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

Jun Liu, Ibrahim Matta, Mark Crovella

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

Jun Liu
Computer Science Department
University of North Dakota
Grand Forks, ND 58202
jliu@cs.und.edu

Ibrahim Matta Mark Crovella
Computer Science Department
Boston University
Boston, MA 02215
{matta,crovella}@cs.bu.edu

***Abstract*— In a hybrid wired/wireless environment, an effective classification technique that identifies the type of a packet loss, *i.e.*, a loss due to wireless link errors or a loss due to congestion, is needed to help a TCP connection take congestion control actions only on congestion-induced losses. Our classification technique is developed based on the *loss pairs* measurement technique and Hidden Markov Models (HMMs). The intuition is that the delay distribution around wireless losses is different from the one around congestion losses. An HMM can be trained to capture the delays observed around each type of loss by different state(s) in the derived HMM. We develop an automated way to associate a loss type with a state based on the delay features it captures. Thus, classification of a loss can be determined by the loss type associated with the state in which the HMM is at that loss. Simulations confirm the effectiveness of our technique under most network conditions, and its superiority over the Vegas predictor. We identify conditions under which our technique does not perform well.**

***Index Terms*— Wireless Communication, HMM, Loss Classification, Queue Occupancy**

I. Introduction

In recent years, as wireless links have become more common in the Internet, the question of TCP behavior over wireless channels has become important [10], [17]. In such a setting, unmodified TCP does not differentiate losses due to buffer overflow from those due to signal fading on the wireless link. In this situation, a TCP connection 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) [8], [6]. However, correct classification based on these metrics has been found to be difficult [8]. It appears that there is lack of correlation between nature of losses and these observable metrics (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 losses in a hybrid wired/wireless environment, *i.e.*, distinguishing losses due to congestion from losses due to wireless channel fading. (From now on, we call packet losses due to congestion *congestion* losses and the losses due to wireless channel fading *wireless* losses.) Specifically, we monitor the most recent RTT before each loss to determine its most likely nature. End-to-end methods like ours have the advantage of not assuming support from the network (*e.g.*, split TCP connections or link layer retransmissions).

Our approach assumes that in order to differentiate congestion losses from the wireless losses, it may help to infer the network state at a packet loss event. Unmodified TCP uses packet loss to infer the presence of network congestion. However, when a network path includes a wireless link, packet loss is no longer an accurate indicator of congestion, and it becomes difficult to determine whether the network path is congested based on packet loss alone. For this reason, we make use of additional information: network delays at the time of the packet loss. Intuitively, congestion losses are associated with buffer overflows, so that packet delays when such losses occur will reflect the full queuing time of at least one buffer along the path. On the other hand, wireless losses are not associated with buffer overflows in general, so delays at such losses may be more variable compared to the delays at congestion losses.

We can formalize this intuition using Bayesian analysis. Let $T = \{C, W\}$ (for "congestion" and "wireless") be the set of loss types. Let L be a random variable ($L \in T$) signifying the type of a packet loss, and let R be a random variable ($R \in \mathbb{R}^+$) signifying the round-trip delay associated with each packet loss. The quality of differentiation possible at a particular delay r can be expressed as $P[L = l|R = r]$. From a Bayesian standpoint,

$$P[L = l|R = r] = \frac{P[L = l] \cdot P[R = r|L = l]}{\sum_{l' \in T} P[R = r|L = l'] \cdot P[L = l']} \quad (1)$$

This exposes the opportunity for making use of prior knowledge in performing the classification. 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 an

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 priors $P[L = l]$ unknown.

This formulation makes clear that the leverage provided by using delay on this problem is maximized when the component distributions $P[R|L = l],\ l \in T$ 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 using the delay measured on the path immediately before the loss. Our technique is based on 1) *loss pairs* [13], which measure the round-trip path delay at the time that a loss happens, and 2) Hidden Markov Models (HMMs) [14]. Loss pairs is a technique that conveys path delay at the time of packet losses to an end node of the path. HMMs are a classification tool that can make use of recent history in classifying present observations.

Our results show that our technique is effective in most network configurations. It is superior in quality of classification of losses compared to the best method reported in [8] (the Vegas predictor).

