

End-to-End Inference of Loss Nature in a Hybrid Wired/Wireless Environment

JUN LIU IBRAHIM MATTA MARK CROVELLA

Computer Science Department
Boston University
Boston, MA 02215

{junliu, matta, crovella}@cs.bu.edu

Technical Report BUCS TR-2002-008

March 14, 2002

Abstract— End-to-End differentiation between wireless and congestion loss can equip TCP control so it operates effectively in a hybrid wired/wireless environment. Our approach integrates two techniques: packet loss pairs (PLP) and Hidden Markov Modeling (HMM). A packet loss pair is formed by two back-to-back packets, where one packet is lost while the second packet is successfully received. The purpose is for the second packet to carry the state of the network path, namely the round trip time (RTT), at the time the other packet is lost. Under realistic conditions, PLP provides strong differentiation between congestion and wireless type of loss based on distinguishable RTT distributions. An HMM is then trained so observed RTTs can be mapped to model states that represent either congestion loss or wireless loss. Extensive simulations confirm the accuracy of our HMM-based technique in classifying the cause of a packet loss. We also show the superiority of our technique over the Vegas predictor, which was recently found to perform best and which exemplifies other existing loss labeling techniques.

Keywords— Wireless Loss, Loss Differentiation, Measurements, Hidden Markov Models, TCP Control, Simulation.

I. INTRODUCTION

In recent years, as wireless communication has become common in the Internet, the question of TCP behavior over wireless channels has become important [9]. In such a setting, unmodified TCP does not differentiate losses due to queue overflow from those due to a lossy wireless medium. As a consequence, undifferentiated packet losses may no longer be valid signals as they always indicate congestion. In this situation, a TCP connection over a wireless medium may over-police itself, and not be able to achieve its fair share of the channel.

Research on improving performance of TCP over hybrid wired/wireless paths has focused on differentiating packet losses using information readily available to TCP: congestion window size, inter-arrival time between ACK packets, and changes in round-trip time (RTT) [6], [5]. However, correct classification based on these metrics has been found to be difficult [6]. It appears that there is lack of correlation between the nature of losses and these observable measures (RTT, congestion window size, and inter-arrival of ACKs).

In this paper, we propose a new end-to-end method to classify the nature of packet loss in a hybrid wired/wireless environment. Specifically, we monitor the most recent RTT at the time each loss occurs to determine its most likely nature. End-to-end loss inference methods have the advantage of not requiring any support from the network.

In a wired network, packet losses are mainly due to congestion, thus the network status can be characterized as either *congested* or *not congested*. However, when the connection's path involves a wireless link, the network can be characterized as being in *congested* state, or in *wireless channel fading* state, or in *neither congested nor wireless channel fading* state. Clearly, we can not model such a mixed wired/wireless communication channel by packet loss events alone, because the *congested* state and the *wireless channel fading* state become indistinguishable. (From now on, we call packet losses due to congestion *congestion* losses and the losses due to wireless channel fading *wireless* losses.) In order to differentiate congestion losses from wireless losses, it may help to consider network delays at the time of packet losses. Intuitively, congestion losses are associated with buffer overflows, so that packet delays when such losses occur contain the full queuing time of at least one buffer along the path. On the other hand, wireless losses are not associated with buffer overflow in general, so delays at such losses may be more variable compared to the delays at congestion losses.

We can formalize this idea using Bayesian analysis. Let $T = \{\text{Congestion (C), Wireless (W)}\}$ be the space of loss types. Let L be a random variable ($L \in T$) signifying the type of a packet loss. Let R be a random variable ($R \in \mathbb{R}^+$) signifying the delay associated with each packet loss. The quality of differentiation at a particular delay r can be formalized as

$P[L = l | R = r]$. From a Bayesian standpoint,

$$P[L = l | R = r] = \frac{P[L = l] \cdot P[R = r | L = l]}{\sum_{l' \in \mathcal{T}} P[R = r | L = l'] \cdot P[L = l']} \quad (\text{I.A})$$

This exposes the opportunity for making use of prior knowledge in performing the estimation. In particular, we can make reasonable prior estimates $P[R = r | L = l]$ for every l : for congestion losses, we expect this distribution to be tightly centered at the delay corresponding to the maximum queueing delay (at the bottleneck), and for wireless losses, we expect this distribution to be approximately the one corresponding to the unconditioned queue occupancy distribution. And, we note that the distribution forming the right hand side denominator can be directly estimated as the observed delay distribution for all losses. This leaves only the $P[L = l]$ priors unknown.

This formulation makes clear that the leverage that using delay provides on this problem is maximized when the component distributions $P[R = r | L = l]$ are sufficiently distinct — that is, when the “typical” queue occupancy is different from the congested queue occupancy.

Using this approach, we propose a differentiation technique for packet loss types by using delays at packet losses. Our technique is based on 1) *loss pairs* [13], which measure the network delay at the time that a loss happens, and 2) Hidden Markov Model (HMM) [15], a powerful modeling tool. Loss pairs is a tool to convey network delay status at losses to the end nodes of a network path. In a wired network, the status of the network can be modeled by a HMM with a very small number of states [17].

Our results show that our technique is effective in most network configurations. It exhibits flexibility in providing applications of different types the ability to choose the classification on losses according to their needs. It also shows its superiority in quality of classification on losses over the best classifier on losses reported in [6], the Vegas predictor.

In the rest of this paper, we describe related work in Section II. We present in Section III the basic introduction on HMM and Loss pairs technique and our labeling framework built on top of these two techniques. The actual labeling technique is described in Section IV. The evaluation of our labeling technique is described in Section V. Section VI concludes the paper.

II. RELATED WORK

Many studies have shown that TCP goodput can be improved if the cause of packet loss is identified [3]. By attributing a

packet loss to wireless transmission errors, the TCP source can refrain from taking unnecessary “congestion” control measures. One set of solutions (I-TCP [2], Snoop [4], WTCP [16], Syndrome [10]) require support from the base station located at the interface between the wired infrastructure and the wireless access infrastructure. The base station can buffer data packets (or just their sequence numbers) as they are received from a wired source. This information can then be used by the base station to recognize if any of those packets are later lost over the wireless link on their way to the destination. These solutions incur the cost of implementation at the base station, or that of explicit feedback messages to inform the source of the cause of loss, and some violate the end-to-end semantics of TCP.

