Mobile Networks and Applications 6, 265–278, 2001

Energy Efficiency of TCP in a Local Wireless Environment

MICHELE ZORZI
Dipartimento di Ingegneria, Università di Ferrara, 44100 Ferrara, Italy

RAMESH R. RAO
Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093-0407, USA

Abstract. The focus of this paper is to analyze the energy consumption performance of various versions of TCP, namely, Tahoe, Reno and NewReno, for bulk data transfer in an environment where channel errors are correlated. We investigate the performance of a single wireless TCP connection by modeling the correlated packet loss/error process (e.g., as induced by a multipath fading channel) as a first-order Markov chain. Based on a unified analytical approach, we compute the throughput and energy performance of various versions of TCP. The main findings of this study are that (1) error correlations significantly affect the energy performance of TCP (consistent with analogous conclusions for throughput), and in particular they result in considerably better performance for Tahoe and NewReno than iid errors, and (2) the congestion control mechanism implemented by TCP does a good job at saving energy as well, by backing off and idling during error bursts. An interesting conclusion is that, unlike throughput, the energy efficiency metric may be very sensitive to the TCP version used and to the choice of the protocol parameters, so that large gains appear possible.

Keywords: energy consumption, wireless TCP, fading, energy efficiency

1. Introduction

Packet switched data services in a wireless environment are rapidly gaining popularity due to the success of mobile communications and the Internet. Provision of popular Internet services, such as e-mail, web browsing and multimedia, will soon become a necessity in the mobile communications market. At the heart of many Internet applications is the Transport Control Protocol (TCP), which is a reliable, end-to-end, transport protocol [17]. Gaining understanding of how this protocol behaves over radio channels is a very important step in extending Internet services to the wireless environment.

TCP was designed for wireline networks where the channel error rates are very low and congestion is the primary cause of packet loss [14]. Since its original deployment, several modifications to TCP, including Reno, NewReno, and Vegas have been proposed and their performance analyzed in wireline networks [1,8,14]. Reno's loss recovery algorithm is optimized for the case when a single packet is lost in a window of data. Hence, Reno can suffer performance problems when multiple packets are lost in a window [8]. NewReno addresses this problem by improving the loss recovery phase to handle this situation [8,12].

The wireless environment often involves portable devices which rely on batteries as their energy supplies. In this scenario, a key objective is the efficient usage of limited power sources. Traditionally, most of the effort in this respect has been focused on battery technology (a slowly improving field) and on low-power circuitry. As better and better performance are being achieved by the RF components, this brings other functions to play a significant role in the power budget [10]. On the other hand, the emergence of various applications and the need to support them in a wireless setting may open up new possibilities for energy saving strategies. For example, delay tolerant traffic may provide some degree of flexibility in that one does not have to fight bad channel conditions by just sending more power, but rather may defer transmission to better times by using channel-dependent scheduling [4]. It therefore makes sense to study how protocol rules and parameters affect the energy consumption of a device. So far, no energy study has been performed for TCP. Such a study is the focus of this paper.

The algorithm used in TCP for congestion control was designed with a wireline environment in mind, where the most likely sources of data losses are congestion and buffer overflow. Therefore, when congestion arises the backoff algorithm shrinks the size of the flow control window, thereby trying to decrease the amount of data entering the network. It has been pointed out by many authors [2,3,5,7,12–15] that the mismatch between the protocol action and the *true* cause of data losses in the wireless environment (i.e., channel errors) leads to performance problems. By looking at it from a different perspective, and considering the memory usually present in the error process induced by a fading channel, we argue that in some cases the window adaptation algorithm used in TCP may in fact be more efficient than predicted by those early studies, most of which neglected the actual characteristics of wireless propagation [11].

For example, it has been shown in [6,19,20] that, in the presence of correlated packet errors, backing off transmissions may in fact be the right thing to do, at least as long as error bursts are long relative to the propagation delay of the connection. This is explained by noting that in this case long error bursts are persistent conditions (as is congestion), and therefore it is appropriate that protocol actions be triggered by them. In fact, the throughput performance of TCP has been shown to be enhanced by the presence of bursty errors, as opposed to random errors [6,19,20].

By the same token, one might expect that this mechanism is efficient from the energy management perspective as well. As discussed in [21,22], efficient usage of energy is achieved when a scheme stops transmitting when the channel conditions become adverse and resumes transmission when they improve. In fact, this is exactly what the window adaptation algorithm of TCP does. That is, even though that algorithm was designed for a wireline environment where energy is usually not an issue, its way of handling congestion turns out to be very efficient in guaranteeing reduced energy consumption as well. This fact is very interesting and sheds new light on the effectiveness of TCP as a transport protocol for wireless environments. Another interesting conclusion of this study is that, unlike throughput, the energy efficiency metric may be very sensitive to the TCP version used and to the choice of the protocol parameters, so that large gains appear possible.

The paper is organized as follows. In section 2, we describe the OldTahoe, Tahoe, Reno and NewReno versions of TCP. The system model and the correlated fading channel model considered in this paper are presented in section 3. The analytical approach is described in section 4, which reports the full analysis for TCP Tahoe. Section 5 discusses the numerical results, and conclusions and topics of future research are provided in section 6.

2. TCP OldTahoe, Tahoe, Reno and NewReno

A detailed description of the receive and transmit processes in TCP OldTahoe, Tahoe, Reno and NewReno has been given in [12,19]. We report it here for the reader's convenience, since the analysis given in the next section will heavily refer to the detailed protocol rules.

While the receive processes are the same for all of the TCP versions considered, their transmit processes are different in the way the *loss recovery phase* is implemented.

The TCP receiver can accept packets out of sequence, but will only deliver them 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 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. If such timer reaches the *round trip timeout* value (derived from a round trip time estimation procedure [17]) before the packet is acknowledged, timeout expiration occurs and 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 [17].

The basic window adaptation procedure, common to all TCP versions [18], works as follows. Let $W(t)$ be the transmitter's *congestion window width* at time t, and $W_{\text{th}}(t)$ be the *slow-start threshold* at time t. The evolution of $W(t)$ and $W_{\text{th}}(t)$ are triggered by ACKs (new ACKs, and not duplicate ACKs) and timeouts as follows:

1. If $W(t) < W_{\text{th}}(t)$, each ACK causes $W(t)$ to be incremented by 1. This is the *slow start* phase.
2. If $W(t) \geqslant W_{\text{th}}(t)$, each ACK causes $W(t)$ to be incremented by $1/W(t)$. This is the *congestion avoidance* phase.
3. If timeout occurs at the transmitter at time t, $W(t^+)$ is set to 1, $W_{\text{th}}(t^+)$ is set to $\lceil W(t)/2 \rceil$, 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 the *loss window*.

Besides running the basic window adaptation algorithm, the transmitter performs other three tasks which are related to packet losses:

- *loss detection:* a mechanism by which the transmitter concludes (correctly or incorrectly) that a packet was lost;
- *loss recovery phase:* a mechanism which allows the protocol to recover lost packets through retransmission;
- *window adaptation during loss recovery:* the way window adaptation is handled while lost packets are being recovered (different than the basic window adaptation in general).

The procedures implemented in TCP OldTahoe, Tahoe, Reno and NewReno are as follows.

- In the case of OldTahoe, loss detection and recovery is performed only through timeout and retransmission. Window adaptation during loss recovery follows the basic algorithm.
- In the case of Tahoe, a *fast retransmit* procedure is implemented for loss detection. If subsequent to a packet loss, the transmitter receives the Kth 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 $W_{\text{th}}(t^+)$ as given in the basic window adaptation algorithm described above.
- In the case of Reno also, the fast retransmit procedure following a packet loss is implemented. However, the subsequent recovery procedure is different. If the Kth duplicate ACK is received at time t, then $W_{\text{th}}(t^+)$ is set to $\lceil W(t)/2 \rceil$, and $W(t^+)$ is set to $W_{\text{th}}(t^+) + K$ (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 [8].
- In the case of NewReno, fast retransmit and congestion window adaptation are as in Reno, but the loss recovery mechanism is as in Tahoe [12]. The NewReno version of the protocol can recover from multiple losses in some cases, and is therefore more suited than Reno to the wireless environment, where packet losses may occur in bursts.

3. System model

We are interested in the performance of TCP during a bulk data transfer. Therefore, we neglect the connection set-up and tear-down phases, and focus only on the transfer of a large amount of data. The transmitter is assumed to always have packets to send. A single TCP connection is present over a channel with very small bandwidth-delay product, assumed to be zero in this study[1]. Acknowledgement messages are therefore assumed to be instantaneously available to the transmitter at the end of each slot, and also error-free[2].

3.1. Error model

Multipath fading exhibits memory and in the presence of short packet durations (e.g., as found on relatively high-data-rate links) this results in correlated packet errors. Therefore, commonly used models where errors occurring in different slots are independent and identically distributed do not apply in this case. Error models which explicitly account for memory are more adequate for the present study.

The issue of how to model fading channels has received much attention, and it is in general very difficult to be able to capture in a very accurate way the artifacts arising in the wireless environment. Following [25], we adopt here a simple two-state Markov model for the process of packet errors. More accurate models for fading channels may involve a number of states larger than two. For the purposes of this study, the two-state model appears to be the one which combines sufficient accuracy with analytical simplicity. For studies in which larger time scales are involved (e.g., delay analysis), more accurate models may have to be used (see [23] for a discussion). The accuracy of the two-state model used will be discussed later in the paper.

