

Markov-based Modeling of Wireless Local Area Networks

Syed A. Khayam and Hayder Radha

Department of Electrical & Computer Engineering / 2120 Engineering Building
Michigan State University, East Lansing, MI 48824, USA
+1-517-432-9958

{khayamsy, radha}@egr.msu.edu

ABSTRACT

Errors introduced by a wireless medium are more frequent and profound than contemporary wired media. Some of these errors, which are not corrected by the physical layer, result in Medium Access Control (MAC) layer bit errors and packet losses. Design of wireless protocols and applications can benefit substantially from a thorough understanding of these MAC layer impairments. This paper evaluates and proposes Markov-based stochastic chains to model the 802.11b MAC-to-MAC channel behavior for both bit errors and packet losses. We introduce an Entropy Normalized Kullback-Leibler measure to evaluate the performance of existing and new bit error and packet loss models. Based on the proposed measure, and contrary to recent results for mobile networks, we demonstrate that the traditional two-state Markov chain provides a very suitable model for the 802.11b MAC-to-MAC packet loss process. However, this simple model is not adequate for bit errors observed at the MAC layer of wireless local area networks. Consequently, we evaluate three other Markov-based chains for modeling these errors: full-state, hidden, and hierarchical Markov chains. Among these chains, we illustrate that the full-state Markov bit error model, evaluated under a wide range of order values, renders the best performance.

Categories and Subject Descriptors

C.2.1 [Network Architecture and Design]: Wireless Communication.

General Terms

Measurement, Performance, Experimentation, Theory.

Keywords

802.11b Networks, MAC-to-MAC, Markov-based Modeling, Information Theoretic Evaluation.

1. INTRODUCTION

Wireless networks suffer from frequent errors and losses due to their vulnerability to interference and transmission medium degradation. In view of these impairments, real-time applications, which can inherently tolerate a certain level of errors and losses, targeted for wireless networks are introducing enhanced error resilience, e.g., slices in JVT, and reversible variable length coding in MPEG-4. In addition, cross-layer protocol schemes have been

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MSWiM'03, September 19, 2003, San Diego, California, USA.
Copyright 2003 ACM 1-58113-766-4/03/0009...\$5.00.

proposed recently to fully exploit the error resilience features [1], [2], [3]. Design of these protocols and applications requires a thorough understanding of the bit error and packet loss patterns encountered at and above the MAC layer of the wireless protocol stack.

Generally, statistical channel models are employed to characterize the error and loss behavior of a network. In addition to the channel insight, these models allow study of the actual source via simulations. Moreover, a model can adapt to varying network conditions and, therefore, can be employed for accurate channel characterization in real-time. This real-time channel characterization yields significant dividend in the context of rate adaptive applications. For example, based on this channel forecast, scalable and/or rate-adaptive multimedia encoders can adapt their overall source bitrate and/or error-control redundancy.

In the context of wireless link layer modeling, Konrad *et al.* performed analysis and presented a Markov-based Trace Analysis (MTA) model algorithm for GSM networks [4]. Ji *et al.* [5] compared the performance of the MTA, a full-state k -order model, a hidden Markov model and an extended ON(error-free)/OFF(error-filled) model in capturing the GSM (link layer) frame losses. Similarly, and in view of the rapidly growing ubiquity of 802.11 networks, we conducted a MAC-to-MAC investigation at the 802.11b link layer in order to facilitate design of effective cross-layer error control schemes for the support of real-time services [6], [7]. Since most error control schemes operate on byte and/or packet boundaries, we proposed Markov-based models at the packet- and byte-level. Although [4] outlines the inadequacy of the two-state Markov chain for GSM-based networks, we proposed that this (rather simplistic) model is adequate for the bursty 802.11b packet loss process. Furthermore, we proposed a hierarchical Markov Model (hMM¹) for the byte-level errors.

In this paper, we evaluate the performance of existing models and, when necessary, propose new models to appropriately represent the 802.11b packet loss and bit error patterns. The analytic nature of this work necessitates an appropriate statistical measure which can accurately quantify the performance of a model. Here, it is should be re-emphasized that one major motivation of this study is to develop models which can simulate the network behavior by generating accurate packet loss and bit error traces. Such models will allow users to perform experiments on the network without having the actual source available. With this consideration, we define a novel information theoretic measure which we refer to as the Entropy Normalized Kullback Leibler (ENK) distance. The ENK indicates an entropy-based source-coding-like overhead incurred by the use of a model instead of the actual source.

¹ In this paper, the *hierarchical* and *hidden* Markov models are respectively abbreviated as *hMM* and *HMM*.

In this paper, we first investigate the ENK-based performance of the two-state packet loss Markov chain proposed in [6]. Our results show that the two-state Markov chain captures the packet loss process very adequately. Interestingly, this result is incongruent with recent (similar) studies since [4] outlines the insufficiency of the two-state model in approximating the GSM loss process. Thus it can be inferred that the GSM and 802.11b networks have dissimilar loss behavior. This substantiates the need for 802.11b link layer investigation at finer levels of granularity.

We, therefore, shift our focus to analysis and modeling at the bit-level. First, we extend the byte-level hierarchical Markov model, proposed in [6], to the bit-level. We show that at the bit-level the hMM exhibits significant improvement potential. Since the byte-level hMM was designed to facilitate delivery of real-time media, its underlying dependence on application-specific parameters deteriorates its performance at the bit-level.