In this paper, we are primarily interested in end-to-end solutions, i.e. those which do not require any support from the network. Proposed end-to-end solutions differ mainly in the measure(s) they use to infer the cause of loss. These measures may be estimated at the sender without any support from the receiver (e.g. round-trip delay), or may require support from the receiver (e.g. one-way delay or delay variance). In the scheme of Biaz and Vaidya [5], the receiving host measures the interarrival times of packets. Assuming the last hop is wireless and the bottleneck, if the time between received packets is close to the minimum, then a lost packet in-between is assumed to have been lost due to wireless errors and not congestion. The receiving host in the Spike scheme [20] measures one-way delays, and switches to congested (wireless) state as the delay exceeds (drops below) a certain threshold. The ZigZag scheme [19] extends Spike to include the mean and deviation of measured one-way delays as well as number of losses in computing the delay thresholds. Intuitively, the higher the number of losses the higher the threshold beyond which congestion is assumed, i.e. the cause of the loss being wireless errors becomes more likely.

Other end-to-end solutions include congestion-avoidance-based loss predictors. Vegas [6] and NCPLD (Non-Congestion Packet Loss Detection) [18] predictors compare the measured RTT with the lowest RTT (or that at the knee of the goodput-load curve). If the former is close to the latter, then the cause of a packet loss is assumed to be wireless errors. The same idea is used in TCP-Probing [21], where the detection of a packet loss triggers the sending of probes to measure the current RTT.

Other end-to-end solutions do not explicitly infer the cause of a packet loss, however they are shown to improve TCP per-

formance over wireless links. TCP Westwood measures goodput (or reception rate) and uses that rate to set the congestion window whenever a packet is detected lost. This is in contrast to regular TCP implementations where the window size is arbitrarily cut in half whenever a loss is detected by duplicate acknowledgments. TCP Santa-Cruz [14] is similar in spirit to TCP Vegas [8], except that it uses one-way measurements of (queueing) delay rather than RTT. TCP-Real [23] combines ideas from Westwood and Santa-Cruz, by measuring the rate at the receiver. Similar to TCP-Real, the receiver in LIMD/H [12] communicates its measured loss rate to the sender, which uses it to compute the goodput. If the current goodput is below a certain band around the mean, then the cause of a packet loss is assumed to be congestion, otherwise the cause of loss is attributed to wireless errors.

Our approach generalizes the two-state view of the network (congestion vs. wireless) taken by the above schemes to multiple states, where each state can encapsulate a different level of wireless errors or congestion. Furthermore, we do not make any assumptions about the location of the bottleneck or wireless links. We use RTT measurements to infer the state of the network at the time of a packet loss. Each state in our model is characterized by the mean and standard deviation of the associated RTT distribution. Our approach is intended for loss inference, and we maintain a clean separation between this inference process and the control that may make use of it. Our approach is based on training a Hidden Markov Model (HMM), which presumes that the set of observations/measurements are generated by a Markovian process. An excellent detailed description of HMMs is given in [15]. Salamatian and Vaton [17] used HMM to model network channel losses and showed that making inference about the state of the channel (lossy or not) can be reasonably accurately predicted. In this paper, we use HMM to predict the *cause* of a lost packet. Our HMM is constructed using only RTTs of interest; those observed at about the same time the packet in question is lost. This RTT filtering is done using the technique of loss pairs [13]. We compare our HMM-based loss predictor to the Vegas loss predictor. The Vegas loss predictor was shown to outperform others [6], and it is representative of the design philosophy of previously proposed predictors. Besides the natural association of multiple levels of wireless errors or congestion to our HMM states, one can consider multiple types of observations. For example, interarrival

time between losses could be considered in addition to RTTs or one-way delays. However we only consider in this paper RTT measurements (of loss pairs) to train our HMM.

III. BACKGROUND AND FRAMEWORK

In this work, we use an HMM to derive a statistical model of the signal of RTTs of loss pairs. Based on the derived model, we propose a technique for differentiating losses into congestion losses and wireless losses. In this section, we give brief introductions to loss pairs and HMMs.

A. Loss pairs

A loss pair is a pair of packets sent back-to-back by the sender such that exactly one of them is lost on the way. Since these two packets travel together close enough to each other up to the point where one of them is lost, the packet that is not lost carries the delay status at current packet loss back to the sender. This pair of packets can be a pair of probing packets in the active probing scheme, or a pair of data packets sent by a TCP agent, like TCP Reno, in a passive measurement scheme.

The use of loss pairs to carry network delays generally relies on three assumptions [13]. Since loss pairs was originally proposed in the wired network environment, we restate these three assumptions for the hybrid wired/wireless environment as follows:

- There is only one most congested point, *i.e.*, queue, called bottleneck, and the number of packet losses and the delays at the bottleneck are significant compared to the ones at other network elements along a path. This makes it possible to ascribe congestion losses and delays seen at the end-point to the internal bottleneck.
- The round-trip path and the location of the bottleneck do not change during measurement. This ensures that the non-dropped packet in a loss pair is likely to see similar queue occupancies along the path at packet losses.
- In order to relate delay in the queue to queue occupancy, we assume that the packet scheduling at the queues along the path is FCFS.

B. Hidden Markov Models and EM Algorithm

B.1 HMMs

HMMs have become a powerful modeling tool for two main reasons: first, an HMM has rich mathematical structure and thus can form the theoretical basis for a wide range of applications;

second, it works very well when it is applied appropriately [15]. An HMM is a statistical signal model which can provide the basis for a theoretical description of a signal processing system. A signal is normally expressed as a time series $\{o_t : t = 1, 2, \dots\}$. The generation of this time series can be imagined as from a Markov chain only taking actions at discrete time points. When it takes action, it generates an observation vector based on the probability distribution associated with the current state. The signal can be either in discrete or in continuous form. In this paper, we only focus on introducing an HMM with Gaussian continuous observations which is characterized by the following elements [15]:

- N , the number of states in the model. We denote $\{S_1, S_2, \dots, S_N\}$ as the state space.
- When the model is in state S_j at time t , the probability density function of the current observation o_t can be described by a Gaussian density of the form

$$b_j(o_t) = \mathcal{N}[o_t; \mu_j, \sigma_j^2]$$

where \mathcal{N} is a Gaussian density function with mean μ_j and covariance σ_j^2 .

- The state transition probability distribution $A = \{a_{ij}\}$ where $a_{ij} = P[s_{t+1} = S_j | s_t = S_i]$ ($1 \leq i, j \leq N$).
- The initial state distribution $\pi = \{\pi_1, \pi_2, \dots, \pi_N\}$ where $\pi_i = P[s_1 = S_i]$ ($1 \leq i \leq N$).

