On-line Estimation of Internet Path Performance: An Application Perspective

Shu Tao and Roch Guérin
Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, PA 19104, USA
Email: {shutao@seas, guerin@ee}.upenn.edu

Abstract— **Estimating end-to-end packet loss on Internet paths is important not only to monitor network performance, but also to assist adaptive applications make the best possible use of available network resources. There has been significant prior work on measuring and modeling packet loss in the Internet, but most of those techniques do not focus on providing real-time information and on assessing path performance from an application standpoint. In this paper, we present an on-line probing-based approach to estimate the loss performance of a network path, and extend this estimate to infer the performance that an application using the path would see. The approach relies on a hidden Markov model constructed from performance estimates generated from probes, which is then used to predict path performance as an application would experience. The accuracy of the model is evaluated using a number of different metrics, including loss rate and loss burstiness. The sensitivity of the results to measurement and computational overhead is also investigated, and an extension of the base approach using a layered model is explored as a possible solution to capturing time-varying channel behavior while keeping computational complexity reasonably low. The results we present show that the approach is capable of generating accurate, real-time estimates of path performance, and of predicting the performance that applications would experience if routed on the path.**

I. Introduction

End-to-end performance of network paths is useful information not only to assess overall network performance, but also as a key input towards improving application quality. In particular, adaptive applications can adjust their behavior in response to changes in network path performance in order to minimize the impact those changes have on application level quality. Furthermore, when multiple paths are available, knowing the performance of each path can help application optimize performance by dynamically selecting which path to use. The latter has been one of the motivations behind the development of application (content) specific distribution networks that are overlaid on top of an existing network infrastructure, and the attempt to adapt the overlay topology and routing to ensure the best possible performance for its application, e.g., [1] and the RON project [2]. The use of such overlays can help overcome sub-optimal routing decisions by the default routing mechanisms that control the network paths on which packets are forwarded. In particular, default routing decisions are typically oblivious to the specific requirements of a given application, and focus instead (rightfully so) on global network performance measures. As a result, paths are often available that offer significantly better performance given the needs of an application.

The material presented in this paper is based upon work supported by the National Science Foundation under Grant No. 9906855.

Taking advantage of the existence of "better" paths calls for both the ability to control routing decision, and for the ongoing monitoring of path quality. As mentioned before, the former has been one of the motivations behind the development of overlay networks, while the latter is the main focus of this paper. In particular, our goals are two-fold. First, we want to develop a practical solution for on-line, i.e., real-time, estimation of end-to-end path performance. Second, we want to ensure that although the probing mechanism used to infer path performance will typically differ significantly from the pattern of application traffic, the performance estimate it generates remains an accurate predictor of the performance the application would experience if it were to use the path. The feasibility of the second goal depends on our ability to construct a model that effectively captures the statistics of the path state. Such a model would allow us to predict the evolution of the path state as seen by the packets of an application. The performance estimates derived from this model will then be used as an input to any control process aimed at deciding if and when the traffic of an application should be switched to a different path. Note that implicit in this approach is the assumption that the traffic of the application itself does not (significantly) affect the path state. This will typically be the case when applications generate an amount of traffic that is small compared to the path bandwidth and the total amount of traffic it carries.

Estimating Internet (path) performance is clearly not a new topic, but most earlier works have had a somewhat different focus. These works can be classified into two major categories. The first category involves studying traces gathered over a period of time to extract a better understanding of traffic and loss patterns. For example, works such as [3], [4], and [5] were aimed at identifying mathematical models to characterize loss traces and reveal temporal dependencies that might exist. Other works such as [6] and [7] were concerned with studying the stationarity of the loss process on Internet paths and analyzing its predictability. Such analyses are clearly useful to understand the general loss characteristics of Internet paths, but they cannot be readily applied to the on-line estimation of path performance.

The second category of works that targeted the estimation of Internet performance is more relevant to this paper. It consists mostly of works that used a variety of probing techniques to infer various characteristics of Internet paths. By nature, probing is well suited to real-time estimation, and is therefore also the basis of the method used in this paper. Probes have been used to obtain various network performance measures such as available bandwidth, delay, and loss. For instance, probing a path using packet pairs [8] or packet trains [9] can provide estimates of bottleneck link speed as well as the available bandwidth. Similarly, in [10] He *et al.* use a probing method to measure end-to-end cross traffic by exploiting the long range dependence nature of Internet traffic. Closer to our goals, [11] introduces a method for measuring network delay using ICMP timestamp probes. Most relevant to this paper, [12] and [13] use various inference techniques and end-to-end multicast/unicast probes to estimate the loss rates on individual links. The main difference with our work is that we are primarily interested in inferring end-to-end loss performance on a given path as opposed to an individual link. More important, we are trying to ensure that our estimates are relevant and accurate from the perspective of applications, so that their traffic can be switched between multiple available paths based on such estimates. In other words, we want to predict the losses that would be seen by an application with very different characteristics compared to the probe traffic. This also means that we need to construct a model that captures more than basic loss estimates and will include other parameters such as loss length distribution, loss distance, etc.

In order to achieve a comprehensive characterization of the end-to-end loss, we use the information gathered by the probes to construct a Hidden Markov Model (HMM) that captures the main characteristics of the loss process. Salamatian *et al.* [14] first proposed using HMM to model the loss performance of network paths. But our work extends the basic model along two dimensions. Specifically, we consider two major issues in constructing the model:

- *The sensitivity of the model accuracy to the probing rate.* A higher rate typically yields a more accurate model but at the cost of a greater overhead.
- *The ability of the model to predict performance for applications with a broad range of characteristics.* This is affected by both the model's ability to predict future path performance, and its dependency on the information obtained from the probes.

In the paper, we first review the method we follow to construct an HMM based on the information obtained from probes. The sensitivity of this model to the characteristics of probing traffic is then assessed, before we evaluate the "performance" of our approach in predicting the end-to-end performance that various applications would experience. This evaluation phase is carried out based on several experiments over different network paths and using a variety of applications as traffic sources.

The remainder of the paper is structured as follows. In section II, we introduce the mathematical model that underlies the approach we use for path state estimation, as well as how to use the model to estimate the performance seen by traffic sources with different characteristics. Section III describes the accuracy of the model in estimating loss rate and loss burstiness. Section IV is devoted to investigating the sensitivity of the model's accuracy to the intensity of the probe traffic, while section V focuses on presenting experimental results aimed at assessing the validity of the approach. Section VI presents an extension to the basic model for loss prediction. Finally, Section VII summarizes the results presented in the paper.

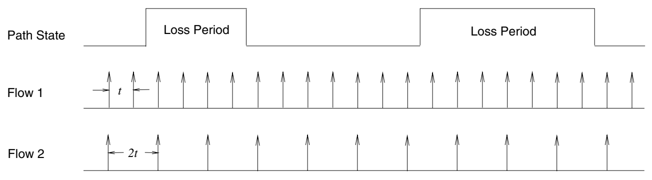


Fig. 1. The difference in sampling the loss process of a network path for different flows.

II. Mathematical Model

What losses a flow would experience if routed on a given path depends on many factors, including when the packets are sent (whether the path is in congestion or not), how the packets are sent (e.g., the inter-packet spacing), the size of packets (a larger packet is more likely to get lost than a smaller one), and so on. Clearly, the most accurate way of determining the performance that a given application flow would experience is to send probes in exactly the same manner as the target application sends its packets. Unfortunately, this is typically infeasible because the target application is often not known ahead of time, and even if it was, the resulting traffic overhead would be too large. In general, active measurement schemes are practical only if the amount of probing traffic remains moderate. Nevertheless, even with different traffic patterns, the losses that two flows following the same network path would experience are not independent. Loss is a function of the path state, e.g., path congestion or path changes, that the two flows share. This simple statement is the basis of our estimation method.

Specifically, our goal is to construct a model that lets us directly estimate the state of a path and its evolution over time. Once the path state is known, it can be used to determine the performance of *any* flow routed over the path. In other words, we consider the state of a path as a continuous-time process, e.g., alternating between congestion periods when loss occurs and loss-free periods, that different flows sample differently. The losses experienced by a flow can then be estimated from the flow's sampling pattern and the evolution of the path state. For example, in Fig. 1, two flows are sampling the same path with different frequencies and, hence, experiencing different loss patterns that can, however, be predicted from the evolution of the path state. Our goal is, therefore, to use a "probing flow" to derive a model describing the state of a path and its evolution, and then use this model to predict the performance experienced by other flows with different sampling patterns.