For a two-state error model, the statistics of the packet errors is fully characterized by the transition matrix of such process:

$$M_{\text{c}} = \begin{pmatrix} p_{BB} & p_{BG} \\ p_{GB} & p_{GG} \end{pmatrix}, \tag{1}$$

where p_{BG} is the transition from bad to good, i.e., the conditional probability that a successful transmission occurs in a slot given that a failure occurred in the previous slot, and the other entries in the matrix are defined similarly. The average probability of a packet loss, ε, can be found from the probabilities in (1), which in turn depend on the physical characterization of the channel, usually expressed in terms of the *fading margin*, F (or, equivalently, the Signal-to-Noise Ratio, averaged over fading), and the *normalized Doppler frequency*, $f_{\text{D}}T$, where T is the packet duration.

More specifically, the model proposed in [25] assumes that whenever the received power falls below a threshold (which depends on the fading margin), the packet is incorrect, and that it is correct otherwise. For this simple model, we have

$$\varepsilon = 1 - \mathrm{e}^{-1/F}, \tag{2}$$

[1] Simulations showed that with round-trip times of one or two slots the results do not differ from those in the ideal case considered.

[2] Here, again, simulations show appreciable differences only for ACK error probabilities higher than about 0.01, so that the assumption of error-free transmission does not appear too restrictive.

and

$$p_{GB} = 1 - \frac{Q(\theta, \rho\theta) - Q(\rho\theta, \theta)}{e^{1/F} - 1}, \tag{3}$$

where

$$\theta = \sqrt{\frac{2/F}{1-\rho^2}}, \tag{4}$$

$\rho = J_0(2\pi f_D T)$ is the correlation coefficient of two successive samples of the complex amplitude of a fading channel with Doppler frequency f_D, taken T seconds apart, $f_D T$ is the normalized Doppler frequency, and

$$Q(x, y) = \int_y^\infty e^{-(x^2+w^2)/2} I_0(xw) w \, dw \tag{5}$$

is the Marcum Q function. In (5), I_0 is the modified Bessel function of the first kind and of zeroth order. The entries of the transition matrix can then be found from (2) and (3), by using the relationship $\varepsilon = p_{GB}/(p_{GB} + p_{BG})$. By choosing different values of $f_D T$, we can establish fading channel models with different degrees of correlation in the fading process.

The above model has the advantage of being completely analytical. The price to be paid in this case amounts to several assumptions, the major being:

(i) the threshold mechanism to decide for success or failure,

(ii) the assumption that the error process is two-state Markov, and

(iii) the assumption (following from (ii)) that the average length of an error burst is given by $1/p_{BG}$.

Of course, a more accurate way to study the behavior of TCP over a wireless channel would be to directly simulate the fading process and the consequent generation of bit errors, to map it into packet errors and then run TCP on top of the error process obtained, which will not be an analytical approximation but the true process, at least within the accuracy of the simulation tool for the fading process. This has the undesirable property of being an all-simulation approach, with the obvious costs in terms of running time. Also, a major drawback of this approach is that it depends on the number of bits in a packet (whereas in the above analytical model everything is summarized in the parameters F and $f_D T$), and on the coding scheme used, if any.

An intermediate approach is to obtain a packet error trace by simulating the fading process, to estimate from it the average packet error rate and the average length of a burst of errors, and to apply the TCP analysis while using the Markov parameters obtained in this way instead of relying on the analytical model of [25]. This approach has the advantage of overcoming assumption (i) above and, partially, also assumption (iii). In fact, the actual bit errors and packet errors are generated. Also, instead of measuring the transition probability between consecutive slots and extrapolating the burst length from it, the actual burst lengths are used. If the error process is not exactly Markov, then the parameter values obtained in the two cases may be considerably different. On the other hand, the assumption that the error process is two-state Markov is still made here.

Later in the paper, all three models will be used to obtain some results and to compare different solutions, also gaining some insight on the behavior of the error process due to fading. In most of the results, we will use the Markov model with simulated parameters. The main reason for using simulated parameters rather than those given by the fully analytical approach is that we want to be as accurate as possible in relating the fading margin (transmitted power) to the packet error rate, to be accurate in investigating energy consumption performance and energy/throughput tradeoffs.

4. Analysis

The analytical approach here described is based on the theory of Markov/renewal reward processes [9]. The joint evolution of the window adaptation algorithm and of the channel state can be represented by a random process. This process has a very complicated evolution in general, due to the timeout mechanism which requires that the ages of the outstanding packets (i.e., packets for which an ACK is still expected) be tracked.

However, if we look at this process at some particular instants, it turns out that the matter is very much simplified, and an analytical approach can be developed and solved. For concreteness, we examine here in detail the case of TCP Tahoe. TCP Reno and NewReno, whose loss recovery phase is a little more elaborate, can be analyzed using similar techniques, although with a little more effort [19].

Let us ignore for the moment the need to track time, transmissions and successes, and let us focus on the joint evolution of the window adaptation parameters, W and W_{th}, and of the channel state. This evolution is composed of *cycles* where a cycle is the time evolution between two loss detection events.

More specifically, let us count time in slots (equal to the packet transmission time) and let us define t_k as the slot immediately following the detection of a packet loss. This detection may be triggered by either a timeout or the reception of the Kth duplicate ACK. In both cases, however, the action taken by the algorithm is the same: the slow start threshold is set to $W_{th}(t_k) = \lceil W(t_k - 1)/2 \rceil$, the window size is set to $W(t_k) = 1$, and all timeouts are reset so that there are no outstanding packets in the system. Then, in slot t_k, the oldest unacknowledged packet is retransmitted. From this point on, as long as there are successful transmissions, the system will keep running, the window size will grow and packets will keep being transmitted and acknowledged until at some point there will be an incorrect transmission. This incorrect transmission will lead to a packet loss detection which will be resolved by either timeout or duplicate ACKs, and in the slot immediately following this detection, i.e., slot t_{k+1}, retransmission of the oldest unacknowledged packet is performed and so on. Thus, cycle k will cover slots $\{t_k, t_k+1, \ldots, t_{k+1}-1\}$, and at time t_{k+1} cycle $k+1$ will start.

Let us define a random process as $X(k) = (C(t_k-1), W(t_k), W_{\text{th}}(t_k))$, where $C(t) \in \{B, G\}$ is the channel state in slot t (with B for a failure and G for a success). From the protocol rules, and from the assumption that the channel error process is Markov, we can conclude that, conditioned on $X(k)$, the future evolution of the process $X(m),\ m > k$, is independent of the past, $X(m),\ m < k$, i.e., the random process $X(k)$ is a Markov process.

In order to get useful results from the above setup, we need to be able to count how many packets are transmitted and how many of them are successfully received, and also to track time. These quantities are not included in the above model, which is therefore to be extended to incorporate this information as well.

Given a Markov chain, it is always possible to define a semi-Markov process as in [9, chapter 10]. This new process admits the original chain as its *embedded Markov chain*, which determines the transition structure and the transition probabilities. In addition, by introducing metrics on the transitions we can track other events and random quantities. In particular, we will be interested in the following quantities associated to a given transition: number of slots, N_{d}, number of transmission attempts, N_{t}, and number of successful transmissions[3], N_{s}. By following the evolution of the embedded Markov chain while cumulating the metrics on each transition, it is possible to compute a variety of quantities, as detailed in the following.

4.1. Markov approach

Let t_k be the kth sampling instant determined according to the above rules (without loss of generality, assume $t_1 = 1$.) *Cycle k* was defined as the time evolution of the system between the two consecutive sampling instants t_k and t_{k+1}. The statistical behavior of a cycle only depends on the channel state at time $t_k - 1$ and on the slow start threshold and window size at time t_k.

When we look at the evolution of the process during a generic cycle k, it is implicit in what follows that system variables are conditioned on $X(k) = (C(t_k-1), W_{\text{th}}(t_k), W(t_k)) \in \Omega_X$, where Ω_X is the set of all possible values of $X(k)$ (state space of the sampled process). For the case under consideration we have

$$\Omega_X = \left\{(C, W_{\text{th}}, 1), C = B, G, 1 \leqslant W_{\text{th}} \leqslant \left\lceil \frac{W_{\max}}{2} \right\rceil \right\}, \quad (6)$$

since after detecting a loss at time $t_k - 1$ we have always $W(t_k) = 1$ and $W_{\text{th}}(t_k) = \lceil W(t_k-1)/2 \rceil \leqslant \lceil W_{\max}/2 \rceil$. Note that the number of states is just $2\lceil W_{\max}/2 \rceil \leqslant W_{\max} + 1$. For simplicity of notation, let $C(t_k - 1) = C$.

For convenience, we divide a cycle into two phases. The first phase starts with slot t_k and terminates when the first error occurs at time $t_k + n$, where the conditional probability distribution of n, given the initial state $X(k)$, is given by

$$\alpha_C(n) = P\big[\text{first error at } t = t_k + n \mid C(t_k) = C\big] = \begin{cases} p_{CB}, & n = 1, \\ p_{CG}\, p_{GG}^{n-2}\, p_{GB}, & n > 1. \end{cases} \quad (7)$$

Note that for a stationary error process (7) is independent of k.

The second phase of cycle k starts at time $t_k + n + 1$ and ends upon detection of the loss which occurred at time $t_k + n$. Cycle $k+1$ starts in the following slot, which by definition is t_{k+1}.