B.2 EM Algorithm

A Hidden Markov Model is usually trained through multiple iterations of the EM algorithm. The EM algorithm is essentially an optimization on the quality parameter of a model based on a fixed series of observations. As the EM iterations proceed, the newly derived HMM becomes more likely than the previous ones in having generated the series of observations. Excellent explanations on how HMM can be estimated by the EM algorithm have been given in [15], [17].

C. State Backtracking

The modeling goal is to explore the state sequence the model follows for a sequence of given observations. If $\hat{\theta}$ is the estimated model after multiple iterations of the EM algorithm, the state backtracking algorithm finds the state sequence $\hat{s} = \{\hat{s}_1, \hat{s}_2, \dots, \hat{s}_T\}$ for a given sequence of observations $\mathbf{o} = \{o_1, o_2, \dots, o_T\}$ [15], [17]. The Viterbi algorithm is a dynamic

programming algorithm which is an efficient state sequence inference technique. The Viterbi algorithm produces a state sequence \hat{s} such that the *a posteriori* log-likelihood $L(\hat{s} | \mathbf{o}; \hat{\theta})$ gets maximized, so that \hat{s} is the most likely sequence of states followed by the model.

IV. LABELING TECHNIQUE

In this section, we further describe how to make use of HMMs described in Section III to establish a model in practice, and how the loss nature labeling technique is constructed based on the estimated model. It has been shown in [17] that $N = 4$ is sufficient to characterize a network communication channel by observations of packet loss events. We use 4 states here. We also set the number of observations used to train a model to be 10000 as in [17].

A. Motivation

We use a 4-state model trained with 10000 loss pair RTT measurements. Our measurements are collected using an ns-2 simulation of a hybrid wired/wireless network. For each loss pair, we also record its nature (congestion or wireless) to allow comparison with our classifier. In this section we concentrate on explaining the overall approach, so discussion of the specific simulation parameters is deferred to the next section.

To motivate our approach, consider the distribution of RTTs shown in the top three plots of Figure 1. These plots show loss pair RTTs for all losses, wireless losses, and congestion losses, respectively. Note that the RTTs of congestion loss pairs are very compactly distributed around the RTT value corresponding to buffer overflow along the network path used in the simulation. On the contrary, the RTTs of wireless loss pairs are wide spread.

We trained our four-state HMM on the set of all loss pairs; the Gaussian models for each state of our trained HMM are shown in order (1 through 4) below the histograms in Figure 1. We note that the distribution corresponding to state 1 looks most similar to the distribution of RTTs of congestion loss pairs. We can conjecture that state 1 corresponds to congestion losses more than any other state. To verify this conjecture, we perform state estimation (using the Viterbi algorithm) to determine the most likely state the model is in for each observation. Then, using our knowledge of the true nature of each loss, we compute the number of times the model is in each state for losses of each type. We plot these numbers in Figure 2. This figure confirms that in fact, state 1 is the most likely state corresponding to con-

gestion losses.

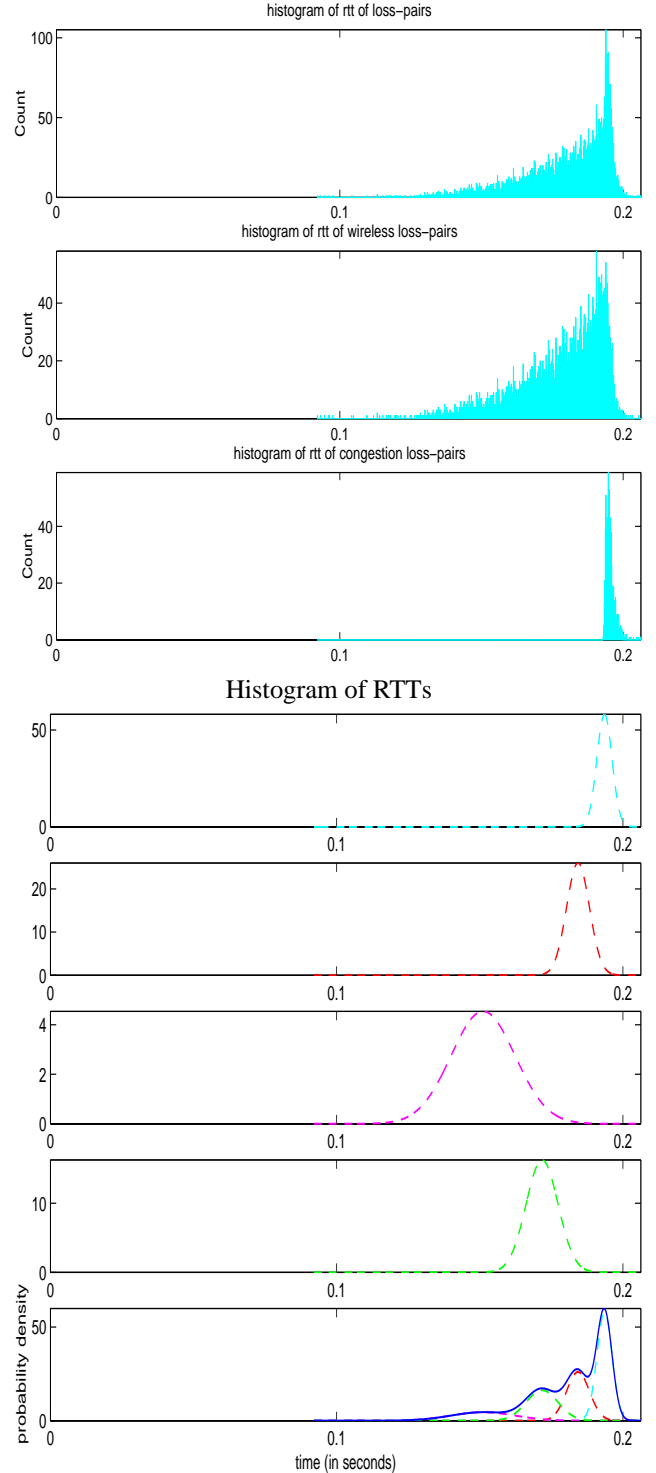
In Figure 1, the distribution of RTTs of congestion loss pairs is quite different from that of wireless loss pairs. This difference is clearly due to the different ways in which losses occur, with congestion losses occurring due to buffer overflow. Thus, any model state corresponding to congestion losses should show a fairly compact distribution of RTTs, with high average value; *i.e.*, its Gaussian distribution should show relatively smaller variance and relatively larger mean. On the other hand, the RTTs of wireless losses may typically be smaller than the those of congestion losses because wireless loss pairs are unlikely to occur during buffer overflow in general. These considerations suggest that the mean, standard deviation, and coefficient of variation of each state’s Gaussian component would appear to be helpful in classifying states into congestion or wireless.