The approach we rely on involves three steps: (1) probing the path to collect loss statistics; (2) use those collected statistics to generate a model describing the evolution of the path state; and (3) derive the characteristics of the losses that a target application would experience based on the path state model and the application traffic pattern.

A. Hidden Markov Model

We model the loss process of a network path as a multi-state continuous-time Markov process. Each state represents a different level of congestion and is, therefore, associated with a different loss probability. Thus, the problem boils down to deriving the model parameters from the results of probing, and we use a Hidden Markov Model (HMM) [14][15][16] for this purpose. We first derive a discrete Markov model from the observation sequence, then convert it to a continuous-time model, which can then be used to predict the performance that a given application flow would see.

An HMM is defined by the following parameters:

- N, the number of states in the model. We denote the individual states as $S = \{S_1, S_1, ..., S_N\}$, and the state of the path at time t as q_t.
- M, the number of distinct observation symbols per state. In our case, there are only two possible observation results—either a correctly received packet or a lost one. We denote the individual symbols as $V = \{v_0, v_1\}$, which represent received and lost packets, respectively.
- The state transition probability distribution $A = \{a_{ij}\}$, where

$$a_{ij} = P\{q_{t+1} = S_j | q_t = S_i\}, i, j \in \{1, 2, ..., N\}.$$

- The observation symbol probability distribution in state j, $B = \{b_k\}$, where

$$b_j(k) = P\{v_k \text{ at } t | q_t = S_j\}, j \in \{1, 2, .., N\}, k \in \{0, 1\}.$$

- The initial state distribution $\psi = \{\psi_i\}$, where

$$\psi_i = P\{q_1 = S_i\}, i \in \{1, 2, ..., N\}.$$

For the convenience of description, we use the compact notation $\lambda = (A, B, \psi)$ to indicate the complete parameter set of the model. Thus, the model formulation problem becomes the following: given the observation sequence $O = O_1 O_2 ... O_T$ (T is the total number of observations), choose the proper model parameters λ to maximize $P(O|\lambda)$. To compute λ in an efficient way, we need to first define the forward variable $\alpha_t(i)$ as

$$\alpha_t(i) = P\{O_1 O_2 ... O_t, q_t = S_i | \lambda\}.$$

Similarly, the backward variable $\beta_t(i)$ is defined as

$$\beta_t(i) = P\{O_{t+1} O_{t+2} ... O_T | q_t = S_i, \lambda\}.$$

Both the forward and the backward variables can be solved inductively given λ and O [17]. We also define $\xi_t(i, j)$, the probability of being in state S_i at time t, and state S_j at time $t + 1$, given the model and the observation sequence, i.e.,

$$\xi_t(i, j) = P\{q_t = S_i, q_{t+1} = S_j | O, \lambda\}.$$

From the definitions of the forward and backward variables, we can write $\xi_t(i, j)$ in the form

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}. \quad (1)$$

Using the variables defined above, we can re-estimate the parameters ($\bar{\lambda}$) of an HMM following the Baum-Welch procedure [17], i.e.,

$$\begin{aligned} \bar{\psi}_i &= \sum_{j=1}^{N} \xi_1(i, j), \\ \bar{a}_{ij} &= \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} \xi_t(i, j)}, \\ \bar{b}_j(k) &= \frac{\sum_{t=1 \text{ s.t. } O_t = v_k}^{T} \sum_{j=1}^{N} \xi_t(i, j)}{\sum_{t=1}^{T} \sum_{j=1}^{N} \xi_t(i, j)}. \end{aligned} \quad (2)$$

It has been proved that starting from an initial parameter λ, the iterations converge to a maximum likelihood estimate of the HMM, because $P(O|\bar{\lambda}) \geq P(O|\lambda)$ is always satisfied [17].

There are several issues we need to explore before we can use HMM in our estimation scheme. First, the forward-backward re-estimation algorithm typically leads to local, instead of global maxima. In some cases, the optimization surface can be complex and have many local maxima. Therefore, the initial estimate of λ plays an important role in determining whether or not the algorithm will converge to the right parameters. Fortunately, in most of the traces we have analyzed, randomly selected initial values typically result in the algorithm converging to the same point. In other words, the number of local maxima appears to be limited in our case. In [14], the authors observed a similar phenomenon. The second concern about HMM is that the convergence process is often slow and the convergence time varies. The major factors that affect the convergence time include the number of states, the number of observations, and the initial parameters. In our implementation, for a 2-state model with 10,000 observations, convergence can be achieved in less than 5 seconds on a PC with 450MHz Pentium processor and 256MB of memory. We expect the on-line decisions using this estimation scheme to be in the context of path switching or re-routing that are unlikely to be extremely frequent. Hence, we believe that a multi-second convergence time will be adequate.

Another issue with an HMM approach is the actual number of states needed to precisely describe the state of an Internet path. In [14], the authors indicate that most loss traces they studied can be represented by a 2 or 3-state model, while very few exhibit 4-state behaviors. In our study, we choose to use a 2-state model in most cases. This is motivated by the fact that a 2-state model is typically sufficient to accurately estimate the first-order statistics (e.g., loss rate, loss length distribution, etc.) that we are interested in. Furthermore, computational complexity grows rapidly with the increase of the number of states, so that there is strong incentive for using model with less states. In Section III, we investigate further the impact of the number of states in the model on the accuracy of the estimates it generates.

B. Inferring Loss Performance for Application Flows

The validity of the above model is based upon the fact that although the path state is "hidden", it can be statistically conjectured from the loss perceived by a flow. This, however, leads to a model that is dependent on the probing interval, which is not our ultimate goal. Instead, we intend to infer from this model the losses experienced by an arbitrary application flow that will typically sample the path with a different rate compared to the probing flow used to construct the HMM. It is, therefore, necessary to determine how to "translate" the HMM constructed from the probing flow into one that is applicable to a target application flow. For simplicity, we initially assume that both the probing flow and the target flow have constant packet size and packet inter-arrival time. The effect of variable packet size and inter-arrival time will be investigated later in section V.

We also assume that both the target application flow and the probing flow see the same loss probability when sampling the path in state S_i. Again, this is because of our assumption that neither flow affects the path state, so that the loss probability experienced by a flow depends only on the underlying path state. However, the state transition probabilities a_{ij} seen by the two flows will be different, because they are directly tied to the packet inter-arrival time. How the two are related can be explored by considering a continuous time Markov model to describe the evolution of the path state. Assuming such a continuous-time model, we first define its discrete state transition matrix $\mathcal{P}$ as $[a_{ij}]$. Then, we can the find its generator matrix $\mathcal{Q}$ by solving the associated *Kolmogorov Backward Equation* $\mathcal{P}'(t) = \mathcal{Q}\mathcal{P}(t)$, which admits the following solution

$$\mathcal{P} = \exp(\mathcal{Q}t) = \sum_{n=0}^{\infty} \frac{(\mathcal{Q}t)^n}{n!}, \tag{3}$$

where t is the sampling interval (packet inter-arrival time). If the probing flow has a sampling interval of t and a transition probability matrix $\mathcal{P}_1$, while the target flow has a sampling interval of $\frac{t}{2}$ and a transition matrix $\mathcal{P}_2$, then $\mathcal{P}_1 = \exp(\mathcal{Q}t)$ and $\mathcal{P}_2 = \exp\left(\mathcal{Q}\frac{t}{2}\right)$. We therefore have $\mathcal{P}_2 = \sqrt{\mathcal{P}_1}$. Similarly, we can compute the transition matrix $\mathcal{P}_k$ of a target application flow that has sampling interval $\frac{t}{k}$ as

$$\mathcal{P}_k = \mathcal{P}_1^{\frac{1}{k}}. \tag{4}$$

According to Sylvester's theorem, for any positive rational number k, the power of matrix $\mathcal{P}_1$ can be numerically computed using its eigenvalues and eigenvectors. In practice, however, we only consider the simplified case where $k = 2^m, (m = 1, 2, 3, ...)$. Thus, $\mathcal{P}_k$ can be derived by recursively computing the square root of matrix $\mathcal{P}_1$. The square root of a matrix with no eigenvalues on $\mathbb{R}^-$ can be quickly found using, e.g., the Denman Beavers iteration [18].

