Performance of TCP on Wireless Fading Links with Memory*

A. Chockalingam[†], **Michele Zorzi**[‡], and **Ramesh R. Rao**[‡]

[†]Qualcomm, Inc., 6455 Lusk Boulevard, San Diego, CA 92121-2779, U.S.A.
fax: +1-619-658-1022 – e-mail: achockal@qualcomm.com

[‡]Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093-0407, U.S.A.
fax: +1-619-534-2486 – e-mail: {zorzi,rao}@ece.ucsd.edu

Abstract— **In this paper, the bulk throughput performance of TCP NewReno over wireless fading links having memory is studied. Like TCP Tahoe, the NewReno version of TCP implements a *fast retransmit* procedure, but it uses a different congestion window adaptation algorithm. In this study, we show that, for the default parameters of the BSD implementation of TCP over a 1.5 Mbps wireless link having a very small bandwidth-delay product, and as long as sufficiently large advertized window sizes are used, the burstiness in packet errors caused by slow multipath fading (experienced by slow moving users) significantly benefits NewReno compared to i.i.d. packet errors (experienced at vehicular user speeds). We further show that, in such slow fading conditions, NewReno performs no better than Tahoe, mainly due to the high degree of correlation in the fading process.**

I. Introduction

Due to rapid advances in the area of wireless communications and the popularity of the Internet, provision of packet data services for applications like e-mail, web browsing, mobile computing, etc., over wireless is gaining importance. Transport Control Protocol (TCP) is a reliable, end-to-end, transport protocol in the Internet Protocol (IP) suite, and is widely used in applications like telnet, ftp, and http [1]. TCP was designed for wireline networks where the channel error rates are very low and congestion is the primary cause of packet loss [2]. However, when used over wireless channels which are characterized by high error rates, the performance of TCP is severeley affected [3]. There have been several recent investigations on different facets of wireless TCP [4]-[11]. Most of these studies do not consider the effect of *correlation in multipath fading* [12] on the performance of TCP over wireless links. Errors are often assumed to occur independently and with the same probability on each packet (i.i.d. packet error model). Yet, because the multipath fading process in a mobile radio environment can be slowly varying for typical values of carrier frequency and user speed, the dependence between transmissions of consecutive packets of data cannot be neglected [12]. This paper addresses the issue of the effect of correlated multipath fading on TCP performance, an issue which has not been adequately addressed in the past. Related work has been presented in [9], where the performance of the OldTahoe and Tahoe versions of TCP is analyzed in the presence of Rayleigh fading, by assuming a two state Markov model to represent the channel status. Also, a simplified analytical model for the throughput of the OldTahoe version in the presence of Markovian packet losses has been presented in [10], where it is shown that correlation has a beneficial effect in general.

Several modifications to TCP, including Reno, NewReno, and Vegas have been proposed and their performance analyzed in wireline networks [2],[13]-[15]. Reno's loss recovery algorithm is optimized for the case when a single packet is dropped from a window of data. Thus, Reno may significantly improve upon the behavior of Tahoe when a single packet is dropped, but can suffer performance problems when multiple packets are dropped from a window of data [13]. As multiple packet losses are common in wireless channels, Reno is expected to perform poorly in such environments [2]. NewReno addresses this problem by improving the loss recovery phase to handle the occurrence of multiple errors within the same congestion window [8],[13]. The performance of these enhanced TCP versions on *correlated* wireless fading links has not been studied so far. In this paper, we investigate the performance of TCP NewReno on a flat Rayleigh fading channel modeled as a first-order Markov chain [16]. We consider a scenario where a large data file is to be transferred from the base station interworking function (IWF) to a mobile terminal, over a 1.5 Mbps wireless link that is characterized by very low delay-bandwidth product (as in [8], we will essentially assume instantaneous ACKs).

The paper is organized as follows. In Section II, we describe the Tahoe, Reno and NewReno versions of TCP. The system model and the correlated fading channel model considered in this paper are presented in Section III. Section IV provides the simulation results comparing the performance of the TCP Tahoe and TCP NewReno on correlated fading channels. Finally, conclusions are provided in Section V.

II. TCP Tahoe, Reno and NewReno

In this section, we provide a description of the receive and transmit processes in TCP Tahoe, Reno and NewReno. While the receive processes are the same for all of them, their transmit processes are different in the way the *loss recovery phase* is implemented. The following description of the receive and transmit processes follow that of [8].

The TCP receiver can accept packets out of sequence, but will deliver them only in sequence to the TCP user. During connection set up, the receiver advertizes a maximum window size, W_{max}, so that the transmitter does not allow more than W_{max} unacknowledged data packets outstanding at any given time. The receiver

*Work partially supported by the Center for Wireless Communications at UC San Diego.

sends back an acknowledgement (ACK) for every data packet it receives correctly. The ACKs are cumulative. That is, an ACK carrying the sequence number m acknowledges all data packets up to, and including, the data packet with sequence number $m-1$. The ACKs will carry the next expected packet sequence number, which is the first among the packets required to complete the in-sequence delivery of packets. Thus, if a packet is lost (after a stream of correctly received packets), then the transmitter keeps receiving ACKs with the sequence number of the first packet lost (called duplicate ACKs), even if packets transmitted after the lost packet succeed in being correctly received at the receiver.