Consequently, we resort to the (traditional) high-order Markov chains to model the MAC-to-MAC bit error behavior. We compute the autocorrelation for different traces and establish that a Markov chain of order 14 (i.e., 2^{14} states) is required to represent the bit error process. Before employing such an involved model, we investigate the suitability of a relatively less-complex model, i.e. the Hidden Markov Model (HMM), for the problem under investigation. We show that short-term energy (i.e., energy within a window) is an appropriate feature to identify trace segments with errors. Therefore, we employ an energy sequence, which is derived from the bit error traces, to train the HMM. We show that the HMM also incurs considerable ENK-based overhead.

Finally, we employ the traditional high-order Markov models, which we refer to as the Full-State Markov (FSM) chains, to model the bit error process. Previous studies limited the order of the Markov chain to constrain the complexity of the model [4], [5]. We, however, evaluate the suitability of all Markov chains from order 1 up to order 14 (i.e., 2^k states where $k=1,2,\dots,14$) to capture the channel bit error behavior. We outline that the models have sparse representations and that for higher order chains some states are never visited. We refer to such states as “unused states” and our analysis employs only the “used states”. We illustrate that the model performance improves with the order and all FSM chains of order 9 and above render accurate models for the bit error process.

The remainder of this paper is structured as follows. Section 2 provides some background information about the use of autocorrelation to determine the order of a Markov chain and the ENK measure. Section 3 explains the data collection set up and procedure. Section 4 evaluates the performance of the two-state packet loss model. Section 5 first evaluates the performance of the hierarchical Markov model. This section then proposes and evaluates two other models, namely the hidden and the full-state Markov models. Section 6 summarizes some key conclusions of this work.

2. BACKGROUND

In this section we provide brief background about autocorrelation of random processes and the use of Entropy Normalized Kullback-Leibler distance to quantify the information theoretic (source-coding-like) overhead of a model.

2.1 Autocorrelation of Random Processes

Let $\bar{X}(n_1)$ and $\bar{X}(n_2)$ be two random variables derived from a random process \bar{X}_n . The “sample” correlation coefficient [8] of these random variables is defined as,

$$\rho(\eta) = \left(\mathbb{E}\{\bar{X}(0)\bar{X}(\eta)\} - \mathbb{E}\{\bar{X}(0)\}\mathbb{E}\{\bar{X}(\eta)\} \right) / \sigma_{\bar{X}(0)}\sigma_{\bar{X}(\eta)}$$

where, $\mathbb{E}\{\bar{X}\}$ and $\sigma_{\bar{X}}$ represent the “sample mean” and “sample

standard deviation” of the random variable \bar{X} . The sample autocorrelation coefficient, when computed for different values of the lag, is a direct metric for the level of temporal dependence in the random process. For a large range of statistical data, autocorrelation between two symbols (random variables) decreases rapidly with the increase in the lag between them. Lag beyond which the autocorrelation coefficient drops to an insignificant value is known as the *memory length* of the process. In slightly relaxed jargon, memory length represents the lag beyond which the symbols (random variables) constituting the random process are (virtually) uncorrelated. Thus, autocorrelation of realizations of a Markov source yields the order of the model required to approximate the respective process.

2.2 Entropy Normalized Kullback-Leibler Distance

From a source coding perspective, entropy provides a measure for the average number of bits required to represent a source completely. Entropy is largely dependent on the random variable used to represent the random experiment. Assuming the availability of an appropriate random variable (\bar{X}) defined over an alphabet set Ψ , entropy provides a weighted average of the minimum information² of the source. Entropy is expressed as,

$$H(p(\bar{X})) = - \sum_{\bar{X} \in \Psi} p(\bar{X}) \log(p(\bar{X})) \quad (1)$$

where, $p(\bar{X})$ is the probability mass function of \bar{X} .

Let $p(\bar{X})$ and $q(\bar{X})$ be two probabilities distributions defined over a common alphabet set Ψ . The *Kullback-Leibler distance* [9] renders a measure of the (statistical) divergence between $p(\bar{X})$ and $q(\bar{X})$ as,

$$D(p(\bar{X}) \parallel q(\bar{X})) = \sum_{\bar{X} \in \Psi} p(\bar{X}) \log(p(\bar{X})/q(\bar{X})) \quad (2)$$

Thus, the Kullback-Leibler distance provides a non-negative statistical divergence measure, which is zero if and only if $p=q$ [9]. When a base-2 logarithmic measure is used, the Kullback-Leibler distance gives the number of overhead bits incurred because a model (represented by $q(\bar{X})$) is used instead of the actual source (represented by $p(\bar{X})$). Since the Kullback-Leibler distance is also completely dependent on the choice of the random variable \bar{X} , utmost care should be exercised in selecting the random variable that represents the underlying random process.

² Here, the term *information* corresponds to the number of bits required to uniquely represent all possible outcomes of the source.

In order to accurately judge the performance of a model, the Kullback-Leibler measure should be weighted with respect to the entropy. For example, let us assume that the entropy of the source is 20 bits whereas the overhead incurred by the model is 0.75 bits. The overhead is relatively insignificant since the source inherently requires a large number of bits to be represented. However, for the same overhead (of 0.75 bits), if the entropy of the source is low, say 1 bit, then an overhead of 0.75 bits is extremely high. Hence, for accurate performance evaluation of a model, both the entropy of the process (represented by some random variable \bar{X}) and the Kullback-Leibler distance should be taken into consideration.

In view of the above considerations, we define a new statistical divergence measure namely the *Entropy Normalized Kullback-Leibler (ENK) distance*. The ENK renders a measure of the source-coding-like overhead incurred by employing a model instead of the actual (random) source. Mathematically,

$$ENK(p(\bar{X})\|q(\bar{X})) = \frac{D(p(\bar{X})\|q(\bar{X}))}{H(p(\bar{X}))} \quad (3)$$

where, $D(p\|q)$ and $H(p)$ are defined in (2) and (1) respectively.