Following the above analysis, one can infer the loss performance of a target application flow from the HMM constructed using the probing flow. As mentioned earlier, a key assumption

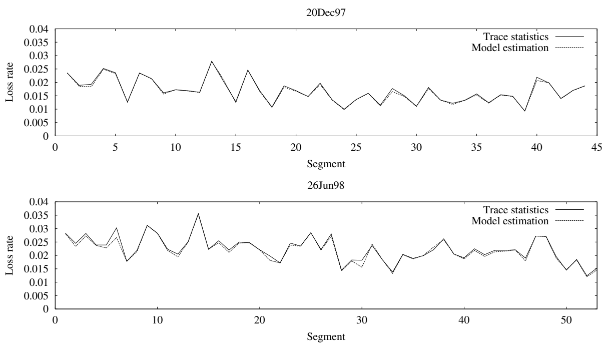


Fig. 2. The actual packet loss rate compared with the results generated by a 2-state HMM: trace *20Dec97* and *26Jun98*.

is that neither the probing flow nor the target flow is a major contributor to congestion on the sampled path. This assumption is key to ensuring that the matrix $\mathcal{Q}$ does not depend on the specific flow and preserving the above simple relation between the discrete state transition matrices of the two flows.

III. Accuracy of the Model

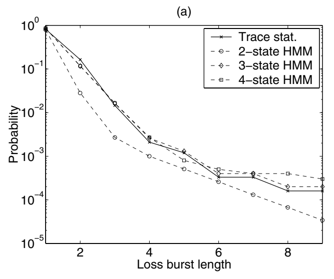
As mentioned in Section I, there have been other models used to describe packet losses, e.g., [3], [19], most of which focused on directly describing the temporal dependency of losses instead of capturing the variation of path state. Representative samples of such models include the Bernoulli model and the Markov model with different orders. The Bernoulli model assumes independent loss probability for each packet, while the Markov model provides for different levels of memory in describing the loss probability of successive packets. Unlike these models, an HMM is a model with parameters that incorporate memory of the entire sampled data. We therefore expect it to yield reasonably accurate estimates even when the number of states used to model the path is limited. In this section, we investigate the accuracy of this model by comparing the estimates it generates to the actual statistics of the loss traces. We are mainly interested in two first-order statistics: *average loss rate* and *loss burstiness*.

A. Average Loss Rate

Average loss rate represents the average number of lost packets out of the transmitted. Given an HMM, this number can be calculated from the steady state probabilities $\pi = [\pi_1, \pi_2, ..., \pi_N]$. Here π_i is the probability of the path being in state i. Then the average loss rate is

$$R = \sum_{i=1}^{N} \pi_i b_i(1). \tag{5}$$

To evaluate the accuracy of the model, we use different loss traces as the input to the HMM estimation algorithm. We use two traces obtained from [3], one is a 2.5-hour trace from Seattle to UMass with a sampling interval of 20ms (*20Dec97*), and the other is a 6-hour trace from Atlanta to UMass with a sampling interval of 40ms (*26Jun98*). We divide the traces



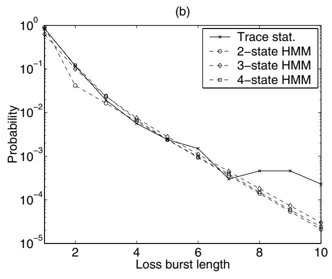


Fig. 3. The loss burst length distribution estimated by HMM compared with the statistics from the original loss traces: (a) *20Dec97* and (b) *21Dec97*.

into segments of 10,000 samples, each of which is used to generate a 2-state HMM. We plot both the measured average loss rate and the values derived from the model in Fig. 2. For both traces, the estimated results and actual values can be seen to match each other well.

B. Loss Burstiness

Another important loss performance metric is loss burstiness, i.e., the number of consecutive packet losses. For some applications (e.g., video or audio), given the same loss rate, variations in loss burstiness can result in dramatic differences in application-level quality [20]. Using HMM, we can compute the burst length distribution of losses from the steady state probability distribution (π) of the path state, the loss probability in each state ($\mathcal{B}$), and the state transition probability matrix ($\mathcal{P}$).

$$P\{x_k = n\} = \frac{\pi(\mathcal{I} - \mathcal{B}(k))(\mathcal{P}\mathcal{B}(k))^n \mathcal{C}(k)}{\pi(\mathcal{I} - \mathcal{B}(k))\mathcal{P}\mathcal{B}(k)\mathbb{I}}, k \in \{0, 1\}, n \geq 1 \tag{6}$$

In the above equation, $P\{x_k = n\}$, $k \in \{0, 1\}$ represents the probability of having a burst (either v_1's or v_0's) of length n; $\mathcal{I}$ is the identity matrix; $\mathcal{P}$ is the state transition probability matrix $[a_{ij}]$; $\mathcal{B}(k) = \text{diag}\{b_i(k)\}$; $\mathcal{C}(k) = [1 - b_0(k), 1 - b_1(k), ..., 1 - b_N(k)]^T$ and $\mathbb{I} = [1, 1, ..., 1]^T$.

We then compare the estimated loss length distribution ($k = 1$) to the actual statistics for two traces: *20Dec97* and *21Dec97*, both from [3]. The results are shown in Fig. 3. It can be seen that the model yields loss length distribution reasonably close to those of the original data traces, especially in the short loss burst region.

To illustrate how the number of states affects the estimation result, we generate 2, 3, and 4-state HMM's and compare their loss length distributions. The results are also plotted in Fig. 3. Notably, 3 or 4-state HMM result in a distribution closer to the statistics of trace *20Dec97*. However, increasing the number of states did not improve the accuracy of the estimates for trace *21Dec97*.

It is worth noting that (7) can also be used to compute the distribution of v_0 burst length, or loss distance [20], simply by letting $k = 0$. This metric is sometimes more convenient for evaluating application quality. Since the distribution of loss distance can be derived from the average loss rate and the loss length distribution, we omit its investigation here.

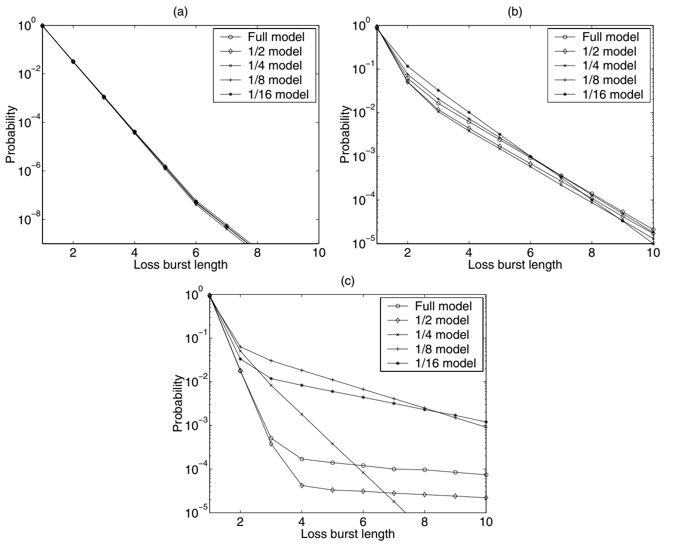


Fig. 4. The estimated loss length distributions from the original trace (full model), and from the trace down-sampled by 2 (1/2 model), by 4 (1/4 model), by 8 (1/8 model) and by 16 (1/16 model): trace (a) *Bernoulli*, (b) *21Dec97*, and (c) *01May03*.

IV. Robustness of the Model

In order to make accurate loss predictions, the HMM is required to properly represent the evolution of path state. Obviously, the more frequently we probe a path, the more accurate is the resulting model. On the other hand, this requirement conflicts with our intent to minimize the probing overhead. In this section, we investigate the trade-off that exists between model accuracy and probing rate. In particular, we investigate the sensitivity of the model to changes in the probing interval and identify key parameters in preserving accuracy. Our study is conducted using both trace analysis and experimental results.