The TCP transmitter operates on a window based transmission strategy as follows. At any given time t, there is a lower window edge $X(t)$, which means that all data packets numbered up to, and including, $X(t)-1$ have been transmitted and acknowledged, and that the transmitter can send data packets from $X(t)$ onwards. The transmitter's congestion window, $W(t)$, defines the maximum amount of data packets the transmitter is permitted to send, starting from $X(t)$. Under normal data transfer, $X(t)$ has non-decreasing sample paths. However, the adaptive window mechanism causes $W(t)$ to increase or decrease, but never exceed W_{max}. Transitions in the processes $X(t)$ and $W(t)$ are triggered by the receipt of ACKs. The receipt of an ACK that acknowledges some data will cause an increase in $X(t)$ by an amount equal to the amount of data acknowledged. The change in $W(t)$, however, depends on the particular version of TCP and the congestion control process. Each time a new packet is transmitted, the transmitter starts a timer and resets the already running transmission timer, if any. The timer is set for a round trip timeout value that is derived from a round trip time estimation procedure [1]. Whenever a timeout occurs, retransmission is initiated from the next packet after the last acknowledged packet. It is noted that the timeout values are set only in multiples of a timer granularity [1].

The basic window adaptation procedure, common to all TCP versions [17], works as follows. Let $W(t)$ be the transmitter's *congestion window width* at time t, and $W_{th}(t)$ be the *slow-start threshold* at time t. The evolution of $W(t)$ and $W_{th}(t)$ are triggered by ACKs (new ACKs, and not duplicate ACKs) and timeouts as follows:

1. If $W(t) < W_{th}(t)$, each ACK causes $W(t)$ to be incremented by 1. This is the *slow start* phase.
2. If $W(t) \geq W_{th}(t)$, each ACK causes $W(t)$ to be incremented by $\frac{1}{W(t)}$. This is the *congestion avoidance* phase.
3. If timeout occurs at the transmitter at time t, $W(t^+)$ is set to 1, $W_{th}(t^+)$ is set to $\lceil \frac{W(t)}{2} \rceil$, and the transmitter begins retransmission from the packet after the last acknowledged packet.

Note that the transmissions after a timeout always start with the first lost packet. The window of packets transmitted from the lost packet onwards, but before retransmission starts, is called as the *loss window*. The congestion control and data loss recovery procedures implemented in TCP Tahoe, Reno and NewReno are as follows.

In the case of Tahoe, *fast retransmit* procedure is implemented subsequent to a packet loss. That is, if subsequent to a packet loss, the transmitter receives the K^{th} duplicate ACK at time t, before the timer expires, then the transmitter behaves as if a timeout has occured and begins retransmission, with $W(t^+)$ and $W_{th}(t^+)$ as given in the basic window adaptation algorithm described above.

In the case of Reno too, the fast retransmit procedure following a packet loss is implemented. However, the subsequent recovery procedure is different. If the K^{th} duplicate ACK is received at time t, then $W_{th}(t^+)$ is set to $\lceil \frac{W(t)}{2} \rceil$, and $W(t^+)$ is set to $W_{th}(t^+)+K$ (the addition of K accounts for the K packets that have successfully left the network). The Reno transmitter then retransmits only the first lost packet. As the transmitter waits for the ACK for the first lost packet retransmission, it may get duplicate ACKs for the outstanding packets. The receipt of each of such duplicate ACK causes $W(t)$ to be incremented by 1. If there was only a single packet loss in the loss window, then the ACK for its retransmission will complete the loss recovery; W at this time would be set to W_{th}, and the transmission resumes according to the basic window control algorithm. If there are multiple packet losses in the loss window, then the ACK for the first lost packet retransmission will advance the left edge of the window, X, by an amount equal to 1 plus the number of good packets between the first lost packet and the next one. In this case, if the loss recovery is not successful due to lack of the duplicate ACKs necessary to trigger multiple fast retransmits, then a timeout has to be waited for. The above data loss recovery strategy in Reno is shown to perform better than Tahoe when single packet losses occur in the loss window, but can suffer performance problems when multiple packets are lost in the loss window [13].

In the case of NewReno, fast retransmit and congestion window adaptation are as in Reno, but the loss recovery mechanism is as in Tahoe [8]. In this paper, we will focus mainly on the NewReno version of TCP, which is more suitable to deal with multiple errors that arise in the wireless environment.