A closer examination reveals that the ENK inherits basic properties of the Kullback-Leibler distance: (a) non-negativity, $ENK(p\|q) \geq 0$, (b) non-symmetry,

$$ENK(p\|q) \neq ENK(q\|p), \text{ and (c) } ENK(p\|q) = 0 \Leftrightarrow p = q.$$

Thus, small values of the ENK indicate that the model renders a good approximate of the (actual) random source. Conversely, large values of the ENK imply that the source-coding-like overhead of the model is large, i.e., the model is not a good approximate of the (actual) source. Note that we would expect the ENK between two ‘‘actual’’ observation (symbol) sequences to be a very small value since both the observations have been realized by the same random source. For instance, the ENK between two traces collected over the 802.11b wireless medium under similar conditions should be quite small. This ENK value can be used as an evaluation reference for the ENK between the actual observations (i.e., traces collected over the wireless network) and the model-based observations (i.e., traces artificially generated by the model).

3. DATA COLLECTION

We collected traces under various network scenarios for over one year. In particular, three scenarios were strenuously investigated: (a) the effect of client positions on the error rate, (b) the packet throughput variations for multiple (overlapping) basic service sets, and (c) the trade-off between the number of clients and the packet throughput³ in a basic service set. While examining (a), it was observed that the even slight changes in client position can drastically degrade/improve the throughput. Nevertheless, the traces collected within a certain (small) radius experienced only meager packet throughput fluctuations. In case of (b) and (c), the (overall) throughput decreased since the available bandwidth was

shared among more wireless stations. However, the error behavior of the channel was largely unaltered.

We generated bit error traces at all bitrates supported by the standard under various settings of an 802.11b network. Network traffic at many constant bitrates was transmitted over the wireless medium. All the bit error traces were collected at the clients by modifying the wireless device drivers. More specifically, the clients were Linux boxes using DLink DWL-650 wireless PC-Cards with prism2 device drivers. The modified client device drivers passed all the packets to a (link layer) raw socket. Thus the traces collected at the clients included successful (i.e., packets with no errors) and unsuccessful (i.e., packets failing the 802.11 MAC layer checksum) transmissions. These link layer traces were copied in the kernel buffer space from where an application thread periodically concatenated them in the user buffer space. The server and clients were always stationary and did not attempt re-transmissions during the course of an experiment.

Previous wireless studies (see for example [4], [5]) used loss traces and, therefore, did not have information about the distribution of errors inside each corrupted (dropped) packet. On the contrary, and due to our interest in analyzing bit errors inside corrupted packets, our traces provided information about all (i.e., successful and unsuccessful) bit transmissions. These bit-level traces were then extended to generate packet loss traces. More specifically, we used a constant packet size of 512 bytes (4096 bits) for all the experiments. After collecting bit error traces, non-overlapping windows of 4096 bits were examined for an unsuccessful bit transmission (i.e., a bit error). Since the retransmissions were disabled, and assuming an operational MAC layer checksum, any window with one or more bit errors was classified as a packet loss. Hence throughout this paper, a packet loss refers to a received packet with one or more bit errors.

In our initial experiments all wireless stations maintained Line of Sight (LoS) with the access point (AP). The AP was forced to transmit at 2, 5.5 and 11 Mbps for each observation. It was observed that with clear LoS, the error rate (at all bitrates) was extremely low. Such excellent performance deemed further LoS study inconsequential. Hence, we positioned the stations in separate rooms to simulate a more realistic business/classroom/home-network wireless setup.

In [6] we demonstrated that, within the notion of realistic network configurations, the 11 Mbps data rate incurs considerable errors which have a profound adverse impact on the throughput. This bitrate is, therefore, rendered inappropriate for high-bitrate multimedia transmission. On the contrary, the 2 Mbps data rate yields extremely good performance thus eliminating the need for further analysis at this bitrate. Furthermore, we illustrated that the 5.5 Mbps data rate, in conjunction with appropriate transport and application layer strategies, exhibits significant promise for the support of high-bitrate communication. Therefore, and in order to limit the scope of this work, we focus on the 5.5 Mbps data rate. The modeling techniques employed in this paper are, nevertheless, directly applicable to other bitrates.

For all analysis and modeling in this paper, we employ three 5.5 Mbps error traces that were collected under realistic home/classroom/e-business settings. For all three error traces, the wireless stations, placed in separate rooms, communicated in infrastructure network configuration as shown in Figure 1. The server always had clear LoS with the AP.

³ Since the 802.11 MAC layer drops all corrupted packets without regard to the number and location of the errors, the term *packet throughput* refers to the ratio between the number of successful packets received (i.e., packets with no errors) at the MAC layer and the total number of packets transmitted.

These traces were collected by transmitting a 512-byte packet every second. The rationale for using this packet rate is rooted in one of the main motivations of this study, that is, to provide the flexibility of real-time model parameter adaptation at the receiver. Such a model can track varying network conditions and, therefore, can be employed for accurate channel characterization in real-time. However, generally it is not possible to determine the exact location of bit errors in the received (actual application data). Thus, the server and the client have to exchange periodic probe packets. The predetermined content of these packets would allow detection of bit errors within the probe packets. The probe packets should, however, utilize minimal bandwidth. In view of such considerations, we believe that a constant probe data rate of 1 packet/sec (i.e., 4 Kbps) is reasonable, i.e., the probe packets use only a small fraction of the total bandwidth thereby sparing sufficient bandwidth for the application data. In order to derive the model parameters, we employ a trace window of approximately 1 million bits. The maximum and mean packet bursts for the traces are 21 and 1.878 respectively.