A. Trace-based Analysis

Given a loss trace obtained using a certain probing frequency, we down-sample it by a variable factor L and use the resulting trace to construct the path state model. The down-sampled model and the original model are then compared[1] in terms of their ability to estimate loss rate and loss length distribution. Clearly, depending on the temporal dependency between losses in the original trace and the value of L, differences between these models can be expected.

The loss process of some traces can be modeled reasonably well by a Bernoulli process, i.e., the loss probability of each packet sample is independent. Assuming that the loss process remains stationary, variations in the probing interval should have only a minor impact on the accuracy of the generated model. This is illustrated in Fig.4(a), which presents the loss length distribution of such a trace collected on a path from

[1]Both of them are discrete-time models that are derived by sampling a continuous Markov model with different intervals (t). In order to compare them, we need to use the aforementioned method to convert them into models with the same sampling interval. In this analysis, we systematically transform the down-sampled model using the same sampling interval as used in the original model.

UMN to UPenn. In this example, the reference model was derived from the original trace with sampling interval 40ms, and several down-sampled models (shown as $\frac{1}{L}$ model in the figure) were generated for $L = 2, 4, 8$, and 16. As can be seen in the figure, the log-scale loss length distribution of this trace is almost linear, and as expected down-sampling did not produce much change, even for L as large as 16.

Although for some traces a Bernoulli model is adequate, we find that in many other instances it is not, e.g., when loss patterns consist of a mixture of both independent and bursty (temporally correlated) losses. In those cases, we find that the accuracy of the model is dependent on the probing rate. More specifically, probing should be done at a frequency high enough to capture the temporal dependencies between losses during bursty loss periods. This means that down-sampled models are likely to become more inaccurate as the probing interval increases.

To demonstrate the above observation, we take $160,000$ samples from trace *21Dec97*, in which 3.05% are loss samples. With L being increased from 2 to 16, the average loss rate computed from the down-sampled models are 3.06%, 2.90%, 2.96% and 3.18%, respectively. Correspondingly, the loss length distributions obtained from all down-sampled models are plotted in Fig.4(b). As seen from the figure, down-sampling does not affect the models' accuracy significantly, although slight variations in loss length distribution can be observed as L increases. This means either the time scale of major congestion events recorded by this particular trace is greater than $16 \times 20 = 320$ms, or the loss process itself is not bursty (Bernoulli-like). Since the tail of the log-scale loss length distribution produced by the reference model is approximately linear, the latter is the more likely scenario. Examining the trace verified this conjecture: most losses are isolated events except for a few rare loss bursts that last longer than 320ms.

The third trace we study is *01May03* collected on a path from UMN to UPenn, with an original probing interval of 20ms. For this trace, the loss length distribution derived from the 1/2 model remains close to that of the original model. However, this does not extend to the cases of down-sampling by a factor $L \geq 4$, where significant differences in loss length distribution can be observed (see Fig.4(c)). To better understand the source of this inaccuracy, it is helpful to look at the HMMs generated by the down-sampled traces. The HMM developed from the original trace has the following parameters:

$$[b_i(1)] = \left[\begin{array}{cc} 0.9952 & 0.0184 \end{array} \right], \mathcal{P} = \left[\begin{array}{cc} 0.8801 & 0.1199 \\ 0.0006 & 0.9994 \end{array} \right]$$

This model captures two distinct states on the underlying path. One is associated with a very high loss probability $b_1(1) = 0.9952$, and the other associated with a relatively low loss probability of $b_2(1) = 0.0184$. Furthermore, the probability of staying in the high loss state is relatively high, i.e., 0.8801, meaning that the loss process is rather bursty. When the trace is down-sampled by factor 4 (i.e., the actual probing interval is 80ms), the model parameters become as follow:

$$[b_i(1)] = \left[\begin{array}{cc} 0.2185 & 0.0174 \end{array} \right], \mathcal{P} = \left[\begin{array}{cc} 0.9964 & 0.0036 \\ 0.0002 & 0.9998 \end{array} \right]$$

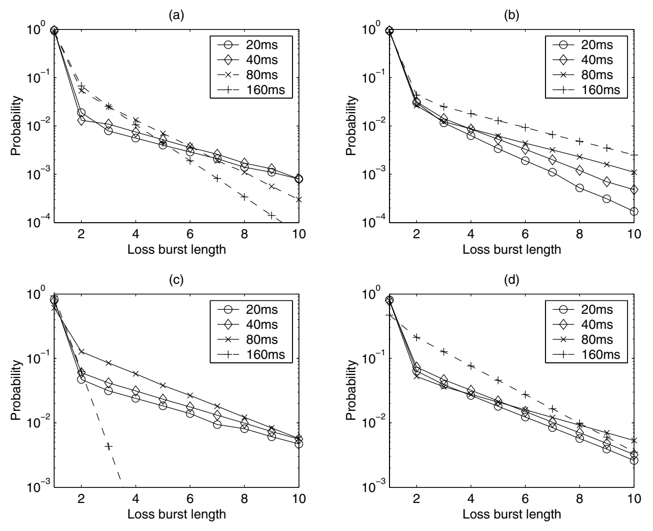


Fig. 5. The loss length distributions estimated by the probing flows with sampling interval 20ms, 40ms, 80ms, and 160ms, respectively. The sampled paths are: (a) from UMN to UPenn, (b) from UPenn to UMN, (c) from UMass to UPenn, and (d) from UPenn to UMass.

in which the distinction between the two states has diminished. The high loss state now has a lower loss probability, and has a duration that is closer to that of the low loss state. The likely explanation is that the gap between subsequent packet samples now extends beyond the duration of the initial loss bursts. In fact, by examining the original trace we find that most loss events in this trace last less than 10 packets (200ms). Thus, this is indeed a typical case that the down-sampling process reduces the correlation between consecutive losses, and when L becomes larger than the time scale of congestion events, the inferred loss model becomes similar to a Bernoulli model rather than a Markov model with 2 different states. As shown in Fig.4(c), the log-scale loss burst length distribution of the 1/4 model is almost linear. It is also worth noting that as L becomes even larger (8 and 16), the loss length distribution starts exhibiting some renewed burstiness, although never close to that of the original model. This is because the probing interval is now so large that consecutive samples may fall in different congestion events. As a consequence, although loss in the down-sampled trace seems "bursty", it does not represent the same loss process as recorded in the original trace.

In some rare cases, we also observed that the distinction between states could be enlarged by down-sampling. This happens when the trace consists of several major loss bursts (longer than the down-sampling factor L) accompanied by a few isolated losses. The long loss bursts survive the down-sampling process, while the other losses are mostly eliminated. As a result, the down-sampling magnifies the initial burstiness of losses, and results in two dramatically different path states: one that introduces very few or no losses, while the other results in long loss bursts.

B. Experimental Validation

To further study the sensitivity of our model to variations in the probing interval, we carry out experiments by initiating simultaneous probing flows along the same path but with different probing periods. The HMMs developed by each probing flow are then compared. Two paths between UMN and UPenn are studied first. The tested path from UMN to UPenn goes through Internet2, while the reverse path is set up through a commercial ISP (Cogent).

On the UMN-to-UPenn path, we set up four probing flows with packet size 100 bytes and sampling interval 20ms, 40ms, 80ms and 160ms, respectively. In this experiment, the loss rates estimated by model are 1.21%, 1.09%, 1.11%, and 1.13%, corresponding to the 4 flows with probing intervals from 20ms to 160ms. The loss length distributions for all flows are plotted in Fig. 5(a). The distributions are close to each other for probing intervals of 20ms and 40ms, but noticeable differences emerge when the probing interval is increased to 80ms or 160ms. This observation coincides with the results obtained from trace *01May03* in the previous analysis.

Similarly, we initiate these probing flows on the UPenn-to-UMN path. In these tests, the loss rates estimated by the probing flows are 0.85% (20ms), 0.93% (40ms), 0.91% (80ms), and 1.02% (160ms). The loss burstiness estimated by an 80ms probing flow is closer to that of the 20ms probing flow, compared with the results of the first path. However, as shown in Fig. 5(b), the probing flow with interval 160ms still significantly differs from the others and is unable to generate a good estimate of the loss burstiness.