Figure 1 shows a sample trace of TCP obtained from simulations, assuming instantaneous and error-free ACKs[1]. On the x-axis is the time in units of packet time, and on the y-axis is the transmitted packet id (i.e., sequence number of transmitted packets). A slope of unity in this trace will imply 100% throughput under ideal conditions. However, less than unity slope results due to packet errors, congestion, and timeouts. In the sample trace in Figure 1, a maximum window size W_{max} of 6 packets, a minimum timeout of 5 packets, a fast retransmit threshold K of 2, and instantaneous and error-free ACKs are assumed. Note that in this trace, the packet with sequence number 4 is retransmitted during $i=7$ subsequent to its loss during $i=4$ and the successful transmission of packets with sequence number 5 and 6 for which two duplicate ACKs are received during $i=5,6$. A similar retransmission occurs for the packet with sequence number 13 during $i=16$. The channel idles during $i=21$ to 24, 29 to 32, and 40 to 42 where the recovery stalls and a timeout is waited for. Also, two consecutive packet losses occur at $i=60,61$ followed by two duplicate ACKs for packets 62 and 63, which trigger a fast retransmit at $i=64$.

Figure 2 shows the evolution of the transmitter's congestion window width, $W[i]$, and the slow-start threshold, $W_{th}[i]$, corresponding to the trace in Figure 1 for TCP Tahoe. Figure 3 shows the evolution of $W[i]$ and $W_{th}[i]$ for TCP NewReno. It is seen

[1] In this specific example, the transmission process happens to be identical for Tahoe and NewReno, even though this is not true in general. The evolution of the congestion window, however, is different for the two protocols, as shown in Figs. 2 and 3.

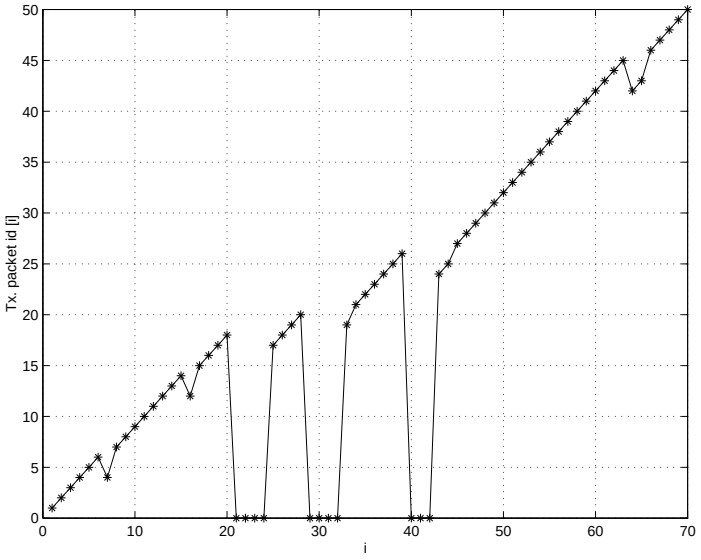

Fig. 1. A sample trace of Tahoe and NewReno TCP. Fast Retransmit Threshold, $K = 2$. Maximum Window Size, $W_{max} = 6$ packets. Minimum Timeout, $MTO = 5$ packets. $Note$: Tx packet id [i] = 0 means $idle$.

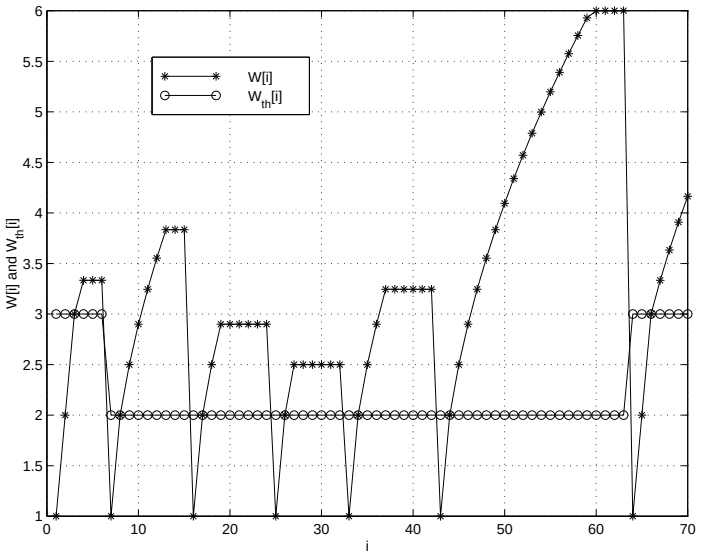

Fig. 2. Evolution of transmitter's congestion window width, $W[i]$, and slow-start threshold, $W_{th}[i]$, for TCP Tahoe. $K = 2$. $W_{max} = 6$. $MTO = 5$.