Let $(C(t_k+n), W_{\text{th}}(t_k+n), W(t_k+n))$ be a variable which tracks the joint channel/window state at the end of phase one. We know that for sure $C(t_k+n) = B$. Also, by looking at the rules of the window adaptation algorithm, we note that after the loss at time $t_k + n$ the window parameters $W_{\text{th}}(t)$, $W(t)$ are not updated until the loss is detected, at which time one sets $W_{\text{th}}(t_{k+1}) = \lceil W(t_k+n)/2 \rceil$, $W(t_{k+1}) = 1$. It is then clear that the state of the Markov chain at the beginning of the next cycle, $X(k+1)$, is independent of $W_{\text{th}}(t_k+n)$, which is therefore irrelevant. An appropriate variable to track the system state between the two phases is therefore $Y(k) = W(t_k + n)$, which takes on values in $\Omega_Y = [1, W_{\max}]$. In order to track the effect of this variable on the future evolution of the system, the state space Ω_Y can be quantized into a finite number of intervals (see [19, appendix] for more details).

The system evolution can then be represented as a sequence of cycles, each composed of two phases. We can study the two phases separately, obtaining the two matrix transition functions $\mathbf{\Phi}^{(1)}(z)$ and $\mathbf{\Phi}^{(2)}(z)$, where the ijth entry of $\mathbf{\Phi}^{(1)}(z)$ is the transition function associated to the transition from state $i \in \Omega_X$ to state $j \in \Omega_Y$, and analogously the $j\ell$th entry of $\mathbf{\Phi}^{(2)}(z)$ is the transition function associated to the transition from state $j \in \Omega_Y$ to state $\ell \in \Omega_X$ [9]. The statistics of the system evolution during a cycle is then characterized by the matrix transition function $\mathbf{\Phi}(z)$, obtained by combining the two phases, i.e.,

$$\mathbf{\Phi}(z) = \mathbf{\Phi}^{(1)}(z)\mathbf{\Phi}^{(2)}(z), \quad (8)$$

whose entries are the transition functions associated to transitions from Ω_X to itself during a full cycle.

The above functions are transform representations of the statistics of the metrics for each transition, and the variable z is in fact a vector of variables, one per metric being tracked. More precisely, let $\xi_{ij}(N_{\text{d}}, N_{\text{t}}, N_{\text{s}})$ be the probability that the system makes a transition to state j in exactly N_{d} slots, and that in $\{1, 2, \ldots, N_{\text{d}}\}$ N_{t} transmission attempts are per-

[3] We remark here that the the notion of successful transmissions is not necessarily the same as that of successfully received packets. It is possible that a packet is successfully received but cannot be acknowledged because of previous losses. Future evolution may lead to another transmission of this packet, again with good channel. In this case, even though we have two successfully received transmissions, only one of them is to be counted towards throughput, as they are the same packet.

formed and N_s transmission successes are counted, given that the system was in state i at time 0. Then, we have

$$\Phi_{ij}(z_\mathrm{d}, z_\mathrm{t}, z_\mathrm{s}) = \sum_{N_\mathrm{d}, N_\mathrm{t}, N_\mathrm{s}} \xi_{ij}(N_\mathrm{d}, N_\mathrm{t}, N_\mathrm{s}) z_\mathrm{d}^{N_\mathrm{d}} z_\mathrm{t}^{N_\mathrm{t}} z_\mathrm{s}^{N_\mathrm{s}}, \qquad (9)$$

where the sum is extended to all possible values of N_d, N_t, N_s.

With this notation, the transition matrix of the embedded Markov chain $X(k)$ is just given by $\boldsymbol{P} = \boldsymbol{\Phi}(1, 1, 1)$. The matrix of average delays can be found as

$$\begin{aligned} \boldsymbol{D} &= \left.\frac{\partial \boldsymbol{\Phi}(z_\mathrm{d}, z_\mathrm{t}, z_\mathrm{s})}{\partial z_\mathrm{d}}\right|_{z_\mathrm{d}, z_\mathrm{t}, z_\mathrm{s}=1} \\ &= \boldsymbol{D}_1 \boldsymbol{\Phi}^{(2)}(1, 1, 1) + \boldsymbol{\Phi}^{(1)}(1, 1, 1)\boldsymbol{D}_2, \end{aligned} \qquad (10)$$

where

$$\boldsymbol{D}_1 = \left.\frac{\partial \boldsymbol{\Phi}^{(1)}(z_\mathrm{d}, z_\mathrm{t}, z_\mathrm{s})}{\partial z_\mathrm{d}}\right|_{z_\mathrm{d}, z_\mathrm{t}, z_\mathrm{s}=1}, \qquad (11)$$

$$\boldsymbol{D}_2 = \left.\frac{\partial \boldsymbol{\Phi}^{(2)}(z_\mathrm{d}, z_\mathrm{t}, z_\mathrm{s})}{\partial z_\mathrm{d}}\right|_{z_\mathrm{d}, z_\mathrm{t}, z_\mathrm{s}=1}. \qquad (12)$$

The averages of the other quantities, $\boldsymbol{T}$, $\boldsymbol{S}$, can be found similarly.

Let $A(k)$ and $B(k)$ be two cumulated metrics during the first k cycles of the system evolution. From the theory of renewal reward processes [9,16] we have that with probability one

$$\lim_{k\to\infty} \frac{A(k)}{B(k)} = \frac{E[A]}{E[B]} = \frac{\sum_{i\in\Omega_X} \pi_i \sum_{j\in\Omega_X} P_{ij} A_{ij}}{\sum_{i\in\Omega_X} \pi_i \sum_{j\in\Omega_X} P_{ij} B_{ij}}, \qquad (13)$$

where π_i, $i \in \Omega_X$, are the steady-state probabilities of the Markov chain with transition matrix $\boldsymbol{P}$, and A_{ij}, B_{ij} are the averages of the corresponding metric during transition ij.

By setting A, B appropriately, one can compute various metrics. For example, we have the following cases:

- $A = S, B = D$: average number of successes per slot (throughput);
- $A = T, B = D$: average number of transmissions per slot (system load);
- $A = S, B = T$: average number of successes per transmission (success probability).

From the second and third cases we can obtain the energy consumption and the energy efficiency, respectively, after multiplying $\boldsymbol{T}$ by a (normalized) measure of the power transmitted when the user is active, for example the fading margin or the average SNR[4].

Note that the above definition for the energy consumption assumes a model in which power is consumed during transmissions only. In reality, the terminal consumes some power also when idle, and if the percentage of idle time is large this contribution cannot be neglected.

A more complete model may be obtained by defining

$$\boldsymbol{C} = (F + w_\mathrm{a})\boldsymbol{T} + w_\mathrm{i}(\boldsymbol{D} - \boldsymbol{T}), \qquad (14)$$

where $\boldsymbol{C}$ is the matrix of the average energy consumption per transition, F is the amount of energy consumed by the terminal during one transmission[5], and w_a, w_i are the amounts of energy consumed by the rest of the circuitry (i.e., excluding the transmission stage) during an active or idle slot, respectively. Note that w_a is assumed here to be independent of the fading margin. Note also that w_a, w_i are to be expressed in appropriate units consistent with the use of F for the energy consumed. Setting $w_\mathrm{a} = w_\mathrm{i} = 0$ gives the simple results as described above.

With the more complete model we can express the energy efficiency of the protocol (bits-per-joule rating) as

$$\text{energy efficiency} = \frac{\sum_{i\in\Omega_X} \pi_i \sum_{j\in\Omega_X} P_{ij} S_{ij}}{\sum_{i\in\Omega_X} \pi_i \sum_{j\in\Omega_X} P_{ij} C_{ij}}. \qquad (15)$$

We may elaborate on this expression to determine the effect of idle consumption on the overall performance. Let η be the average throughput, and let λ denote the energy efficiency and λ_0 its value when all consumption is due to transmissions. We have

$$\begin{aligned} \lambda &= \frac{E[S]}{E[C]} = \frac{E[S]}{(F + w_\mathrm{a} - w_\mathrm{i})E[T] + w_\mathrm{i}E[D]} \\ &= \frac{E[S]}{FE[T]}\left(1 + \frac{w_\mathrm{a} - w_\mathrm{i}}{F} + \frac{w_\mathrm{i}E[D]}{FE[T]}\right)^{-1} \\ &= \lambda_0\left(1 + \frac{w_\mathrm{a} - w_\mathrm{i}}{F} + \frac{w_\mathrm{i}\lambda_0}{\eta}\right)^{-1} \\ &\geqslant \lambda_0\left(1 - \frac{w_\mathrm{a} - w_\mathrm{i}}{F} - \frac{w_\mathrm{i}\lambda_0}{\eta}\right) \\ &\geqslant \lambda_0\left(1 - \frac{w_\mathrm{a}}{F} - \frac{w_\mathrm{i}}{F}\frac{1-\eta}{\eta}\right), \end{aligned} \qquad (16)$$

where we used the fact that

$$\frac{\lambda_0}{\eta} = \frac{E[S]}{FE[T]}\frac{E[D]}{E[S]} \leqslant \frac{E[D]}{FE[S]} = \frac{1}{F\eta}. \qquad (17)$$

Therefore, the actual value of the energy efficiency can be bounded as follows:

$$\lambda_0\left(1 - \frac{w_\mathrm{a}}{F} - \frac{w_\mathrm{i}}{F}\frac{1-\eta}{\eta}\right) \leqslant \lambda \leqslant \lambda_0. \qquad (18)$$

It can be clearly seen from (18) that as long as $F \gg w_\mathrm{i}$, w_a and for the cases of interest where the throughput is not too small (say, $\eta \geqslant 0.5$), λ_0 is a good approximation for λ. In this paper, we will only consider λ_0, since application of the above formulas requires detailed knowledge of the energy consumption quantities, which is beyond the scope of the

[4] What is of interest here is not the absolute value of the energy efficiency, which depends on the conventional way to define the unit of transmitted energy, but rather the relative comparison among different solutions. In this view, any quantity which is proportional to the transmitted power will do, as long as it is used consistently.