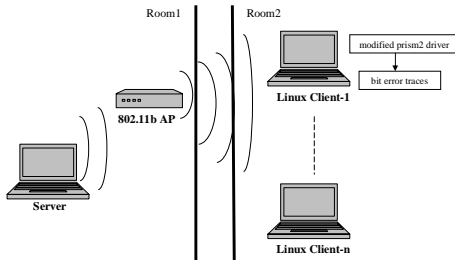


Figure 1. Set up for collection of bit error traces.

4. PACKET LOSS MODELING

Packet loss modeling is important for the design of both reliable and unreliable applications. In this section we evaluate the performance of the two-state Markov chain, proposed in [6], to capture the packet loss process. The two states of the model correspond to: 1) a *good* state representing a successful packet transmission (i.e., packet received at time n had no errors); and 2) a *bad* state representing a packet loss (i.e., packet received at time n had one or more errors and was dropped).

4.1 Performance of the Packet Loss Model

Throughout this work, Entropy Normalized Kullback-Leibler measure, defined in (3), will be used for statistical performance evaluation of the packet loss and bit error models. Recall that ENK is dependent on the choice of an appropriate random variable to represent the stochastic process. Hence a random variable that efficiently represents the packet loss process should be identified here.

We employ two random variables to represent the packet loss process: 1) inter-arrival-rate of packet loss bursts (\bar{I}), where \bar{I} takes on non-zero positive integers; 2) burst-length of packet losses (\bar{B}), where \bar{B} also takes on non-zero positive integers. Here, it is important to justify the rationale for selecting these two random variables. The first random variable (\bar{I}) characterizes (inversely) the frequency of occurrences of the packet drops. Small values of \bar{I} imply large numbers of (or frequent) packet loss events, and vice versa. Thus, the inter-arrival-rate random variable (\bar{I}) is tantamount to the number of good packets (i.e., packets with no errors) received between occurrences of cor-

rupted packets. In other words, the inter-arrival-rate random variable represents the burst-length of good packets. The second random variable (\bar{B}) represents the length of consecutive packet losses. Bursty losses are more detrimental to data/multimedia transmissions than isolated losses. The burstiness of the loss process is adequately characterized by these two random variables.

Due to the non-symmetric nature of our divergence measure, we exercised care in maintaining a specific order while computing the ENK between two traces. In our evaluation, we employ a particular *source-based*⁴ trace as the reference. Thus, a source-based trace S_{ref} provides the “reference” probability distribution (p) of the ENK measure. Another source-based trace S_{sec} is used for computing the “secondary” probability distribution (q) of the ENK measure. This ENK between two source-based traces serves as a performance criterion while evaluating the ENK between a source- and a model-based trace. While evaluating the performance of a model, a source-based trace again serves as the “reference” probability distribution of the ENK measure. However, the “secondary” probability distribution (q_m) of the ENK measure is

rendered by the model-based trace m . The random variable \bar{X} in (3) can be either the inter-arrival-rate or the burst-length, i.e., $\bar{X} = \bar{I}$ or $\bar{X} = \bar{B}$. We present comparison between $ENK(p_{s_1}||q_{s_3})$, $ENK(p_{s_2}||q_{s_3})$, $ENK(p_{s_1}||q_m)$ and $ENK(p_{s_2}||q_m)$. Note again that $ENK(p_{s_1}||q_{s_3})$ and $ENK(p_{s_2}||q_{s_3})$ render reference values against which $ENK(p_{s_1}||q_m)$ and $ENK(p_{s_2}||q_m)$ can be evaluated. Table 1 tabulates the ENK of the source- and model-based observations for the two random variables.

Table 1. Performance of the two-state packet loss model

	\bar{I}	\bar{B}
$ENK(p_{s_1} q_{s_3})$	0.02535	0.03849
$ENK(p_{s_2} q_{s_3})$	0.03621	0.056467
$ENK(p_{s_1} q_m)$	0.006113	0.019183
$ENK(p_{s_2} q_m)$	0.01883	0.04829

For both of the random variables, the ENK-based overhead incurred by the model is nominal. In fact, the overhead incurred by the model is always lower than the reference overhead values, i.e.,

$$ENK(p_{s_1}||q_m) < ENK(p_{s_1}||q_{s_3}) \text{ and } ENK(p_{s_2}||q_m) < ENK(p_{s_2}||q_{s_3})$$

for both \bar{I} and \bar{B} . These results clearly depict the appropriateness of the two-state model in approximating the packet loss patterns. Hence, we conclude that the two-state (packet-level) model is adequate to capture the packet loss patterns of an 802.11b MAC-to-MAC channel.

3.2.1. Discussion

The above discussion clearly demonstrates the adequacy of the two-state Markov chain in capturing the packet loss process. This result is incongruent with a similar study which proves that a two-state Markov chain is incapable of capturing the loss phenomenon

⁴ We refer to the traces collected over the network as *source-based* since they are generated by the actual random source, i.e., the 802.11b MAC-to-MAC channel. The traces artificially generated by the model are referred to as the *model-based* traces.

observed over GSM networks [4]. Thus, it can be concluded that the loss patterns observed over the 802.11b networks are quite different from GSM networks. The extremely good performance of the loss model renders further packet loss analysis and modeling inconsequential. This model can be used as a generator of packet-level sequences and/or to study the behavior of the packet loss channel. Therefore, in the subsequent sections we focus our attention on bit error modeling and analysis.

5. BIT ERROR MODELING

In this section, we first extend the byte-level hierarchical Markov model (hMM) presented in [6] to the bit-level and evaluate its performance. We show that the model can be improved further and, therefore, propose a hidden Markov model. The hidden Markov model also yields a significant improvement potential. Hence, we propose a full-state Markov chain and demonstrate that the FSM renders the most accurate model for the bit error process.