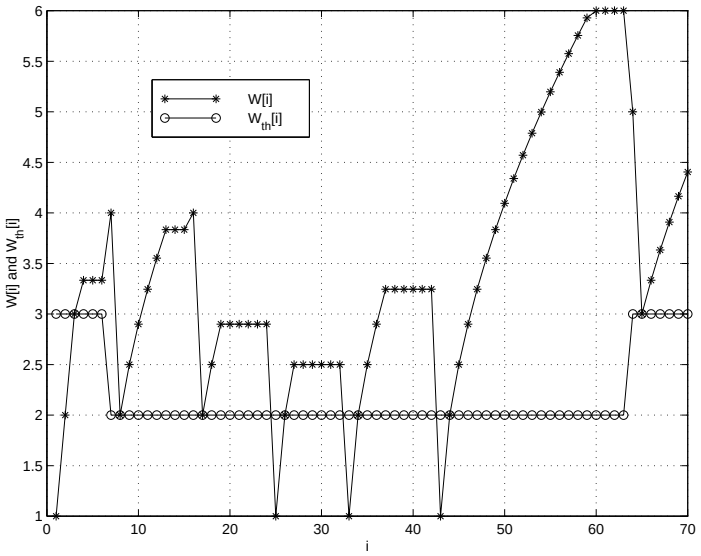

Fig. 3. Evolution of $W[i]$ and $W_{th}[i]$ for TCP NewReno. $K = 2$. $W_{max} = 6$. $MTO = 5$.

from Figure 2 that each time there is a retransmission of the first lost packet, either after K duplicate ACKs ($i = 7, 16, 64$) or after a timeout ($i = 25, 33, 43$), Tahoe resets the congestion window width $W[i]$ to 1 (as per the basic window adaptation procedure – in this case, *slow-start*). However, in NewReno (Figure 3), note that because of its different congestion window adaptation procedure, its evolution pattern of $W[i]$ and $W_{th}[i]$ is different from that of Tahoe's. For example, following the single loss during $i = 4$ and receipt of two duplicate ACKs during $i = 5, 6$, the window width and the slow-start threshold during $i = 7$ are updated as $W_{th}^{+} = \lceil \frac{W}{2} \rceil = \lceil \frac{3.3}{2} \rceil = 2$, and $W^{+} = W_{th}^{+} + K = 2 + 2 = 4$. The retransmission during $i = 7$ is successful thus completing the loss recovery. Subsequently, W during $i = 8$ is set to W_{th} (i.e., $W = W_{th} = 2$). Similar loss recovery scenarios are observed during $i = 16$, and $i = 64$[2].

III. System Model

Data exchange using TCP involves a connection set up phase, a data transfer phase, and a call tear down phase. In this paper, we are primarily interested in the bulk throughput performance of TCP, for example, when a large data file is transferred from the base station to a mobile terminal. Consequently, we model only the data transfer phase of the protocol which dominates in the resulting performance. Unlike in [8], where the base station side TCP/IP stack is placed on a fixed LAN (10 Mbps Ethernet) host from which packets are routed through an Intermediate System (an IP router) at the base station to the mobile terminal over a 1.5 Mbps wireless link, our model assumes no IP router at the base station and the TCP/IP stack is directly placed at the base station interworking function (IWF). Thus, in our model, there are no queueing and delays due to the IS. A wireless data link operating at 1.5 Mbps transmission rate is considered. The bandwidth-delay product of the link is assumed to be negligibly small. In other words, the acknowledgements (ACKs) from the mobile receiver arrive instantaneously at the base station IWF. Also the ACK packets are assumed to arrive error-free. These assumptions are reasonable because in wireless local environments the propagation delays are small, and the ACK packets are relatively smaller in size than data packets (40 bytes vs 500-1500 bytes). We assume a TCP packet size of 1400 bytes. At 1.5 Mbps rate, this corresponds to a packet transmission time of about 7.5 ms. A minimum timeout (MTO) value of 100 packets (i.e., 750 ms) is used. This value corresponds to an average of 2 ticks of timeout granularity in a BSD implementation of TCP where the timeout granularity is 500 ms [8]. We do not consider the issue of mobility in the present study. See [7] for an integrated performance study of TCP/IP including the impact of mobility. Finally, we model the correlation in the multipath fading process using a first-order

[2]Note that, because of two errors at $i = 60, 61$, the original Reno would not allow the retransmission of the second lost packet, stalling the recovery and waiting for a timeout.

f_dT	P_E	F (dB)	p	q	$(1-q)^{-1}$
0.01	0.001	29.9978	0.999329	0.329452	1.49132
	0.01	19.9782	0.997518	0.754308	4.07013
	0.1	9.77322	0.991872	0.926851	13.6708
0.08	0.001	29.9978	0.999007	0.008239	1.00831
	0.01	19.9782	0.990680	0.077286	1.08376
	0.1	9.77322	0.940354	0.463188	1.86285
0.64	0.001	29.9978	0.999000	0.001185	1.00238
	0.01	19.9782	0.990019	0.011834	1.01198
	0.1	9.77322	0.90182	0.116376	1.13170

Table 1. Markov parameters p and q at different values of P_E and f_dT.

Markov model as follows.