[5] That is, the energy consumption unit in this case is the energy spent to transmit one packet with fading margin of 0 dB.

present work. It is clear, however, that as soon as such quantities are known, application of the analysis readily provides numerical results.

In order to compute steady-state performance from these relationships, knowledge of $\boldsymbol{P} = \boldsymbol{\Phi}(1, 1, 1)$ and of $\boldsymbol{D}$, $\boldsymbol{T}$ and $\boldsymbol{S}$ is sufficient. On the other hand, the full transform expression of the transition functions, $\boldsymbol{\Phi}(z)$, may be useful in further elaborations (e.g., computation of higher moments [9]).

4.2. Computation of the transition functions

We first compute $\boldsymbol{\Phi}^{(1)}(z)$. Let $X = X(k) = (C, W_{\text{th}}, W)$. The first part of the cycle has a duration of $N_{\text{d}} = n$ slots with probability $\alpha_C(n)$, $n \geqslant 1$. Conditioned on the value of n, the value of the window size at time n is a deterministic function $W(n) = w(n, W, W_{\text{th}}) \triangleq Y$ of W_{th} and W, and is assumed to be known[6].

Conditioned on the value of n, we have that $N_{\text{d}} = n$ slots, $N_{\text{t}} = n$ packet transmissions and $N_{\text{s}} = n - 1$ packet successes. Therefore,

$$\boldsymbol{\Phi}^{(1)}_{XY}(z_{\text{d}}, z_{\text{t}}, z_{\text{s}}) = \sum_{n \in \mathcal{C}(X,Y)} \alpha_C(n) z_{\text{d}}^n z_{\text{t}}^n z_{\text{s}}^{n-1}, \tag{19}$$

where $\mathcal{C}(X, Y) = \{n\colon\ w(n, W, W_{\text{th}}) = Y\}$ is the set of all values of n which result in the same window size.

As to the computation of $\boldsymbol{\Phi}^{(2)}(z)$, note that phase two may end in two different ways. If after the loss at time n and before the window fills up, there occur K successful transmissions, fast retransmit is triggered and a cycle starts in the slot following the Kth success. On the other hand, if the window fills up before K successful transmissions occur, then the transmitter stalls until the timeout timer for the oldest loss expires at time $n + T_0 - 1$ and the next cycle will start at time $n + T_0$. For simplicity of notation, and without loss of generality, in the following we set $n = 0$.

4.2.1. Timeout

Let $\varphi_{m\ell}(j, x)$ be the probability that there are j successful transmissions in slots 1 through x and that the channel is in state ℓ at time x, given that the channel was in state m at time 0.[7] Note that $\varphi_{m\ell}(j, x) = 0$ for $j > x$. For a given value Y, consider the event that there are j successful slots in $\{1, 2, \ldots, \lfloor Y \rfloor - 1\}$ and that the channel state in slot $\lfloor Y \rfloor - 1$ is γ, with probability $\varphi_{B\gamma}(j, \lfloor Y \rfloor - 1)$. If $j < K$, this event will lead to timeout, and the system state at the beginning of the next cycle will be equal to $X(k + 1) = (C, \lceil Y/2 \rceil, 1)$. The transition metrics N_{d} and N_{t} are deterministic and given by $T_0 - 1$ and $\lfloor Y \rfloor - 1$, respectively. Note that on the other hand, N_{s} is a random variable. Moreover, since the number of successful transmissions to be counted as true successes (i.e., contributing to throughput) depends on what happens in the following cycles, considering the exact value of N_{s} in the analysis leads to a process which is not semi-Markov. A possible way around this problem is to define two slightly different ways to count successes, which provide stochastic bounds and result in semi-Markov processes. An obvious pessimistic bound is $N_{\text{s}} \geqslant 0$, whereas an optimistic bound is $N_{\text{s}} \leqslant j$, since the total contribution to throughput cannot exceed the total number of successful transmissions. For a specific event as defined above, these two bounds do not depend on past and future evolution, so that the semi-Markov character of the resulting processes is guaranteed.

The transition function which incorporates all the above events leading from Y to $X(k+1) = (C, \lceil Y/2 \rceil, 1)$ can then be written as

$$\sum_{\gamma=B,G} p_{\gamma C}\big(T_0 - \lfloor Y \rfloor\big) z_{\text{d}}^{T_0-1} z_{\text{t}}^{\lfloor Y \rfloor - 1} \times \sum_{j=0}^{K-1} \varphi_{B\gamma}\big(j, \lfloor Y \rfloor - 1\big) z_{\text{s}}^{N_{\text{s}}(\gamma, j)}, \tag{20}$$

where $0 \leqslant N_{\text{s}}(\gamma, j) \leqslant j$ and the appropriate $(T_0 - \lfloor Y \rfloor)$-step channel transition probability has been included to account for the channel evolution during the time when transmissions do not occur. These probabilities are given by

$$\begin{pmatrix} p_{BB}(i) & p_{BG}(i) \\ p_{GB}(i) & p_{GG}(i) \end{pmatrix} = \begin{pmatrix} p_{BB} & p_{BG} \\ p_{GB} & p_{GG} \end{pmatrix}^i. \tag{21}$$

4.2.2. Fast retransmit

Consider now the case in which fast retransmit is triggered, i.e., K successful transmissions occur before the window is filled. Let i be the slot in which the Kth successful transmission occurs. Since only $\lfloor Y \rfloor - 1$ transmissions are allowed after the loss, we have $K \leqslant i < \lfloor Y \rfloor$.

For the case $i = K$ we have that the loss at time 0 is followed by K successful transmissions, and the resulting duplicate ACKs trigger fast retransmit with the consequent start of a new cycle at time $K + 1$. The probability of this event is $p_{BG} p_{GG}^{K-1}$, and the associated time is $N_{\text{d}} = K$ slots, equal to the number of transmissions $N_{\text{t}} = K$. The window parameters at time $K + 1$ will be $W(K + 1) = 1$, $W_{\text{th}}(K + 1) = \lceil Y/2 \rceil$. As to the number of successes, we also have $N_{\text{s}} = K$. In fact, if the next cycle starts with a success, this will ACK all K packets along with the retransmitted packet. On the other hand, if the next cycle starts with a failure, since the window size has been set to 1 the algorithm will have to wait for the timeout, and will do so as many times as needed until at last a successful retransmission will occur. At that time, all K packets will be acknowledged along with the retransmitted packet. Therefore, we conclude that in this case all K successful transmissions must be counted as successes since they will eventually be acknowledged without being retransmitted. We therefore conclude that the event under consideration corresponds to a transition from Y to $X = (G, \lceil Y/2 \rceil, 1)$ with transition function

$$p_{BG} p_{GG}^{K-1} z_{\text{d}}^K z_{\text{t}}^K z_{\text{s}}^K. \tag{22}$$

[6] For example, it can be tabulated by a simple program.

[7] Both a recursive technique and explicit expressions for these probabilities are given in [24]. Note that the functions ϕ in that paper are defined in a slightly different way. However, it is straightforward to relate them by noting that $\varphi_{BG}(j, x) = \phi_{10}(j - 1, x)$ and $\varphi_{GB}(j, x) = \phi_{01}(j + 1, x)$, whereas in the other two cases they are the same.

Note that in the definition of $X(k)$ the channel state refers to time $t_k - 1$, which in this case is necessarily a success.

For the case $i > K$, we have $i - K$ losses in $\{1, 2, \ldots, i\}$. The above argument about counting successes applies, but in this case the packets which will certainly be acknowledged without being retransmitted are only those transmitted in $\{1, 2, \ldots, \ell_1 - 1\}$, where ℓ_1 is the slot in which the first loss after the one at time 0 occurs. Other successfully transmitted packets in $\{\ell_1 + 1, \ell_1 + 2, \ldots, i\}$ will not be acknowledged upon correct retransmission of the packet lost at time 0, and they may or may not be acknowledged without being retransmitted, depending on the specific evolution. A detailed study would be too tedious in this case, so we use a bounding technique to estimate the number of successes to be counted. Of course, we have the upper bound $N_s \leqslant K$. For the lower bound, based on the above discussion we have that at least $\ell_1 - 1$ packets will eventually be acknowledged when a successful retransmission of the packet lost at time 0 occurs. Therefore, we have $N_s \geqslant \ell_1 - 1$. For the other metrics, we have $N_d = N_t = i$.

Finally, let $\mathcal{B}(i, \ell_1)$, $K < i < \lfloor Y \rfloor$, $0 < \ell_1 \leqslant K$ be the event that the Kth success occurs at time i and the first loss after the loss in 0 occurs at time ℓ_1. We have

$$P\big[\mathcal{B}(i, \ell_1)\big] = \begin{cases} p_{BB}\varphi_{BG}(K, i-1), \\ \quad \ell_1 = 1, \\ \quad i = K+1, \ldots, \lfloor Y \rfloor - 1, \\ p_{BG}p_{GG}^{\ell_1 - 2}p_{GB}\varphi_{BG}(K - \ell_1 + 1, i - \ell_1), \\ \quad \ell_1 = 2, \ldots, K, \\ \quad i = K+1, \ldots, \lfloor Y \rfloor - 1. \end{cases} \tag{23}$$

The transition which corresponds to $\mathcal{B}(i, \ell_1)$ again goes from Y to $X = (G, \lceil Y/2 \rceil, 1)$ with transition function

$$P\big[\mathcal{B}(i, \ell_1)\big] z_d^i z_t^i z_s^{N_s(\ell_1)}, \tag{24}$$

where $\ell_1 - 1 \leqslant N_s(\ell_1) \leqslant K$.