5.1 The Hierarchical Markov Model

Under the proposed hMM [6], “severe-burst” and “low-burst” regions were identified in the byte-level error traces. Furthermore, each of the aforementioned states had a two-state Markov model embedded in it (as depicted in Figure 2). One of the challenges of the hMM model is the delineation of the high-level “severe-burst” and “low-burst” states. A state demarcation heuristic [6] was employed to delineate the low- and severe-burst states in the error traces. The heuristic utilized application-specific thresholds for identifying upper bounds for the runs of good and bad bytes.

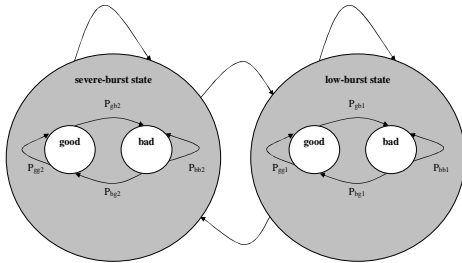


Figure 2. The hierarchical Markov model.

5.1.1 Performance of the Hierarchical Model

We again employ the same random variables, i.e., inter-arrival-rate (\bar{I}) and burst-length (\bar{B}), to evaluate the performance of the bit error models. However in this (bit-level) scenario, \bar{I} and \bar{B} respectively represent the inter-arrival-rate and burst-length of the bad bits. In accordance with our previous performance evaluation example, the primary (reference) distribution is always source-based and is denoted as p_{s_i} . The secondary distribution can be either source-based or model-based which are respectively referred to here as q_{s_j} and q_{hMM} . The performance of the hMM is tabulated in Table 2.

As mentioned previously, the ENK between the source-based traces (i.e., row 1 and 2 of Table 2) provide a reference value for performance evaluation of the hMM. It is obvious that the hMM incurs approximately 60% (Table 2 column \bar{I} - row 3 and 4) overhead for the inter-arrival-rate and, therefore, is rendering unsatisfactory performance. The burst-length random variable usually takes on small values since most of the bits are not corrupted during transmission and, hence, result in small (bit error) bursts. Therefore, it is important to quantify the hMM burst-length performance with respect to the source-based traces. It is obvious that

for the burst-length random variable, the ENK distance between the hMM- and source-based traces (Table 2 column \bar{B} - row 3 and 4) is much larger as opposed to the ENK between two source-based traces (Table 2 column \bar{B} - row 1 and 2). We conclude that although the hMM performs adequately in characterizing hybrid (i.e., bursty and isolated) byte-level error patterns, errors at finer (bit-level) granularity have significantly different behaviour which is not captured by this model.

Table 2. Performance of the hierarchical Markov model

	\bar{I}	\bar{B}
$ENK(p_{s_1} \ q_{s_3})$	0.008645	0.002903
$ENK(p_{s_2} \ q_{s_3})$	0.014066	0.001978
$ENK(p_{s_1} \ q_{hMM})$	0.587209	0.009064
$ENK(p_{s_2} \ q_{hMM})$	0.594563	0.013115

5.1.2 Discussion

Clearly, the hMM performance can be improved significantly. At this point, an obvious direction is to apply the traditional high-order Markov modeling technique which has shown ample promise for similar problems [4], [5]. Let us first determine the order of the Markov chain using autocorrelation analysis. We represent the bit error traces as a binary time-series $\{x(i)\}_{i=1}^l$, $x(i) \in \{0,1\}$, where $x(i)=1$ represents a bit error and l is the length of the time-series. For a memory length, k , let us define a full-state Markov (FSM) chain which corresponds to the 2^k different possible combinations of k consecutive bits. Transition probabilities between states are computed by the number of times a bit-pattern $\underline{x} = [x_1 x_2 \dots x_k]$ is followed by another bit-pattern $\underline{y} = [y_1 y_2 \dots y_k]$ in the bit error traces. The sample autocorrelation for the three 5.5 Mbps traces is illustrated in Figure 3.

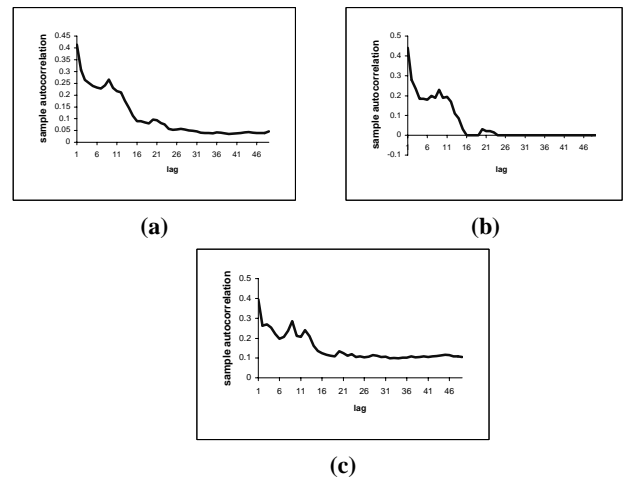


Figure 3. Autocorrelation of the bit error traces.

Clearly, the autocorrelation is a decaying function, i.e., the level of temporal dependence is decreasing with time. From the examples provided in Figure 3 we assume that the memory length is determined by the lag beyond which the correlation drops below 0.15 and stays within that bound. This correlation threshold of 0.15 was chosen after examination of a number of traces. We observed that in some traces, see for example Figure 3(c), the corre-

lation does not drop significantly below 0.15 even for very large lags. However, in general the bit errors exhibit rapidly decaying correlation as outlined by Figure 3(a) and (b). Hence, and as will be substantiated by the following performance evaluation, correlation of less than 15% does not play a significant role in the error process characteristics. Based on this correlation threshold of 0.15, the memory length for the traces of Figure 3 is 13, 12 and 14 respectively. Hence, we use memory length 14 as the maximum order of our Markov chain.