A. Correlated Fading Channel Model

The multipath fading on a mobile radio channel is considered to follow a Rayleigh distribution [12]. The issue of modeling block error process on a correlated Rayleigh fading channel has been addressed in [16], where it was shown that, for a broad range of parameters, the sequence of data block success and failure can be approximated by means of a simple two-state Markov chain whose transition probability matrix is given by

$$M_c = \begin{pmatrix} p & 1-p \\ 1-q & q \end{pmatrix}, \tag{1}$$

where p and $1-q$ are the probabilities that the j^{th} block transmission is successful, given that the $(j-1)^{st}$ block transmission was successful or unsuccessful, respectively. Given the matrix M_c, the channel properties are completely characterized. In particular, it is possible to find the steady state distribution of the chain. The steady state probability that a packet error occurs, P_E, is

$$P_E = \frac{1-p}{2-p-q}. \tag{2}$$

Note that $(1-q)^{-1}$ represents the average length of a burst of packet errors, which is described by a geometric random variable. Also, for a Rayleigh fading channel with a fading margin F, the average packet error rate can be found as [16]

$$P_E = 1 - e^{-1/F}, \tag{3}$$

and the Markov parameter q can be derived as [16]

$$q = 1 - \frac{Q(\theta, \rho\theta) - Q(\rho\theta, \theta)}{e^{1/F} - 1}, \text{ where } \theta = \sqrt{\frac{2/F}{1-\rho^2}}. \tag{4}$$

In the above, $\rho = J_0(2\pi f_d T)$ is the Gaussian correlation coefficient of two successive samples of the complex amplitude of a fading channel with Doppler frequency f_d, taken T seconds apart. The parameter f_dT is the normalized Doppler bandwidth which describes the correlation in the fading process. $J_0(.)$ is the Bessel function of the first kind and zero order. $Q(\cdot,\cdot)$ is the Marcum Q function. From P_E and q in (3) and (4), the Markov parameter p can be obtained using (2). In this paper, we apply this Markov model at the TCP packet level (i.e., $T = 7.5$ ms). By choosing different P_E and f_dT values, we can establish fading channel models with different degree of correlation in the fading process. When f_dT is small, the fading process is very correlated (long bursts of packet errors); on the other hand, for larger values of f_dT, successive samples of the channel are almost independent (short bursts of packet errors). Table 1 shows the Markov parameters p and q, and the average burst length for different values of P_E and f_dT.

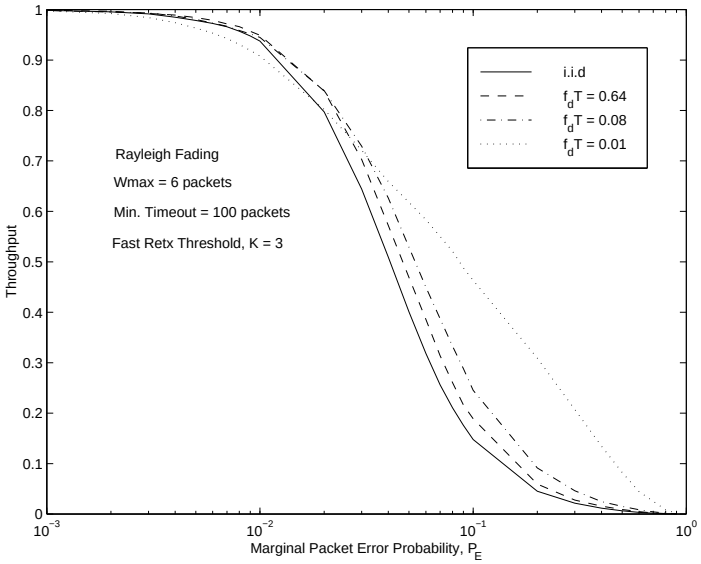

Fig. 4. Throughput performance of TCP NewReno in correlated Rayleigh fading at different values of normalized Doppler bandwidth, f_dT. $W_{max} = 6$. $K = 3$. $MTO = 100$.

IV. Simulation Results

The throughput performance of TCP NewReno was evaluated through simulations both in i.i.d. and correlated packet errors on Rayleigh fading channels. The values of the normalized Doppler bandwidth, f_dT, considered are 0.01, 0.08, and 0.64. At a carrier frequency of 900 MHz and the considered packet duration of about 7.5 ms, an f_dT value of 0.01 corresponds to a user moving at a speed of about 1.5 km/h (pedestrian user), and an f_dT value of 0.64 corresponds to a user speed of about 100 km/h (vehicular user). When $f_dT = 0.01$, the fading process is very much correlated (e.g., average packet error burst length of 13 packets for $f_dT = 0.01$ and $P_E = 0.1$ in Table 1). However, when $f_dT = 0.64$, the fading is fast such that the packet errors are more like i.i.d. (e.g., average packet error burst length of 1 packet for $f_dT = 0.64$ in Table 1). In Figure 4, the throughput performance of TCP NewReno is plotted as a function of the marginal packet error probabilty, P_E, for different values of f_dT. The performance in i.i.d. packet errors is also plotted for comparison. A maximum advertized window size, W_{max}, of 6 packets is used in Figure 4. The minimum timeout (MTO) is set at 100 packets and the fast retransmit threshold, K, is set at 3.