We also repeat the above experiments on a non-Internet2 path from UMass to UPenn, and vice-versa. It turns out that in all these tests, the estimate of loss rate is not sensitive to changes in the probing interval. For example, in two runs of experiments on 05/17/2003, the estimated loss rates are 1.70%/1.65%/1.59%/1.51% (UMass to UPenn) and 1.84%/1.70%/1.91%/1.75% (UPenn to UMass), respectively. As far as loss burstiness is concerned, a probing interval of 40ms gives a rather stable estimation in both experiments. The results are shown in Fig. 5(c) and (d).

The above experimental results confirm that when the probing interval exceeds a certain threshold, the HMM cannot properly capture the variations of path state, and therefore be an accurate estimator of performance. This threshold is determined by the time scale of queue dynamics and congestion events, which is in turn dependent on the speed and buffer size of the bottleneck link. However, from all the experiments we have conducted, including others not presented here due to lack of space, a probing interval of 40ms appears adequate for the kind of network configurations we are considering. Hence, the resulting HMMs should allow reasonably accurate inference of performance for different applications. In the next section, we test this hypothesis by verifying the inference accuracy of our approach for a range of target application flows.

V. Inferring Application Performance

As introduced in Section II, our approach to inferring the performance of a target application flow assumes that the packets of both the probing flow and the target flow are identical and periodically transmitted, and packet spacing is their only difference. Clearly, this is too restrictive an assumption as application flows can have a much broader range of traffic patterns. For example, probes are often kept as small as possible to minimize the resulting overhead, while no such constraints exist on applications. Meanwhile, applications can generate packet sequences that are far from periodic. The challenge is that in the absence of a well-defined traffic pattern, it is impossible to formulate a precise mathematical relation between the channel model derived from probing results and the one an application flow would experience[2]. Fortunately, many (real-time) applications of interests can be well approximated by either a pseudo-periodic flow with limited randomization in packet spacing, or as an ON-OFF flow with alternating ON and OFF periods. In this section, we investigate how good a job our approach does in inferring the performance seen by such application flows.

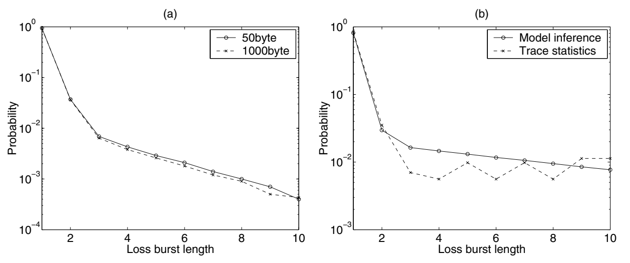


Fig. 6. The effect of packet size on the performance of inferring loss length distribution: (a) two flows with different packet size; (b) a probing flow with fixed packet size and a target flow with variable packet size.

A. The Effect of Packet Size

We conduct the following experiments to study the effect of packet size on the accuracy of performance inference. In the first experiment, we initiated two periodic flows, on a path from UPenn to UMass on 04/03/2003, with the same sampling interval of 40ms, but different packet size 50bytes and 1000bytes. An HMM is constructed from the loss samples recorded by each flow. The average loss rates seen by these two flows are 1.08% and 1.17%, respectively. And as shown in Fig. 6(a), the loss length distributions estimated by the two models are almost identical.

In the second experiment, we study two periodic flows: a probing flow with probe size 50bytes and probing interval 40ms, and a target flow with packet size uniformly distributed in the range of $500-1500$bytes, and a sampling interval of 20ms. The loss traces were collected on a path between UMN and UPenn on 06/20/2003. From the probing trace, we derive an HMM, which estimates loss rate as 2.27%. The target flow experiences a loss rate of 2.35%. In Fig. 6(b), the loss length distribution inferred from the probing trace is compared to the statistics of the target flow. Clearly, the effect of packet size on the accuracy of performance inference is again negligible.

The implications of the above findings are two-fold. On one hand, it means that decreasing probe size should not affect the accuracy of the resulting HMM. Hence, we can reduce the

[2]For example, there is no easy way to infer the loss process seen by a rate-adaptive flow such as TCP.

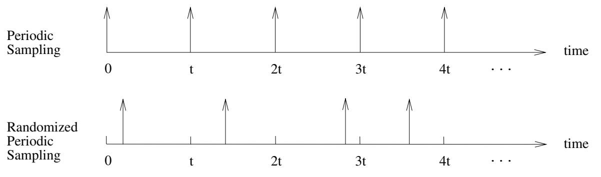


Fig. 7. Examples of periodic flow and randomized periodic flow, both with sampling period t.

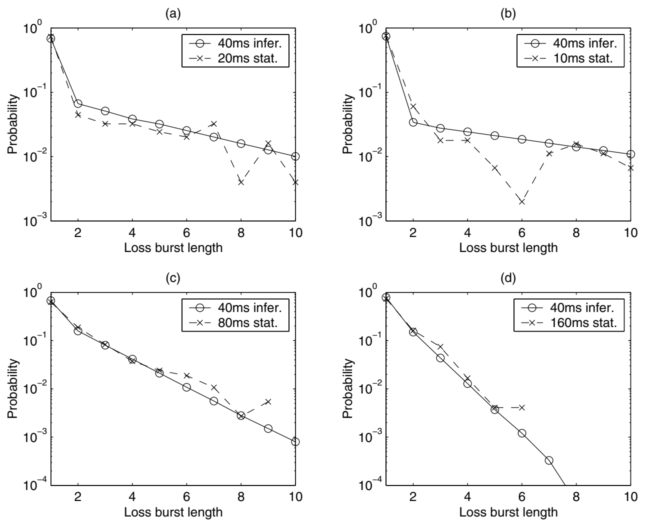


Fig. 8. Inferring the loss length distribution of periodic flow with sampling interval (a) 20ms, (b) 10ms, (c) 80ms, and (d) 160ms.

probing overhead by using smaller probes. For instance, with a probe size of 50bytes and a probing interval of 40ms, the traffic generated by a single probe stream is only 10Kbit/s. On the other hand, it also shows that our inference approach is applicable to application flows with variable packet size. This also partially validates our assumption that path state and sampling patterns are the only deterministic factors that affect the loss performance of a flow.

B. Inferring the Performance of a Periodic Flow

As discussed earlier, the sampling pattern of the application flow has a significant effect on the accuracy of inference. The simplest case is a periodic application flow that deterministically samples the path every t units of time, as shown in Fig. 7. Note that this is a different perspective from that of the previous section where we investigated the accuracy of the HMM itself as a function of the probing period. Here, we assume that the constructed model[3] is reasonably accurate, and our focus is on its ability to accurately predict the losses experienced by flows with a different sampling period.

To show the validity of our inference approach, we conducted two experiments on the UMN to UPenn path on 06/25/2003. In the first experiment, we set up two target periodic flows and the probing flow. The sampling period t of the target flows are 20ms and 10ms, which are smaller than the probing interval of 40ms. In the second experiment, their

[3]From this point onward, unless stated otherwise, we always use a probing flow with a packet size of 50bytes and a sampling interval of 40ms.

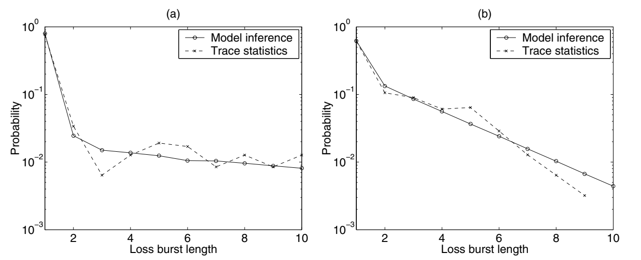


Fig. 9. Inferring the loss length distribution experienced by randomized periodic flows with sampling interval (a) $t = 20ms$ and (b) $t = 80ms$.

sampling periods are 80ms and 160ms, while the configuration of the probing flow remains unchanged. The inferred loss rate in the first experiment is 1.44%, while the actual values are 1.53% and 1.51%. In the second experiment, these values are 1.70% (inferred), against 1.66% (80ms) and 1.69% (160ms). Fig. 8 shows the inferred and measured loss length distributions of all flows. The results show that the model is capable of accurately inferring the performance of periodic flows across a broad range of sampling periods that differ from that of the probing flow. Note that deviations such as the one observed in Fig. 8(b) are unavoidable, as we are dealing with a single sample path that can occasionally deviate from the expected long term statistics.