5. Numerical results

In this section, we present numerical results obtained via the analytical approach previously discussed. In the following, the Tahoe, Reno and NewReno versions of TCP will be considered, whereas TCP OldTahoe will not be considered due to poor performance. As discussed in section 3, we consider the flat Rayleigh fading channel [11], whose error process is approximated by a two-state Markov chain. In particular, the degree of memory of the channel directly depends on the normalized value of the Doppler frequency, $f_D T$, where f_D is the maximum Doppler shift [11] and T is the packet duration.

We assume here a wireless link running at 1.5 Mbps. TCP packets are 1024 bytes long, which results in a packet length T of a little more than 5 ms. As the focus here is on slow fading channels, the considered range of values for the normalized Doppler frequency, $f_D T$, is between 0.001 and 0.256. In the 900 MHz band, the largest value corresponds to about 5 km/h, whereas the lowest corresponds to essentially stationary terminals. With the assumed packet size, $f_D T = 0.256$ already gives results close to the iid error case, so that the range considered covers all cases of interest. The baseline case has $K = 3$ for the fast retransmit threshold, and a minimum timeout $T_0 = 100$ slots, which is close to the usual 500-ms timer granularity. For the size of the maximum advertized window, $W_{\max} = 6$ and 24 packets have been considered.

5.1. Effect of the channel model

Before we present and discuss TCP performance results, we take a brief look at the issue of channel modeling. Our goal here is to model the channel error process in a simple way, while retaining sufficient accuracy in the evaluations.

In figures 1–4 we compare the throughput performance of TCP Tahoe for various values of the Doppler frequency and for $W_{\max} = 24$, as obtained via the following three approaches:

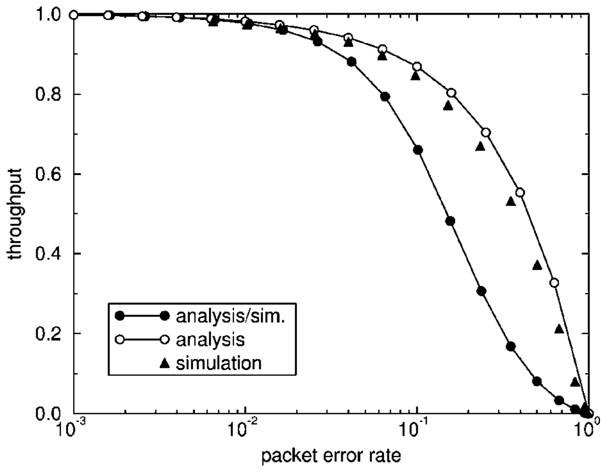

Figure 1. Throughput versus average error rate for TCP Tahoe with $W_{\max} = 24$, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.001$. Comparison of three models.

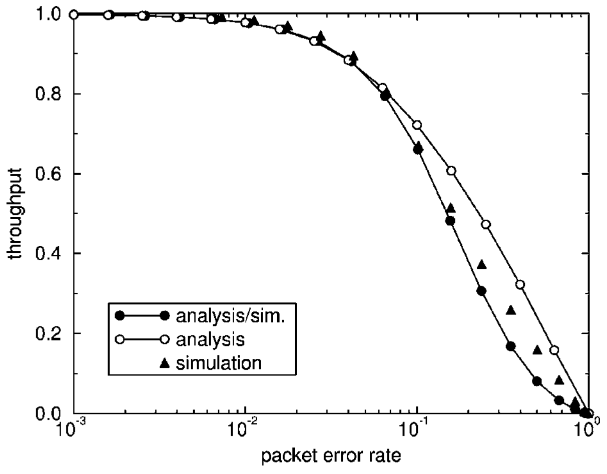

Figure 2. Throughput versus average error rate for TCP Tahoe with $W_{\max} = 24$, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.008$. Comparison of three models.

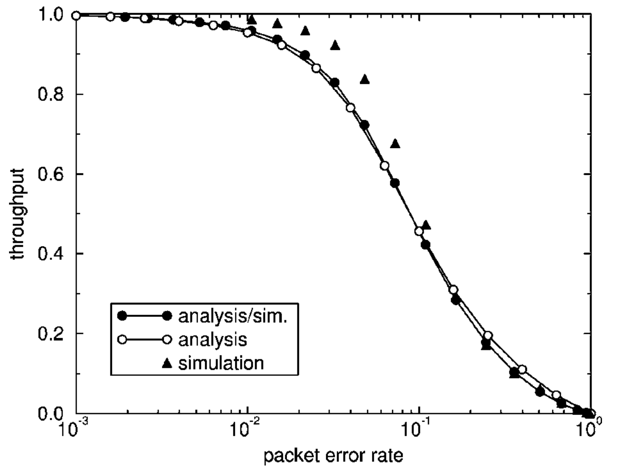

Figure 3. Throughput versus average error rate for TCP Tahoe with $W_{\max} = 24$, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.032$. Comparison of three models.

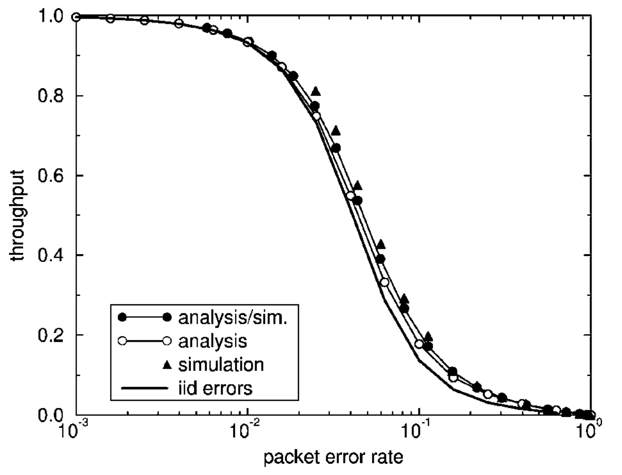

Figure 4. Throughput versus average error rate for TCP Tahoe with $W_{\max} = 24$, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.128$. Comparison of three models.

- the fully analytical approach, i.e., the Markov analysis of section 4 with the parameters of the error process computed as in [25] (marked as “analysis” in the plots);
- the fully simulated approach, where the actual TCP protocol is run over a trace of packet errors obtained through fading simulation according to Jakes [11] (marked as “simulation” in the plots);
- the hybrid approach, i.e., the Markov analysis of section 4 with the parameters of the error process estimated from the trace of packet errors obtained through fading simulation according to Jakes [11] (marked as “analysis/sim.” in the plots).

The values used for the normalized Doppler frequency are $f_D T = 0.001, 0.008, 0.032, 0.128$, each corresponding to a different plot. For the simulation approach, which depends on the physical layer details, a packet length of 1024 bytes with no coding has been assumed. From plots in figures 1–4, different behaviors can be observed. When the fading is very slow ($f_D T = 0.001$, figure 1), the actual simulated performance is in fact closer to the all analytical approach than to the (supposedly more accurate) hybrid approach. By comparing the values of the burst length obtained from (3) with those actually measured, one finds that the former overestimate the burstiness, and this explains the better performance obtained in this case with respect to the hybrid case, in which the true (lower) burstiness is used. However, as discussed in [23], in some cases the tail of the error burst distribution is heavier than predicted by the two-state Markov approach, so that the simulated performance is better than the hybrid results due to this increased burstiness. It appears that in this case the heavy tail of the simulated error process and the overestimation of the average burst length in the analytical approach lead to a similar effect on the numerical results.

For faster (while still slow in general) fading, this behavior is attenuated and the two analytical approaches get closer together. The agreement of the simulated results is good in general, although analytical predictions may turn out to be a little pessimistic for intermediate values of $f_D T$ (see figure 3). Finally, for $f_D T = 0.128$ all three approches agree, since the error process is weakly correlated, and the results obtained are close to the iid case (see figure 4).

In the following we will use the hybrid model, mainly because it provides the correct relationship between the transmitted power and the error parameters. The results of the above figures give an indication on the accuracy of the results presented in the rest of this section. In particular, most results closely match simulations, and are conservative in general.

5.2. TCP performance results

Figures 5 and 6 show the throughput performance of TCP Tahoe and TCP Reno as a function of the average packet error rate. Three values of the Doppler frequency are considered, namely, $f_D T = 0.001, 0.016$ and 0.256.

In the presence of bursty packet errors, two conflicting effects influence the performance. For a given value of the average packet error rate, clustered errors correspond to fewer error *events* (each comprising multiple packet errors), and therefore the congestion window is shrunk less frequently than for iid 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, $K = 3$ duplicate ACKs must be successfully received after a packet loss. If a packet is corrupted by the channel at time t, only $W(t) - 1$ more packets can be transmitted, with $W(t) - 1 \leqslant W_{\max} - 1$ (= 5 in the case of figure 5). 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. This explains why the results for Tahoe

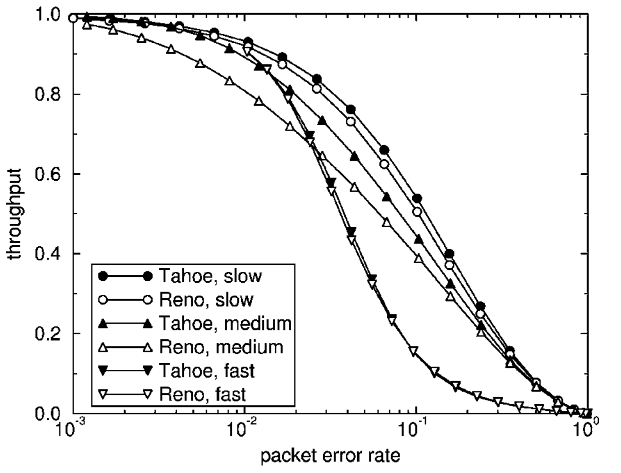

Figure 5. Throughput versus average error rate for TCP Tahoe and TCP Reno with $W_{\max} = 6$, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.001$ (slow), 0.016 (medium) and 0.256 (fast).