As just described, each observation of loss pair at a particular time can be associated with a most likely state of the HMM. Thus to use the trained HMM, we need to determine a “labeling” for each state, *i.e.*, an assignment of a loss type to each state of the HMM. When $N = 4$, there are 2^4 possible ways to make this assignment. Of course, different assignments are not equal in the quality of the classification they produce.

To measure the quality of a classification method (*i.e.*, an assignment scheme) we propose to use $P[t|t']$ as a set of metrics for evaluating the quality of labeling loss pairs. $P[t|t']$ is the probability of a loss pair being classified as of loss type t given that it is in fact due to loss type t' , for $t, t' \in T$. These metrics (which previous work have not clearly exposed) make clear the fundamental tradeoffs inherent in developing a method of loss classification.

More specifically let the event “being labeled as a wireless loss” be denoted as W and “being labeled as a congestion loss” be denoted C ; and let the event “the loss is actually due to congestion” be denoted \mathcal{C} and “the loss is actually due to wireless channel fading” be denoted \mathcal{W} . Among these metrics, $P[W|\mathcal{W}]$ is appropriate for evaluating the accuracy of labeling wireless loss pairs, and $P[C|\mathcal{C}]$ is appropriate for congestion loss pairs. To give a feeling for how these two metrics can vary together, in Figure 3, we plot $P[C|\mathcal{C}]$ and $P[W|\mathcal{W}]$ under all possible type assignments to states for three network configurations. These figures show that it is hard, in general, to simultaneously have $P[C|\mathcal{C}] = 1$ and $P[W|\mathcal{W}] = 1$.

The ability to separately consider $P[C|\mathcal{C}]$ and $P[W|\mathcal{W}]$ is a



Gaussian component in each state and their composition.

	S_1	S_2	S_3	S_4
Mean μ	0.19	0.19	0.16	0.18
Std. Dev. σ	0.0027	0.0043	0.012	0.0063
Indicator μ/σ	71	45	13	28

Parameters of Gaussian Component in each state of the model

Fig. 1. Distribution of RTTs of loss pairs by type (all, wireless, congestion) and the Gaussian components for each state (1 through 4) of the trained HMM.

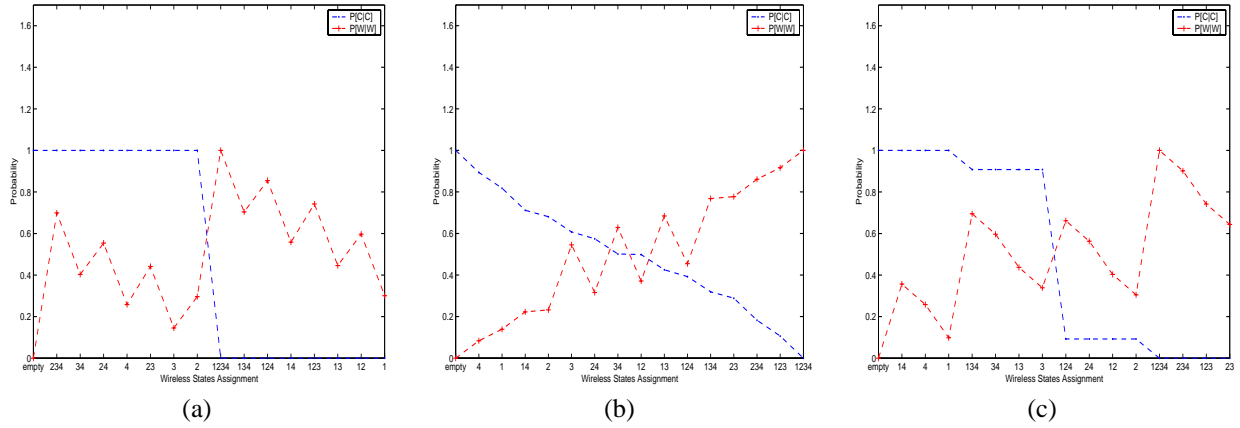


Fig. 3. Accuracy of classification on losses by type under all possible wireless state(s) assignments.

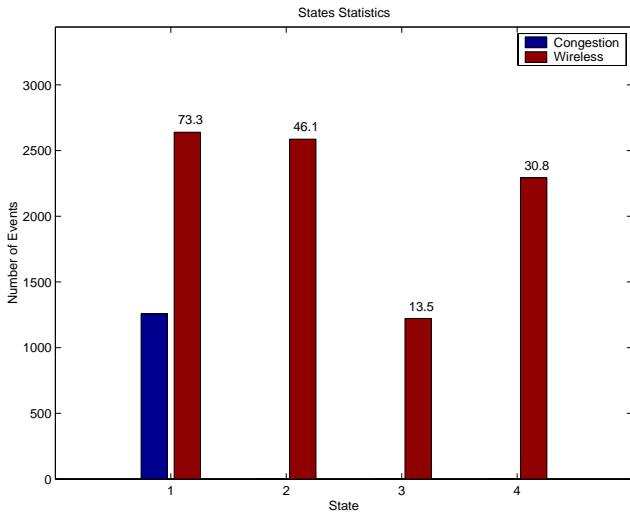


Fig. 2. Number of times a state is visited by loss pairs. In each state, the left bar corresponds to losses due to congestion and right bar corresponds to losses due to wireless channel fading.

strength of our approach. In general, different applications may wish to emphasize accuracy in one of these metrics at the expense of the other metric. These different emphases can be thought of as leading to different points in the tradeoff graphs shown in Figure 3. It is encouraging that Figure 3 does show that for high $P[C|C]$, it is often possible to obtain reasonably high $P[W|W]$ as well. Referring to Figure 2, if we assign state 1 as Congestion, and the remaining three states as Wireless, then we achieve $P[C|C] = 1$ as well as a high $P[W|W]$.

The state assignment we use in the rest of this paper is one that emphasizes high $P[C|C]$. That is, we seek to find the state assignment with highest $P[W|W]$ given that $P[C|C]$ is near 1. The choice reflects the importance of correctly responding to congestion for a TCP connection to share a channel fairly. The process of automatically selecting such a state assignment is

necessarily heuristic. We discuss our heuristic here and evaluate it in the next section.