Our technique has the advantage that it does not disrupt the end-to-end semantics of the IP path (and so, for example, could co-exist with IPsec). Furthermore, we show that there is generally a tradeoff in accuracy of identification of congestion losses versus identification of wireless losses. Our approach has the advantage that is can be tuned to favor accuracy of one type of identification over another. For example, in a setting where TCP-friendliness is important, accuracy of identification of congestion losses can be favored.

The rest of this paper is organized as follows. We describe related work in Section II, and in Section III we present a basic introduction to HMMs and the loss pairs technique. The classification technique itself is described in Section IV, and its evaluation is described in Section V. In Section VI we conclude.

II. Related Work

Many studies have shown that TCP goodput can be improved if the cause of packet loss is identified [3], [17]. By attributing a packet loss to wireless transmission errors, the TCP source can refrain from taking unnecessary "congestion" control measures. One set of solutions (e.g., I-TCP [2], Snoop [4], WTCP [15]) require support from the base station located at the interface between the wired infrastructure and the wireless access infrastructure. 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). For example, the Flip Flop-based loss classification in [5] and the Vegas predictor [8] attribute a packet loss to congestion if the measured delays exceed a certain delay threshold, otherwise the cause of a packet loss is assumed to be 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 (unlike implicit schemes such as TCP Westwood [11]). 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 [14]. Salamatian and Vaton [16] 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 [8], 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

1) HMM Structure: 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 [14]. 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 $\{o_t : t = 1, 2, \cdots\}$. The generation of this time series can be imagined as coming from a discrete time Markov chain. At each state change, the chain generates an observation (signal) based on a probability distribution associated with the current state. In general, the observation can be either in discrete or in continuous form; in this paper, we only focus on HMMs with Gaussian (continuous) observations.

More formally, an HMM is defined by the following elements [14]:

- N, the number of states in the model. We denote the state space as $\{S_1, S_2, \cdots, S_N\}$.
- For each state S_j, a probability density function $b_j(o)$ over the set of possible observations. In our work we take this to be a Gaussian density of the form

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

  where $\mathcal{N}$ is a Gaussian density function with mean μ_j and variance σ_j^2.
- The state transition probability distribution $A = \{a_{ij}\}$ where $a_{ij} = P[s_{t+1} = S_j | s_t = S_i],\ 1 \le i, j \le N$.
- The initial state distribution $\pi = \{\pi_1, \pi_2, \cdots, \pi_N\}$ where $\pi_i = P[s_1 = S_i],\ 1 \le i \le N$.

2) HMM Training: Training the HMM requires finding appropriate values for $\{a_{ij}\}$ and $\{b_j\}$ (which in our case means $\{(\mu_j, \sigma_j^2)\}$). This inverse problem is usually attacked using the EM (Expectation-Maximization) approach. The EM approach seeks to find the model parameters that make the set of observations most likely. EM is an iterative approach; as the EM iterations proceed, each newly derived HMM becomes more likely than the previous one to have generated the series of observations. Excellent explanations of how HMMs can be trained via the EM algorithm are given in [14], [16].

3) State Sequence Inference: Once we have a trained model, we seek to find the state sequence that corresponds to the given sequence of observations. That is, the final goal is to find the state sequence $\hat{\mathbf{s}} = \{\hat{s}_1, \hat{s}_2, \cdots, \hat{s}_T\}$ for a given sequence of observations $\mathbf{o} = \{o_1, o_2, \cdots, o_T\}$. For this purpose we use the Viterbi algorithm [14], which uses dynamic programming to perform efficient inference of state sequences. If $\hat{\theta}$ is the trained model (*i.e.,* the model after multiple iterations of the EM algorithm), the Viterbi algorithm produces a state sequence $\hat{\mathbf{s}}$ such that the *a posteriori* log-likelihood $L(\hat{\mathbf{s}}|\mathbf{o}; \hat{\theta})$ is maximized, meaning that $\hat{\mathbf{s}}$ is the most likely sequence of states followed by the model $\hat{\theta}$.

IV. Loss Differentiation Technique

In this section, we describe how we apply Hidden Markov Models and loss pairs to differentiate wireless from congestion losses.