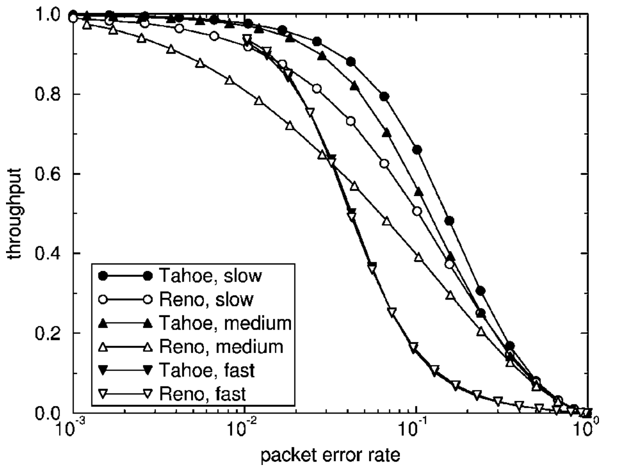

Figure 6. Throughput versus average error rate for TCP Tahoe and TCP Reno with $W_{\max} = 24$, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.001$ (slow), 0.016 (medium) and 0.256 (fast).

in figure 6 (where $W_{\max} = 24$) are noticeably better than those in figure 6 (where $W_{\max} = 6$). On the other hand, the rules of Reno are such that when two or more errors occur in the same window the system will timeout regardless of the window size, and this is why the performance of Reno in figures 5 and 6 is essentially the same. Finally, as to NewReno, results similar to Tahoe have been observed (not shown in the figures for clarity), with a little advantage of NewReno over Tahoe for fast fading [19].

The results of figures 5 and 6 clearly show that for relatively large values of the error rate the decreased frequency of error events induced by error clustering overweighs the negative effect of multiple errors, and the throughput is higher for slower fading. On the other hand, for small values of the error rate, the increased fraction of error events which trigger a timeout because of window filling may prevail, and this explains why, in the left-hand side of the graphs, correlated fading corresponds to worse performance than fast fading.

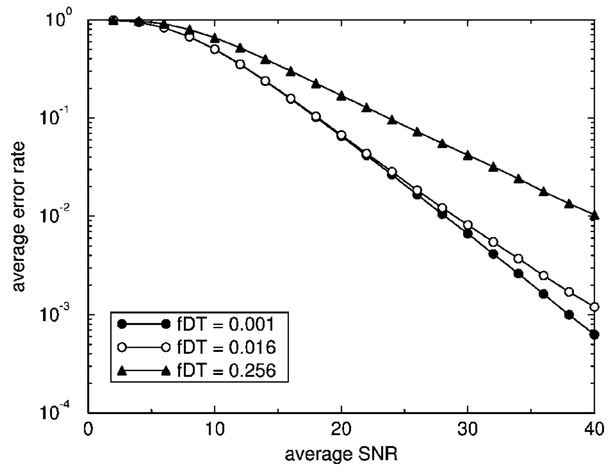

Figure 7. Average packet error rate versus average Signal-to-Noise Ratio, $f_D T = 0.001, 0.016$ and 0.256, packet size 1024 bytes.

It should be noted here that the plots shown have the average error rate on the horizontal axis, as is commonly done in the literature on this topic. However, different values of the Doppler frequency may result in different error rates for the same average SNR on the radio channel. In fact, a packet is more likely to be hit by an error on a rapidly varying channel than it is on a slowly varying channel. The average error rate versus the average SNR over the radio channel (i.e., averaged over the fading process) is shown in figure 7 for the same values of the parameter $f_D T$ as used previously. It is clearly seen here that fast fading leads to worse performance in general. By the same token, the plot of figure 5 is redrawn in figure 8 with the average SNR rather than the error rate on the horizontal axis. It is clearly seen here that faster fading is always worse than slow fading. As a last remark, we note that this conclusion also depends on the absence of error correction codes in the case considered. It is well known that in the presence of error correction codes the interaction between the error correction capability of the code and the time diversity inherent in fast fading channels may give some advantage.

Another important conclusion to be drawn from the above results is that TCP Reno performs worse than Tahoe in virtually all cases (with the possible exception of infrequent and iid errors [19]). This fact is due to the bad response of TCP Reno to multiple errors in the same loss window, which leads to many timeouts in the presence of even slightly correlated fading. TCP NewReno does not suffer from this limitation, and performs close to Tahoe in this environment.

To deepen our understanding of the energy consumption performance of TCP, we look at the average number of transmissions per success, which is the inverse of the success probability of the transmissions. This metric is related to how much energy is spent to get a packet across the channel. Figure 9 shows an example of the results obtained in this case for $W_{\max} = 24$ and various fading rates. It is interesting to note that, unlike throughput, this metric is not very sensitive to the error correlation, except for situations close to iid errors. Also, it is interesting to note that, although

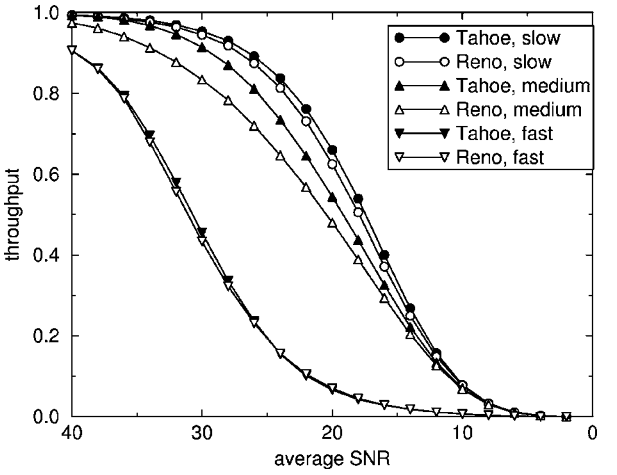

Figure 8. Throughput versus average SNR for TCP Tahoe and TCP Reno with $W_{\max} = 6$, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.001$ (slow), 0.016 (medium) and 0.256 (fast).

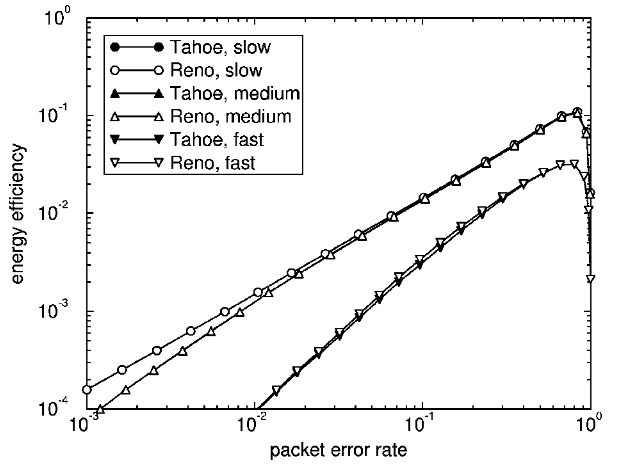

Figure 10. Energy efficiency versus average error rate for TCP Tahoe and TCP Reno with $W_{\max} = 24$, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.001$ (slow), 0.016 (medium) and 0.256 (fast).

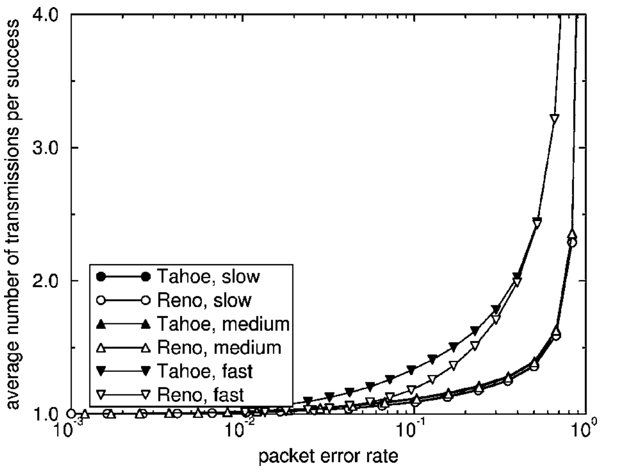

Figure 9. Average number of transmissions per success versus average error rate for TCP Tahoe and TCP Reno with $W_{\max} = 24$, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.001$ (slow), 0.016 (medium) and 0.256 (fast).

less efficient than Tahoe in terms of throughput, Reno totals fewer transmissions than Tahoe, due to its more conservative loss recovery algorithm. In any case, unless the error rate exceeds 0.1, all protocols appear to be efficient in that they do not transmit packets many times. As before, iid errors represent a worst case. Similar results for $W_{\max} = 6$ (not presented here) show that this metric is also relatively insensitive to the maximum window size.

Various authors have observed that the TCP congestion control mechanism, which stops or slows down transmissions in the face of detected losses, does the wrong thing since it reacts to channel errors with a scheme designed for congestion. It is interesting to note that, from the energy efficiency point of view, the congestion control algorithm does in fact the *right* thing, as it avoids bad channel conditions.