B. Labeling Losses by Type

The parameters contained in the estimated model are inputs to the labeling algorithm. By studying the relationship between the model parameters and the accuracy metrics, we develop the following heuristic labeling technique:

- Given parameters:
 - Initial state distribution π , state transition matrix $A = [a_{ij}]$;
 - Mean μ , std. dev. σ , congestion indicator μ/σ in each state.
- Step 1 (Bipartite partition among states of the model)
 - (i) Compute the flow matrix $F = [f_{ij}]$ where

$$f_{ij} = \pi_i \cdot a_{ij} \quad (1 \leq i, j \leq N)$$

- (ii) Partition the states into two disjoint sets with the restriction that each set should contain at least one state, then compute the flow volume between these two sets. There are $2^{N-1} - 1$ such bipartite partitions. Suppose that $\{P_1, P_2\}$ is one of the bipartite partition, then the flow between P_1 and P_2 can be expressed as

$$f(P_1, P_2) = \sum_{S_i \in P_1, S_j \in P_2} (f_{ij} + f_{ji})$$

- (iii) Sort the resulting $2^{N-1} - 1$ flows in order, and select the bipartite partition associated with the median value of inter-partition flow.

- Step 2 (Label assignment to each partition)
 - (i) Sort $[\mu_i], [\sigma_i]$ and $[\mu_i/\sigma_i]$ and determine their ranks as follows:
 - $[\mu_i]$ is in increasing order, such that $\text{Rank}(\mu_i) \geq \text{Rank}(\mu_j)$ iff $\mu_i \geq \mu_j$;

- $[\sigma_i]$ is in decreasing order, such that $\text{Rank}(\sigma_i) \geq \text{Rank}(\sigma_j)$ iff $\sigma_i \leq \sigma_j$;
 - $[\mu_i/\sigma_i]$ is increasing order, such that $\text{Rank}(\mu_i/\sigma_i) \geq \text{Rank}(\mu_j/\sigma_j)$ iff $\mu_i/\sigma_i \geq \mu_j/\sigma_j$.
- (ii) Assign a weight w to each state as follows:

$$w(S_i) = \alpha_1 \text{Rank}(\mu_i) + \alpha_2 \text{Rank}(\sigma_i) + \text{Rank}(\mu_i/\sigma_i).$$

Experimentation suggests the use of $\alpha_1 = N^2$ and $\alpha_2 = N$.

(iii) The states in the partition with the largest weight are labeled as *Congestion* (C), and the states in the other partition are labeled as *Wireless* (W).

We note that this method is certainly subject to refinement for different application settings. The motivation for this method is:

- The bipartite partition with the median flow appears to maximize $P[C|C]$ while keeping $P[W|W]$ high, and
- the weight assignment for each state constructs a metric as a function of its associated mean and standard deviation, with the property that the state with the largest metric is most likely to be a congestion state; using rank instead of actual value makes the metric more robust.

V. EVALUATION

To evaluate the effectiveness of labeling losses by type, we conducted extensive experiments using the `ns-2` network simulator [11]. The network topology used in the simulations is shown in Figure 4. Nodes n_1, n_2, n_3, n_4 represent backbone routers. The link between nodes n_2 and n_3 is set to be the bottleneck, which means that a significant number of packet losses due to congestion happen at this link. We also vary the location of the wireless medium to be at any of the links connecting neighboring routers. A set of TCP communication agents are attached to each router. Among these TCP agents, we treat the ones that use the longest path on the backbone as the observable TCP agents (connections) in the evaluation, and all the others are treated as TCP connections that create cross traffic. Furthermore, we set all the cross traffic to only go across one network link on the backbone. This topology setting provides us the flexibility of varying the location and loss rate of the wireless medium and the ability of varying the strength of both observable traffic and cross traffic. In Table I, we list two sets of network configurations we used in the simulation. These configurations represent a “high” speed network (setting I), and a “slow” speed network (setting II). Unless specified otherwise,

we use 500 TCP observable connections in network configuration I to create a strong traffic load scenario, and 200 TCP observable connections in network configuration II for a moderate traffic load scenario. In studying the impact of cross traffic, we use 30 TCP connections in each group crossing one link. Each TCP connection follows an ON/OFF activity pattern with heavy-tailed ON and OFF durations. The OFF duration follows a Pareto distribution with shape parameter of 1.5 and scale parameter of 10. The ON duration follows a Pareto distribution with shape parameter of 1.5 and scale parameter of 1000.

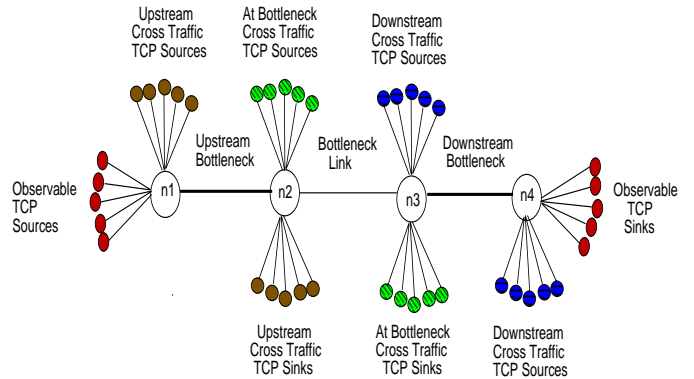


Fig. 4. Topology of simulated network.

Setting	Upstream Bottleneck	At Bottleneck	Downstream Bottleneck	Access link
Network Configuration I				
Bandwidth	1.5Mbps	1.3Mbps	1.5Mbps	10Mbps
Buffer Size	10KB	8KB	10KB	∞
Latency	4ms	4ms	4ms	2ms
Network Configuration II				
Bandwidth	1Mbps	0.5Mbps	1Mbps	10Mbps
Buffer Size	10KB	10KB	10KB	∞
Latency	4ms	4ms	4ms	2ms

TABLE I
NETWORK SETTINGS USED IN THE `ns-2` SIMULATION

We make use of both the simple memoryless uniform error model and the two-state Markovian-Gilbert model to simulate wireless errors. When we focus our discussion on network contributing factors other than the wireless link error model, we consider the simple memoryless error model as the wireless link error model as in earlier studies (e.g. [6], [5]). When we use a two-state wireless link error model, we use the two-state Markovian-Gilbert model in [1], which better characterizes the bursty error nature due to wireless channel fading [22], [24].

