. This means that a long-range dependent process is always an asymptotically second-order self-similar process, and long-range dependence implies an asymptotically second-order self-similarity. Note that a long-range dependent process can also be an exactly second-order self-similar since the latter one implies asymptotically second-order self-similarity. For $H = 1/2$, all correlations at non-zero lags are zero, i.e., the samples (observations) X_t are uncorrelated. For $0 < H < \frac{1}{2}$, the correlations sum up to zero (i.e., $\sum_{k=1}^{\infty} r(k) = 0$), and the process X has short-range dependence. [12] In this study, we focus on stochastic processes that have long-range dependence ($\frac{1}{2} < H < 1$).

3.4 Properties of Long-Range Dependent Stochastic Processes

3.4.1 Nondegenerate Correlation Structure

The aggregated processes $X^{(m)}$ of long-range dependent processes possess a nondegenerate correlation structure as $m \to \infty$. This is in entirely contrast to conventional packet traffic models which all have the property that their aggregated processes $X^{(m)}$ tend to second-order pure noise, i.e., $\forall K \geq 1$,

$$r^{(m)}(k) \to 0 \qquad as \quad m \to \infty.$$

3.4.2 Slowly Decaying Variance

An important feature of asymptotically second-order self-similar (and thus long-range dependent) processes is that the variance of the aggregated time series $X^{(m)}$ decreases more slowly than the reciprocal of the sample size n. That is,

$$Var[X^{(m)}] \sim am^{-\beta} \qquad as \quad m \to \infty \tag{11}$$

where a is a positive constant independent of m and $0 < \beta < 1$. This implies that $H = 1 - (\beta/2)$ for long-range dependency, and as $\beta \to 0$, and thus $H \to 1$, the autocorrelation

[12] For other values of H see [5], page 53

function decays more slowly, i.e., the degree of self-similarity increases. In contrast, for short-range dependent processes,

$$Var[X^{(m)}] \sim bm^{-1} \qquad as \quad m \to \infty \tag{12}$$

where b is a finite constant independent of m. The slowly decaying variance feature of self-similar process can be used to identify and estimate the degree of self-similarity of a time series. As shown by Section 3.3, the degree of self-similarity or the intensity of long-range dependence of a time series can be captured by a single parameter, the *Hurst* parameter H.

3.5 Methods of Estimating H

There are several methods for estimating the self-similarity parameter H or the intensity of long-range dependence in a time series [5, 46]. In this simulation study we used only two of them, the Aggregated Variance and the R/S methods, as described in [27, 5]. We validated our implementation of the two estimators with the Ethernet data used in [24]. For the robustness of these estimator, see [5, 47, 46]. Both methods are two of the better known methods and they have good robustness properties, in particular, with respect to long-tailed distributions [20, 5, 46]. Since confidence intervals can not be obtained with either method, we conlude that a timeseries has long-range dependence if the estimated H is well greater than 0.5.

3.5.1 Aggregated Variance Method

Consider the aggregated series $X^{(m)}$ as described in Sec. 3.3, obtained by dividing a given series of length N into blocks of length m, and averaging the series over each block. Its sample variance is then given by:

$$\hat{Var}[X^{(m)}] = \frac{\sum_{k=1}^{\frac{N}{m}} (X^{(m)}(k) - \bar{X})^2}{\frac{N}{m}} \tag{13}$$

where

$$\bar{X} = \frac{\sum_{t=1}^{N} X_t}{N} \tag{14}$$

is the sample mean of the series. For successive values of m that are equidistant on a log scale, the sample variance of the aggregated series is plotted versus m on a log-log plot. By

fitting a least-squares line to the points of the plot and then calculating its slope, an estimate of the Hurst parameter is obtained as follow:

$$\hat{H} = 1 - \frac{slope}{2}.$$

3.5.2 Rescaled Adjusted Range Statistic (R/S) Method

Let $X_t, \cdots, X_{t+n-1}$ be n observations of the process X with a sample variance,

$$S^2(t,n) = \frac{\sum_{i=t}^{t+n-1} X_i^2}{n} - (\bar{X}(t,n))^2$$

where $\bar{X}(t,n) = \frac{1}{n}\sum_{i=t}^{t+n-1} X_i$ is the sample mean. The R/S statistic, or the *rescaled adjusted range*, is given by:

$$\frac{R(t,n)}{S(t,n)} = \frac{\max\limits_{1\le u\le n}[\sum_{i=t}^{t+u-1} X_i - u\bar{X}(t,n)] - \min\limits_{1\le u\le n}[\sum_{i=t}^{t+u-1} X_i - u\bar{X}(t,n)]}{S(t,n)}.$$

For successive logarithmically spaced values of n, the average

$$Q(n) = \frac{1}{n}\sum_{i=1}^{n} \frac{R(t_i,n)}{S(t_i,n)}$$

for different values of t is plotted versus n on a log-log plot. The estimated H is given by the slope of a fitted least-squares line to the points of the plot. For a more detail discussion on both methods see [5].

3.6 Self-Similar Stochastic Modeling

Since the observation of self-similarity in several areas, several self-similar stochastic models have been developed [5, 9, 40]. Two well known models that yield exquisite representations of the self-similarity phenomenon but do not provide any physical explanation of self-similarity are *fractional Gaussian noise* (FGN) and the class of *fractional autoregressive integrated moving-average* (ARIMA) processes [24].

An important model which constructs self-similar processes is based on aggregating many simple renewal reward processes which exhibit inter-renewal times with infinite variances [44]. In other words, a self-similar process is generated by superimposing many simple renewal reward[13] processes, in which the rewards are restricted to the values 0 and 1 (OFF/ON), and

[13]See [48] for the definition

in which the inter-renewal times are heavy-tailed. As the number of such processes grows large, such a construction yields a self-similar FGN process. In contrast to FGN and ARIMA, this method does provide a physical explanation of self-similarity and it became appealing in the context of high-speed packet traffic [49, 24]. As shown in [50], the superposition of many independent ON/OFF traffic sources with strictly alternating ON- and OFF-periods and whose ON-periods or OFF-periods are heavy-tailed results in aggregate packet streams that are consistent with measured local-area network (LAN) traffic and exhibits the same self-similar property as can be observed in the data [24].

3.6.1 Heavy-Tailed Probability Distributions

Since heavy-tailed distributions are very important in self-similar modeling, a brief description is given here as presented in [10]. A probability distribution is heavy-tailed if

$$P[X > x] \sim x^{-\alpha} \qquad as \quad x \to \infty \tag{15}$$

where $0 < \alpha < 2$. Thus, regardless of the behavior of the distribution for small values of the random variable, if the asymptotic shape of the distribution is hyperbolic, it is heavy-tailed. A simple heavy-tailed distribution is the *Pareto* distribution [36]. Its distribution is hyperbolic over its entire range. Its probability density function is given by

$$p(x) = \alpha k^{\alpha} x^{-\alpha-1} \qquad \alpha, k > 0 \quad x \geq k \tag{16}$$

and its cumulative distribution function is given by

$$F(x) = P[X \leq x] = 1 - (k/x)^{\alpha}. \tag{17}$$

The parameter k represents the smallest possible value of the random variable.

Heavy-tailed distributions have a number of properties that are qualitatively different from distributions more commonly encountered such as exponential or normal distributions. If $\alpha \leq 2$, then the distribution has infinite variance. If $\alpha \leq 1$, then it has infinite mean. Thus, as α decreases, a large portion of the probability mass is present in the tail of the distribution. That is, a random variable that follows a heavy-tailed distribution can give rise to extremely large values with non-negligible probability.

4 Network Model and Simulation Scenarios

The purpose of our simulation study was to validate our assumption that the primary factor contributing to long-range dependence in TCP traffic is the dynamics of TCP. The dynamics of TCP, congestion control and avoidance, are in effect only in response to packet losses and when a new TCP connection is established, where TCP starts transmitting data with the slow start phase. Therefore, simulations were designed and run a) for cases where the dynamics of TCP were frequently in effect, and b) for cases where the dynamics of TCP were not frequently in effect.

4.1 Network Model

The network model used here is shown in Figure 5. N TCP sources transmit over an IP WAN through a shared bottleneck node with capacity of μ bits per second and a FIFO buffer of size B packets to an equal number of TCP receivers. Importantly, the study was

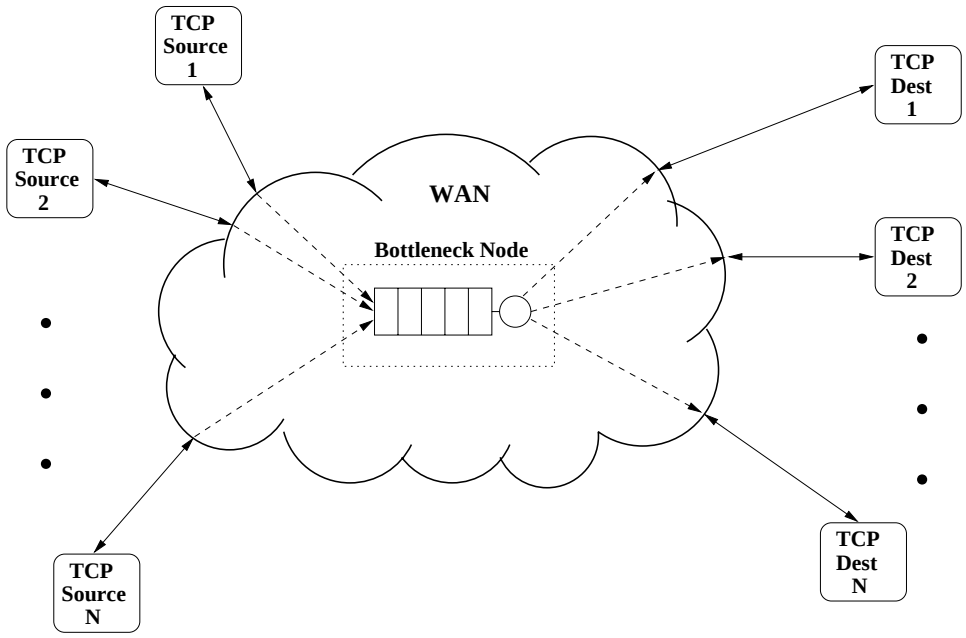

Figure 5: WAN Network Model with N TCP Traffic Flows via Bottleneck Node

focused on *TCP-Reno* which is currently the de facto standard implementation of TCP [42]. TCP-Reno implements Slow Start and Congestion Avoidance Schemes in the manner

described in Section 2.1.2. It also includes the Fast Retransmit, Fast Recovery[14], and Delayed Acknowledgements mechanisms. However, in some cases studied these mechanisms were disabled in order to detect the effect of Slow Start and Congestion Avoidance mechanisms on the TCP traffic's long-range dependency. When these mechanisms are disabled, we call this implementation of TCP as *TCP-Tahoe*. Also, several studies were conducted with UDP as the transport protocol to see if indeed the dynamics of TCP have any effect on long-range dependence in network traffic. The round-trip times (RTT) for each connection were set in the range from about 20 ms to about 450 ms. High RRTs results in longer Slow Start and Congestion Avoidance periods, and thus burstier traffic at the bottleneck node.

Several types of TCP and UDP connections were considered: a) connections with *greedy* sources, b) client/server connections (ON/OFF model): 1) with file sizes and OFF times exponentially distributed, 2) with file sizes Pareto distributed and OFF times exponentially distributed, and, 3) with files sizes and OFF times uniformly distributed. In the greedy-source cases, TCP layer had always data to send, i.e., the transmitted messages were infinity long. In the client/server cases, the client was modeled as "TCP Dest" and the server was modeled as "TCP Source". When UDP was used as the transport protocol, only client/server connections were considered.

In order to see the effect of TCP congestion control and avoidance on long-range dependency, it was necessary to create packet losses. Two methods for packet losses were used. First, by setting the bottleneck queue size small, packets were discarded due to queue overflow. Second, packets were discarded randomly (Random Loss) with a probability p drawn from a uniform distribution.

4.2 Simulation Model

The simulation models in this study were implemented using *BONeS DESIGNER*, a commercial software package for modeling and simulating event-driven systems [43]. The TCP BONeS module used in all simulation models was created for the study in [23]. Figure 6 shows the the top level of the simulation model created for the case of 64 TCP connections with greedy sources. The BONeS simulation model for a greedy TCP source is shown in Figure 7. For the client/server cases, the *TCP User* module was replaced by a *Client/Server User* module. The top model of the simulation model created for the case of 64 UDP

[14]See Section 2.1.3.