From Figure 4, for the chosen advertized window size of 6 packets, i.i.d. packet errors (i.e., mostly single packet errors) are seen to result in better performance than correlated packet errors (e.g., $f_dT = 0.01$) at small values of P_E. This is because at low values of P_E, i.i.d. errors are dealt with effectively by NewReno's congestion window adaptation algorithm (same as that of Reno's),

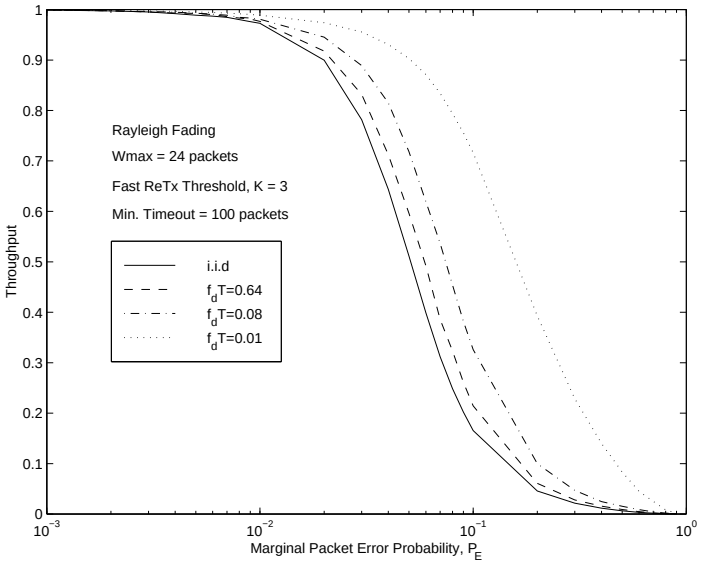

Fig. 5. Throughput performance of TCP NewReno in correlated Rayleigh fading at different values of f_dT. $W_{max} = 24$. $K = 3$. $MTO = 100$.

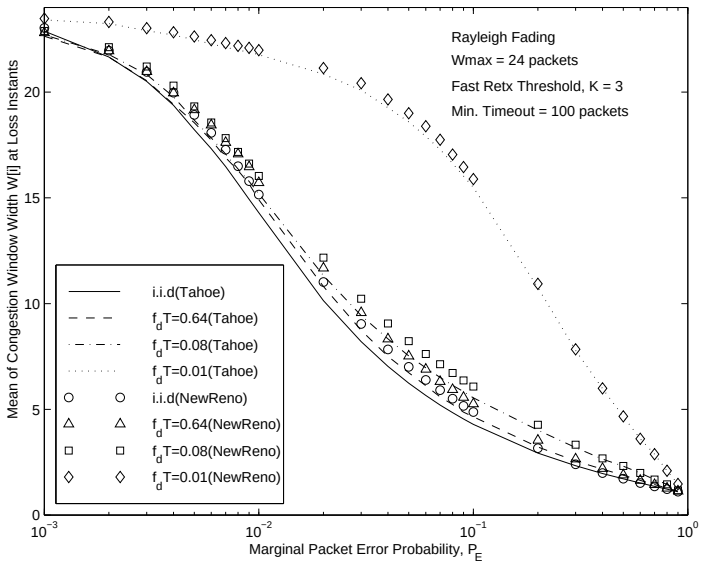

Fig. 6. Mean congestion window width $W[i]$ at loss instants in TCP Tahoe and NewReno for different values of f_dT. $W_{max} = 24$. $K = 3$. $MTO = 100$.

which is optimized for single packet errors.

In the presence of clustered errors, however, two effects tend to influence the performance in opposite directions. For a given value of the average packet error rate, P_E, correlated errors correspond to fewer error *events* (each comprising multiple packet errors), and therefore the congestion window is shrunk less frequently than for i.i.d. errors. A second effect takes place depending on the relationship between channel error burstiness and size of the advertized window. In order to trigger a fast retransmit, K duplicate ACKs must be received, i.e., K (= 3 in this case) packets must be successfully received after a packet loss. If packet n is corrupted by the channel at time t, only $W(t) - 1$ more packets can be transmitted, with $W(t) - 1 \leq W_{max} - 1$ (= 5 in this case). Therefore, if $W(t) - K$ or more packets are also corrupted, then the congestion window will be exhausted before K duplicate ACKs can be generated. The transmitter will then stop and wait for a timer expiration, i.e., the fast retransmit feature is not triggered. This undesirable event is fairly likely if the channel error burstiness is comparable with the congestion window size.

As can be seen in Figure 4, for high values of P_E, the beneficial effect overweighs the negative effect, resulting in better performance for correlated errors than for i.i.d errors. On the other hand, for small values of P_E, the effect of fast retransmit failures may dominate the performance, resulting in degraded performance due to error correlation. According to the above discussion, this effect can be expected to be significant if the channel error burstiness is comparable with the congestion window size (which can grow as large as W_{max}), and to become negligible otherwise. For example, in Figure 4, a W_{max} value of 6 provides improvement in throughput when f_dT is 0.08 or 0.64, where the average packet error burst length is less than W_{max} (see Table 1), whereas the correlation benefit is not fully exploited for $f_dT = 0.01$, where packet error bursts span 13 packets on average for $P_E = 0.1$. In this case, providing a W_{max} larger than 13 is beneficial, as clearly confirmed by the performance plots for $W_{max} = 24$ in Figure 5, where the $f_dT = 0.01$ case is found to perform significantly bet-