A. Motivation

In all the work described in this paper, we use HMMs with 4 states, trained on 10000 observations. These parameters are chosen based on [16], which showed that in a number of settings, $N = 4$ was sufficient to characterize a network communication channel using observations of packet loss events, and that 10000 observations was sufficient to train such a model.

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 portion (a) 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 much more widely spread.

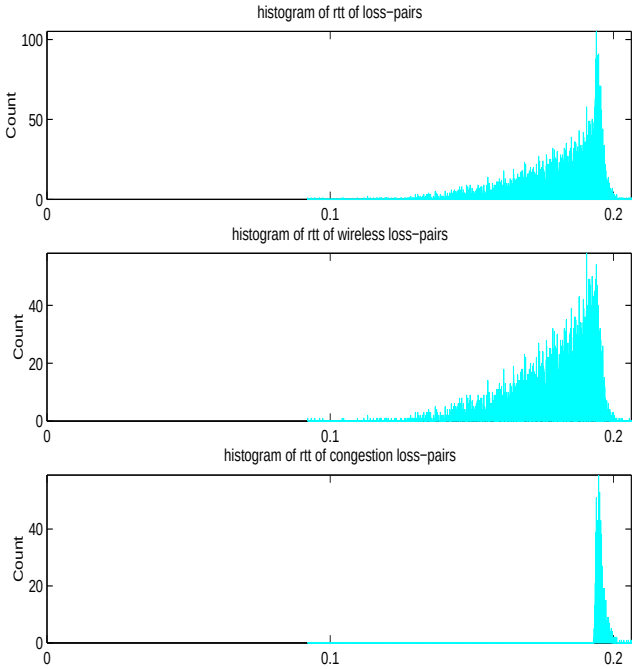
Using these loss pair RTT measurements, we trained our four-state HMM; the Gaussian models for each state of our trained HMM are shown in order (1 through 4) below the histograms in Figure 1. We note that the distribution corresponding to state 3 looks most similar to the distribution of RTTs of congestion loss pairs. We can conjecture that state 3 corresponds to congestion losses more strongly 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 3 is the state most strongly corresponding to congestion losses.

In Figure 1(a), 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 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.

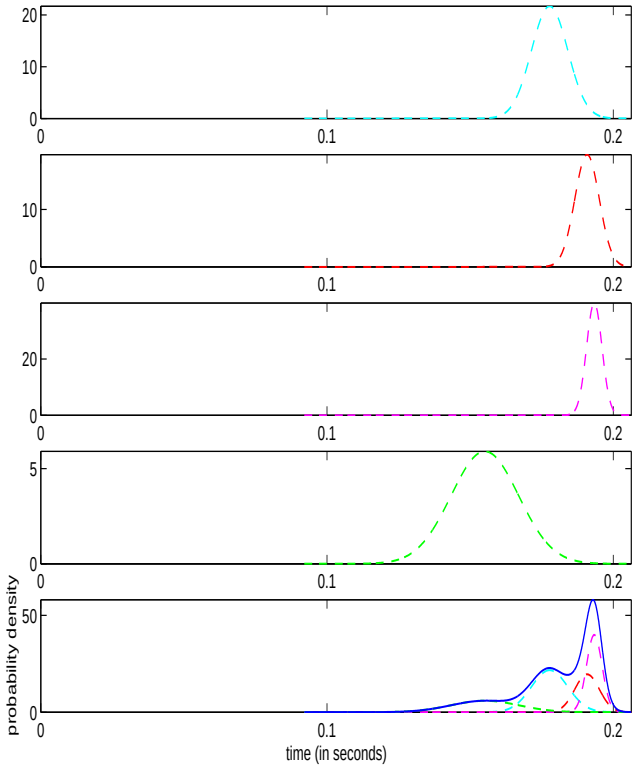
As just described, each observation of loss pair at a particular time can be associated with a most likely state of the HMM. Thus to use the trained HMM, we need to determing a "labeling" for each state, *i.e.*, an assignment of a loss type to each state of the HMM. When $N = 4$, there are 2^4 possible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