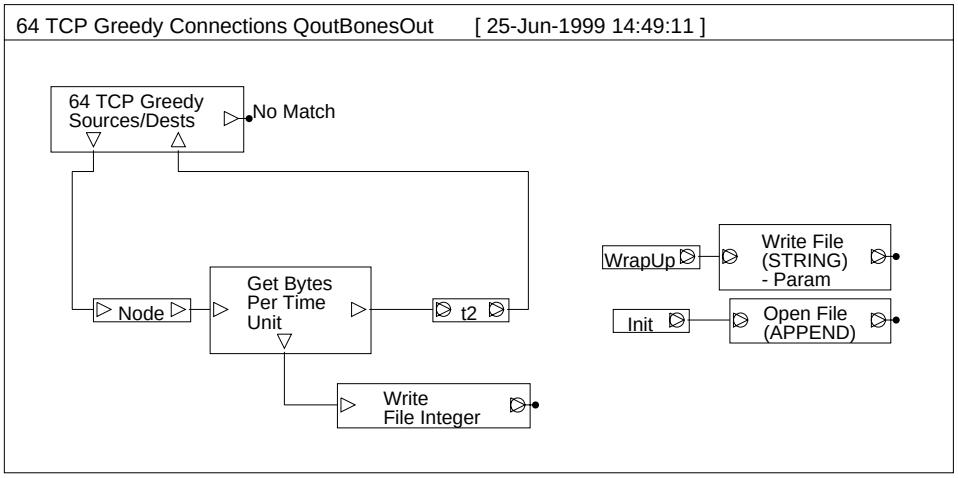

Figure 6: BONeS Simulation Model for 64 Greedy TCP Connections

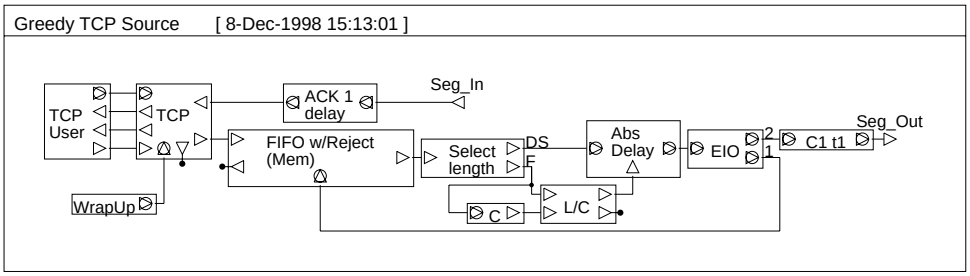

Figure 7: BONeS Module for a Greedy TCP Source

client/server connections with file sizes and OFF times exponentially distributed is shown in Figure 8 The maximum packet size for each simulation was set to 1000 bytes. Simulations were run for different number of TCP/UDP connections. The maximum number of connections in a simulation was 512.

In all the cases studied, the traffic flows were measured by collecting the number of bytes sent by all active sources (aggregated byte traffic) per aggregation time intervals (10 ms or greater) at the ingress (see Figure 8) or at the egress (see Figure 6) of the bottleneck node . As shown in Figures 6 and 8, the collection of byte counts was done by the "Get Bytes Per Time Unit" module. Each simulation was run for at least 10000 simulated seconds resulting in one million or more byte counts in the timeseries. The run times of each simulation on a SPARC Ultra-60 with 1 GByte of RAM were in the range from about 10 hours to several weeks. The intensity of long-range dependency (H) in each timeseries was estimated by the methods described in Sec. 3.5.

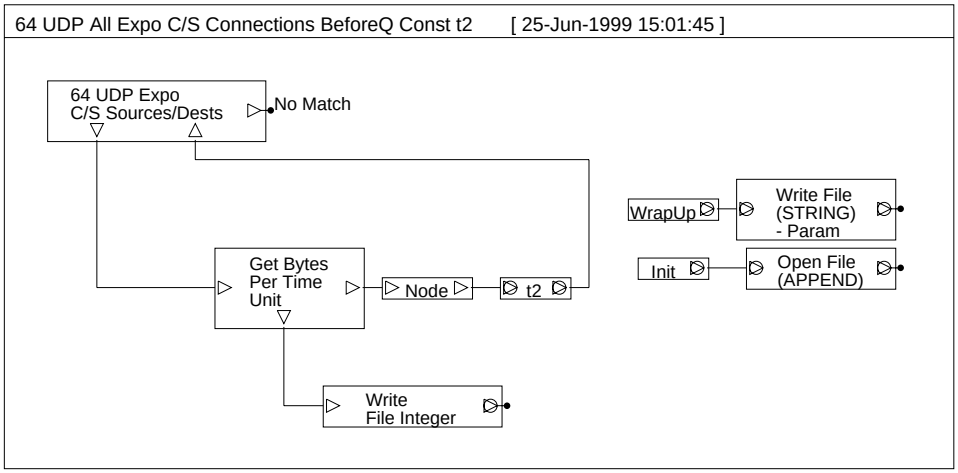

Figure 8: BONeS Simulation Model for 64 UDP Client/Server Connections

5 Results

5.1 Cases of Connections with Greedy Sources

Studies, [50, 34, 20, 10], have shown so far that some application-level characteristics (i.e., file-size distributions) can cause network traffic to be long-range dependent. The goal of this part of the study was to identify other factors contributing to long-range dependence in network traffic. This set of cases addresses TCP connections with greedy sources. In these cases, it is very clear that all application-level factors contributing to long-range dependency are eliminated. Specifically, the primary goal of these simulations was to validate our assumption that long-range dependence in aggregated TCP traffic can be induced by the dynamics of TCP.

5.1.1 Case of No Packet Losses

The congestion control and avoidance mechanism of TCP are activated only in response to packet losses. Without the effect of these TCP mechanisms, is the generated traffic with greedy sources long-range dependent? To answer the question, we run several simulations with different number of TCP greedy sources and system parameters. In all simulations there were no packet losses and round-trip times were kept constant. Therefore, the generated traffic was equivalent of a traffic generated by a constant bit rate (CBR) source. Obviously, such traffic patterns (constant rate) do not exhibit long-range dependence (LRD).

Part of the dynamics of TCP is flow control. The operation of flow control depends very much on the round-trip time (RTT). If the RTT is made random, does this make the traffic LRD? To answer this question, we ran a simulation in which the RTTs were uniformly distributed random variables in the range of 300 ms to 600 ms. The bit rate generated was fluctuating within a small range when byte aggregation was done every 10 ms, but it was fairly constant when the byte aggregation was done every 1 sec (Figure 9). Thus, the answer to the above question is no.

5.1.2 Case of Packet Losses Due to Queue Overflows

Does the activation of TCP's dynamics by packet losses give rise to long-range dependence (LRD) in TCP traffic? To answer this question, several simulations were performed with the effect of TCP's dynamics being the only factor that could contribute to the possible

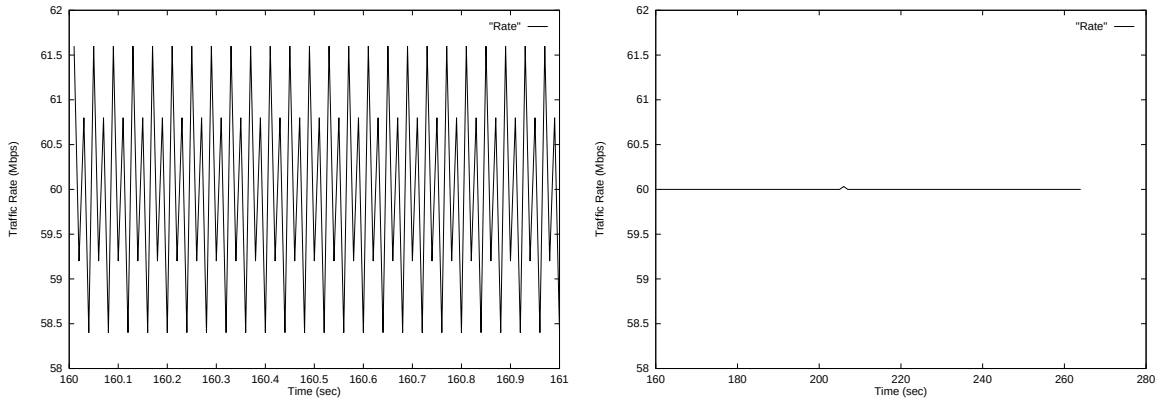

Figure 9: Data Rate vs Time for the Cases of 64 TCP Connections with Greedy Sources, No Packet Losses, and Random RTT. Left Plot) Aggregation Time = 10ms, Right Plot) Aggregation Time = 1 sec.

long-range dependency. The values of the most important TCP and network parameters are shown in Table 1. Table 2 shows the values of the estimated self-similar parameter $\hat{H}$ for the simulations described in Table 1. As shown in Table 2, for each simulation, the long-range intensity ($\hat{H}$) of the generated aggregated traffic was estimated for three different aggregation time intervals: 10 ms, 100ms and 1 sec. Let the traffic sequences formed by sampling the traffic every 10 ms, 100 ms and 1 sec be $X(n)$, $Y(i)$, and $Z(j)$, respectively. Clearly, $Y(i)$ and $Z(j)$ are aggregates of $X(n)$. Therefore, $Y(i)$ and $Z(j)$ should have the same LRD properties as $X(n)$. Looking the values of $\hat{H}$ estimated by both estimators, R/S and variance methods, in Table 2, we observe that in most cases seemingly $Z(j)$ doesn't have LRD. That means that the corresponding $X(n)$ and $Y(i)$ sequences should not have LRD either. However, the estimators show evidence of weak LRD in the traffic sequences of some cases.

Then, for the cases where the sequences $Z(j)$ are shown to have no LRD, why do both estimators show evidence of LRD for $X(n)$? Note that the definition of long-range dependence applies only to infinite time series. Here we attempt to estimate the intensity of long-range dependence in a time series by using only a finite number of samples. The question now is: how many samples required for a time series to get a good estimate of its LRD intensity, if there exists? Figures 10 and 11 show plots of $\hat{H}$ versus number of samples for $X(n)$ for the cases 19 and 10 described in Table 1. It is clear from these plots that for these traffic sequences a much greater number of samples is required to obtain more accurate values of

$\hat{H}$. Also, since not enough samples are collected for the sequences $Z(j)$, it is not possible by these results to conclude that for cases 2, 3, 4, 5, and 14, the traffic sequences exhibit LRD.

Case #	Number of Flows	Bottleneck Queue Size (packets)	TCP Max Window Size(KB)	Link Speeds (Mbps)	Bottleneck Link Speed (Mbps)	RTT (ms)
1	2	20	1000	134	100	30–50
2	2	50	1000	10	15	200–250
3	4	20	1000	134	100	30–50
4	6	40	1000	134	100	30–50
5	8	40	1000	134	100	30–50
6	8	60	1000	134	100	30–50
7	16	40	1000	134	100	30–50
8	16	60	1000	134	100	30–50
9	20	60	1000	134	100	30–50
10	24	60	1000	134	100	30–50
11	28	80	1000	134	100	30–50
12	64	30	100	100	80	30–50
13	64	100	100	100	80	30–50
14	64	300	100	100	80	30–50
15	64	100	64, 90, 300	1.5, 10, 20, 40, 98, 136	136	30–50
16	64	400	64	1.5, 10, 20, 40, 98, 136	136	30–50
17	64	50	1000	5	45	200–250
18	64	50	1000	5	45	20-30
19	64	100	1000	10	45	400-450
20	64	100	1000	10	45	400-450

Table 1: Run-Time Simulation Parameters for the Cases of TCP Connections with Greedy Sources Undergoing Packet Losses.

Case #	TCP Type	Data Collection Place	$X(n)$ Aggregation 10 ms Estimated		$Y(i)$ Time 100 ms $\hat{H}$		$Z(j)$ Interval 1 sec Parameter	
			R/S	Var	R/S	Var	R/S	Var
1	Tahoe	ingress	0.75	0.71	0.59	0.62	0.45	0.52
2	Reno	egress	–	–	–	–	0.67	0.67
3	Tahoe	ingress	0.80	0.73	0.73	0.67	0.66	0.59
4	Tahoe	ingress	0.76	0.71	0.69	0.65	0.61	0.55
5	Tahoe	ingress	0.76	0.72	0.69	0.69	0.63	0.58
6	Tahoe	ingress	0.72	0.65	0.61	0.54	0.52	0.43
7	Tahoe	ingress	0.70	0.66	0.64	0.58	0.56	0.50
8	Tahoe	ingress	0.71	0.64	0.61	0.56	0.54	0.47
9	Tahoe	ingress	0.71	0.66	0.64	0.59	0.58	0.52
10	Tahoe	ingress	0.69	0.65	0.62	0.58	0.57	0.51
11	Tahoe	ingress	0.65	0.53	0.57	0.52	0.54	0.49
12	Tahoe	ingress	0.61	0.56	0.57	0.52	0.52	0.48
13	Tahoe	ingress	0.62	0.57	0.56	0.53	0.55	0.52
14	Tahoe	ingress	0.64	0.66	0.59	0.65	0.59	0.69
15	Reno	ingress	0.60	0.57	0.56	0.53	0.57	0.50
		egress	0.60	0.50	0.55	0.49	0.56	0.48
16	Tahoe	ingress	0.69	0.62	0.60	0.55	0.56	0.50
17	Reno	egress	0.65	0.61	0.59	0.54	0.52	0.47
18	Reno	egress	0.63	0.58	0.55	0.54	0.57	0.55
19	Reno	egress	0.70	0.67	0.66	0.61	0.55	0.50
20	Tahoe	egress	0.71	0.64	0.62	0.55	0.52	0.45

Table 2: Estimated Values of H for the Cases of TCP Connections with Greedy Sources Undergoing Packet Losses and Described in Table 1.