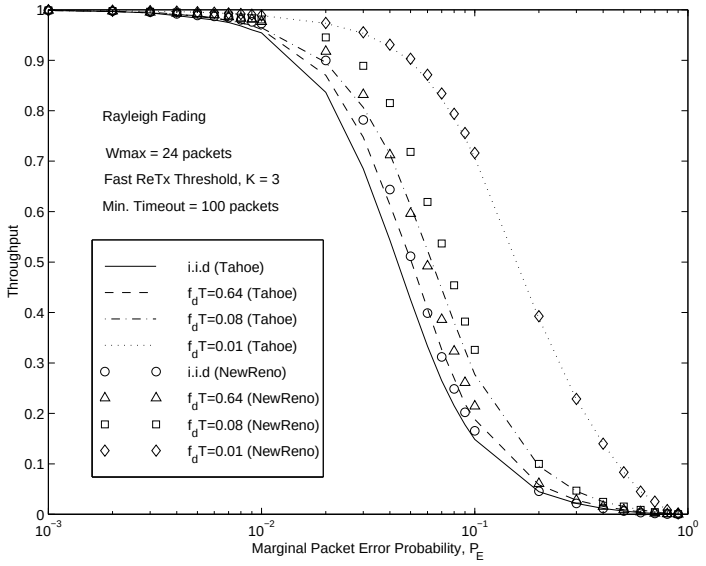

Fig. 7. Throughput comparison of TCP Tahoe and NewReno at different values of f_dT. $W_{max} = 24$. $K = 3$. $MTO = 100$.

ter than the i.i.d. case at all values of P_E. It is to be noted that larger W_{max} means larger buffer sizes at the receiver (for the considered packet size of 1400 bytes, $W_{max} = 6$ and 24 correspond to receiver buffer sizes of about 8kB and 32kB, respectively). It is further observed that increasing W_{max} beyond 24 does not offer any additional improvement (the results we obtained for the case of $W_{max} = 48$ were indistinguishable from those of Figure 5 for $W_{max} = 24$). In summary, by keeping the receiver buffer size in excess of the average packet error burst length ($\frac{1}{1-q}$ in Table 1), TCP NewReno can offer improved throughputs in slow fading compared to i.i.d. fading.

In Figures 6 and 7, we provide a performance comparison of TCP NewReno with TCP Tahoe under different degrees of correlation in the fading process for $W_{max} = 24$, $MTO = 100$, and

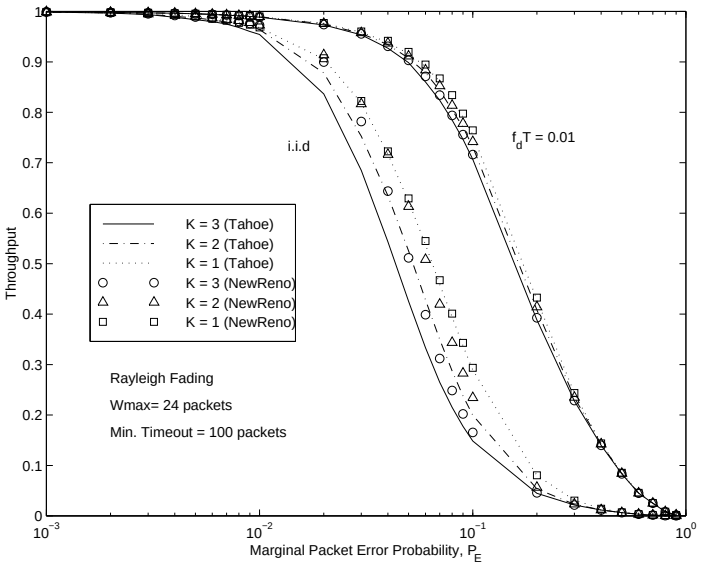

Fig. 8. Throughput comparison of TCP Tahoe and NewReno at different values of fast retransmit threshold threshold, K (1,2,3) at $f_dT = 0.01$ and i.i.d. $W_{max} = 24$. $MTO = 100$.

$K = 3$. Figure 6 shows the mean value of the congestion window width $W(t)$ at the *loss instants*. Note that the minimum and maximum values $W(t)$ at loss instants can take are 1 and W_{max} (= 24), respectively. Two important observations can be made from these figures. First, both in Tahoe and NewReno, there is a substantial increase in the mean congestion window width at loss instants when $f_dT = 0.01$ compared to that in the i.i.d. case. This essentially translates into increased throughput in slow fading compared to i.i.d. fading as can be seen from Figure 7. Second, conforming to the results reported in [8], with i.i.d. packet errors, TCP NewReno is found to perform better than TCP Tahoe at low and medium values of P_E, because of its recovery optimization to single packet errors. At high values of P_E, Tahoe and NewReno exhibit similar performance. Like in i.i.d. errors, NewReno is found to perform better than Tahoe in correlated errors as well, but only at large values of f_dT (e.g., f_dT = 0.08, 0.64). However, as the value of f_dT is decreased, the difference between NewReno and Tahoe diminishes. For example, when $f_dT = 0.01$ both NewReno and Tahoe exhibit almost identical performance.