Our evaluation was conducted for two purposes: first, we

want to study the effect of different factors on the effectiveness of our labeling technique; second, we compare our technique to the Vegas predictor which is shown to outperform others in terms of capability to accurately classify losses [6].

A. Factors Contributing to Labeling Accuracy

The labeling accuracy, *i.e.*, probability $P[L = l|R = r]$, formalized by formula (I.A) depends on two factors: $P[L = l]$, the relative frequency of the losses of each type, and the prior probability $P[R = r|L = l]$, the distinctiveness of RTTs of each type. Some other possible contributing factors include: bottleneck utilization, bandwidth-delay product of the bottleneck link, location of the wireless link, strength of observable traffic and cross traffic, and the wireless link error characteristics. In the rest of this section, we evaluate our technique with respect to these factors.

To evaluate the network-related factors contributing to the labeling accuracy, we examine how these factors impact the prior density of different loss types. These factors affect the prior density by affecting the variation in bottleneck queue occupancy. Highly variable queue occupancy distributions will allow the two loss types to be distinguishable more easily. In Figure 5, we show two sets of prior density with different features and the corresponding posterior probabilities to support the fact that distinctive prior density leads to good posterior probabilities. The two prior densities are taken from the empirical distributions of RTTs, *i.e.*, like the ones in Figure 1, and the posterior probabilities are computed by formula (I.A). By further examining the labeling accuracy in both cases as shown in Table II, we conclude that distinctive prior density of RTTs of both loss types leads to better labeling quality than otherwise.

Case	Overall Loss Rate	$P[W]$	$P[W W]$	$P[C C]$
Fig. 5 (a)	20%	0.344	0.327	0.726
Fig. 5 (b)	18.1%	0.797	0.705	1

TABLE II

LABELING ACCURACY ON LOSSES BY TYPE UNDER DISTINCTIVE OR INDISTINCTIVE PRIOR DENSITIES OF RTTS

These network related contributing factors generally have a joint impact on labeling accuracy, and thus we discuss such joint impact by evaluating labeling accuracy under different utilization levels of the bottleneck link. To create different levels of bottleneck link utilization, we adopt network setting I as shown in Table I and we set the wireless medium between nodes n_2

and n_3 with a 2% loss rate and vary the number of observable TCP connections from 50 to 250 in increments of 50. Under different levels of utilization, we list the quality of labeling by our classification technique in Table III. We note that the labeling quality deteriorates with increasing utilization of the bottleneck link. This shows that wireless losses and congestion losses are more difficult to differentiate by RTTs under high load because the bottleneck queue occupancy shows less variation under very high utilization ($> 90\%$).

Utilization ρ	Overall Loss Rate	$P[C]$	$P[C C]$	$P[W W]$
0.514	4%	0.5	1	0.536
0.921	4%	0.5	1	0.33
0.995	8.7%	0.23	0.706	0.171

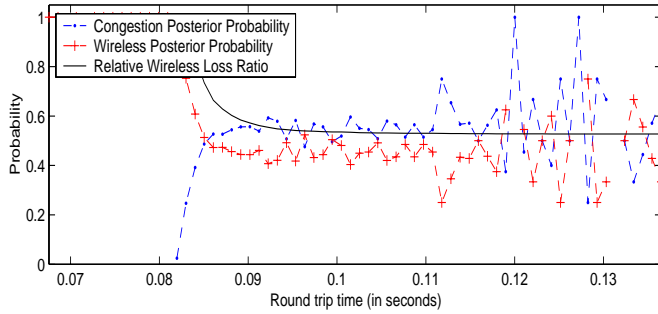
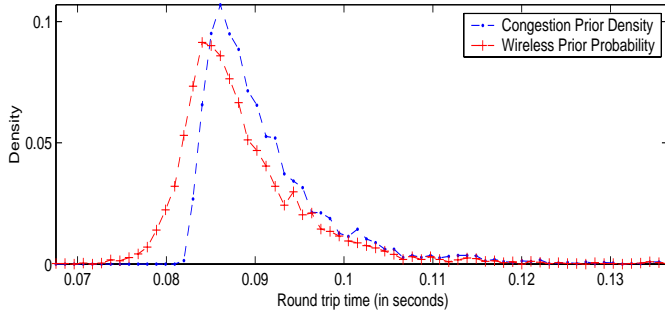
TABLE III

Labeling accuracy under different utilization levels of the bottleneck link.

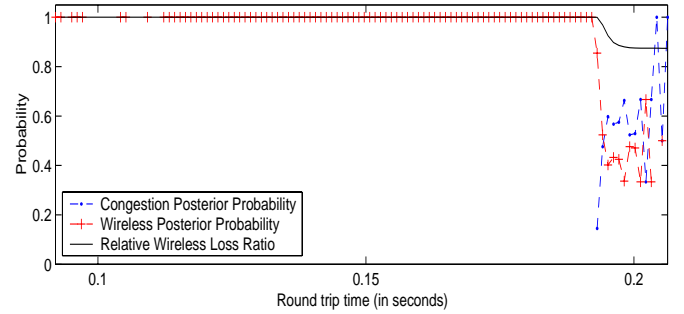
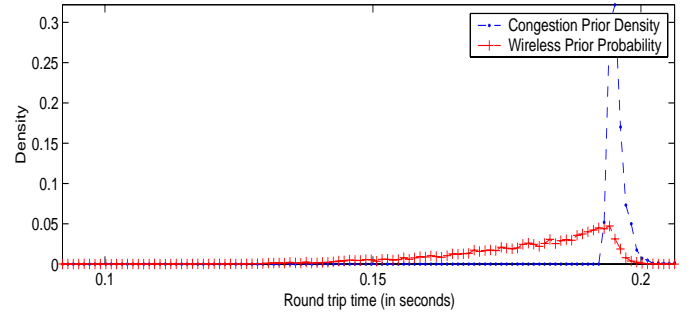
We can also evaluate the marginal impact of other contributing factors on loss classification accuracy. We first look at the impact of different settings of bandwidth and buffer size of the bottleneck link by looking at the bandwidth-delay product of the bottleneck link as shown in Table IV. We obtain different values of bandwidth-delay product of the bottleneck link by varying bandwidth from $0.5Mbps$ to $2.5Mbps$ in increments of $0.5Mbps$. In each case, the buffer size at the bottleneck link is set equal to the bandwidth-delay product. We can see in Table IV that the classification quality for both loss types is generally improved as the value of bandwidth-delay product becomes larger. The reason is that congestion losses can be more easily differentiated from wireless losses by their RTTs under larger communication pipes.