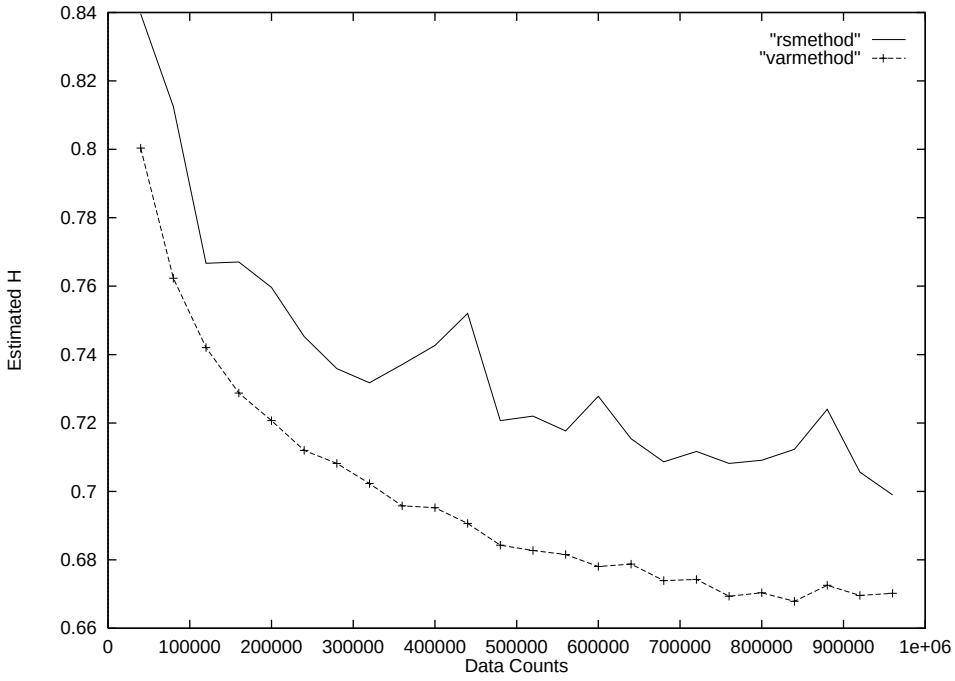

Figure 10: $\hat{H}$ vs Number of Samples for the Traffic Sequence $X(n)$ of Case 19 Described in Table 1

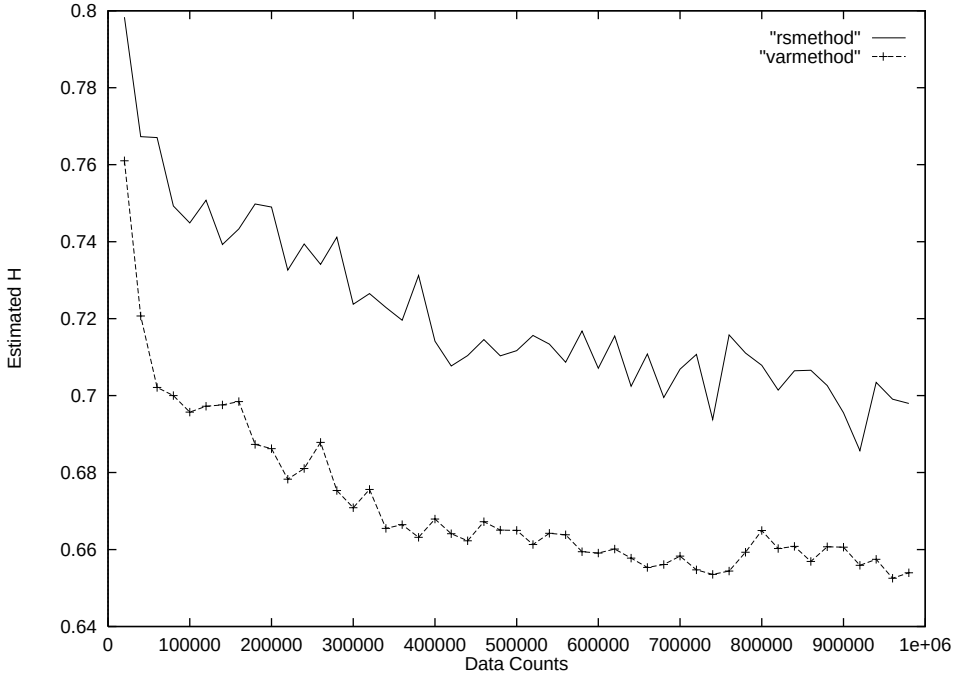

Figure 11: $\hat{H}$ vs Number of Samples for the Traffic Sequence $X(n)$ of Case 10 Described in Table 1

5.1.3 Case of TCP Connections with Greedy Sources Undergoing Random Packet Loss

TCP's dynamics (congestion control and avoidance) are in effect in response to packet loss regardless the source of loss. If TCP packets are discarded randomly within the network, does the aggregated TCP traffic exhibit long-range dependence? A study was conducted to resolve the above question. In all cases considered in this section, the queue size at the bottleneck node was set to infinity and the maximum TCP window size was set to 1 MByte.

Three set of simulations were performed. In the first set, packets were discarded randomly with a probability p from the aggregated traffic after the network node in consideration. Fig. 12 shows exactly the position ("Random Loss" module) where packets were getting lost in a random fashion. In the second set of simulations, all connections were combined

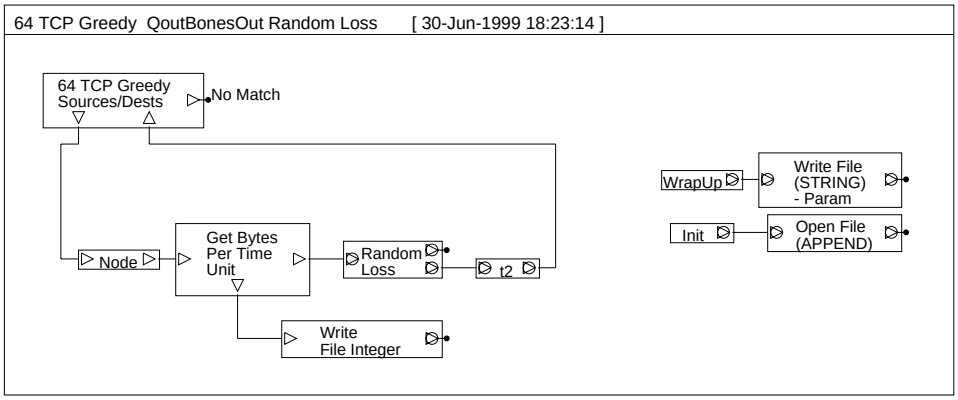

Figure 12: BONeS Simulation Model for 64 TCP Connections with Greedy Sources Undergoing Random Packet Loss.

in groups of two and packets were discarded randomly with a probability p per aggregated pair, as shown by Fig 13. All simulations in this set were carried out only for 16 connections (See Table 3, Simulation Numbers 9 to 13). For the last set of simulations in the case of random loss, connections were joined in groups of four and packets were dropped randomly with a probability p per group. All simulations in the last set were carried out only for 28 connections (See Table 3, Simulation Numbers 14 and 15). The values of the most important TCP and network parameters are shown in Table 3. Table 4 shows the values of the estimated self-similar parameter $\hat{H}$ for the simulations described in Table 3. Again, for each simulation, the long-range intensity ($\hat{H}$) of the aggregated traffic was estimated for

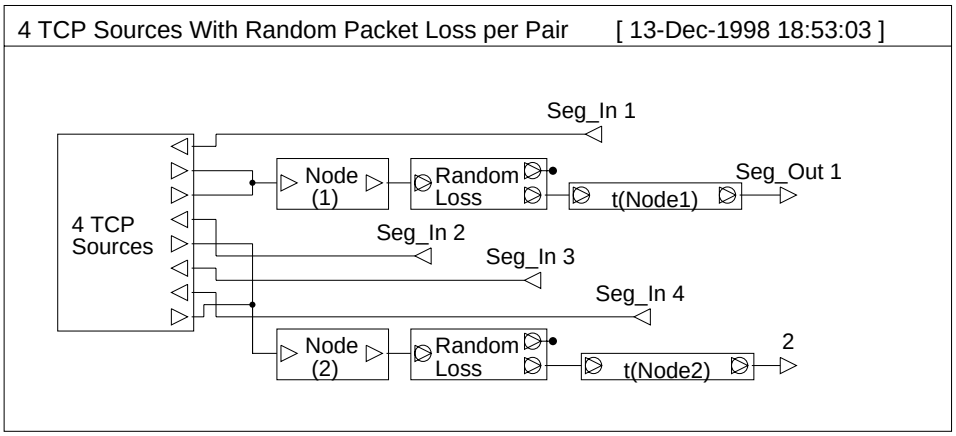

Figure 13: BONeS Simulation Module for 4 TCP Connections with Greedy Sources Undergoing Random Packet Loss per Aggregated Pair.

three different aggregation time intervals: 10 ms, 100ms and 1 sec. Analyzing the results of Table 4 and applying the same reasoning of the previous section, we arrive to the conclusion that for the cases considered in this section, the generated TCP traffic traces do not have long-range dependence.

Case #	Number of Flows	Packet Loss Probability	MTU (KB)	Link Speeds (Mbps)	Bottleneck Link Speed (Mbps)	RTT (ms)
1	2	0.01	1	134	100	30–50
2	2	0.001	1	134	100	30–50
3	2	0.0001	1	134	100	30–50
4	2	0.00001	1	134	100	30–50
5	28	0.01	1	134	100	30–50
6	28	0.001	1	134	100	30–50
7	64	0.001	8	134	134	30–50
8	64	0.001	1, 1.5 8, 9	1, 2, 10, 15, 34, 40, 50, 70, 90, 100, 110, 120, 134	134	30–50
		Per 2 Flows				
9	16	0.0 (2 Pairs) 0.01 (6 pairs)	1	134	134	30–50
10	16	0.01	1	134	134	30–50
11	16	0.01, 0.05, 0.001, 0.0001	1	134	134	30–50
12	16	0.01	1, 4, 8, 9	134	134	30–50
13	16	0.01, 0.05, 0.001, 0.0001	1, 4, 8, 9	134	134	30–50
		Per 4 Flows				
14	28	0.01	1	134	134	30–50
15	28	0.001	1	134	134	30–50

Table 3: Run-Time Simulation Parameters for the Cases of TCP Connections with Greedy Sources Undergoing Random Packet Losses.

Case #	TCP Type	Data Collection Place	$X(n)$ Aggregation 10 ms Estimated $\hat{H}$ R/S	Var	$Y(i)$ Time 100 ms $\hat{H}$ R/S	Var	$Z(j)$ Interval 1 sec Parameter R/S	Var
1	Tahoe	ingress	0.61	0.57	0.57	0.51	0.53	0.43
2	Reno	ingress	0.65	0.60	0.59	0.53	0.50	0.41
3	Tahoe	ingress	0.71	0.59	0.58	0.49	0.43	0.31
4	Tahoe	ingress	0.79	0.65	0.67	0.53	—	—
5	Tahoe	ingress	0.58	0.52	0.52	0.46	0.46	0.36
6	Tahoe	ingress	0.63	0.55	0.55	0.46	0.43	0.33
7	Tahoe	ingress	0.46	0.35	0.41	0.26	0.37	0.15
8	Tahoe	ingress	0.45	0.33	0.42	0.25	0.36	0.14
9	Tahoe	ingress	0.34	0.44	0.36	0.47	0.35	0.46
10	Tahoe	ingress	0.61	0.57	0.55	0.50	0.50	0.43
		egress	0.61	0.57	0.55	0.50	0.50	0.43
11	Tahoe	ingress	0.71	0.56	0.63	0.51	0.51	0.38
12	Tahoe	ingress	0.63	0.58	0.58	0.51	0.52	0.44
13	Tahoe	ingress	0.73	0.66	0.66	0.57	0.54	0.45
14	Tahoe	ingress	0.59	0.54	0.54	0.47	0.48	0.39
15	Tahoe	ingress	0.64	0.56	0.56	0.46	0.44	0.32

Table 4: Estimated Values of H for the Cases of TCP Connections with Greedy Sources Undergoing Random Packet Losses and Described in Table 3.

5.2 Cases of Client/Server Connections (ON/OFF Model)

We saw in the previous sections that the dynamics of TCP alone may not give rise to LRD in TCP traffic. The simulation study performed in [34] on a specific client/server scenario shows that the presence of long-range dependence in network traffic depends on whether reliable and flow-controlled communication (such as the dynamics of TCP) is employed at the transport layer. Is this true in general? In this section, simulations were performed to re-evaluate a) the effect of different file size's distributions on the long-range dependency and b) whether the dynamics of TCP have any effect on the intensity of LRD in a client/server network environment. Several simulations were run with TCP Reno, TCP Tahoe, and UDP as the transport protocols.

5.2.1 Case of File Sizes with Heavy-tailed Distribution

Several studies have already shown that when the distribution of file sizes being transferred is heavy-tailed, then the aggregation of many such connections results in a long-range dependent network traffic [50, 34, 20, 10]. However, it was necessary to validate this observation with our network model. In this set of simulations, the file sizes were Pareto distributed with the heavy-tailed parameter α set to 1.06 and the OFF times were exponentially distributed. The run-time simulation parameters for the cases considered here are shown in Table 5. Note that simulations 1, 2, and 5 were run long enough to collect 14, 75, and 92 hours of continuous traffic sequences, respectively. These sequences are much greater than any empirical traffic trace study published so far. Adopting the notation defined in Section 5.1.2, it is very clear by the values of $\hat{H}$ shown in Table 6, $X(n)$, $Y(i)$, and $Z(j)$ exhibit strong LRD properties. That is, the traffic sequences generated by the simulations performed in this section have strong long-range dependence.