Finally, the effect of varying the fast retransmit threshold, K, on the NewReno and Tahoe versions of TCP for i.i.d. and $f_dT = 0.01$ is shown in Figure 8. Throughput curves are plotted for $K = 1, 2, 3$. It is observed that, in i.i.d. errors, the performance with $K = 1$ is better than with $K = 3$. However, when $f_dT = 0.01$, there is practically no benefit from going from $K = 3$ to $K = 1$, both for Tahoe and NewReno. Hence it is envisioned that newer versions of TCP like, for example, the TCP Vegas, which proposes to retransmit at the instance of the receipt of the first duplicate ACK itself (i.e., $K = 1$), may not benefit from such modifications on highly correlated wireless fading channels. A promising feature, currently under investigation, is the use of selective ACKs (SACK TCP [13]).

V. Conclusions

We studied the bulk throughput performance of TCP NewReno over wireless fading links with memory. We considered a scenario where a large data file is to be transferred from the base station to a mobile terminal over a wireless link having negligible bandwidth-delay product. We showed that, for the default parameters of the BSD implementation of TCP over a 1.5 Mbps wireless link, the burstiness in the packet errors on the wireless link caused by slow multipath fading significantly benefited NewReno compared to i.i.d. packet errors, as long as sufficiently large advertized window sizes are used. We further showed that in such slow fading scenarios NewReno performed no better than Tahoe, mainly due to the high degree of correlation in the fading process. Based on the results, for example, at different values of the fast retransmit threshold, it may be argued that other enhanced versions of TCP (e.g., Vegas) may not offer any significant improvement on highly correlated wireless fading channels. This fact, together with the observation that TCP performance significantly depends on the channel error correlation, suggests that a clever design of the lower layers where memory in the packet error process (naturally present on wireless links because of the fading behavior) is preserved may turn out to be much more effective than the development of more complex TCP versions.

References

[1] W. R. Stevens, *TCP/IP Illustrated, Volume 1*, Addison Wesley, 1994.

[2] T. V. Lakshman, U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Trans. on Networking*, pp. 336–350, June 1997.

[3] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *ACM/IEEE Trans. on Networking*, Dec. 1997.

[4] A. DeSimone, M. C. Chuah, and O. C. Yue, "Throughput performance of transport layer protocols over wireless LANs," *Proc. IEEE Globecom'93*, pp. 542-549, December 1993.

[5] R. Caceres, and L. Iftode, "Improving the performance of reliable transport protocols in mobile computing environments," *IEEE Jl. Sel. Areas Commun.*, vol. 13, no. 5, pp. 850-857, June 1995.

[6] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz, "Improving TCP/IP performance over wireless networks," 1^{st} *Intl. Conf. on Mobile Computing and Networking*, November 1995.

[7] P. Manzoni, D. Ghosal, and G. Serazzi, "Impact of moblity on TCP/IP : An integrated performance study," *IEEE Jl. Sel. Areas Commun.*, vol. 13, no.5, pp. 858-867, June 1995.

[8] A. Kumar, "Comparative performance analysis of versions of TCP in a local network with a lossy link," *Tech. Rep. WINLAB-TR-129*, October 1996.

[9] A. Kumar, and J. Holtzman, "Comparative performance analysis of versions of TCP in a local network with a lossy link, Part II: Rayleigh fading mobile radio link," *Tech. Rep. WINLAB-TR-133*, November 1996.

[10] M. Zorzi, and R. R. Rao, "Effect of correlated errors on TCP," in *Proc. 1997 CISS*, pp. 666–671, March 1997.

[11] P. Bhagwat, P. Bhattacharya, A. Krishna, and K. Tripathi, "Using channel state dependent packet scheduling to improve TCP throughput over wireless LANs," *Wireless Networks*, pp. 91-102, 1997.

[12] W. C. Jakes, Jr., *Microwave mobile communications*, New York: John Wiley & Sons, 1974.

[13] K. Fall, and S. Floyd, "Comparisons of Tahoe, Reno, and Sack TCP," manuscript, ftp://ftp.ee.lbl.gov, March 1996.

[14] L. S. Brakmo, and L. L. Peterson, "TCP Vegas: End to end congestion avoidance on a global internet," *IEEE Jl. Sel. Areas Commun.*, vol. 13, no. 8, pp. 1465-1480, October 1995.

[15] O. Ait-Hellal, and E. Altman, "Analysis of TCP Vegas and TCP Reno," *Proc. IEEE ICC'97*, 1997.

[16] M. Zorzi, R. R. Rao, and L. B. Milstein, "On the accuracy of a first-order Markov Model for data transmission on fading channels," *Proc. IEEE ICUPC'95*, pp. 211-215, November 1995.

[17] Van Jacobson, "Congestion avoidance and control," *Proc. ACM Sigcomm'88*, pp. 314-329, August 1988.