Energy consumption by itself does not tell the whole story, since one must consider also how much data is being delivered for a given power level. A more significant parameter is the energy efficiency, which is the ratio between throughput and average energy consumption, i.e., the bits-per-joule rating of the protocol [22]. It relates consumed energy to actual useful throughput, and therefore captures the way energy is spent in delivering data. Figure 10 shows the energy efficiency, i.e., the average number of successes per unit energy, which is obtained by taking the inverse of the average number of transmissions per success (as given in figure 9) and dividing it by the SNR, which is proportional to the transmitted power. It is seen that the energy efficiency is an increasing function of the error rate, i.e., that worse channel conditions correspond to better energy usage. Although this may sound surprising, in fact it is not, since worse channel conditions result from less energy being used per transmission, and the increased number of transmissions which results overweighs the decreased success probability of each [22]. In this view, a small error rate clearly corresponds to a small energy efficiency due to the large SNR required in this case. On the other hand, the energy efficiency does not grow indefinitely as the error rate increases, because at some point the effect of too many errors and of little delivered data becomes dominant. What is interesting is that Tahoe and Reno do not have essentially different performance in terms of energy efficiency. Also, we can note the degradation of the energy efficiency as the fading becomes faster, due to the higher SNR necessary to achieve a given value of the average error rate. Accordingly, the slope of the curves for slow and medium fading in the left-hand side of the plot is about unity, which corresponds to a relationship between error rate and SNR of one order of magnitude per decade. For fast fading, this slope is increased, consistent with the results shown in figure 7.

Note that the energy efficiency improvement as fading margin is decreased comes at the expense of degraded delay and throughput performance. Even though we do not address delay here, it is intuitively clear that when more errors are allowed the delay incurred by a packet is larger. Therefore, in discussing the possible benefits on energy ef-

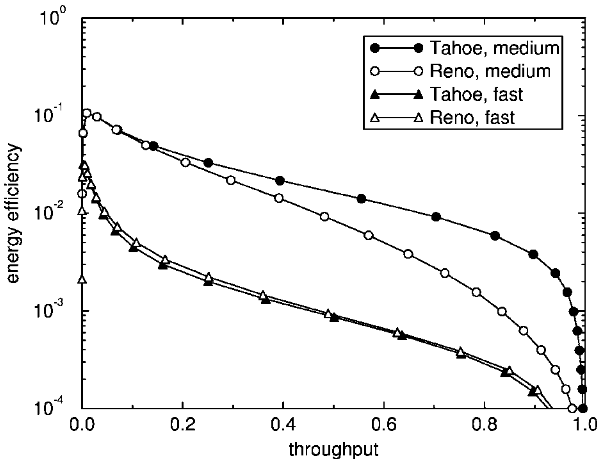

Figure 11. Energy efficiency versus throughput for TCP Tahoe and TCP Reno with $W_{\max} = 24$, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.016$ (medium) and 0.256 (fast).

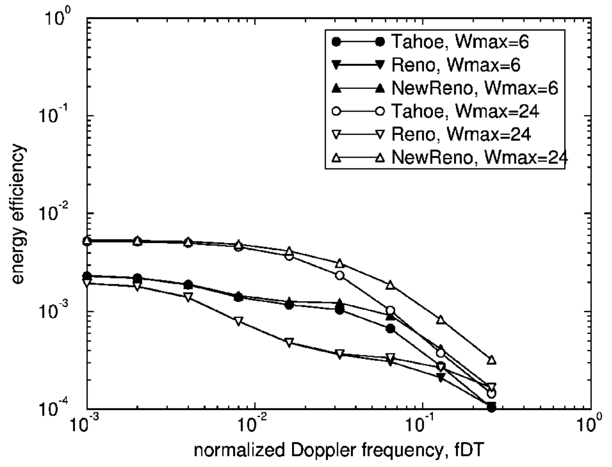

Figure 13. Energy efficiency versus normalized Doppler frequency for TCP Tahoe, Reno and NewReno with $W_{\max} = 6$ and 24, $K = 3$, $T_0 = 100$, packet size 1024 bytes. All points correspond to 90% throughput.

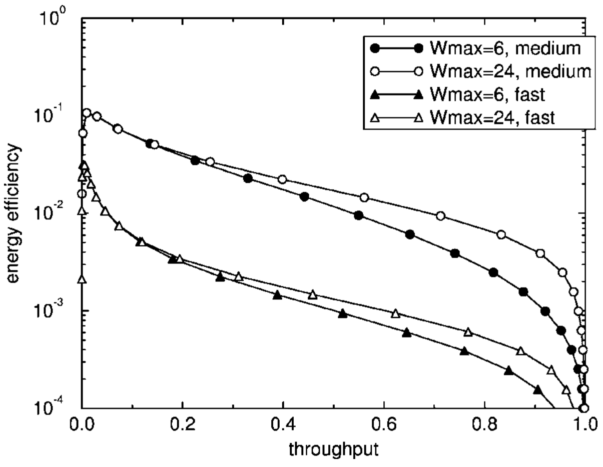

Figure 12. Energy efficiency versus throughput for TCP NewReno with $W_{\max} = 6$ and 24, $K = 3$, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.016$ (medium) and 0.256 (fast).

ficiency achieved by using reduced transmission power, we must keep in mind that delay may suffer. Evaluation of the tradeoff between energy efficiency and delay is left for future study.

Figures 11 and 12 show some examples of what we see as the most interesting way to present the energy efficiency results, i.e., through the energy efficiency versus throughput tradeoff. Curves in the upper right portions of the plots correspond to more energy efficient protocols. This representation gives a clear understanding of how instantaneous rate of data delivery (in good packets per slots) can be traded off for better usage of the energy source, and may help in assessing the energy savings achievable according to the application's requirements. Not surprisingly, higher throughput comes at the expense of reduced energy efficiency. Figures 11 and 12 illustrate quantitatively the trade-off between throughput and energy efficiency, i.e., the price to pay in terms of energy efficiency if one wants a larger throughput. Interestingly, this tradeoff is less advantageous for slow fading than it is for fast fading, although the former case results in better performance in general. It is also to be noted that for the medium fading case considered, TCP Tahoe and TCP NewReno perform about the same (compare figures 11 and 12), whereas for fast fading some difference can be observed.

From the above results, it is clear that the Doppler frequency plays a major role in determining the performance of the system, since it affects both the burstiness and the average error rate. As we already observed, a faster fading process has a doubly negative effect, since it corresponds to higher error rate (in the absence of coding) and to shorter bursts, and both effects result in decreased performance. This conclusion is supported by the results of figure 13, where the energy efficiency of TCP Tahoe, Reno and NewReno is plotted as a function of the normalized Doppler frequency, $f_D T$. All curves plotted correspond to throughput equal to 0.9, and $W_{\max} = 6$ and 24 is assumed. It can be seen that, as the fading rate is increased, the energy efficiency suffers in all cases, with a degradation in excess of an order of magnitude across the range considered. The worst performance of course corresponds to TCP Reno, which is known to be the poorest. TCP Tahoe and NewReno, as already observed, exhibit similar performance, except for the case of fast fading, where NewReno performs better because of its generally larger window size due to the algorithm rules.

Another design parameter is the timeout, which was set to $T_0 = 100$ slots in all the above results. As an example of the results obtained for different values of T_0, figure 14 shows the energy efficiency versus throughput tradeoff for Tahoe and Reno for $f_D T = 0.016$ and $W_{\max} = 24$. The results show that shorter timeouts result in better performance in general. TCP Reno is more sensitive to the timeout value than TCP Tahoe, since timeouts are more likely to occur. Likewise, we found that the case $W_{\max} = 6$ is more sensitive to T_0 than the case $W_{\max} = 24$ since again in this case more timeouts will occur. As a general comment, the difference in performance between the cases $T_0 = 50$ and $T_0 = 150$ may be significant, e.g., a factor of 3 or even more for the energy efficiency. This clearly indicates that the smallest possible

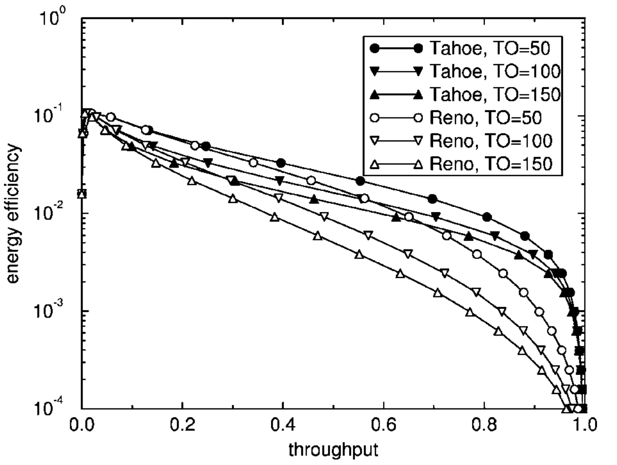

Figure 14. Energy efficiency versus throughput for TCP Tahoe and TCP Reno with $W_{\max} = 24$, $K = 3$, $T_0 = 50$, 100 and 150, packet size 1024 bytes, $f_D T = 0.016$.

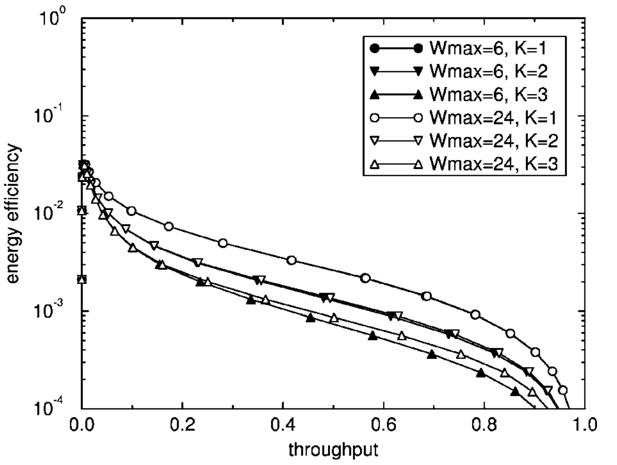

Figure 15. Energy efficiency versus throughput for TCP Tahoe with $W_{\max} = 6$ and 24, $K = 1, 2$ and 3, $T_0 = 100$, packet size 1024 bytes, $f_D T = 0.256$.

timeout should be chosen, which was to be expected in this environment where round-trip times are negligible. On the other hand, granularity of the possible timeout values will pose a limit on T_0 (in fact our choice of $T_0 = 100$ roughly corresponds to 500 ms, a commonly assumed value of the timer granularity).