Figure 14 shows a plot of $\hat{H}$ as a function of the number of samples for the traffic sequence $X(n)$ generated by the case 1. Despite the relative small fluctuations, both estimators seem to converge to a value well greater than 0.75, suggesting strong LRD.

Do the dynamics of TCP have any effect on the intensity of LRD? As shown by Table 6 three cases were considered with TCP Reno as the transport, two with UDP, and four with TCP Tahoe. The values of $\hat{H}$ between the two TCP flavors do not significantly vary. Disabling the slow start mechanism in case 7 had no effect on the intensity of LRD. Comparing the values of $\hat{H}$ between the UDP and TCP cases, we see no significant variation.

Case #	Number of Flows	Mean File Size	Mean OFF Time	Packet Loss	TCP Max Window Size (KB)	Link Speeds (Mbps)	Bottleneck Link Speed (Mbps)	RTT (ms)
	14	Hours	Traffic					
1	64	4 KB	600 ms	Yes	1000	20	10	200–250
	75	Hours	Traffic					
2	64	4 KB	600 ms	No	1000	134	622	200-250
3	64	4 KB	600 ms	No	1000	Inf	Inf	200-250
4	64	4 KB	600 ms	No	—	10	100	—
	92	Hours	Traffic					
5	64	4 KB	600 ms	No	—	134	622	—
6	64	4 KB	600 ms	No	64	10	134	30–50
7	64	4 KB	600 ms	No	64	10	134	30–50
8	64	100 KB	1 sec	No	64	10	134	30–50
9	512	4 KB	600 ms	No	64	10	134	30–50

Table 5: Run-Time Simulation Parameters for the Cases of ON/OFF Traffic Sources with File Sizes Pareto Distributed and OFF Times Exponentially Distributed.

Note that the values of $\hat{H}$ listed for case 5 (UDP case) are for a 92-hour traffic trace, and comparing them with the 75-hour traffic trace of case 2 (TCP Reno case) we can see that there is not much difference. Extending the 75-hour traffic sequence to 92 hours, as the UDP case, the values of $\hat{H}$ for both would probably be similar. Also, the small difference in $\hat{H}$ values between cases 2 and 5 can be due to the fluctuation of $\hat{H}$ versus number of samples.

Comparing a section of the traffic sequence $X(n)$ between cases 2 and 5 shown in Figure 15 by the plots (e) and (f), we can observe that the traffic pattern of the UDP case, case 5, resembles more to a typical plot of file size versus number of observations when generated by a Pareto distribution with mean 4 KB and $\alpha = 1.06$ than the traffic pattern of case 2 (TCP case) (see Figure 1 in [11]). We believe that the traffic pattern of case 2 shown by the plot (e) in Figure 15 is altered by the dynamics of TCP without affecting the LRD properties. Plots (c) and (d) in Figure 15 compare the autocorrelation functions (ACF) of $X(n)$ for cases 2 and 5, respectively. The ACF of plot (c) shows evidence of periodicity in

Case #	Transport Protocol Type	Data Collection Place	$X(n)$ Aggregation 10 ms Estimated $\hat{H}$ R/S	Var	$Y(i)$ Time 100 ms $\hat{H}$ R/S	Var	$Z(j)$ Interval 1 sec Parameter R/S	Var
1	Reno	egress	0.84	0.85	0.86	0.85	0.82	0.83
2	Reno	egress	0.82	0.83	0.82	0.81	0.77	0.75
3	Reno	egress	0.81	0.92	0.85	0.92	0.76	0.89
4	UDP	egress	0.87	0.88	0.85	0.86	0.81	0.84
5	UDP	egress	0.79	0.78	0.76	0.73	0.71	0.67
6	Tahoe	ingress	0.86	0.91	0.81	0.89	0.79	0.86
7	Tahoe:	Disabled	Slow	Start				
		ingress	0.85	0.90	0.82	0.89	0.79	0.86
8	Tahoe	ingress	0.93	0.92	0.89	0.90	0.86	0.89
9	Tahoe	ingress	0.88	0.89	0.84	0.87	0.81	0.84

Table 6: Estimated Values of H for the Cases of ON/OFF Traffic Sources with File Sizes Pareto Distributed and OFF Times Exponentially Distributed and Described in Table 5.

TCP traffic (as expected), but the ACF of plot (d) shows no evidence of periodicity in UDP traffic. Obviously, the periodicity observed in TCP traffic is caused by the dynamics of TCP. Do the dynamics of TCP always cause a periodicity in TCP traffic? As we can see from the plots of Figure 16 the answere is no. Comparing the ACF of plot (d) in Figure 15 with the ACF plots in Figure 16 and the round-trip times (RTT) listed by the last column of Table 5 we can say that the periodicity in TCP traffic is caused by the window flow control of TCP when the round-trip times are very large. The plots generated by the R/S statistics are shown by the plots (a) and (b) in Figure 15 for the cases 2 and 5, respectively.

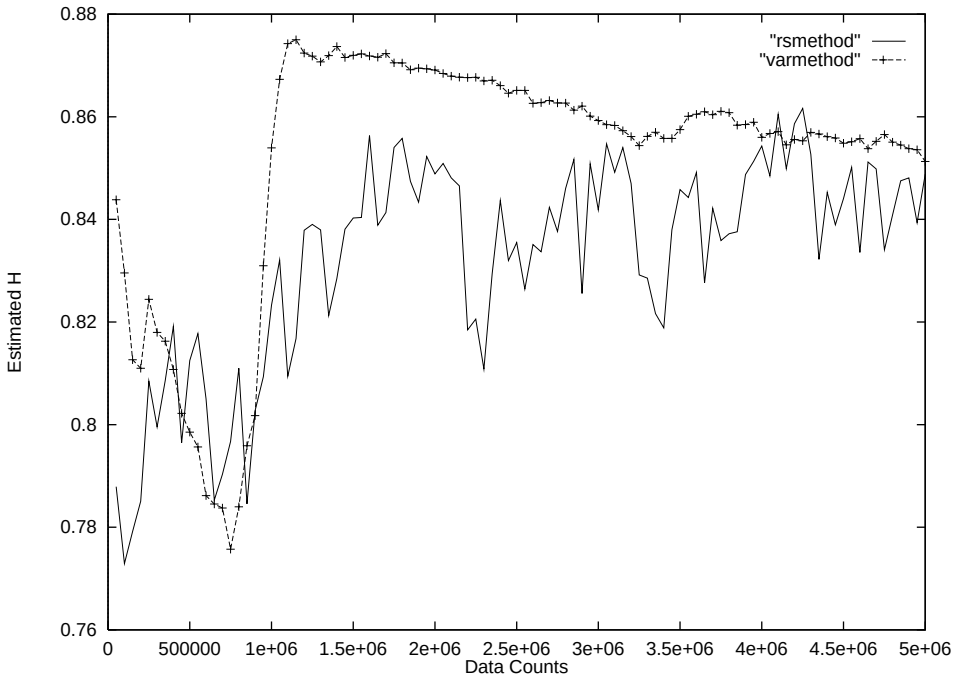

Figure 14: $\hat{H}$ vs Number of Samples for the Traffic Sequence $X(n)$ of Case 1 Described in Table 5

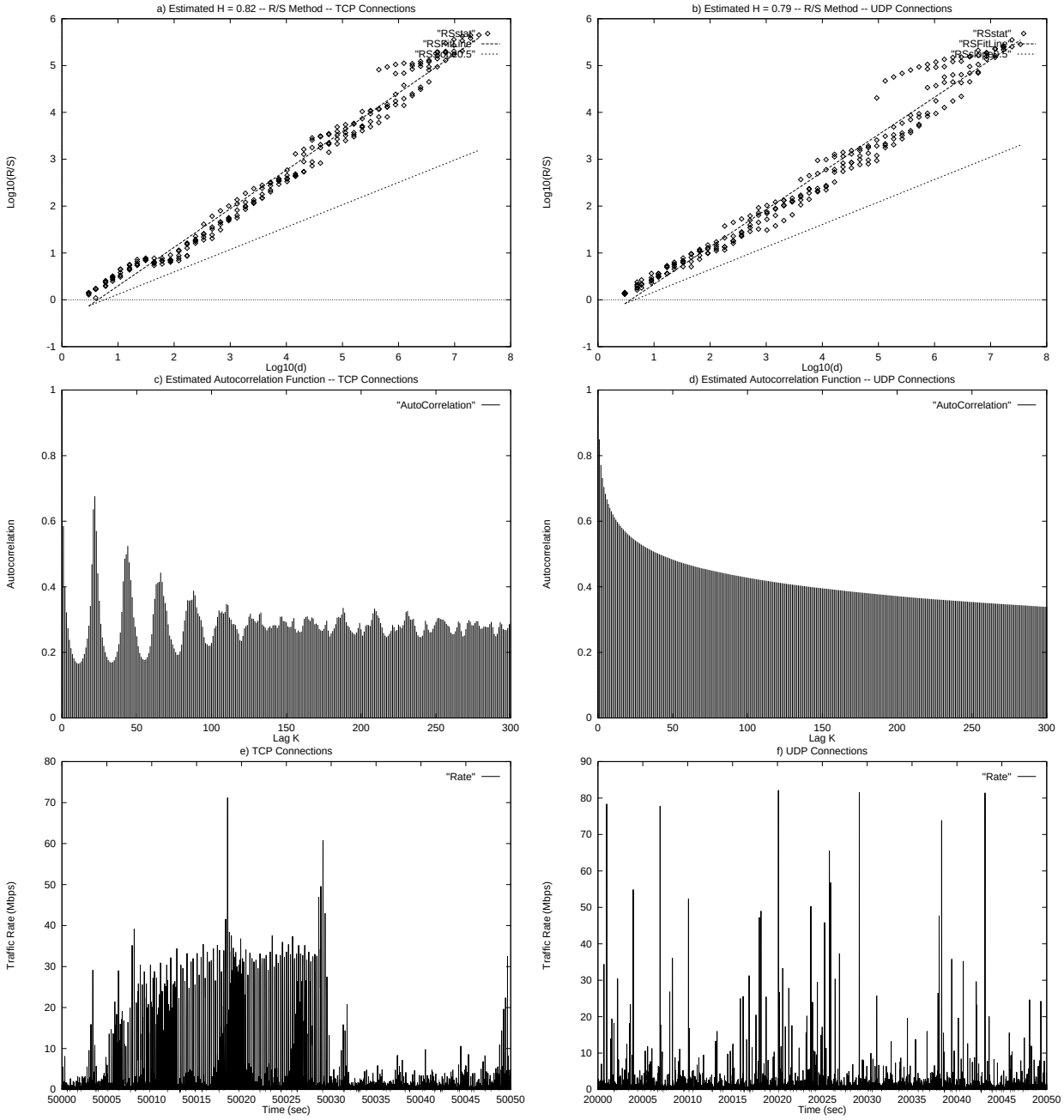

Figure 15: Plots for Cases 2 and 5 Described in Table 5. Case 2: plots (a), (b), and (e). Case 5: plots (b), (d), and (f). Plots (a) and (b): pox plots of R/S Estimator. Plots (c) and (d): Estimated ACFs. Plots (e) and (f): Sections of Traffic Showing the Rate (Mbps) vs Time. All Plots were Generated by the Traffic Sequences $(X(n)s)$ Collected Every 10 ms Intervals.

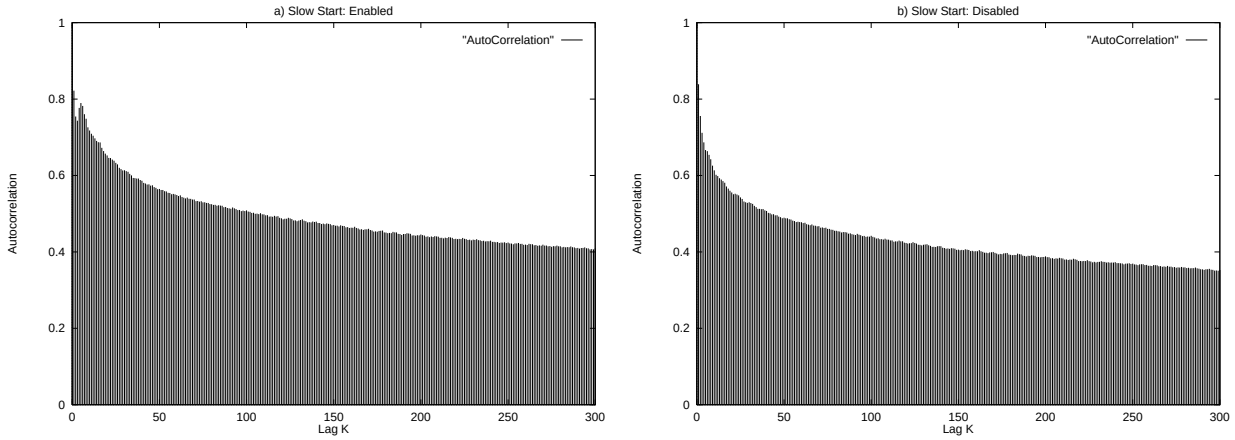

Figure 16: Estimated Autocorrelation Functions of $X(n)$ for Cases 6 (plot a) and 7 (plot b) Described in Table 5.