To measure the quality of a classification method (*i.e.*, a particular state labeling) we propose to use $P[t|t']$ as a set of metrics. $P[t|t']$ is the probability of a loss 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 classified as a wireless loss" be denoted as W and "being classified 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 classifying 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 labeling of states for three network



(a) Histogram of RTTs at losses

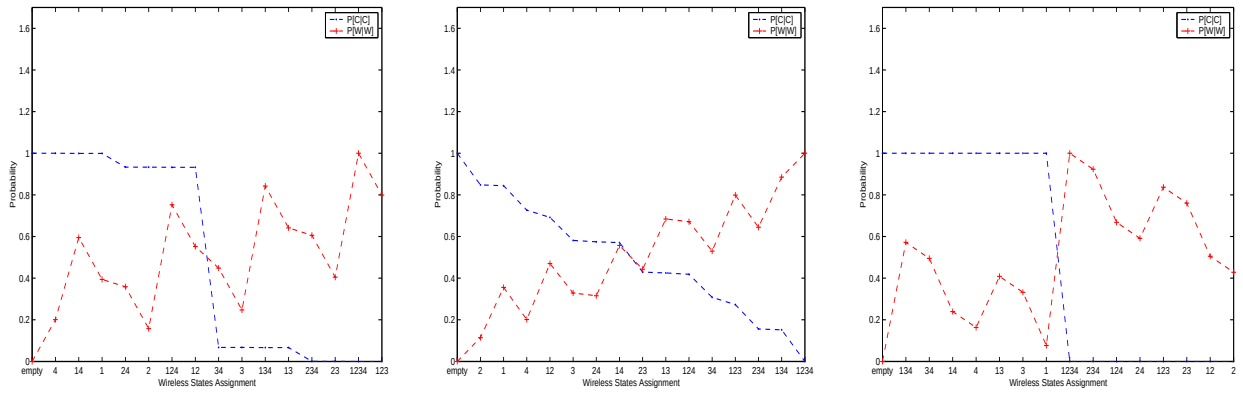


(b) Gaussian corresponding to each state, and their composition

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

(c) Parameters of Gaussian for each state

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.



(a) (b) (c)

Fig. 3. Accuracy of classifi cation on losses by type under all possible wireless state assignments. State assignments are labeled according to which states are assigned to wireless errors, and are presented in decreasing order of $P[C|\mathcal{C}]$.

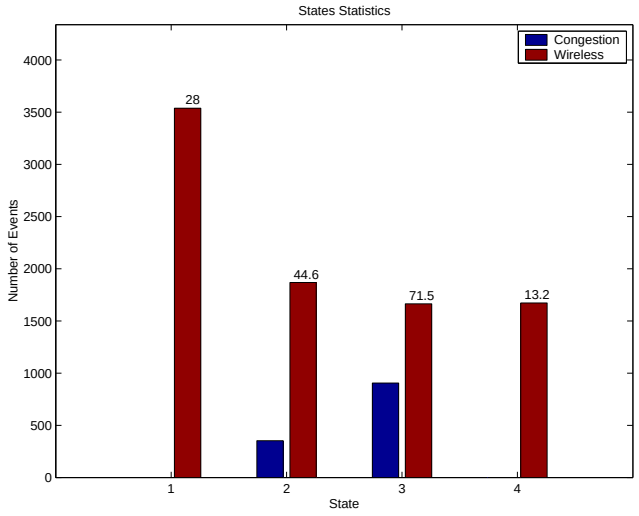


Fig. 2. Number of times a state is visited by loss pairs. For each state, the left bar denotes losses due to congestion and right bar denotes losses due to wireless channel fading.

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 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 $P[C|\mathcal{C}]$, it is often possible to obtain reasonably high $P[W|\mathcal{W}]$ as well. Referring to Figure 2, if we label states 2 and 3 as Congestion, and the remaining two states as Wireless, then we achieve $P[C|\mathcal{C}] = 1$ as well as a high $P[W|\mathcal{W}]$.