C. Inferring the Performance of a Randomized Periodic Flow

A randomized periodic flow is similar to a periodic flow, except that rather than sampling the path at the same instant in each period, it samples the path at a random instant in each t units of time (see also Fig. 7). The ability to accurately estimate the performance of randomized periodic flows is of practical importance for several reasons. First, even constant rate applications will often not generate packets in a perfectly periodic manner. Fluctuations are introduced by many factors. For example, it is difficult for a server to transmit packets precisely spaced to each client because of the granularity of system clock and the scheduling of processor time. Similarly, media flows generated by a CBR encoder often exhibit a certain level of rate fluctuation. In addition, even if the application flow was strictly periodic, interactions with other flows will typically perturb its original period and introduce random fluctuations as packets traverse the network. A pseudo-periodic model is thus well suited to capture the fluctuations of most constant rate applications.

To understand how such irregular packet spacing would affect the performance of inference, we set up a probing flow and a randomized periodic application flow, on the path from UMN to UPenn on 05/08/2003. In our first experiment, the randomized periodic flow had a sampling period of 20ms and its sampling time uniformly distributed within each 20ms interval. The experiment was run for 800 seconds. The inferred and the actual loss rates were 4.27% and 4.20%, respectively. The comparison of the inferred loss burst length distribution and the actual statistics is shown in Fig. 9(a). In the second experiment, we changed the period of the application flow to 80ms. The inferred loss rate was then 3.54%, and the corre-

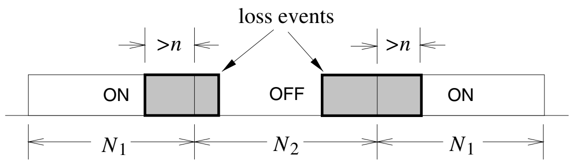


Fig. 10. The example of ON-OFF sampling: loss burst length depends on not only the duration of loss events, but also the position of where it starts.

sponding actual value was 3.42%. The loss length distributions are shown in Fig. 9(b).

As shown in Fig. 9, the introduction of randomization did not significantly affect the accuracy of the model. We expect this to hold in general, except in the case of very large periods where randomization introduces variability that the model is unable to predict. Such scenarios of very low rate applications are, however, unlikely to be common or important in practice.

D. Inferring the Performance of an ON-OFF Flow

An ON-OFF flow generates packets periodically in the ON state and stays silent in the OFF state. It is also representative of a wide range of real-time traffic sources. For example, a packet-based voice flow is typically composed of active and inactive periods, corresponding to the activity of human speech. Moreover, in some video servers [21], streaming is implemented in bursts of packets (typically with a burst duration of hundreds of milliseconds), in order to lower the overhead on the server.

We consider a flow with ON-OFF ratio $N_1 : N_2$ (in number of sampling intervals). The first step in inferring losses experienced by such an ON-OFF flow, is to derive from the HMM constructed from the probing results an HMM that assumes a sampling period equal to the packet spacing during its ON period. Once this HMM is obtained, we use it to derive estimates for the loss rate and burstiness of the ON-OFF flow. Since the path state variation is a random process independent of the duration of ON and OFF periods, the average loss rate of the ON-OFF flow can again be estimated from Eq. (5). The computation of loss length distribution is more complicated. In our derivation, we define a loss burst to be consecutive losses *within a single ON period*. This not only simplifies the mathematical derivation, but is also more meaningful from an application perspective. For example, the application content contained in a given packet burst, e.g., a video frame or a talk spurt, is often self-contained. Therefore, considering only loss bursts in each ON period is not unreasonable.

We first study the complementary cumulative distribution function (CCDF) of loss length, which can be computed in a manner similar to how Eq. (6) was derived.

$$P\{x_k \geq n\} = \frac{P_k^n}{P_k^1} = \frac{\pi(\mathcal{I} - \mathcal{B}(k))(\mathcal{P}\mathcal{B}(k))^n \mathbb{I}}{\pi(\mathcal{I} - \mathcal{B}(k))\mathcal{P}\mathcal{B}(k)\mathbb{I}} \qquad (7)$$

for periodically sampled loss sequence. Here, P_k^n denotes the probability of having a burst (either loss or loss-free) longer than $n-1$, while P_k^1 denotes the probability of having a burst of any length in the sequence.

For an ON-OFF flow, the loss event can start either in the ON period or in the OFF period. If it starts in the ON period, the loss burst can be longer than n only if its starting point is more than n samples prior to the end of this ON period (see Fig. 10). If the loss event starts in the OFF period, the loss burst can be longer than n only if the end of the loss event extends more than n samples past the beginning of the next ON period. The same argument can be applied to loss-free periods. Assuming the starting time of a loss event is random, the probability of having a burst of length $\geq n$ in an ON-OFF sampling sequence can be approximated as

$$\tilde{P}_k^n = \frac{N_1 - n + 1}{N_1 + N_2} P_k^n + \sum_{s=1}^{N_2} \frac{1}{N_1 + N_2} P_k^{n+s}. \qquad (8)$$

From Eq. (8), we can derive the CCDF of the burst length distribution as

$$\tilde{P}\{x_k \geq n\} = \frac{\tilde{P}_k^n}{\tilde{P}_k^1}. \qquad (9)$$

Note that as alluded to earlier, we do not consider cases where a loss burst extends over several ON periods. In general, the probability of having such long loss bursts is very small, so that they can safely be ignored in the above computation.

To verify the validity and accuracy of the above expressions, we first collected loss traces of an ON-OFF flow with ratio 20:20, together with a probing flow that was used to construct the HMM of the path. Subsequently, we conducted a similar experiment but for an ON-OFF flow with ON-OFF ratio of 15:45. All traces were collected on a path from UMass to UPenn on 06/23/2002 and had a duration of 800 seconds. All the tested flows have sampling interval 20ms during their ON periods. The inferred/actual loss rates for the two tests are 3.54%/3.42% and 4.27%/4.20%, respectively. From the loss length distributions shown in Fig. 11, we can observe that the inference based on periodic probing is capable of providing estimates reasonably close to the actual statistics of the ON-OFF flows. But the estimation is not as accurate as in the cases studied before. The estimation error is mainly caused by two factors. First, for an ON-OFF flow, we are only interested in the loss events happening in its ON periods. Compared with the previous cases, this means that we are using the model to infer statistics based on fewer loss event samples, which could result in a less accurate estimation. Second, in deriving the loss length distribution we assume a random starting time for loss events, which could introduce additional inaccuracy in the estimation. As a result and in particular if the observation sequence is short, the estimations produced by Eqs. (8) and (9) can exhibit non-trivial deviations from the actual statistics. In the above example, the two ON-OFF traces we used contain 10,000 and 5,000 packet samples during the ON periods, but only 195 and 228 corresponding loss events, respectively. Thus, the inaccuracy of the estimation observed in Fig. 11 is not unexpected.

E. A Video Streaming Flow Example

All the previous examples used *synthetic* applications flows, and were aimed at assessing the accuracy of our approach for common models representing real-time applications. In this

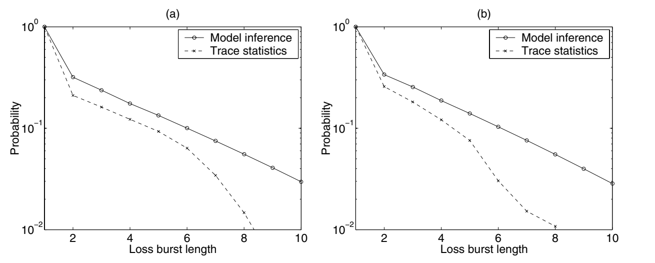


Fig. 11. Comparison of the inferred CCDF (based on periodic probing) of loss length distribution and the actual statistics from the ON-OFF sampling trace with ON-OFF ratio (a) 20:20 and (b) 15:45.