5.2.2 Case of File Sizes and OFF Times Exponentially Distributed

In the previous section we saw that when the distribution of file sizes being transferred is heavy-tailed, then the aggregation of many such connections results in a network traffic that exhibits long-range dependence. What happens when the distribution of file sizes being transferred is exponentially? The simulation study in [34] shows that when files sizes are exponentially distributed with mean 4 KB and the OFF times are also exponentially distributed with mean 600 ms, the aggregated traffic does not have LRD. In this section, we study the case of with TCP (or UDP) client/server connections with both file sizes and OFF times exponentially distributed, but with different and much higher means.

Case #	Number of Flows	Mean File Size	Mean OFF Time	Packet Loss	TCP Max Window Size (KB)	Link Speeds (Mbps)	Bottleneck Link Speed (Mbps)	RTT (ms)
1	64	50 KB	0.6 s	Yes	1000	20	10	200–250
2	64	100 KB	10 s	No	1000	5	134	200–250
3	64	100 KB	240 s	No	1000	5	134	200–250
4	64	100 KB	240 s	No	1000	5	134	200–250
5	64	1 MB	240 s	No	1000	5	134	200–250
6	64	10 MB	240 s	No	1000	5	134	200–250
7	64	10 MB	240 s	Yes	1000	5	134	20–30
8	64	10 MB	240 s	Yes	1000	5	134	200–250
9	64	5 MB	360 s	No	1000	5	45	200–250
10	64	4 KB	0.6 s	No	–	5	45	–
11	64	1 MB	120 s	Yes	–	5	45	–
12	64	2 MB	120 s	Yes	–	5	45	–
13	64	10 MB	240 s	Yes	–	5	45	–
14	64	10 MB	240 s	Yes	–	5	134	–
15	64	10 MB	240 s	Yes	–	5	134	–
16	64	5 MB	360 s	Yes	–	5	45	–

Table 7: Run-Time Simulation Parameters for the Cases of ON/OFF Traffic Sources with File Sizes and OFF Times Exponentially Distributed.

Table 7 lists the run-time simulation parameters for 16 different cases. In each case, all 64 TCP or UDP connections were identical. The estimated values of the H parameter are listed in Table 8. Figure 17 compares case 4 described in Table 5 with case 10 described in Table 7 in terms of the variance-time curve, autocorrelation function, and aggregated traffic rate. As we can see, although in both cases the files sizes and OFF times have the same means, the characteristics of the aggregated traffic are different. In addition, this verifies again the fact that when files sizes are exponentially distributed with mean 4 KB and the OFF times are also exponentially distributed with mean 600 ms, the aggregated traffic does not exhibit LRD. It is clear by the results in Table 8 that the traffic sequences generated by cases 1, 2, 3, 4, 10, and 11 do no exhibit LRD. However, both estimators show evidence of LRD in the other traffic sequences generated by the rest of the cases listed in Table 8. Case 8 was run long enough to generate a 27-hour traffic sequence, again much longer that all previously reported results.

We considered nine more cases, but this time not all connections were similar. Connections were splitted in two or more groups where in each group all connections were identical. In the first group, the mean file size was set to 4 KB and the mean OFF time set to 600 ms. In the other groups the means were set to much higher values. The run-time simulation parameter are shown in Table 9, and the corresponding estimated values of H are shown in Table 10. Again, results show that the traffic sequences generated by these cases have LRD. Note also that cases 6 to 9 were run long enough to collect 92-, 60-, 49-, and 18-hour traffic traces, respectively.

The results of Table 10 are unexpected. The common conception is that the superposition of many ON/OFF traffic sources whose ON-periods or/and OFF-periods have heavy-tailed probability distribrution functions yields an aggregate network traffic that exhibits long-range dependence. Here are cases of LRD where both ON and OFF times are not heavy-tailed. We then plotted sections of traffic rates at different time scales (0.01, 1, and 10 seconds) for the traffic sequences generated by cases 4 and 5 of Table 9 (exponential cases, means 4 KB and 5 MB) and compared them with similar plots constructed by case 4 of Table 5 (heavy-tailed case, mean 4 KB). Comparing the traffic patterns shown in Figures 18 and 19, we observe that the traffic patterns created by the exponential cases are very similar with the traffic patterns that were created by the heavy-tailed case. Also, we can see clearly the presence of high burstiness in traffic pattern of case 4 of Table 9 at all three different time scales. Importantly, both cases were run with UDP as the transport protocol.

This shows again that the presence of long-range dependence in traffic does not necessarily depend on whether a reliable and flow- and congestion-controlled protocol is employed at the transport layer. The plots (c)-(f) of Figure 19 compare the two cases in terms of the plots of the R/S estimator and autocorrelation functions.

Further, the variance-time plot, autocorrelation function, and traffic patterns at four different time scales (0.01, 1, 10, and 100 seconds) for case 7 (exponential case) of Table 9 are compared with those of case 2 (heavy-tailed case) of Table 5 in Figures 20 and 21. Both cases used with TCP Reno as the transport protocol. Although the variance-time plots and autocorrelation functions look very similar, the traffic patterns do not. We expected the traffic patterns of the two cases to be different since in the exponential case there were more connections (128) and the source link speeds were much lower than those in the heavy-tailed case. Despite the fact that the traffic patterns between the two cases look very different, this shows that the traffic for the exponential case is not smoothed out at high time scales (10 and 100 seconds) which it is consistent with the presence of long-range dependence. This also shows that even though the values of LRD intensity (H) of two traffic streams might be similar, their pattern could be very different.

Case #	Transport Protocol Type	Data Collection Place	$X(n)$ Aggregation 10 ms Estimated R/S	Var	$Y(i)$ Time 100 ms $\hat{H}$ R/S	Var	$Z(j)$ Interval 1 sec Parameter R/S	Var
1	Reno	egress	0.62	0.59	0.62	0.57	0.60	0.55
2	Reno	egress	0.66	0.64	0.62	0.57	0.56	0.51
3	Reno	egress	0.68	0.66	0.62	0.59	0.55	0.52
	Disabled	Slow	Start					
4	Reno	egress	0.63	0.62	0.57	0.54	0.54	0.49
5	Reno	egress	0.79	0.78	0.71	0.70	0.63	0.60
6	Reno	egress	0.94	0.90	0.88	0.85	0.80	0.79
7	Reno	egress	0.96	0.90	0.85	0.85	0.79	0.79
	27	Hours	Traffic					
8	Reno	egress	0.95	0.81	0.81	0.76	0.72	0.68
9	Reno	egress	0.95	0.86	0.82	0.81	0.73	0.73
10	UDP	egress	0.55	0.51	0.54	0.49	0.56	0.49
11	UDP	egress	0.85	0.76	0.73	0.68	–	–
	227	Hours	Traffic					
			–	–	–	–	0.57	0.53
	227	Hours	Traffic					
12	UDP	egress	–	–	–	–	0.56	0.50
13	UDP	egress	0.98	0.92	0.90	0.84	0.71	0.65
14	UDP	egress	0.99	0.90	0.88	0.85	0.80	0.78
15	UDP	ingress	0.96	0.94	0.84	0.85	0.79	0.78
16	UDP	egress	0.98	0.85	0.86	0.80	0.74	0.70

Table 8: Estimated Values of H for the Cases Described in Table 7 of ON/OFF Traffic Sources with File Sizes and OFF Times Exponentially Distributed.

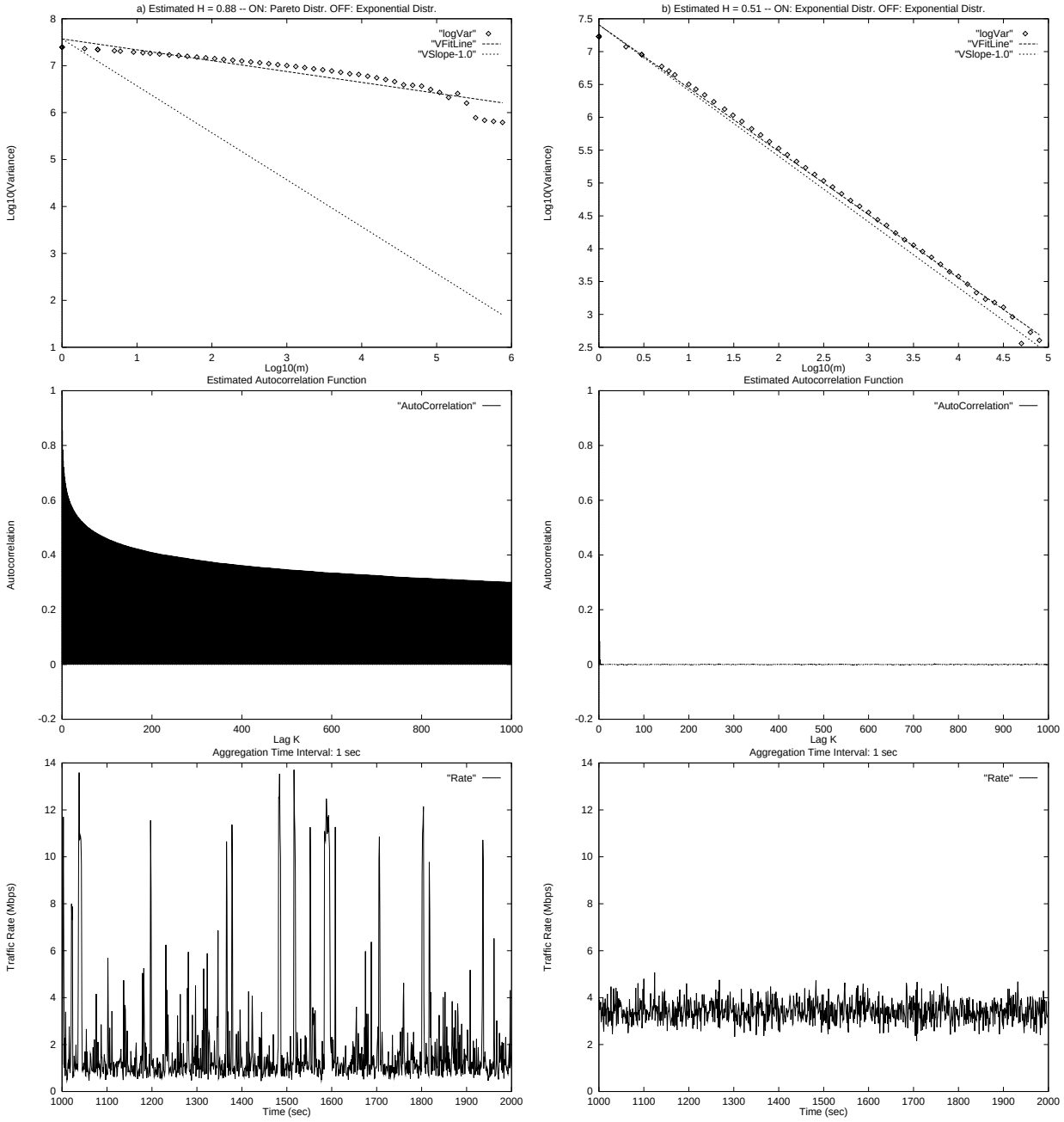

Figure 17: Comparison of Case 4 – Table 5 (a) With Case 10 – Table 7 (b) in Terms of Estimated $\hat{H}$, Estimated Autocorrelation Function, and Aggregated Traffic Rate.

Case #	Number of Flows	Mean File Size	Mean OFF Time	Packet Loss	TCP Max Window Size (KB)	Link Speeds (Mbps)	Bottleneck Link Speed (Mbps)	RTT (ms)
1	56	4 KB	0.6 s	No	1000	5	45	200–250
	8	2 MB	240 s					
2	56	4 KB	0.6 s	No	1000	5	134	200–250
	8	10 MB	240 s					
3	32	4 KB	0.6 s	No	1000	5	134	200–250
	32	10 MB	240 s					
4	56	4 KB	0.6 s	No	–	5	45	–
	8	5 MB	360 s					
5	32	4 KB	0.6 s	Yes	–	5	45	–
	32	5 MB	360 s					
6	32	4 KB	0.6 s	No	1000	Inf	Inf	200–250
	32	10 MB	240 s					
7	64	4 KB	0.6 s	No	64	10	134	200-250
	64	10 MB	900 s					
8	64	4 KB	0.6 s	No	1000	10	134	200-250
	64	10 MB	900 s					
9	128	4 KB	0.6 s	No	64	10	134	200-250
	64	1 MB	240 s					
	64	10 MB	900 s					

Table 9: Run-Time Simulation Parameters for the Cases of ON/OFF Traffic Sources with File Sizes and OFF Times Exponentially Distributed. Cases of Two or More Connection Groups.

Case #	Transport Protocol Type	Data Collection Place	$X(n)$ Aggregation 10 ms Estimated R/S	Var	$Y(i)$ Time 100 ms $\hat{H}$ R/S	Var	$Z(j)$ Interval 1 sec Parameter R/S	Var
1	Reno	egress	0.82	0.70	0.81	0.73	0.70	0.64
2	Reno	egress	0.93	0.88	0.89	0.82	0.79	0.74
3	Reno	egress	0.93	0.88	0.86	0.83	0.78	0.74
4	UDP	egress	0.84	0.82	0.84	0.79	0.76	0.73
5	UDP	egress	0.91	0.86	0.86	0.80	0.75	0.71
6	Reno 92 Hours	egress	0.84 Traffic	0.82 –	0.93 –	0.83 –	– 0.70	– 0.66
7	Reno 60 Hours	egress	0.90 Traffic	0.87 –	0.93 –	0.84 –	– 0.74	– 0.71
8	Reno 49 Hours	egress	0.91 Traffic	0.83 –	0.85 –	0.79 –	– 0.68	– 0.61
9	18 Hours Reno	Traffic egress	–	–	0.85	0.79	0.76	0.73

Table 10: Estimated Values of H for the Cases Described in Table 9 of ON/OFF Traffic Sources with File Sizes and OFF Times Exponentially Distributed. Cases of Two or More Connection Groups.