The state labeling we use in the rest of this paper is one that emphasizes high $P[C|\mathcal{C}]$. That is, we seek to find the state labeling with highest $P[W|\mathcal{W}]$ given that $P[C|\mathcal{C}]$ 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 labeling is thus necessarily heuristic. We discuss our heuristic next and evaluate it in the following 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
$$f_{ij} = \pi_i \cdot a_{ij} \quad (1 \le i, j \le N)$$
  - (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 $P[C|\mathcal{C}]$ while keeping $P[W|\mathcal{W}]$ 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

We first examine the accuracy of our loss classification technique under different network settings, we then discuss the possible contributing factors to its accuracy.

We evaluate our technique by classifying losses collected in simulations using the ns-2 network simulator [12]. The simulations are configured to have a network topology as shown in Figure 4, where nodes n_1, n_2, n_3, n_4 represent routers on the backbone path $n_1 - n_4$ with a bottleneck link between routers n_2 and n_3.[1] Furthermore, one link on the backbone path is wireless; its location, wireless error model, and loss rate are varied.

To generate network traffic, several sets of TCP sources/sinks are used to send packets through a portion of or the whole backbone network path. Among the TCP source/sink pairs, the ones that use the whole backbone path are treated as observable TCP source/sink pairs in the simulations, and all the others which only use a portion of the backbone path are treated as unobservable TCP source/sink pairs used to create the cross traffic. In order to better control the strength of the cross traffic on each link, each of the unobservable TCP source/sink pairs only goes across one network link of the backbone.

[1] A significant number of packet losses due to congestion happen at the bottleneck link.

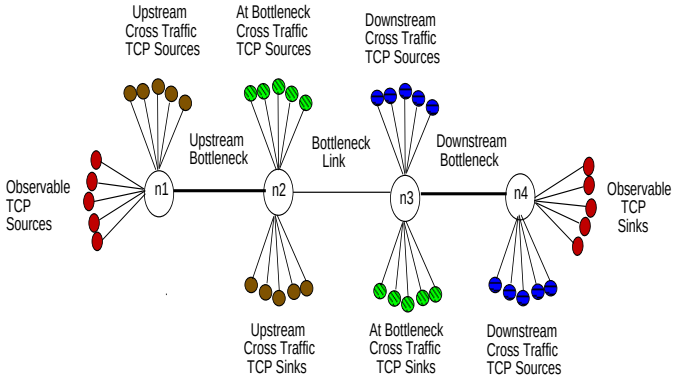


Fig. 4. The network topology used in the ns-2 simulations with a hybrid wired/wireless network setting.

We list the parameters of two network configurations in Table I. Unless specified otherwise, 500 and 200 observable TCP source/sink pairs are used in configurations I and II, respectively. 30 unobservable TCP source/sink pairs go across each link of the backbone path to create cross traffic in both network scenarios. Furthermore, in order to better mimic the traffic behavior in the Internet, each TCP source follows an ON/OFF activity pattern where the OFF duration is heavy-tail distributed drawn from a Pareto distribution.

In the rest of this section, we first examine the accuracy of our loss classification technique and how it is affected by various network parameters, we then compare our loss classification technique to the Vegas predictor [8]. Finally, we present results with IEEE 802.11 wireless interfaces.

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 SIMULATIONS.

A. Contributing Factors to the Classification Accuracy

Various factors may affect the accuracy of loss classification by affecting the distinctiveness of delay distributions around different types of losses. These factors include: utilization of the bottleneck link, the capacity of the backbone path (*i.e.*,

the bandwidth-delay product), the location and the link error model of the wireless link, and the strength of the observable traffic and the cross traffic. In our evaluation, the wireless error models over the wireless link include both the simple memoryless uniform error model used to mimic random packet losses [8], [6] and the two-state Markovian-Gilbert model used to mimic bursty packet losses [1], [18], [19].

1) Network Configuration Related Factors: The basic idea behind our classification technique is that the distinct delay distributions (or equivalently, the convolved queue occupancies on the backbone path) around different types of losses can be used to classify losses. Therefore, we examine the impact of network parameters on queue occupancy fluctuations, which in turn affect classification accuracy. In that context, the utilization of the bottleneck link is a critical factor; a heavily loaded bottleneck may result in high packet loss rate and steady queue occupancy (close to a full queue), while a lightly loaded bottleneck may result in low packet loss rate and highly oscillatory queue occupancy.