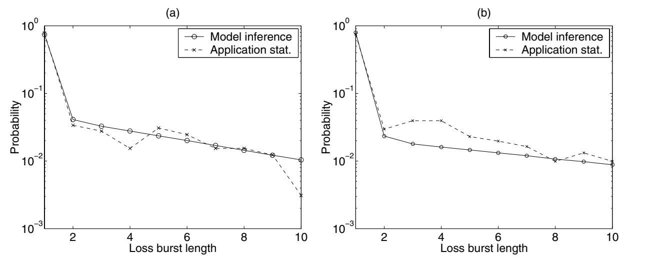


Fig. 12. Inferring the loss length distribution of video streaming flow with equivalent sampling interval 10ms. The encoding bit rates of the video sources are (a) 150Kbit/s and (b) 500Kbit/s.

section, we extend the investigation to a *real* application flow. Specifically, we set up a probing flow and a target flow in the same manner as before, but this time the latter is generated by a video streaming software. Two sets of experiments were carried out separately on the UMN-to-UPenn path on 06/27/2003, with the difference that the first video flow had an average bit rate of 150Kbit/s, while the second had a bit rate of 500Kbit/s. The traffic generated by this video software has features of both variable packet size (one major reason is that the data in a slice of an MPEG frame is designed to be contained in one or several packets, while no packet should cross the boundary of two slices) and randomized periodic sampling (the MPEG video source is CBR-encoded, and roughly speaking, the number of packets generated in a certain time interval can be considered as constant).

In these two experiments, the loss rates estimated by the probing flow are 1.27% and 1.86%, respectively, which are very close to the values collected at the video client (1.24% and 1.84%). We also infer the loss burstiness using 10ms as the equivalent sampling interval for both video flows (the major difference in bit rate for these two flows turns out to be the result of different packet sizes). As shown in Fig. 12, the model performs well in estimating the application flow's loss length distribution.

VI. Predictability of Loss

In the previous sections, we have introduced a method that relies on an HMM constructed from the loss information gathered by probes, with which we then infer the performance different applications would experience. This is still one step short of our ultimate goal, which calls for using the inferred performance as an input to a control process that decides whether to switch application flows to an alternative path. The missing step is to study whether the performance estimates generated by the model using past statistics allow us to accurately predict future performance, as an application switched onto the path would experience it. In this section, we introduce an extension to the basic model that addresses this issue by allowing relatively accurate loss *predictions* over different time scales.

A. Layered HMM

Due to the bursty nature of Internet traffic, the time series of network losses represent a fairly complex random process, in which stationarity and non-stationarity coexist. The level of stationarity often varies with the observation time scale and the level of congestion on the path. For example, in [7], the majority of the loss processes exhibit some level of stationarity when the observation is restricted to traces with loss rate less than 1%, while such stationarity decreases dramatically when it comes to traces with loss rate $\geq 1\%$. Periodicity also exists in the loss process. Such periodicity could for example come from the diurnal behavior of network users. In [6], the author also observed periodicities because of the timer synchronization between routers. Based on such observations, we believe that it is difficult to capture loss variations across a broad range of time scales using a simple "flat" model. However, if we use different models for different time scales, we may be able to construct a simple, yet reasonably accurate, compound or layered model.

Specifically, we consider two time scales or layers. At the coarse time scale associated with observation interval T, we define a number of loss states or phases. Those phases can be classified as "no loss", "minor loss", "tolerable loss", "serious loss", "very serious loss", and "unacceptable loss"[4], which quantitatively correspond to loss rates in the following ranges: 0, $0-0.5\%$, $0.5\%-1\%$, $1\%-5\%$, $5\%-10\%$, and $>10\%$. Within each phase an HMM is constructed from the observed loss samples. In other words, the HMM parameters are derived from observations in the corresponding phase. This process is structured as a dynamic process with the system parameters being continuously refined based on observations.

B. Parameter Estimation

At the coarse time scale, what we use to describe transitions between phases is a first-order Markov model, which is characterized by the state transition probabilities

$$\phi_{ij} = P\{X_{t+1} = x_j | X_t = x_i\}, \qquad i, j \in \{0, 1, ..., 5\}$$

where X_t is the phase of the loss process in interval t. The transition probabilities can be estimated on-line by counting in the observed time series the number of times that phase i is followed by phase j (n_{ij}), and the number of times the loss process is in phase i (n_i), i.e.,

$$\phi_{ij} = \frac{n_{ij}}{n_i}. \tag{10}$$

[4]These qualitative notations are defined roughly in accordance to the loss tolerant capability of real-time applications, such as streaming video.

The values of ϕ_{ij}'s are adjusted after each observation interval T. Then given that the current phase is $X_t = x_i$, the next phase X_{t+1} can be predicted as $X_{t+1} = x_k$, using a maximum likelihood predictor [22]:

$$k = \arg\max_j \phi_{ij}. \tag{11}$$

The purpose of constructing an HMM in each phase is to derive estimates for the loss rate and burstiness given the phase prediction. In keeping with standard estimation techniques, we use past observations within the corresponding phase to derive the model parameters. A window of size W, i.e., the number of observation intervals used in constructing the HMM for a particular phase, governs the amount of memory or history used by the estimator. Too long a memory will unnecessarily increase the length of the training sequence for the HMM, which results in long convergence time, while potentially preserving irrelevant data if the statistical behavior of a given phase has evolved over time. On the other hand, too short a memory cannot provide sufficient training data for the HMM, and therefore leads to an "over-fitted" model that could yield poor accuracy. Our experience shows that only limited amount of history is necessary, as the dependency of losses observed in different intervals decreases dramatically when increasing the time lag between these intervals. The impact of W on the accuracy of prediction is investigated further below.

Assuming that the loss processes in W observation intervals of a particular phase are represented by the same HMM, we consider the problem as forming an HMM from multiple observation sequences

$$O = [O^{(1)}, O^{(2)}, ..., O^{(W)}].$$

Thus the goal of the re-estimation procedure becomes adjusting λ to maximize

$$P(O|\lambda) = \prod_{k=1}^{W} P(O^{(k)}|\lambda) = \prod_{k=1}^{W} P_k$$

Consequently, the re-estimation formulas of a_{ij} and $b_j(l)$ should be modified as [17]:

$$a_{ij} = \frac{\sum_{k=1}^{W} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) a_{ij} b_j(O_{t+1}^{(k)}) \beta_{t+1}^k(j)}{\sum_{k=1}^{W} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)}, \tag{12}$$

$$b_j(l) = \frac{\sum_{k=1}^{W} \frac{1}{P_k} \sum_{t=1 s.t. O_t = v_t}^{T_k-1} \alpha_t^k(j) \beta_t^k(j)}{\sum_{k=1}^{W} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(j) \beta_t^k(j)}. \tag{13}$$

Here $\alpha_t^k(i)$ and $\beta_t^k(i)$ are the forward and backward variables corresponding to observation sequence k, and P_k is the probability of sequence k being observed with current parameters λ. $\psi_i^{(k)}$ is estimated based only on sequence $O^{(k)}$ and the related re-estimation procedure remains intact. The data in window W is updated after each interval T. So that the HMM for each phase keeps "learning" from the most updated observations, while "forgetting" outdated ones.

TABLE I
THE PERFORMANCE OF PREDICTION WITH DIFFERENT W

W	2	3	4	5	6
Error in R	0.55%	0.49%	0.38%	0.44%	0.45%
Error in P_I	33.0%	36.7%	31.3%	35.3%	30.9%

C. Prediction Accuracy

In this section, we evaluate the ability of the approach we just described to accurately predict the evolution of path state and, therefore, the losses that an application flow would experience. We implemented the probing-prediction scheme as a UDP sender-receiver pair. The sender transmits probes periodically with a pre-configured interval. The estimation/prediction mechanism is integrated in the receiver. The loss sequences are first collected by the receiver. Then, after each observation interval T, the most likely phase in the next observation interval is predicted, as well as its loss rate and loss length distribution. After finishing collecting the loss information in each time interval T, the HMM parameters of the corresponding phase are updated.