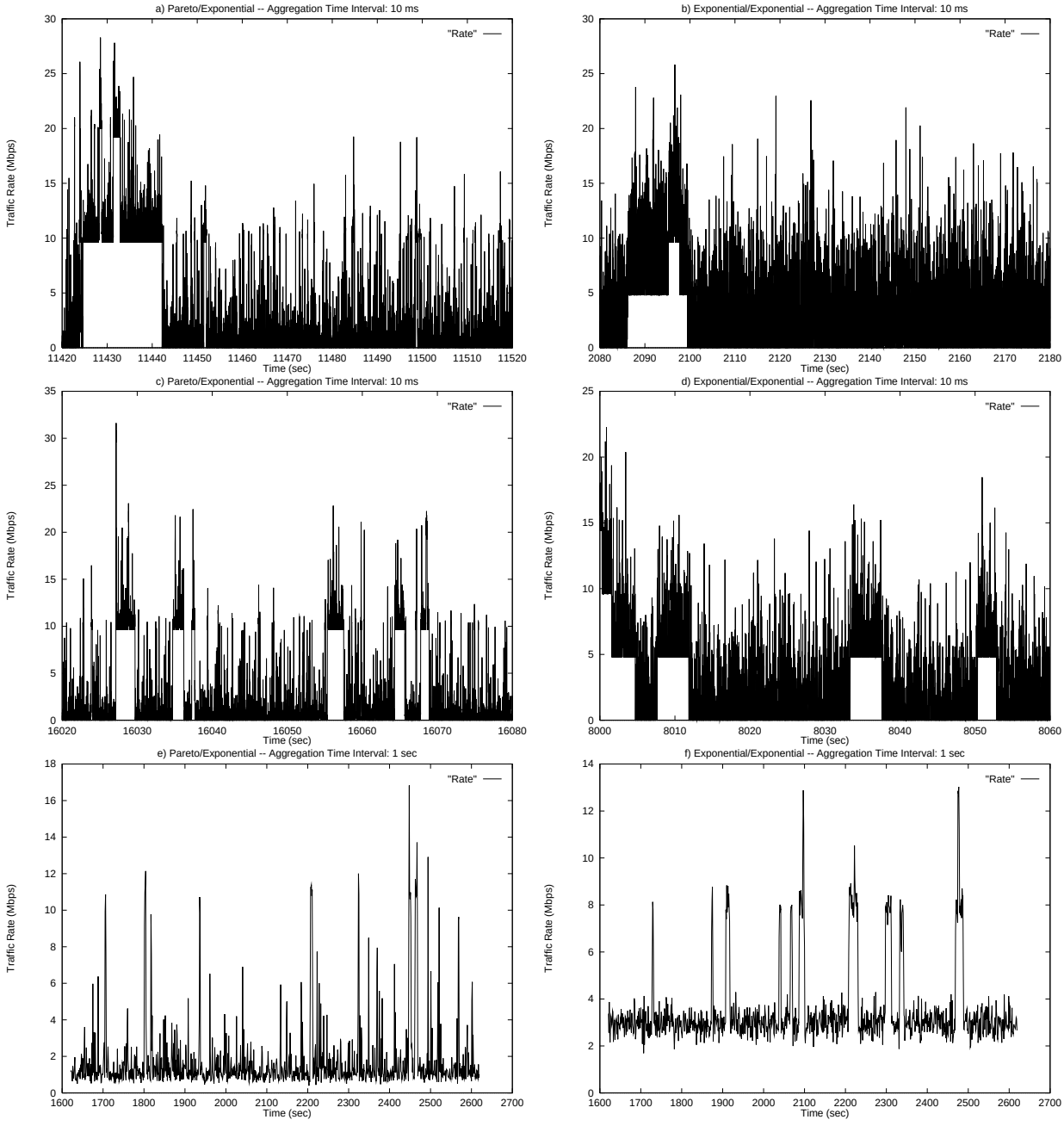

Figure 18: Comparison of Case 4 Described in Table 5 With Cases 4 and 5 Described in Table 9 in Terms of Aggregated Traffic Rate Patterns. Plots (a), (c), & (e): Case 4 of Table 5. Plots (b), & (f): Simulation 4 of Table 9. Plot (d): Case 5 of Table 9.

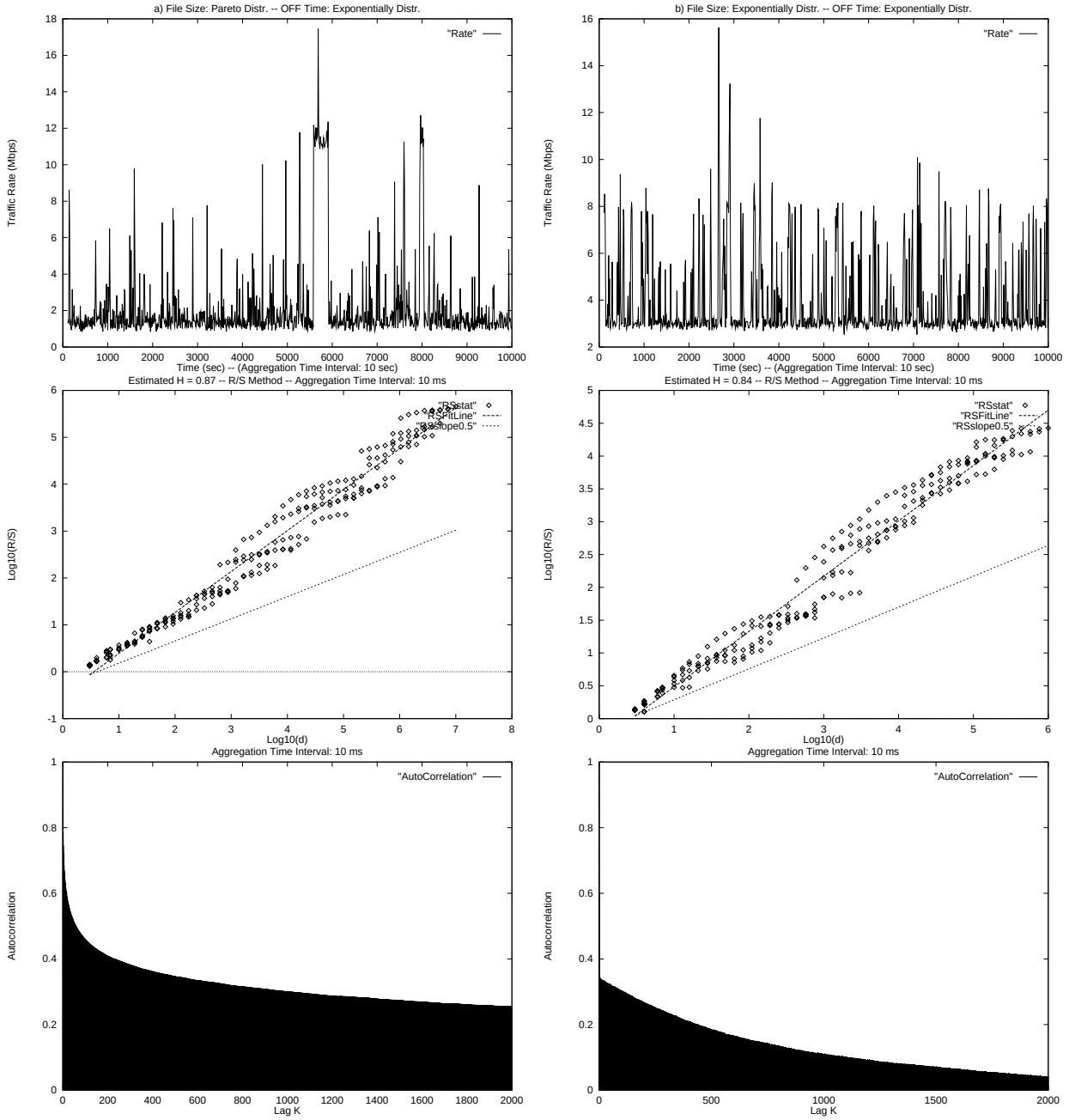

Figure 19: Comparison of Case 4 (a) Described in Table 5 With Case 4 (b) Described in Table 9 in Terms of 1) Aggregated Traffic Rate Patterns (ATI = 10 sec), 2) Estimation of H using the R/S Method (ATI = 10 ms), and 3) Estimated Autocorrelation Function (ATI = 10 ms)– (ATI: Aggregated Time Interval).

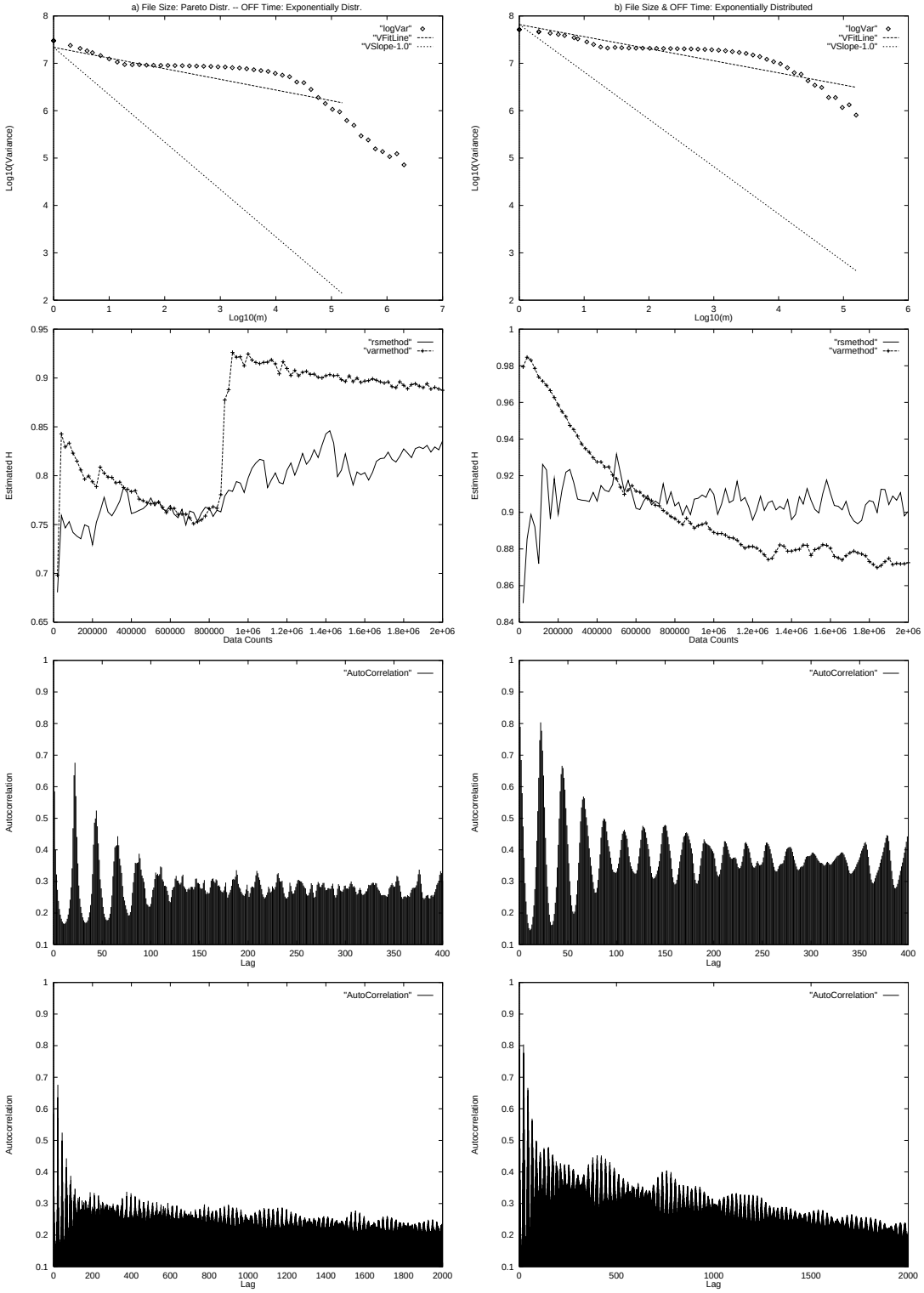

Figure 20: Statistic Plots Created by (a) Case 2 Described in Table 5 (b) Case 7 Described in Table 9 – Aggregation Time Interval: 10 ms