In order to create different levels of utilization of the bottleneck link, using the network configuration I shown in Table I, we vary the number of observable TCP connections from 50 to 250 in increments of 50. We set the (n_3,n_4) link as wireless with a uniform error model of 2% error rate. The classification accuracies under different levels of utilization are listed in Table II. We note that the classification accuracies deteriorate with increasing utilization of the bottleneck link. This fact shows that wireless losses and congestion losses become difficult to differentiate by delays around losses since the bottleneck queue occupancy is less oscillatory under very high link utilizations ($> 90\%$).

Utilization ρ	Overall Loss Rate	$P[\mathcal{C}]$	$P[C\|\mathcal{C}]$	$P[W\|\mathcal{W}]$
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 II

THE CLASSIFICATION ACCURACY UNDER DIFFERENT LEVELS OF UTILIZATION AT THE BOTTLENECK LINK. THE OVERALL LOSS RATE IS DEFINED AS THE RATIO OF THE TOTAL NUMBER OF LOSS EVENTS TO THE TOTAL NUMBER OF PACKETS SENT BY ALL TCP SOURCES.

We next examine the impact of the capacity (end-to-end bandwidth-delay product) of the backbone path. The classification accuracies for different values of the bandwidth-delay product are shown in Table III by varying the bottleneck bandwidth from 0.5 Mbps to 2.5 Mbps at increments of 0.5 Mbps. We observe that the classification accuracy is generally improved for increasing values of the bandwidth-delay product. This is because a higher capacity of the network path generally results in more highly oscillatory (variable) queue occupancies on the backbone path (especially at the bottleneck link). Therefore, the convolved queue occupancy on the backbone path spreads over a wider range, whereas the queue occupancy distribution around congestion is still compactly localized around the full queue size of the bottleneck link.

BW×D factor	Overall Loss Rate	$P[\mathcal{C}]$	$P[C\|\mathcal{C}]$	$P[W\|\mathcal{W}]$
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 III

THE CLASSIFICATION ACCURACY UNDER DIFFERENT VALUES OF THE BANDWIDTH-DELAY PRODUCT OF THE BACKBONE PATH FOR CONFIGURATION II SHOWN IN TABLE I. THE ACTUAL VALUE OF THE BANDWIDTH-DELAY PRODUCT IN EACH ROW IS THE PRODUCT OF THE BW×D FACTOR AND A BASE VALUE OF $16Kb$.

The cross traffic can also affect the queue occupancies on the backbone path and thus affect the classification accuracy. Table IV shows that a *static* cross traffic has a negative impact on the classification accuracy, whereas a *bursty* cross traffic has a positive impact on the classification accuracy. This is because a bursty cross traffic helps make the queue occupancies more oscillatory such that the delay distribution around the two types of losses are more distinctive.

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

TABLE IV

THE CLASSIFICATION ACCURACY UNDER DIFFERENT TYPES OF CROSS TRAFFIC.

We next vary the location of the wireless link on the backbone path in both network configurations I and II shown in Table I. We assume a uniform error model of a fixed 5% error rate over the wireless link. Table V shows no noticeable impact of the location of the wireless medium on the classification accuracy.[2]

2) The Effect of the Wireless Link Error Model: We first examine the classification accuracy for the uniform error model under different link error rates. We can clearly observe from Table VI that the classification accuracy is generally increased under higher wireless link error rate. A possible reason is that under a higher link error rate, TCP connections tend to synchronize their backoffs and probing of bandwidth,

[2]The same is true for a higher 10% wireless link error rate (not shown here).

Location	Overall Loss Rate	$P[\mathcal{C}]$	$P[C\|\mathcal{C}]$	$P[W\|\mathcal{W}]$
Heavy Traffic under Network Configuration I				
Upstream Bottleneck	15%	0.655	0.726	0.327
At Bottleneck	14.6%	0.77	0.36	0.713
Downstream Bottleneck	10.2%	0.539	0.796	0.26
Moderate Traffic under Network Configuration II				
Upstream Bottleneck	9.5%	0.626	0.928	0.673
At Bottleneck	7.8%	0.771	0.943	0.396
Downstream Bottleneck	6%	0.453	1	0.207

TABLE V

THE CLASSIFICATION ACCURACY UNDER VARIOUS WIRELESS MEDIUM LOCATIONS IN BOTH NETWORK CONFIGURATIONS SHOWN IN TABLE I. THE WIRELESS LINK ERROR RATE IS SET TO 5%.

resuling in more oscillatory convolved queue occupancy on the backbone path.