The total number of states for an order-14 FSM will be 2^{14} (=16384). Before proceeding with this somewhat involved modeling paradigm, it is important to explore other Markov-based techniques which might limit the overall model complexity while yielding appropriate performance. One such technique, which has been successfully applied to other fading channels, is hidden Markov modeling [10], [11], [12]. In the following section we develop a suitable HMM for the 802.11b bit error process.

5.2 The Hidden Markov Model

In order to apply the hidden Markov model to this problem, we first analyze trace segments with low and high error rates for (discriminative) statistical feature(s). We, then, use these feature(s) as input to the Expectation-Modification training algorithm [13].

5.2.1 Feature Selection

In this sub-section, we evaluate discriminative features that can differentiate trace segments with high and (relatively) low error rates. Examples of trace regions with low and high error rates computed over a 2000-bit window are presented in Figure 4. We compute the error rate as,

$$\text{error rate} = \frac{\text{total number of bad bits in a window}}{\text{total number of bits (good + bad) in a window}} \quad (4)$$

Figure 4(a) represents a low error rate trace segment since it comprises of a large number of good bits with some sporadic instances of bad bits. Figure 4(b) and (c) represent (relatively) high-error trace regions. The error rates are tabulated in Table 3.

Table 3. Error rate of trace segments

	Error Rate
Figure 4(a): low-error	0.0003628
Figure 4(b): high-error	0.1000723
Figure 4(c): high-error	0.0550724

Clearly, the channel can be classified as operating in two high-level conditions, i.e., the low- and high-error conditions. At this point, it is essential that discriminative features of these high-level conditions be determined. Note that the error rate of high- and low-error conditions varies by order of magnitude (0.0003 versus 0.1 and 0.05 in the above example). Hence, it can be concluded that the error rate formulates one such discriminative feature. A closer examination reveals that the error rate represents the normalized short-term energy (also referred to as *short-term power*) of a trace segment,

$$\text{normalized short_term energy} = \frac{1}{l} \sum_{i=1}^l x^2(i) \quad (5)$$

where, l represents the total number of bits in the window. Since $x(i) \in \{0,1\}$, we obtain $x^2(i) = x(i)$ which yields,

$$\text{normalized short_term energy} = \frac{1}{l} \sum_{i=1}^l x(i)$$

or,

$$\text{normalized short_term energy} = \frac{\text{total number of bad bits}}{l} = \text{error rate}$$

Thus, equations (4) and (5) represent the same feature which will, henceforth, be referred to as the *energy*.

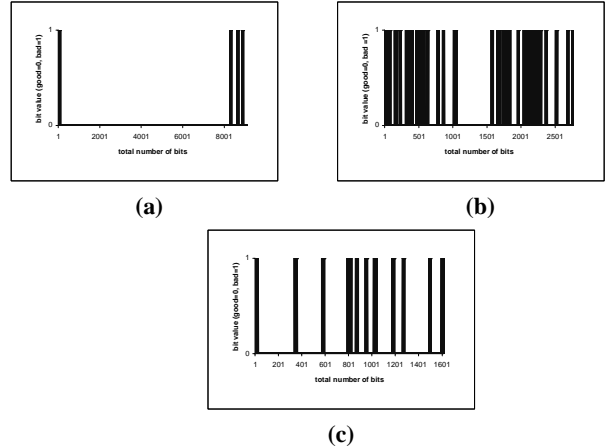


Figure 4. Trace segments with low and high error rates.

5.2.2 Hidden Markov Model Parameters

Rabiner [14] characterized the theoretical aspect of hidden Markov modeling in terms of solving three fundamental problems. Since our main concern is to use the HMM to artificially generate error traces, we just focus on one problem, and that is, the adjustment of model parameters so as to best account for the observed signal. Let us define the fundamental HMM elements:

1. N , the number of (hidden) states in the model. The error-conditions can be classified into three sub-states based on the error rate magnitude. For example, Figure 4(a) (error rate 0.0003) can be classified as low-error, Figure 4(b) (error rate 0.1) as very-high-error, and Figure 4(c) (error rate 0.055) as medium-error. Thus, we use $N = 3$, where individual states are denoted as $Z = \{Z_1, Z_2, Z_3\}$ and the (unknown hidden) state at time n is denoted by w_n .
2. M , the number of distinct observation symbols per state, i.e., the discrete alphabet size. In our case, the HMM operates on the energy values generated from the collected traces which we denote as $\Delta = \{v_1, v_2, \dots, v_M\}$, $0 \leq v_i \leq 1 \quad \forall i$. Thus each discrete observation gives a floating-point value which represents the energy in a trace window of 2000-bits.
3. The state transition probability distribution $P = \{P_{ij}\}$ where,

$$P_{ij} = P\{w_{n+1} = Z_j | w_n = Z_i\}, \quad 1 \leq i, j \leq 3$$

The transition probabilities are initialized randomly and the actual values result as a consequence of the HMM training. We employ the Expectation-Modification algorithm [13] to maximize the probability of occurrence of an observation. The transition probability matrix of the HMM is given in Table 4. It can be observed that the probability of staying in Z_1 is quite high. Any reasonable channel should have a high probability of successful (uncorrupted) bit transmissions. Thus, Z_1 should have a very low error rate. This observation will be substantiated by the next HMM parameter.