Finally, the effect of the fast retransmit threshold is examined in figure 15. It is seen that a more prompt trigger for fast retransmit results in better performance, which again was to be expected in an environment where ACK are instantaneously available. While this difference is very significant for the case shown of fast fading, for slow fading it was observed to be minimal [19].

Additional results have been obtained for a number of cases, although not shown here due to lack of space. In any event, the qualitative behavior is similar, and the main findings are summarized below. The results obtained confirm in the energy efficiency domain what has already been found for throughput performance, i.e.:

- TCP Tahoe performs better than TCP Reno, and TCP NewReno performs slightly better than Tahoe;
- correlated fading is more benign than fast fading;
- a larger advertized maximum window, $W_{\max}$, allows to fully exploit the advantages of correlated errors;
- the advantage of using NewReno instead of Tahoe vanishes as error correlation is increased;
- the advantage of using smaller K (number of duplicate ACKs after which a retransmission is triggered) may be significant for fast fading, but vanishes as error correlation is increased.

Even though the choice of a logarithmic scale for the energy efficiency axis somewhat attenuates large variations, a closer look at some of the graphs reveals that some protocol or parameter choices may have a major impact on the energy performance. For example:

- at 90% throughput and for correlated fading, using $W_{\max} = 24$ instead of $W_{\max} = 6$ in NewReno triples the battery life (figure 12);
- in the same conditions, using Reno instead of Tahoe leads to a factor of about ten in energy efficiency loss (figure 11);
- decreasing the timeout value T_0, if possible, leads to significant savings;
- for fast fading and Tahoe at 90% throughput the energy efficiency is doubled when passing from $K = 3$ to $K = 2$, and almost again doubled when passing from $K = 2$ to $K = 1$. That is, using $K = 1$ instead of $K = 3$ increases battery life three- to fourfold (figure 15).

In the above, "double battery life" should really be interpreted as double energy efficiency. In this case, the same amount of data is transmitted across the channel (and correctly received) with half the energy or, equivalently, twice as much data can be received with the same energy expense. It should be noted that the classic notion of "battery life" as applies to circuit-switching (where there is a one-to-one correspondence between transmitted data and battery life in actual time) does not apply here directly, since packet switching is used instead. For example, a very long battery life could always be achieved by almost never transmitting, but this would be of little use, showing that a metric which focuses on the useful data instead of time duration of a battery charge is much more meaningful here.

We may conclude that if choosing the "right" TCP version may lead to some incremental performance advantage in terms of throughput, the energy efficiency performance can be dramatically increased by the appropriate choice of the protocol and its parameters.

6. Summary

In this paper, we have analyzed the energy consumption performance of various versions of TCP, for bulk data transfer in an environment where channel errors are correlated. We investigated the performance of a single wireless TCP connection by modeling the correlated packet loss/error process as a first-order Markov chain. The main findings of our study were that (1) error correlations significantly affect the energy performance of TCP (consistent with analogous conclusions for throughput), and in particular slow fading may result in considerably better performance for Tahoe and NewReno than fast fading, and (2) the congestion control mechanism implemented by TCP does a good job at saving energy as well, by backing off and idling during error bursts. An interesting conclusion is that, unlike throughput, the energy efficiency metric may be very sensitive to the TCP version used and to the choice of the protocol parameters, so that large gains appear possible.

Topics of ongoing investigation include the study of the effect of using a link-layer FEC/ARQ scheme, and the effect of other physical layer parameters (e.g., packet size) on the overall TCP performance. Other performance metrics besides throughput (e.g., delay) are also being considered.

References

[1] O. Ait-Hellal and E. Altman, Analysis of TCP Vegas and TCP Reno, in: *Proc. IEEE ICC'97* (1997).

[2] H. Balakrishnan, V.N. Padmanabhan, S. Seshan and R.H. Katz, A comparison of mechanisms for improving TCP performance over wireless links, ACM/IEEE Transactions on Networking (December 1997).

[3] H. Balakrishnan, S. Seshan, E. Amir and R.H. Katz, Improving TCP/IP performance over wireless networks, in: *1st Intl. Conf. on Mobile Computing and Networking* (November 1995).

[4] P. Bhagwat, P. Bhattacharya, A. Krishna and K. Tripathi, Using channel state dependent packet scheduling to improve TCP throughput over wireless LANs, Wireless Networks (1997) 91–102.

[5] R. Caceres and L. Iftode, Improving the performance of reliable transport protocols in mobile computing environments, IEEE Journal on Selected Areas in Communications 13(5) (June 1995) 850–857.

[6] A. Chockalingam, M. Zorzi and R.R. Rao, Performance of TCP on Wireless Fading Links with Memory, in: *Proc. IEEE ICC'98* (June 1998).

[7] A. DeSimone, M.C. Chuah and O.C. Yue, Throughput performance of transport layer protocols over wireless LANs, in: *Proc. IEEE Globecom'93* (December 1993) pp. 542–549.

[8] K. Fall and S. Floyd, Comparisons of Tahoe, Reno, and Sack TCP (March 1996) `ftp://ftp.ee.lbl.gov`

[9] R.A. Howard, *Dynamic Probabilistic Systems* (Wiley, 1971).

[10] IEEE Personal Communications Magazine, ed. M. Zorzi, Special issue on Energy Management in Personal Communications and Mobile Computing 5(3) (June 1998).

[11] W.C. Jakes, Jr., *Microwave Mobile Communications* (Wiley, New York, 1974).

[12] A. Kumar, Comparative performance analysis of versions of TCP in a local network with a lossy link, ACM/IEEE Transactions on Networking (August 1998).

[13] 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, Technical report WINLAB-TR-133 (November 1996).

[14] T.V. Lakshman and U. Madhow, The performance of TCP/IP for networks with high bandwidth-delay products and random loss, IEEE/ACM Transactions on Networking (June 1997) 336–350.

[15] P. Manzoni, D. Ghosal and G. Serazzi, Impact of mobility on TCP/IP: An integrated performance study, IEEE Journal on Selected Areas in Communications 13(5) (June 1995) 858–867.

[16] S.H. Ross, *Stochastic Processes* (Wiley, 1983).

[17] W.R. Stevens, *TCP/IP Illustrated*, Vol. 1 (Addison Wesley, 1994).

[18] Van Jacobson, Congestion avoidance and control, in: *Proc. ACM Sigcomm'88* (August 1988) pp. 314–329.

[19] M. Zorzi, A. Chockalingam and R.R. Rao, Performance analysis of TCP on channels with memory, IEEE Journal on Selected Areas in Communications (July 2000).

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

[21] M. Zorzi and R.R. Rao, Error control and energy consumption in communications for nomadic computing, IEEE Transactions on Computers 46, Special Issue on Mobile Computing (March 1997) 279–289.

[22] M. Zorzi and R.R. Rao, Energy constrained error control for wireless channels, IEEE Personal Communications Magazine 4 (December 1997) 27–33.

[23] M. Zorzi and R.R. Rao, On channel modeling for delay analysis of packet communications over wireless links, in: *Proc. 36th Annual Allerton Conference* (September 1998).

[24] M. Zorzi and R.R. Rao, Lateness probability of a retransmission scheme for error control on a two-state Markov channel, IEEE Transactions on Communications 47 (October 1999).

[25] M. Zorzi, R.R. Rao and L.B. Milstein, On the accuracy of a first-order Markov model for data transmission on fading channels, in: *Proc. IEEE ICUPC'95* (November 1995) pp. 211–215.

Michele Zorzi received the Laurea Degree and the Ph.D. in electrical engineering from the University of Padova, Italy, in 1990 and 1994, respectively. During the academic year 1992/93, he was on leave at the University of California, San Diego (UCSD), attending graduate courses and doing research on multiple access in mobile radio networks. In 1993, he joined the faculty of the Dipartimento di Elettronica e Informazione, Politecnico di Milano, Italy. After spending three years with the Center for Wireless Communications at UCSD, in 1998 he joined the School of Engineering of the Università di Ferrara, Italy, where he is currently a Professor. His present research interests include performance evaluation in mobile communications systems, random access in mobile radio networks, and energy constrained communications protocols. Dr. Zorzi currently serves on the Editorial Boards of the IEEE Personal Communications Magazine and of the Journal of Wireless Networks. He is also a guest editor for special issues in the IEEE Personal Communications Magazine (Energy Management in Personal Communications Systems) and the IEEE Journal on Selected Areas in Communications (Multi-media Network Radios).
E-mail: zorzi@ing.unife.it

Ramesh R. Rao received his Honours Bachelor's degree in electrical and electronics engineering from the University of Madras in 1980. He did his graduate work at the University of Maryland, College Park, Maryland, receiving the MS degree in 1982 and the Ph.D. degree in 1984. Since then he has been on the faculty of the Department of Electrical and Computer Engineering at the University of California, San Diego. His research interests include architectures, protocols and performance analysis of computer and communication networks.
E-mail: rao@ece.ucsd.edu