Wireless Loss Rate	Overall Loss Rate	$P[\mathcal{W}]$	$P[C\|\mathcal{C}]$	$P[W\|\mathcal{W}]$
1%	7.3%	0.059	0.76	0.513
2%	8%	0.111	0.513	0.666
5%	7.8%	0.414	0.943	0.396
10%	8.4%	0.383	0.952	0.218
15%	8.3%	0.475	1	0.673
20%	7%	0.5	1	0.556

TABLE VI

THE CLASSIFICATION ACCURACY UNDER DIFFERENT LINK ERROR RATES WHEN A UNIFORM WIRELESS LINK ERROR MODEL IS USED BETWEEN NODES n_2 AND n_3 FOR CONFIGURATION II.

We next consider a more realistic wireless link error model—the two-state Markovian-Gilbert error model. In this model, the wireless link is either in a GOOD state in which no wireless error happens, or in a BAD state in which wireless errors happen at a predefined loss rate [1]. The wireless medium transits between these two states according to predefined state transition probabilities. Table VII shows that bursty wireless link errors (longer duration in the BAD state) generally lead to accurate loss classification.

Loss Burstiness	Overall Loss Rate	$P[\mathcal{C}]$	$P[C\|\mathcal{C}]$	$P[W\|\mathcal{W}]$
Smooth	4.1%	0.756	0.568	1
Bursty	6.2%	0.512	0.76	0.755

TABLE VII

THE CLASSIFICATION ACCURACY UNDER WIRELESS LINK LOSSES MODELED BY A TWO-STATE MARKOVIAN-GILBERT ERROR MODEL. IN THE "SMOOTH" ERROR MODE, THE RESIDENCE DURATION IN THE *GOOD* STATE AND THE *BAD* STATE IS 99 PACKETS AND 1 PACKET, RESPECTIVELY. IN THE "BURSTY" ERROR MODE, THE DURATION IN THE *GOOD* STATE AND THE *BAD* STATE IS 90 PACKETS AND 10 PACKETS, RESPECTIVELY. NETWORK CONFIGURATION II IS USED.

B. Accuracy Comparison to the Vegas Predictor

We compare the classification accuracy of our loss classification technique to the Vegas predictor [7], [9]. The Vegas predictor works as follows: When an ACK packet is received by the TCP source, a new predictor value is computed as:

$$Vegas\ predictor \quad = \quad W \cdot \left(1 - \frac{\mathrm{Base\,RTT}}{\mathrm{RTT}}\right) \qquad (2)$$

where W is the current unacknowledged congestion window size, $\mathrm{Base\,RTT}$ is the minimum round-trip time observed so far, and RTT is the current round-trip time.

The TCP source then categorizes the current state of the path using two predefined thresholds, α and β ($\alpha < \beta$). When the newly computed value of the Vegas predictor is smaller than α, the path is categorized as *underutilized*. If the value of the predictor is between α and β, the path is categorized as *normal*. If the value of the predictor is higher than β, the path is categorized as *congested*. Since the state of the path (congested or not) is only determined by the value of the predictor relative to β, we let $\alpha = \beta$ in the comparison. Table VIII shows that our classification technique exhibits great flexibility and superiority over the Vegas predictor. In some sense, no matter how the parameter β is set, the Vegas predictor always favors one type of losses over the other type of losses. On the contrary, our technique shows high accuracy in classifying congestion losses, while maintaining a reasonably high accuracy in classifying wireless losses.

Vegas Predictor			Our Technique	
β	$P[C\|\mathcal{C}]$	$P[W\|\mathcal{W}]$	$P[C\|\mathcal{C}]$	$P[W\|\mathcal{W}]$
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 VIII

ACCURACY COMPARISON OF OUR TECHNIQUE TO THE VEGAS PREDICTOR UNDER DIFFERENT VALUES OF β. NETWORK CONFIGURATION II IS USED.