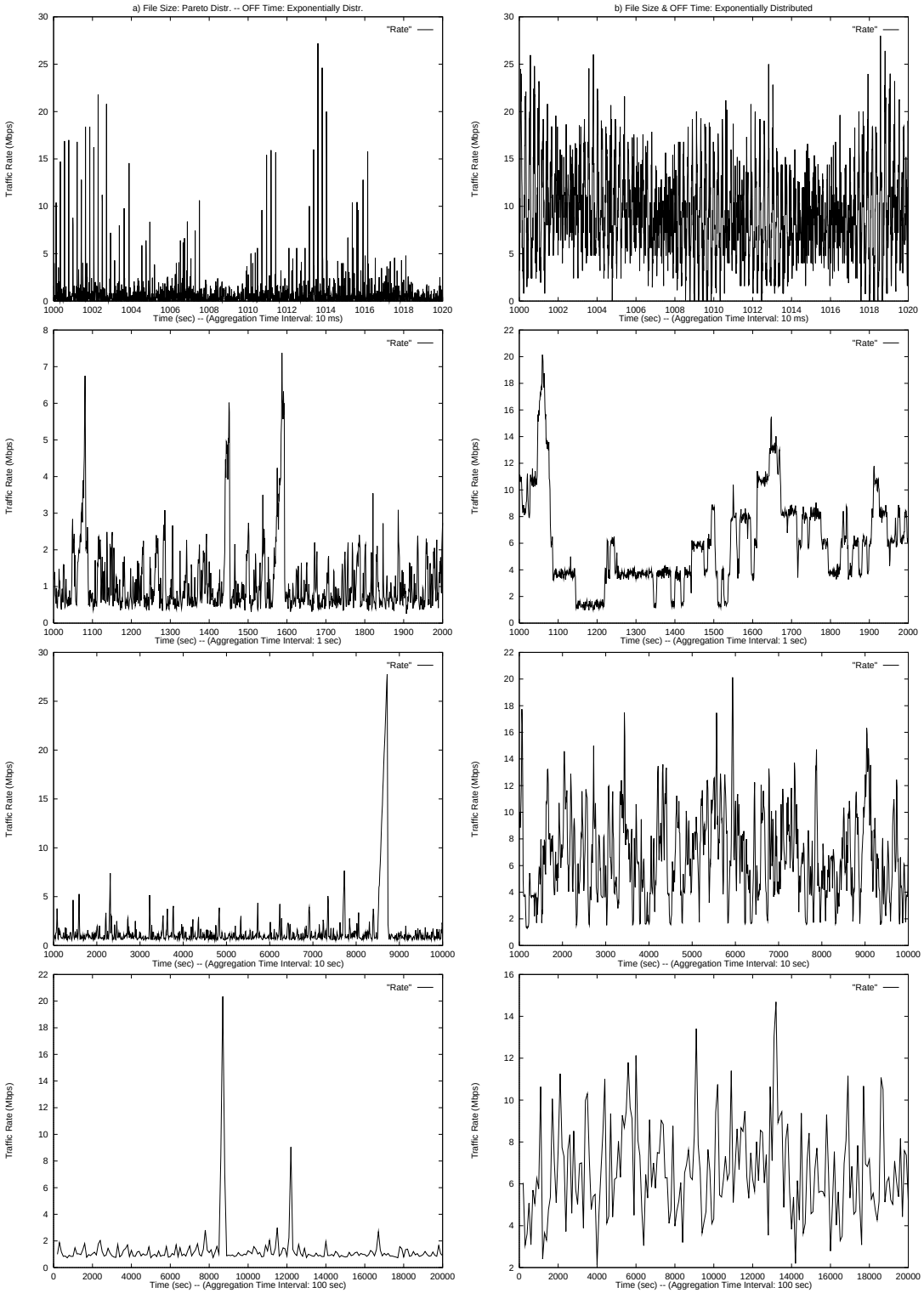

Figure 21: Traffic Patterns at Different Time Scales Generated by (a) Case 2 Described in Table 5 and (b) Case 7 Described in Table 9

5.2.3 Case of File Sizes and OFF Times Uniformly Distributed

In the previous section, we observed cases in which the file sizes and OFF times were exponentially distributed and both variance and R/S estimator showed evidence that the aggregated traffic had LRD. In this section we considered seven cases having the file sizes and OFF times uniformly distributed. The run-time simulation parameters are shown in Table 11 and the corresponding estimated values of H are presented in Table 11. Obviously from the results of Table 11, again, both estimators suggest the presence of LRD in traffic formed by connections whose file sizes and OFF times are uniformly distributed. Figure 22 shows plots of the estimated partial autocorrelation function for cases 1, 2, 5, and 6.

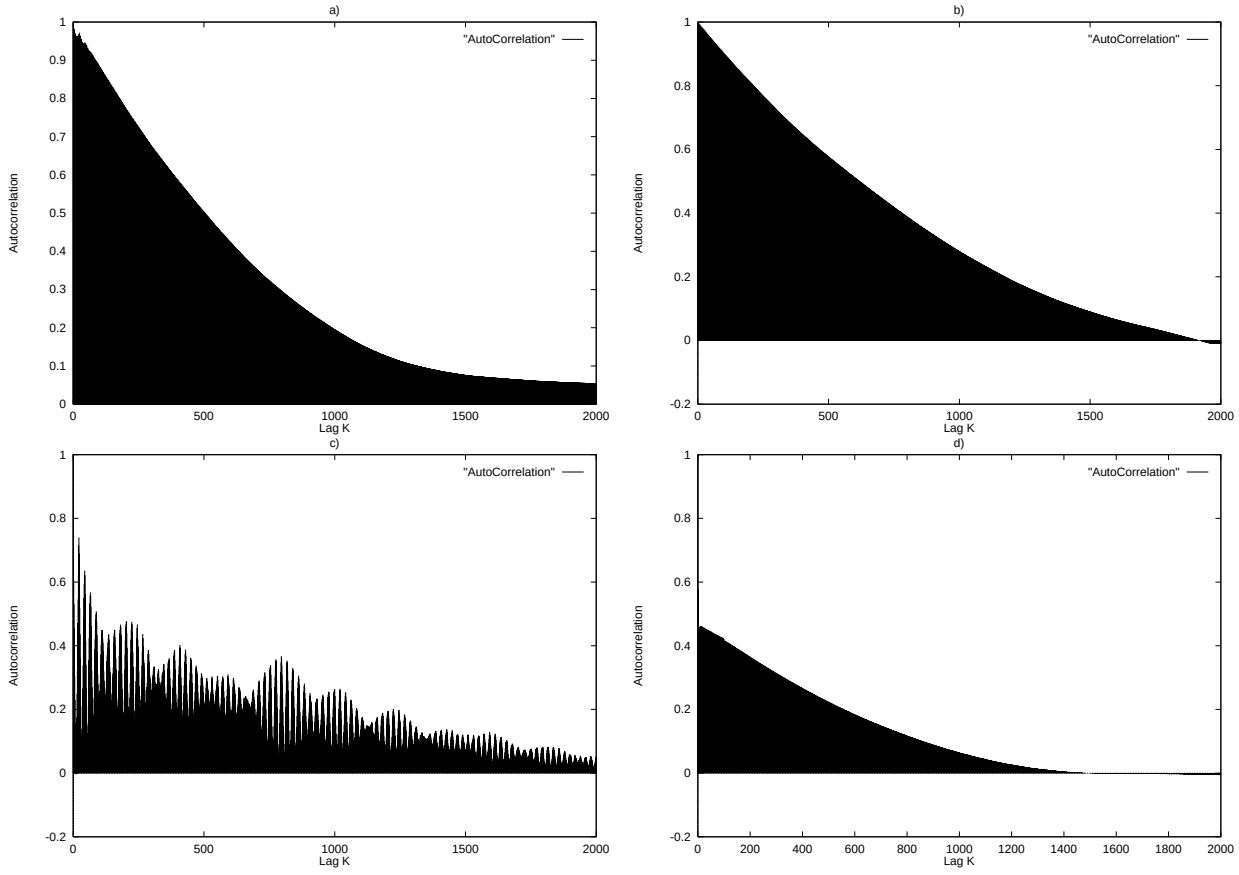

Figure 22: Estimated Autocorrelation Functions (ACF) for Traffic Sequences Formed by Cases of of Table 11. Plots: (a) Case 1, (b) Case 3, (c) Case 5, (d) Case 6.

Case #	Number of Flows	Min/Max File Size	Min/Max OFF Time	Packet Loss	TCP Max Window Size (KB)	Link Speeds (Mbps)	Bottleneck Link Speed (Mbps)
1	64	10B/10MB	0.1/480 sec	No	1000	5	45
2	64	10B/10MB	0.1/480 sec	No	1000	5	134
3	64	10B/10MB	0.1/480 sec	No	1000	5	134
4	29 128 64 64	Hours 0.1/8 KB 0.1/4 MB 5/15 MB	Traffic 0.01/1 sec 10/490 sec 100/1900 sec	No	64	10	134
5	14 56 8	Hours 0.1/8 KB 1/10 MB	Traffic 0.1/1 sec 10/400 sec	No	64	5	134
6	14 56 8	Hours 0.1/8 KB 1/10 MB	Traffic 0.1/1 sec 10/400 sec	No	–	5	134
7	14 48 8 8	Hours 0.1/8 KB 0.01/1 MB 1/10 MB	Traffic 0.1/1 sec 1/10 sec 10/400 sec	No	64	5	134

Table 11: Run-Time Simulation Parameters for the Cases of ON/OFF Traffic Sources with File Sizes and OFF Times Uniformly Distributed.

Case #	Transport Protocol Type	Data Collection Place	$X(n)$ Aggregation 10 ms Estimated $\hat{H}$		$Y(i)$ Time 100 ms Estimated $\hat{H}$		$Z(j)$ Interval 1 sec Estimated $\hat{H}$	
			R/S	Var	R/S	Var	R/S	Var
1	Reno	egress	0.91	0.81	0.80	0.74	0.70	0.63
2	Reno	egress	0.91	0.80	0.77	0.72	0.68	0.60
3	UDP	egress	0.95	0.82	0.81	0.76	0.72	0.64
4	Reno	egress	–	–	0.79	0.68	0.66	0.58
5	Reno	egress	0.81	0.75	0.79	0.70	0.68	0.60
6	UDP	egress	0.80	0.74	0.75	0.68	0.66	0.58
7	Reno	egress	0.73	0.70	0.71	0.64	0.62	0.56

Table 12: Estimated Values of H for the Cases of ON/OFF Traffic Sources with File Sizes and OFF Times Uniformly Distributed and Described in Table 11.

5.2.4 Case of File Sizes and OFF Times Exponentially Distributed: Alternating Mean for File Sizes

In this set of scenarios, for each TCP or UDP connection, the file size was drawn by either of two exponential random generator with different means. With probability p the file size being transmitted was given by generator A with mean A, and with probability $1-p$ it was given by generator B with mean B. For all six cases considered in this section, the OFF times were exponentially distributed with mean 600 ms. Table 13 shows the run-time simulation parameters, and the corresponding estimated values of LRD intensity (H) are shown in Table 14. Again, the results presented here indicate LRD in the traffic traces generated by the simulations of this section.

Case #	Number of Flows	Mean File Size A	Mean File Size B	Packet Loss	TCP Max Window Size (KB)	Link Speeds (Mbps)	Bottleneck Speed (Mbps)
1	64	4 KB	5 MB	No	64	10	134
2	64	4 KB	5 MB	No	64	10	134
3	64	4 KB	5 MB	No	64	10	134
4	64	4 KB	5 MB	No	64	10	134
5	64	4 KB	10 MB	No	64	10	134
6	64	4 KB	10 MB	No	64	10	134

Table 13: Run-Time Simulation Parameters for the Cases of ON/OFF Traffic Sources with File Sizes and OFF Times Exponentially Distributed: Alternating Mean for File Sizes. Mean OFF Time: 600 ms

Case #	Transport Protocol Type	Probability of File Size w/ Mean B	$X(n)$ Aggregation 10 ms Estimated R/S	Var	$Y(i)$ Time 100 ms $\hat{H}$ R/S	Var	$Z(j)$ Interval 1 sec Parameter R/S	Var
1	5.5 Reno	Hours 0.01	Traffic 0.83	0.79	0.85	0.78	0.75	0.70
2	5.5 Reno 102	Hours 0.001 Hours	Traffic 0.84 –	0.82 –	0.85 –	0.81 –	0.79 0.71	0.76 0.65
3	5.5 Reno 43	Hours 0.0001 Hours	Traffic 0.78 –	0.80 –	0.80 –	0.81 –	0.79 0.73	0.75 0.66
4	5.5 UDP 40	Hours 0.0001 Hours	Traffic 0.76 –	0.74 –	0.75 –	0.70 –	0.69 0.66	0.61 0.59
5	5.5 Reno 58	Hours 0.001 Hours	Traffic 0.87 –	0.86 –	0.91 –	0.86 –	0.84 0.75	0.81 0.71
6	5.5 Reno 229	Hours 0.0001 Hours	Traffic 0.84 –	0.85 –	0.89 –	0.86 –	0.84 0.73	0.82 0.63

Table 14: Estimated Values of H for the Cases of ON/OFF Traffic Sources with File Sizes and OFF Times Exponentially Distributed (Alternating Mean for File Sizes, Mean OFF Time: 600 ms) and Described in Table 13

6 Discussion

Understanding the nature of traffic of large internets or high-speed networks is essential for engineering, operations, and performance evaluation of these networks. Studies on a variety of networks have empirically show that network traffic is bursty over a broad range of time scales suggesting long-range dependence. In this study we attempted to give answers to the following questions by analyzing network traffic generated by simulations.