Table 4. Transition probabilities ($P = \{P_{ij}\}$) for the HMM

	$P\{w_{n+1} = Z_1\}$	$P\{w_{n+1} = Z_2\}$	$P\{w_{n+1} = Z_3\}$
$w_n = Z_1$	0.9722	0.0091	0.0187
$w_n = Z_2$	0.2052	0.5443	0.2506
$w_n = Z_3$	0.2379	0.0721	0.69

4. The observation symbol probability distribution in state j ,

$$B = \{b_j(k)\}, \text{ where}$$

$$b_j(k) = P\{v_k \text{ at } n | w_n = Z_j\}, \quad 1 \leq j \leq 3, \quad 1 \leq k \leq M$$

We initialized the observation distribution randomly. During training, we fit a Gaussian distribution as the observation symbol distribution in each state. Since the Gaussian distribution is completely characterized by the mean and standard deviation, for each state we obtained a set of these parameters as tabulated in Table 5.

Table 5. Symbol probability distribution for the HMM

	(mean, standard deviation)
Z_1	(0.0003, 8.7269×10^{-6})
Z_2	(0.0731, 0.0019)
Z_3	(0.0174, 2.7364×10^{-4})

Clearly, the mean error rate in Z_1 is very low. This observation, in conjunction with the transition probability of staying in Z_1 , outlines that most of the bit transmissions are successful. However, sometimes the channel (briefly) transits to Z_2 or Z_3 and then the error rate increases.

5. The initial state distribution ($\pi = \{\pi_i\}$) after the training is $\pi_1 = 0.3877$, $\pi_2 = 0.3763$, and $\pi_3 = 0.236$ where,

$$\pi_j = P\{w_1 = Z_j\}, \quad 1 \leq i \leq 3$$

Thus, the HMM can be represented as $\lambda = (P, B, \pi)$. Our goal is to maximize $P\{O|\lambda\}$, where O represents the observations generated from the collected traces. Hence, the training algorithm tries to maximize the probability that the collected traces were generated from the model.

After the training phase, the HMM was used to generate (simulated) observation sequences ($O = O_1 O_2 \dots O_l$, where $O_n \in \Delta$). In other words, each observation O_n gives an energy value based on the current state, w_n , and the symbol distribution in the current state, $b_j(k)$. Recall that the energy directly corresponds to the density of bad bits in a window. Thus, it represents the overall probability of receiving a corrupted bit. This probability was used to generate model-based error traces.

5.2.3 Performance of the Hidden Markov Model

For the HMM, the probability distributions derived from the secondary model-based traces are referred to as q_{HMM} . Table 6 enumerates the performance of the HMM. Comparing the \bar{I} column of Table 2 (row 3 and 4) with Table 6 outlines that the HMM shows clear improvement in the inter-arrival-rate performance, for instance, 40.33% as opposed to 58.72% for the hMM. However, the ENK for the burst-length random variable in the HMM case

(Table 6 column \bar{B}) is orders of magnitude greater than the respective ENK for the hMM traces (Table 2 column \bar{B} - row 3 and 4). Hence we conclude that, while the HMM improves the modeling of “good bursts”, (when compared to the hMM) the hidden Markov model can not approximate the “bad bursts” adequately.

Table 6. Performance of the HMM

	\bar{I}	\bar{B}
$ENK(p_{s_1} q_{HMM})$	0.403356	0.684794
$ENK(p_{s_2} q_{HMM})$	0.408725	0.730848

More importantly, the overall performance of HMM modeling for our error traces is not acceptable. The failure of HMM in our case may be attributed to the following. In other problem areas well-defined characteristics of the input data are available for pre-processing and training, e.g., cepstral features in speech. However, in this work, the corrupted traces regions exhibit highly random behaviour (see Figure 4). Therefore, and while ignoring simple energy-like features, it is not possible to generate subtle, yet distinct, training sequences. This results in inaccurate HMM parameters and, consequently, the model is unable to approximate the actual source.

Based on the results of the preceding bit error models, we now focus our attention solely to the traditional high-order full-state Markov chains. The following section evaluates the performance of the FSM for varying orders.

5.3 The Full-State Markov Chain

We generated Markov chains with varying memory lengths (k) such that each chain had 2^k states, where $k = 1, 2, \dots, 14$. The number of states in the FSM increases exponentially with the increase in memory length. In order to limit the complexity, previous (related) studies employed a manageable (relatively small) order of the Markov chain [4], [5]. We, however, present analysis for all models from order 1 up to 14.

5.3.1 Performance Evaluation of the Full-State Markov Chain

A sliding window was used to compute the FSM transition probability matrices. Due to the sliding window, the transition possibilities between the states were restricted. Let us consider a simple example to elaborate this idea. Let the current state, in a process with memory length of 4-bits, be $(0110)_2 = (6)_{10}$. As the window slides, the 0 in the most significant bit position will be dropped and a bit will be added to the least significant bit position, that is, the chain can transit to either $(1100)_2 = (12)_{10}$ or $(1101)_2 = (13)_{10}$. Thus from any current state, the process can jump to only two possible next states (current state inclusive). The transition probability matrix computed by this procedure will be sparse thus providing an upper bound on the model complexity.

In order to efficiently and accurately represent the transition probability data, and to reduce the complexity, we examined the transition probability matrices for bit-patterns that never occur in the collected traces. We refer to such bit-patterns as the *unused states*. In other words, such states result in all-zero columns of the transition probability matrix. An all-zero column implies that the probability of jumping to that state from any state is zero. The percentage unused states for each order are shown in Figure 5.

We observed that the number of unused states grew as the order of the Markov chain increased, for a 2^{14} state model approxi-

mately 70% of the states are unused (4848 used states). We lay special emphasis on this observation since the total number of states directly corresponds to the complexity of the model. Therefore, all following results will employ the “used states” only. Performance of each FSM model is given in Figure 6.