We ran the probing program on the UMN-to-UPenn path for several weeks and selected two traces with relatively high loss rate for our analysis. One is a 4-day trace starting from 03/14/2003 with loss rate 1.78%, and the other is a 3-day trace starting from 04/28/2003 with loss rate 1.25%. Both traces show some diurnal patterns, namely the loss rate and loss burstiness vary in accordance with the time of a day. To demonstrate the performance of loss burstiness prediction, we used the percentage of isolated loss events (P_I), which is defined as the percentage of loss bursts shorter than 3 packets.

The window size W is the first parameter that needs to be carefully tuned, because it affects not only the accuracy of the estimator, but also the computational load during the prediction process. We measure the overall performance for the predictors with W = 2, 4, and 6, using the aforementioned traces. The probing interval was 80ms, while the observation interval was set to T = 10min to yield enough samples in each interval for training the model. The prediction error is measured as $|\tilde{f} - f|$, given that f is the target parameter, and $\tilde{f}$ is the predicted value. Table I shows the overall errors for predicting the loss rate R and the percentage of isolated loss events P_I among all loss events. It is notable that $W = 4$ (i.e., a memory of 40 minutes) gives the best overall prediction performance. Further increasing this value does not yield much improvement, while decreasing it may result in worse prediction in either loss rate or burstiness.

Using window size $W = 4$, the receiver predicts both the loss rate and the loss burstiness in the next interval T. We scatterplot the prediction results against actual statistics in Fig. 13. Fig. 13(a) shows the performance of loss rate prediction, while Fig. 13(b) provides a similar comparison for the percentage of isolated losses. From these figures, we see that the layered model can provide reasonably good predictions, especially in terms of loss rate. However, the model tends to underestimate the loss burstiness (as shown in Fig. 13(b), the predicted percentage of isolated loss is

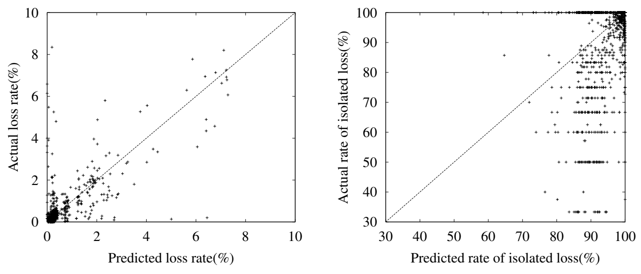


Fig. 13. The performance of predicting loss using the layered HMM: (a) average loss rate, and (b) loss burstiness.

generally lower than the actual value). It is worth noting that there are a few sample points in Fig. 13(b) where the actual value of P_I is 100%, while the predicted values are much lower. These points correspond to intervals that have very low loss rate (typically less than 0.1%).

VII. Conclusion

In this paper, we have introduced an approach for on-line estimation of end-to-end loss performance of network paths. This approach models the path loss process using a continuous-time hidden Markov model and constructs the model based on the loss trace collected using an active probing scheme. From the resulting model, we show that it is possible to infer the loss performance that an application flow routed over the path would experience. We investigated both the accuracy of the proposed approach and the associated trade-off in the model's complexity, with an eye on ensuring its suitability for on-line estimation. This included guidelines for minimizing the overhead of probe traffic, while keeping estimation accuracy at a meaningful level. We also extended the basic model to a two-tier layered model, attempting to not only give accurate on-line estimations, but to also incorporate the ability to predict the evolution of path performance over time. The results in the paper show that the proposed approach appears to hold some promises in timely and accurately estimating path performance and quality, as seen by application flows.

We believe that there are several potential applications for such an on-line estimation method. First, it can serve as an end-to-end measurement facility to monitor loss performance of network paths as it affects different applications. The measurement results can be used to trigger application-specific alarms. Second, it can be used as input to the admission control that might be performed by an application gateway responsible for certain resources. Third, it can also be integrated into a dynamic path selection solution in a multihoming or overlay environment, where the information it provides can be used to decide when to switch the traffic of various applications to an alternative path. The last aspect is the one that we are currently investigating in the context of a wide-area testbed aimed at serving real-time applications.

VIII. Acknowledgments

The authors would like to acknowledge the assistance of their colleagues at University of Minnesota (UMN), University of Massachusetts (UMass) and University of Pennsylvania (UPenn), in developing the testbed infrastructure that was used to perform many of the measurements studied in the paper.

References

[1] S. Aavage, A. Collins, and E. Hoffman, "The end-to-end effects of Internet path selection," in *Proc. ACM SIGCOMM*, Cambridge, MA, August 1999.

[2] D. G. Anderson, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. 18th ACM SOSP*, Banff, Canada, October 2001.

[3] M. Yajnik, S. Moon, J. Kurose, and D. Towsley, "Measurement and modeling of the temporal dependence in packet loss," in *Proc. IEEE INFOCOM*, New York, March 1999.

[4] V. Paxon, "End-to-end Internet packet dynamics," *IEEE/ACM Trans. Networking*, vol. 7, no. 3, pp. 277–292, June 1999.

[5] J. Bolot, "End-to-end packet delay and loss behavior in the Internet," in *Proc. ACM SIGCOMM*, San Francisco, CA, September 1993.

[6] Y. Zhang, V. Paxson, and S. Shenker, "The stationarity of Internet path properties: Routing, loss and throughput," in *ACIRI Technical Report*, 2000.

[7] Y. Zhang, N. Duffield, V. Paxon, and S. Shenker, "On the constancy of Internet path properties," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, San Francisco, CA, November 2001.

[8] V. Jacobson, "Pathchar - a tool to infer characteristics of Internet paths," in *ftp://ftp.ee.lbl.gov/pathchar/*, 1997.

[9] M. Jain and C. Dovrolis, "End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput," in *Proc. ACM SIGCOMM*, Pittsburgh, PA, August 2002.

[10] G. He and J. C. Hou, "On exploiting long range dependence of network traffic in measuring cross traffic on an end-to-end basis," in *Proc. IEEE INFOCOM*, San Francisco, CA, March 2003.

[11] K. G. Anagnostakis, M. Greenwald, and R. Ryger, "Cing: Measuring network-internal delay using only existing infrastructure," in *Proc. IEEE INFOCOM*, San Francisco, CA, March 2003.

[12] R. Caceres, N. G. Duffield, J. Horowitz, D. Towsley, and T. Bu, "Multicast-based inference of network-internal characteristics: Accuracy of packet loss estimation," in *Proc. IEEE INFOCOM*, New York, March 1999.

[13] N. G. Duffield, F. Lo Presti, V. Paxson, and D. Towsley, "Inferring link loss using striped unicast probes," in *Proc. IEEE INFOCOM*, Anchorage, AL, April 2001.

[14] K. Salamatian and S. Vaton, "Hidden markov modeling for network communication channels," in *Proc. ACM SIGMETRICS*, Cambridge, MA, June 2001.

[15] W. Wei, B. Wang, and D. Towsley, "Continuous-time hidden markov models for network performance evaluation," in *Performance Evaluation*, 2002, vol. 49, pp. 129–146.

[16] P. Ji, B. Liu, D. Towsley, and J. Kurose, "Modeling frame-level errors in GSM wireless channels," in *Special Issue of Performance Evaluation, in press*.

[17] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in *Proc. the IEEE*, February 1989.

[18] E. D. Denman and Jr. A. N. Beavers, "The matrix sign function and computations in systems," *Applied Mathematics and Computation*, vol. 2:63-64, 1976.

[19] H. Sanneck and G. Carle, "A framework model for packet loss metrics based on loss runlengths," in *Proc. Multimedia Computing and Networking*, San Jose, CA, January 2000.

[20] W. Jiang and H. Schulzrinne, "Modeling of packet loss and delay and their effect on real-time multimedia service quality," in *Proc. NOSSDAV*, Chapel Hill, NC, June 2000.

[21] D. Loguinov and H. Radha, "Measurement study of low-bitrate Internet video streaming," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, San Francisco, CA, November 2001.

[22] G. Dangelmayr, S. Gadaleta, D. Hundley, and M. Kirby, "Time series prediction by estimating markov probabilities through topology preserving maps," in *Proc. SPIE 44th Annual Meeting*, Denver, CO, July 1999.