C. Evaluations with 802.11 Wireless Connectivities

Finally, we examine our classification technique in simulations with IEEE 802.11 wireless interfaces configured. The

Overall Loss Rate	$P[\mathcal{C}]$	$P[C\|\mathcal{C}]$	$P[W\|\mathcal{W}]$
8.7%	0.756	0.867	0.71

TABLE IX

THE CLASSIFICATION ACCURACY IN A NETWORK ENVIRONMENT WITH IEEE 802.11 WIRELESS INTERFACES CONFIGURED.

topology used in this simulation is the same as the one shown in Figure 4 except that n_4 acts as a base station that communicates with the set of observable TCP sinks to its right via a common wireless medium. The parameters used are those of configuration I in Table I. Table IX confirms the effectiveness of our classification accuracy for 300 observable TCP sources/sinks.

VI. CONCLUSION

In this paper, we have described a heuristic technique for classifying losses into different types, *i.e.*, losses due to congestion and losses due to wireless channel fading. This technique is developed based on the observation that the distributions of round-trip delays around different types of loss have different features which can be used to classify losses. These different features can be captured by the Gaussian component(s) in a derived HMM model. More specifically, based on Bayesian analysis, we associate a loss type with each state in an HMM trained by delay observations around losses. The type of an observed loss is identified by the loss type associated with the state in which the HMM stays at that loss. We evaluated our classification technique under various network conditions, including different network configurations, workload, locations and error rates of the wireless medium, and different wireless error models. We found that our classification technique is effective in most network environments, and that it is superior to the Vegas predictor.

We are currently investigating tradeoffs between classification accuracy and time/computation complexity, and the interaction between loss classification and recovery control.

ACKNOWLEDGEMENTS

The authors gratefully thank Kavé Salamatian for helpful discussions, and Sandrine Vaton for sharing her code for HMM training and Viterbi state estimation. This work was supported in part by NSF grants ANI-9986397, ANI-0095988 and EIA-0202067, and grants from Sprint and Motorola Labs.

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] Dhiman Barman and Ibrahim Matta. Effectiveness of Loss Labeling in Improving TCP Performance in Wired/Wireless Networks. In *Proceedings of ICNP'2002: The 10th IEEE International Conference on Network Protocols*, Paris, France, November 2002.

[6] Saad Biaz and Nitin Vaidya. Discriminating congestion losses from wireless losses using inter-arrival times at the receiver. *IEEE Symposium ASSET'99, Richardson, TX, USA*, 1999.

[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] Saad Biaz and Nitin H. Vaidya. Distinguishing congestion and corruption losses: A negative result. *Seventh International Conference on Computer Communications and Networks (IC3N)*, 1998.

[9] 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.

[10] 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.

[11] C. Casetti, M. Gerla, S. Mascolo, M. Sanadidi, and R. Wang. TCP westwood: Bandwidth estimation for enhanced transport over wireless links. In *ACM Mobicom 2001*, pages 287–297, 2001.

[12] Information Sciences Institute. The `ns-2` simulator. Available at `http://www.isi.edu/nsnam/ns/`.

[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] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[15] K. Ratnam and I. Matta. WTCP: An efficient transmission control protocol for networks with wireless links. *Proc. Third IEEE Symposium on Computers and Communications (ISCC '98), Athens, Greece*, 1998.

[16] Kavé Salamatian and Sandrine Vaton. Hidden markov modeling for network communication channels. In *Proceedings of ACM SIGMETRICS 2001 / Performance 2001*, Cambridge, MA, June 2001.

[17] Vassilis Tsaoussidis and Ibrahim Matta. Open Issues on TCP for Mobile Computing. *Journal of Wireless Communications and Mobile Computing – Special Issue on Reliable Transport Protocols for Mobile Computing*, 2(1), February 2002.

[18] H. Wang and P. Chang. On verifying the first-order markovian assumption for a rayleigh fading channel model, May 1996.

[19] 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.