Does the activation of TCP's dynamics give rise to long-range dependence in TCP traffic? For the case of greedy sources, the results in Section 5.1 indicate that the activation of TCP's dynamics by packet losses due to overflows at a bottleneck node within the network might give rise to LRD in TCP traffic. A definitely answer to this question can only be given by a rigorous mathematical analysis. To get an accurate answer to this question by simulations requires a very large number ($>>$ 1000000) of traffic samples be collected.

Is the long-range dependence observed by empirical studies in real network traffic induced by the dynamics of TCP? In real network scenarios, there are no connections with greedy sources. Real network traffic sources are best described by the ON/OFF model where the transmission of a finite file (ON time) is followed by an idle time (OFF time) and so on. The results presented in Section 5.2 and obtained by running several simulations with TCP Reno, TCP Tahoe, TCP Tahoe with Slow Start Disabled, and UDP as the transport protocols suggest that a) the LRD observed in many traffic sequences was not caused by the dynamics of TCP, and b) the dynamics of TCP had no effect on the estimated intensity of LRD detected in simulated TCP traffic.

If the distribution of file sizes is not heavy-tailed, does the aggregation of many connections result in traffic that exhibit LRD? In several cases studied in Section 5.2, the file sizes and/or OFF times were not heavy-tailed distributed but both variance and R/S estimators showed strong evidence of LRD in the generated traffic sequences. Specifically, it was shown that when the file sizes are either exponentially or uniformly distributed with high means (or a combination of low and high means), the resulting traffic has properties indicating LRD. Importantly, we showed that the combination of connections whose file sizes are exponentially distributed with low mean (4 KB) with connections whose file sizes are also exponentially distributed but with high mean (5 MB or higher) can generate traffic with similar burstiness with traffic created by a combination of connections whose files sizes are heavy-tailed distributed, i.e., pareto distributed.

In most cases we have no prior knowledge of how the empirical network traffic traces were created. *Can we conclude that a network traffic trace has LRD by examining only the results from the variance and R/S estimators*? The definition of long-range dependence applies only to infinite time series; a process has LRD **iff** $\sum_{k=1}^{\infty} r(k) = \infty$. Since traffic traces are finite, the question now is how many samples are required to get a good estimate of the Hurst parameter. We believe based on our results that the number of samples require to get a very good estimate of the intensity of long-range dependence (H) by using the variance or the R/S estimators depends very much on the process being study. For example, if the file sizes being transferred are heavy-tailed, then 1-hour traffic trace is enough to detect the LRD by both estimators. However, if the distribution of files sizes and OFF times are unknown, then several hours of traffic must be collected in order to get an accurate estimate of the Hurst parameter. The results of this study show that it is possible for the self-similar parameter H estimators to show evidence of LRD at short time scales but show no evidence of LRD at large time scales. Since the results also show that for many traffic processes a very large number of samples is required for an accurate detection of LRD presence, to detect possible LRD in traffic traces, many hours of traffic traces must be collected to enable estimation of H at different time scales. We recommend that ten or more hours of traffic should be collected and then the long-range dependence should be estimated at different time scales. In most empirical and simulation studies [24, 10, 20, 34, 36, 38, 49, 50] performed to either detect the presence of long-range dependence (LRD) or give a possible explanation of what causes the LRD in TCP traffic, at most three hours of TCP traffic was used. Thus, some of the results reported by these studies could be wrong.

Although, results from this study show evidence of LRD in simulated traffic for cases of TCP connections with greedy sources and TCP/UDP connections with ON/OFF sources whose file size distribution and OFF times were not heavy-tailed, we can not arrive to a conlusion based merely on these results, that for these cases the traffic indeed had LRD. In future study, we will attempt to verify whether or not the traffic generated by such cases exhibit LRD by using rigorous mathematical analysis.

References

[1] Furquan A. Ansari, "Adapting TCP/IP to ATM," Master's Thesis, Electrical Engineering and Computer Science, University of Kansas, 1996.

[2] M. F. Arlitt, C. L. Williamson, " Web server workload characterization: The search for invariants, In Proc. ACM SIGMETRICS, 1996.

[3] A. Baiocchi, N. Blefari Melazzi, M. Listanti, A. Roveri, R. Winkler, "Modeling Isssues on a ATM Multiplexer Within a Bursty Traffic Environment," INFOCOM'91, Vol. 1, pp. 2c.2.2, 1991.

[4] J. Beran, R. Sherman, and W. Willinger, "Long Range Dependence in Variable Bit Rate Video Traffic," IEEE Transactions on Communications, 43(3):1566-1579, Feb. 1995.

[5] J. Beran, *Statistics for Long-Memory Processes. Monographs on Statistics and Applied Probability*, Chapman and Hall, New York, NY, 1994.

[6] Dimitri Bertsekas, Robert Gallager, *Data Networks*, 2^{nd} edition, Prentice Hall, Englewood Cliffs, New Jersey 07632, 1992.

[7] Denis Bosq and Hung T. Nguyen, *A COURSE IN STOCHASTIC PROCESSES, Stochastic Models and Statistical Inference,* Kluwer Academic Publishers, 1996.

[8] Douglas E. Comer, *Internetworking with TCP/IP, Volume I: Principles, Protocols, and Architecture,* 2^{nd} edition, Prentice Hall, 1991.

[9] D. R. Cox, "Long-Range Dependence: A Review," Statistics: An Appraisal, Proceedings 50th Anniversary Conference, Iowa State Statistical Laboratory, H. A. David and H. T. David (Editors), The Iowa State University Press, pp. 55-74, 1984.

[10] M. E. Crovella, A. Bestavros, "Explaining World Wide Web Traffic Self-Similarity," Computer Science Dep. Technical Report TR-95-015, Boston University, October 1995.

[11] M. E. Crovella, Lester Lipsky, "Long-Lasting Transient Conditions in Simulations with Heavy-Tailed Workloads," In Proceedings of the 1997 Winter Simulation Conference, S. Andradottir, K. J. healy, D. H. Withers, and B. L. Nelson, eds.

[12] N. G. Duffield, N. O'Connell, "Large deviations and overflow probabilities for the general single-server queue, with applications," in Math, Proc. Cambridge Philos. Soc., pp. 363-375, 1995.

[13] D. Duffy, A. McIntosh, M. Rosenstein, and D. Wilson, "Statistical Analysis of CCSN/SS7 Traffic Data from Working CCS Subnetworks," IEEE *JSAC*, 12(3):544-551, 1994.

[14] A. Erramilli, O. Narayan, W. Willinger, "Experimental Queueing Analysis with Long-Range Dependent Packet Traffic," IEEE/ACM Transactions on Networking, Vol. 4, No. 2, April 1996.

[15] H. J. Fowler, W. E. Leland, "Local Area Network Traffic Characteristics, with Implication for Network Congestion Management," IEEE JSAC, Vol. 9, No. 7, Sept 1991.

[16] Janey C. Hoe, "Startup Dynamics of TCP's Congestion Control and Avoidance Schemes," Master's Thesis, Massachusetts Institute of Technology, June 1995.

[17] G. Irlam, Unix file survey. Available at http://www.base.com/gordoni/ufs93.html, September 1994.

[18] Van Jacobson, "Congestion Avoidance and Control," Proceedings of the ACM SIGCOMM'88, August 1988.

[19] Raj Jain, K. K. Ramakrishnan, Dah-Ming Chiu, "Congestion Avoidance in Computer Networks With a Connectionless Network Layer,", Digital Equipment Corporation, Technical Report, DEC-TR-506, August 1987. Also in C. Partridge, Ed., *Innovations in Internetworking,* Artech House, Norwood, MA, 1998, pp. 140-156.

[20] S. M. Klivansky, A. Mukherjee, C. Song, "On Long-Range Dependence in NSFNET Traffic," Technical Report GIT-CC-94-61, Georgia Institute of Technology, 1994.

[21] T. V. Lakshman, Upamanyu Madhow, "The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss," IEEE/ACM Transactions on Networking, Vol. 5, No. 3, June 1997.

[22] Georgios Y. Lazarou, and Victor S. Frost, "Simulation Study: The Effect of TCP's Dynamics on TCP's Data Traffic Long-Range Dependence Property," ITTC, The University of Kansas, Technical Report: ITTC-FY99-TR-10980-27, Jan. 4, 1999.

[23] Georgios Y. Lazarou, Victor S. Frost, Joseph B. Evans, and Douglas Niehaus, "Simulation & Measurement of TCP/IP over ATM Wide Area Networks," Special Issue on ATM Switching Systems for future B-ISDN, IEICE Trans. Commun. Vol. E81-B, No. 2, February 1998.

[24] W. E. Leland, M. S. Taqqu, W. Willinger, D. V. Wilson, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," IEEE/ACM Transactions on Networking, Vol. 2, No. 1, February 1994.

[25] W. E. Leland, D. V. Wilson, "High Time-Resolution Measurement and Analysis of LAN Traffic: Implication for LAN Interconnection," INFOCOM'91, Vol. 3, pp. 11d.3.1, 1991.

[26] Yong-Qing Lu, "Characterization and Modeling of Long-Range Dependent Telecommunication Traffic," Master's Thesis, EECS, University of Kansas, 1994.

[27] Mihaela Teodora Matache, "Estimation of the Hurst Parameter for Self-Similar Processes," presented in the Seminar of Stochastic Adaptive Control, Department of Mathematics, The University of Kansas, March 7, 1997.

[28] K. Meier-Hellstern, P Wirth, Y-L. Yang, and D. Hoeflin, "Traffic Models for ISDN Data Users: Office Automation Application," In Proc. ITC'13, pp 167-172, Copenhagen, June 1991.

[29] Patrick R. Morin, "The Impact of Self-Similarity on Network Performance Analysis," Computer Science Technical Report 95.495, Carleton University, Dec. 4, 1995.

[30] Robert Morris, "TCP Behavior with Many Flows," IEEE International Conference on Network Protocols, Atlanta, Georgia, October 1997.

[31] I. Norros, "A storage model with self-similar input," Queueing Syst. Vol. 16. pp. 387-396, 1994.

[32] Raif O. Onvural, *ASYNCHRONOUS TRANSFER MODE NETWORKS Performance Issues*, Second Edition, Artech House, Boston-London, 1995.

[33] K. Park, G. Kim, M. Crovella, "On the Effect of Traffic Self-Similarity on Network Performance," Computer Science Dep. Technical Report CSD-TR 97-024, Purdue University, 1997.

[34] K. Park, G. Kim, M. Crovella, "On the relationship between file sizes, transport protocols, and self-similar network traffic," Computer Science Dep. Technical Report BU-CS-96-016, Boston University, August 1996.

[35] Craig Partridge and Timothy J. Shepard, "TCP/IP Performance over Satellite Links," IEEE Network, Vol. 11, No. 5, September/October 1997.

[36] Vern Paxson, Sally Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling," IEEE/ACM Transactions on Networking, Vol. 3, No. 3, June 1995.

[37] Peter Zimmer, Mihaela Teodora Matache, Personal Communication, May 1998.

[38] Rudolf H. Riedi, and Jacques Levy Vehel, "TCP traffic is multifractal: a numerical study," Project Fractales, INRIA Rocquencourt, France, (submitted IEEE Transactions of Networking, October 1997).

[39] James Roberts, Ugo Mocci, Jorma Virtamo, ***Broadband Network Teletraffic****: Performance Evaluation and Design of Broadband Multiservice Networks; Final Report of Action COST 242*, Springer, 1996.

[40] G. Samorodnitsky, M. S. Taqqu, *STABLE NON-GAUSSIAN RANDOM PROCESSES: Stochastic Models with Infinite Variance*, Chapman & Hall, New York, NY, 1994.

[41] K. Sam Shanmugan and A. M. Breipohl, *RANDOM SIGNALS Detection, Estimation and Data Analysis,* John Wiley & Sons, 1988.

[42] W.R. Stevens, *TCP/IP Illustrated, Volume 1,2*, Addison-Wesley, Readings, Massachusetts, 1994.

[43] Systems & Networks, *BONeS DESIGNER 3.0 Modeling Guide*, Lawrence, Kansas, 1995.

[44] M. S. Taqqu, J. B. Levy, "Using renewal processes to generate long-range dependence and high variability," in *Dependence in Probability and Statistics*, E. Eberlein and M. S. Taqqu. eds. Boston. MA: Birkhauser, 1996, vol. 11, pp. 73-89.

[45] Murad S. Taqqu, Vadim Teverovsky, and Walter Willinger, "Is network traffic self-similar or multifractal?," Preprint 1996. To appear in the journal "Fractals".

[46] Murad S. Taqqu, Vadim Teverovsky, and Walter Willinger, "Estimators for long-range dependence: an emperical study," Appeared in "Fractals", Vol. 3, No. 4, 1995.

[47] Murad S. Taqqu, Vadim Teverovsky, "Robustness of Whittle-type Estimators for Time Series with Estimators for Time Series with Long-Range Dependence," *Stochastic Models,* 1997.

[48] Sheldon M. Ross, *STOCHASTIC PROCESSES*, Second Edition, John Wiley & Sons, New York, 1996.

[49] W. Willinger, M.S. Taqqu, W. E. Leland, D. V. Wilson, "Self-Similarity in High-Speed Packet Traffic: Analysis and Modeling of Ethernet Traffic Measurements," Statistical Science, Vol. 10, No 1, pp. 67-85, 1995.

[50] W. Willinger, M.S. Taqqu, R. Sherman, D. V. Wilson, "Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level," IEEE/ACM Transactions on Networking, Vol. 5, No. 1, February 1997.