In general, there is no noticeable impact on classification accuracy by the location and loss rate of the wireless medium. This fact is shown in Table V by varying the location of the wireless medium with a fixed 5% wireless loss rate under both heavy and mild traffic. The same conclusion was reached in another set of simulations under a fixed 10% wireless loss rate (not shown here).

We tested the labeling quality of our technique under different wireless link loss rate as shown in Table VI. We can clearly see that the labeling quality is generally better under higher wireless link loss rates. The reason is due to the fact that there are only finite number of TCP sources used in the simulation. The higher



(a) Under Heavy Traffic, Network Configuration I



(b) Under Moderate Traffic, Network Configuration II

Fig. 5. Distribution of RTTs of loss pairs for each type of loss

BW×D factor	Overall Loss Rate	$P[C]$	$P[C C]$	$P[W W]$
1	10.5%	0.762	0.653	0.22
2	7%	0.714	1	0.331
3	7%	0.714	0.989	0.704
4	4%	0.5	1	0.456
5	4%	0.5	1	0.679

TABLE IV

Labeling accuracy under different values of bandwidth-delay product of the bottleneck link. Network configuration II is used in these simulations. The actual bandwidth-delay product at each row equals the product of the BW×D factor and the base value 16Kb.

the wireless link loss rate is, the higher the probability that the congestion window size of a TCP source is small due to larger number of packet losses. Thus the network pipe is unlikely to be filled to capacity, such that it's easy to differentiate the two types of losses.

Cross traffic is another factor that may affect the classification accuracy by affecting queue occupancies. Cross traffic can be roughly classified into static one and bursty one. Table VII shows the impact of strong cross traffic that is static and bursty. The ON duration of the static (bursty) cross traffic follows a Pareto distribution with shape parameter of 1.5 (0.5) and scale parameter of 1000. We found that strong and static cross traffic has negative impact on classification accuracy, whereas strong

Location	Overall Loss Rate	$P[C]$	$P[C C]$	$P[W W]$
Heavy Traffic under Network Configuration I				
Upstream Bottleneck	29%	0.655	0.726	0.327
At Bottleneck	19.5%	0.77	0.36	0.713
Downstream Bottleneck	19.3%	0.539	0.796	0.26
Moderate Traffic under Network Configuration II				
Upstream Bottleneck	26.5%	0.626	0.928	0.673
At Bottleneck	21%	0.771	0.943	0.396
Downstream Bottleneck	17%	0.453	1	0.207

TABLE V

Impact of location of wireless medium on classification accuracy under different traffic load and different wireless medium location. The 5% wireless loss rate is used in all simulations shown in this table.

but bursty cross traffic has positive impact. The reason is that bursty cross traffic helps to make the queue occupancies more variable such that the RTT distribution of wireless losses is more wide spread, and thus more distinct from the compactly localized RTT distribution of congestion losses.

Up to this point, we only evaluated our technique under uniform wireless losses. We now consider the two-state Markovian-

Wireless Loss Rate	Overall Loss Rate	$P[W]$	$P[C C]$	$P[W W]$
1%	17.1%	0.059	0.76	0.513
2%	18%	0.111	0.513	0.666
5%	12%	0.414	0.943	0.396
10%	26%	0.383	0.952	0.218
15%	31.5%	0.475	1	0.673
20%	40%	0.5	1	0.556

TABLE VI

LABELING ACCURACY UNDER DIFFERENT WIRELESS LOSS RATES SET IN THE SIMULATION. NETWORK CONFIGURATION II IS USED WITH THE WIRELESS LINK BETWEEN NODES n_2 AND n_3 .

Traffic Type	Overall Loss Rate	$P[C]$	$P[W W]$	$P[C C]$
Static	2.6%	0.5	0.0966	1
Bursty	2.1%	0.524	0.978	1

TABLE VII

LABELING ACCURACY ON LOSSES BY TYPE IN THE PRESENCE OF STRONG STATIC AND BURSTY CROSS TRAFFIC. THE RELATIVE WIRELESS LOSS RATIO IS 1.

Gilbert error model. We used the model implemented in the ns-2.1b6 simulator. There are two states, GOOD and BAD, in the model in which no wireless error happens when the model is in GOOD state, whereas it happens at a predefined loss rate in the BAD state. Transition happens between these two states according to a predefined transition matrix. We tested our technique under two scenarios: light wireless loss mode and bursty wireless loss mode, and the results are shown in Table VIII.

B. Comparison to Vegas Predictor

We also compared the labeling accuracy obtained by our technique to the one obtained by the Vegas predictor [7], [8]. The Vegas predictor was found to be the best loss predictor among the set of predictors studied in [7]. It also encompasses the de-

Loss Burstiness	Overall Loss Rate	$P[C]$	$P[C C]$	$P[W W]$
Smooth	4.1%	0.756	0.568	1
Bursty	20.9%	0.512	0.76	0.755

TABLE VIII

Labeling accuracy under 2-state Markovian-Gilbert link error model. In the ‘‘Smooth’’ case, the time duration distribution in GOOD and BAD states is $\{\text{GOOD} = 0.99, \text{BAD} = 0.01\}$. In the ‘‘Bursty’’ case, the distribution is $\{\text{GOOD} = 0.9, \text{BAD} = 0.1\}$. Network configuration II is used in both cases.

sign philosophy of other existing loss predictors (cf. Section II). The Vegas predictor is defined as follows:

$$\text{Vegas predictor} = W \cdot \left(1 - \frac{\text{Base RTT}}{\text{RTT}}\right) \quad (\text{V.B})$$

where W is the current unacknowledged congestion window size and the Base RTT is the propagation round-trip delay. When Vegas predictor $> \beta$, a packet loss is attributed to congestion, otherwise a loss is attributed to wireless. We tried a set of values of β in the comparison as shown in Table IX. We can see that our labeling technique exhibits greater flexibility over the Vegas predictor. The Vegas predictor can only accurately predict one loss type at the expense of the other type.

Vegas Predictor			Our Labeling	
β	$P[C C]$	$P[W W]$	$P[C C]$	$P[W W]$
1	0.994	0.00332	0.932	0.753
2	0.225	0.819		
3	0.103	0.896		
4	0.0382	0.95		

TABLE IX

Comparison of labeling accuracy by our technique to the one by the Vegas predictor under different β . Network Configuration II is used.