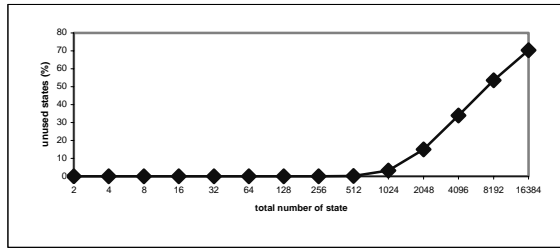


Figure 5. Unused states in the FSM.

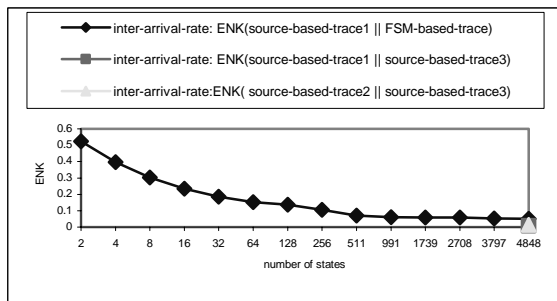
Clearly, the FSM performs remarkably for the burst-length random variable. Note that even smaller order chains perform adequately with the source coding overhead of less than 1% for all cases. However, the inter-arrival-rate random variable incurs profound overhead for smaller order chains. For example, the two-state chain renders an overhead of approximately 50% and is, therefore, not a viable option. As we move to higher order chains, the overhead decreases and drops to a reasonable level at and after the 511-state model. Here, due to data over-fitting considerations, we assume that any overhead less than 10% is acceptable. The best performance is rendered by the highest order (4848-state chain) which incurs an overhead of 5.163% and 0.1531% for the inter-arrival-rate and burst-length random variables respectively. Thus we conclude that all FSM chains of order 9 and above render appropriate models for the bit error process.

6. CONCLUSIONS & FUTURE WORK

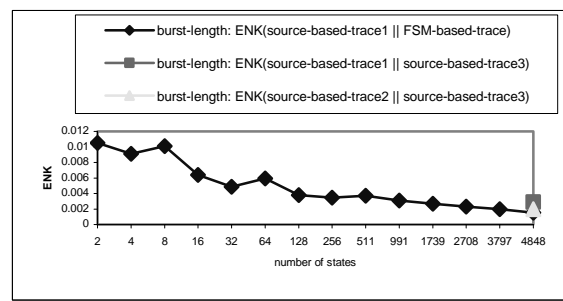
In this paper we analyzed Markov-based modeling techniques for bit error and packet loss patterns observed over 802.11b networks. We demonstrated that the packet loss process can be adequately modeled using a two-state Markov chain. We also showed that the hierarchical and hidden Markov models can not represent the 802.11b MAC-to-MAC bit error process accurately. On the contrary, full-state chains of order 9 and above render very good models for the bit error process. However, due to its complexity, this modeling paradigm can not be adapted in real-time. In order to render real-time channel characterization, we are currently investigating approaches which can constrain the exponentially increasing (with respect to the order) FSM complexity.

7. REFERENCES

- [1] L. Larzon, M. Degermark, and S. Pink, “Efficient Use of Wireless Bandwidth for Multimedia Applications,” *IEEE MoMUC*, November 1999.
- [2] S. Khayam, S. Karande, M. Krappel, and H. Radha, “Cross-Layer Protocol Design for Realtime Multimedia Applications over 802.11b Networks,” *IEEE ICME*, July 2003.
- [3] Haitao Zheng, “Optimizing Wireless Multimedia Transmissions through Cross Layer Design,” *IEEE ICME*, July 2003.
- [4] A. Konrad, B. Y. Zhao, A. D. Joseph, and R. Ludwig, “A Markov-based Channel Model Algorithm for Wireless Networks,” *ACM MSWiM*, July 2001.
- [5] P. Ji, B. Liu, D. Towsley, and J. Kurose, “Modeling Frame-level Errors in GSM Wireless Channels,” *IEEE Globecom*, November 2002.
- [6] S. Khayam, S. Karande, H. Radha, and D. Loguinov, “Performance Analysis and Modeling of Errors and Losses over 802.11b LANs for High-Bitrate Real-time Multimedia,” *Signal Processing: Image Communication*, vol. 18, no.7, pp. 575–595, August 2003.
- [7] S. Karande, S. Khayam, M. Krappel, and H. Radha, “Analysis and Modeling of Errors at the 802.11b Link Layer,” *IEEE ICME*, July 2003.
- [8] P. Brockwell and R. Davis, “Introduction to Time Series and Forecasting,” Springer: Verlag, 1996.
- [9] T. Cover and J. Thomas, “Elements of Information Theory,” Wiley: New York, 1991.
- [10] C. Anton-Haro, J. A. Fonollosa, and J. R. Fonollosa, “Blind Channel Estimation and Data Detection Using Hidden Markov Models,” *IEEE Trans. on Signal Processing*, vol. 45, no.1, pp. 241–246, January 1997.
- [11] C. Anton-Haro, J. A. Fonollosa, C. Fauli, and J. R. Fonollosa, “On the Inclusion of Channel’s Time Dependence in a Hidden Markov Model for Blind Channel Estimation,” *IEEE Trans. on Vehicular Technology*, vol. 50, no.3, pp. 867–873, May 2001.
- [12] W. Turin and R. van Nobelen, “Hidden Markov Modeling of Flat Fading Channels,” *IEEE JSAC*, vol. 16, no.7, pp. 1809–1817, December 1998.
- [13] L. E. Baum, “An inequality and associated maximization technique in statistical estimation of probabilistic functions of Markov processes,” *Inequalities*, vol. 3, no. 1, pp. 1–8, 1972.
- [14] L. R. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.



(a)



(b)

Figure 6. Performance of k -order FSMs, where $k = 1, 2, \dots, 14$, for the inter-arrival-rate and burst-length random variables.