VI. CONCLUSION

In this paper, we have described a heuristic labeling technique for differentiating losses due to congestion from losses due to wireless channel fading. This technique is developed based on the observation that the distribution of RTTs measured at the time of wireless losses has different characteristics from the one of congestion losses. Such difference in their RTT distributions can be exhibited in an HMM by different Gaussian components at different states. Based on the characteristics of the Gaussian components in a well estimated HMM, we proposed a labeling technique of assigning a very likely loss type to each state. The type of a packet loss can then be inferred by the loss type associated with a state that best characterizes the RTT observed at the time of occurrence of this packet loss.

We evaluated our labeling technique under different network configurations and traffic load. We found that our technique is effective in providing good labeling accuracy on losses under most practical network configurations. In particular, when buffers are not always at capacity, our technique is able of very accurately attributing congestion losses to congestion (*i.e.*, $P[C|C]$ is close to 1) while simultaneously providing high accuracy of labeling wireless losses (*i.e.*, $P[W|W]$ is high). Thus,

transport protocols equipped with our technique can react to congestion correctly as they now would assuming all losses are due to congestion. The high $P[W|\mathcal{W}]$ further allows a transport protocol to avoid unnecessary/incorrect application of “congestion control” actions at times of wireless losses. We have also shown the superiority of our labeling technique over the Vegas predictor, which was recently found to perform best and which exemplifies other existing loss labeling techniques.

ACKNOWLEDGEMENT

The authors gratefully thank Kavé Salamatian for helpful discussions and for sharing his code for HMM training.

REFERENCES

- [1] Alhussein A. Abouzeid, Sumit Roy, and Murat Azizoglu. Stochastic modeling of TCP over lossy links. In *INFOCOM (3)*, pages 1724–1733, 2000.
- [2] Ajay V. Bakre and B. R. Badrinath. I-TCP: Indirect TCP for mobile hosts. In *International Conference on Distributed Computing Systems*, pages 136–143, 1995.
- [3] Hari Balakrishnan, Venkata N. Padmanabhan, Srinivasan Seshan, and Randy H. Katz. A comparison of mechanisms for improving TCP performance over wireless links. *IEEE/ACM Transactions on Networking*, 5(6):756–769, 1997.
- [4] Hari Balakrishnan, Srinivasan Seshan, Elan Amir, and Randy H. Katz. Improving TCP/IP performance over wireless networks. In *proc. 1st ACM Int'l Conf. on Mobile Computing and Networking (Mobicom)*, page 10, 1995.
- [5] S. Biaz and N. Vaidya. Discriminating congestion losses from wireless losses using inter-arrival times at the receiver. *IEEE Symposium ASSET'99, Richardson, TX, USA*, 1999.
- [6] Saad Biaz and Nitin H. Vaidya. Distinguishing congestion and corruption losses: A negative result.
- [7] Saad Biaz and Nitin H. Vaidya. Performance of tcp congestion predictors as loss predictors. *Technical Report 98-007, Department of Computer Science, Texas A&M University*.
- [8] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. TCP vegas: New techniques for congestion detection and avoidance. In *SIGCOMM*, pages 24–35, 1994.
- [9] Ramon Caceres and Liviu Iftode. Improving the performance of reliable transport protocols in mobile computing environments. *IEEE Journal of Selected Areas in Communications*, 13(5):850–857, 1995.
- [10] W-P. Chen, Y-C. Hsiao, Y. Ge, and J. Hou. Syndrome: A light-weight approach to improving tcp performance in mobile wireless networks. *The Journal of Wireless Communications and Mobile Computing – Special Issue on Reliable Transport Protocols for Mobile Computing*, 2(2), 2002. Guest Editors: Vassilis Tsaoussidis and Ibrahim Matta.
- [11] Information Sciences Institute. The ns-2 simulator. Available at <http://www.isi.edu/nsnam/ns/>.
- [12] T. Kim, S. Lu, and V. Bharghavan. Improving congestion control performance through loss differentiation. In *ICCCN'99 (Eighth International Conference on Computer Communications and Networks)*, October 1999.
- [13] Jun Liu and Mark Crovella. Using loss pairs to discover network properties. In *ACM SIGCOMM Internet Measurement Workshop 2001*, San Francisco, CA, November 2001. ACM SIGCOMM.
- [14] C. Parsa and J.J. Garcia-Luna-Aceves. Improving TCP congestion control over internets with heterogeneous transmission media. In *IEEE ICNP 99: 7th International Conference on Network Protocols*, 1999.
- [15] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [16] K. Ratnam and I. Matta. WTCP: An efficient mechanism for improving tcp performance over wireless links. *Proc. Third IEEE Symposium on Computers and Communications (ISCC '98), Athens, Greece*, 1998.
- [17] Kavé Salamatian and Sandrine Vatou. Hidden markov modeling for network communication channels. In *Proceedings of ACM SIGMETRICS 2001 / Performance 2001*, Cambridge, MA, June 2001.
- [18] N. Samaraweera. Non-congestion packet loss detection for TCP error recovery using wireless links. In *IEE Communications*, volume 146, pages 222–230, 1999.
- [19] S.Cen, P. Cosman, and G. Voelker. End-to-end differentiation of congestion and wireless losses. In *Multimedia Computing and Networking 2002 (MMCN'02)*, 2002.
- [20] Y. Tobe, Y. Tamura, A. Molano, S. Ghosh, and H. Tokuda. Achieving moderate fairness for UDP flows by path-status classification. In *25th Annual IEEE Conf. on Local Computer Networks (LCN 2000)*, pages 252–261, 2000.
- [21] V. Tsaoussidis and H. Badr. Tcp-probing: Towards an error control schema with energy and throughput performance gains. In *Proceedings of the 8th IEEE Conference on Network Protocols, Japan*, November 2000.
- [22] H. Wang and P. Chang. On verifying the first-order markovian assumption for a rayleigh fading channel model, May 1996.
- [23] C. Zhang and V. Tsaoussidis. TCP-real: Improving real-time capabilities of TCP over heterogeneous networks. In *11th IEEE/ACM NOSSDAV 2001*, June 2001.
- [24] M. Zorzi, R.R. Rao, and L.B. Milstein. On the accuracy of a first-order markov model for data block transmission on fading channels. *Proc. IEEE ICUPC*, pages 211–215, Nov. 1995.