Computer Networks 70 (2014) 384–399

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

A two-level Markov model for packet loss in UDP/IP-based real-time video applications targeting residential users

Martin Ellis [a], Dimitrios P. Pezaros [a], Theodore Kypraios [b], Colin Perkins [a,*]

[a] School of Computing Science, University of Glasgow, Glasgow G12 8QQ, UK
[b] School of Mathematical Sciences, University of Nottingham, Nottingham NG7 2RD, UK

ARTICLE INFO

Article history:
Received 21 November 2013
Received in revised form 26 April 2014
Accepted 26 May 2014
Available online 24 June 2014

Keywords:
Packet loss models
Markov models
Video streaming
IPTV
Residential broadband
FEC

ABSTRACT

The packet loss characteristics of Internet paths that include residential broadband links are not well understood, and there are no good models for their behaviour. This complicates the design of real-time video applications targeting home users, since it is difficult to choose appropriate error correction and concealment algorithms without a good model for the types of loss observed. Using measurements of residential broadband networks in the UK and Finland, we show that existing models for packet loss, such as the Gilbert model and simple hidden Markov models, do not effectively model the loss patterns seen in this environment. We present a new two-level Markov model for packet loss that can more accurately describe the characteristics of these links, and quantify the effectiveness of this model. We demonstrate that our new packet loss model allows for improved application design, by using it to model the performance of forward error correction on such links.

1. Introduction

Studies show that video comprises more than 50% of global consumer Internet traffic, and it is predicted that this will rise to 80–90% by 2017 [1]. This video traffic can be split into two categories: on-demand streaming of near real-time content, primarily movies and catch-up TV (e.g., NetFlix, BBC iPlayer), and real-time traffic in the form of live TV broadcasts and video conferencing. Much has been written about near real-time applications. These typically make use of dynamic adaptive streaming over HTTP (DASH) [2] and accept TCP congestion control and in-order reliable delivery, with the multi-second play-out delays this entails, in return for being able to build on the web infrastructure of content delivery networks. By way of contrast, real-time applications require shorter play-out buffers, typically on the order of tens of milliseconds, and so eschew the DASH model and instead build custom UDP/IP transport protocols and infrastructure. By accepting this complexity, real-time applications can gain visibility into the timing and reliability of the packet delivery process, allowing them to optimise the transport to reduce latency.

In order to design effective low-latency real-time transport protocols, new forward error correction (FEC) schemes, or improved packet loss concealment algorithms, we need to create an accurate model of the packet delivery process. Such a model will allow analytic and simulation studies to evaluate the performance of new applications and services in advance of full-scale deployments, testing the effectiveness of the design. Previous studies have used Markov models, such as the classical Gilbert model [3], to evaluate video streaming applications [4] and forward error correction [5]. These models are popular due to their relative simplicity, and because they are easy to implement and understand. Using measurements of packet loss between academic sites, more complex Markov models [6,7] and hidden Markov models [8,9] have also been

* Corresponding author. Tel.: +44 141 330 2525.
E-mail address: csp@csperkins.org (C. Perkins).

http://dx.doi.org/10.1016/j.comnet.2014.05.013
1389-1286/

proposed to describe Internet packet loss. However, the accuracy of these models for characterising packet loss on residential broadband networks remains poorly understood. There is some evidence that packet loss characteristics of residential networks are different to those on academic networks [10], so it is important to study the accuracy of the models in this context.

In this paper, we use measurements of packet loss from real-time video on residential broadband links [10] to show that well-known Markov models of network packet loss do not effectively capture the characteristics of such links. We introduce a new two-level model that uses both packet loss and delay information to better understand the state of the network, using delay data to give insight into whether packet loss is due to congestion or link noise. Finally, we show that this new model captures packet loss processes seen on residential ADSL and cable links more accurately than the previous models. This paper is an extended version of [11] that refines and expands our analysis, and makes two new contributions: (1) we present a new technique for classifying congestion for use in the two-level model and (2) we apply the two-level model to simulation of FEC performance, to evaluate its improved accuracy over existing models in a realistic scenario.

We structure the remainder of this paper as follows. We begin, in Section 2, by describing the experimental dataset used in our work. Section 3 then outlines existing approaches to modelling packet loss patterns in the Internet, and Section 4 explores the limitations of these existing approaches, based on an evaluation using our dataset. In Section 5, we introduce a new type of packet loss model, then show how this improves accuracy when modelling residential links in Section 6. Section 7 compares the different packet loss models using simulation of FEC performance, and Section 8 outlines how the two-level model can be applied to other scenarios. We discuss related work in Section 9, and conclude in Section 10.

2. Measurement of packet loss on residential links

We use a dataset we have previously collected, which is described in [10], to evaluate the performance of the existing packet loss models against traces of packet loss on residential links. This dataset contains end-to-end measurements of one-way Real-time Transport Protocol (RTP) [12] traffic sent over UDP/IP at a range of bit-rates (1–8.5 Mbps) from a well-connected server to measurement hosts located in residential premises. The server was located at the University of Glasgow, while the residential measurement hosts were located in the UK and Finland. A range of experimental factors were considered: the traces were collected at different bit-rates, at different times of day, and from residences connected to the Internet via several different Internet Service Providers (ISPs) using a mix of ADSL and cable modem links. The measurements were taken in 2009 and 2010.

A sequence number and transmission timestamp was attached to each packet, i, allowing us to obtain an observation sequence of packet losses, Z_i ($Z_i = 0$ for received packets, $Z_i = 1$ for lost packets) and arrival timestamps, T_i, for those packets that were received. We use the timestamps to estimate the queueing delay (DQ) experienced by each received packet, correcting for clock skew as in [13], to understand the correlation between the queueing conditions experienced by the stream and the loss behaviour. These packet loss and delay observations are then used as input to the packet loss models.

The dataset contains ~3800 RTP packet traces varying between one and ten minutes in duration, with sending data rates ranging from 1–8.5 Mb/s depending on the edge link capacity. This gives between 6000 and approximately 600,000 packets per trace. Within the traces, there are a range of different loss and delay characteristics, showing cases where the network is overloaded and causing packet loss due to congestion, cases of time-of-day variation in packet loss behaviour, as well as many traces where no packet loss at all was encountered. More information on the dataset, including how to download the raw traces, can be found in [10].

Few consistent patterns can be identified in the packet loss characteristics of the traces. Different links exhibited quite different behaviours. For example, some ADSL links showed signs of time-of-day variation in packet loss, with significant spikes during the evening (perhaps associated with more load on the network), while other ADSL links showed no such behaviour. Similarly, in some cases, the same links showed different behaviours across the two years when the measurements were taken (possibly due to network upgrades by the ISP during this period). For this reason, the analysis presented here uses each trace individually, as a separate observation of the behaviour of the network, and does not aggregate across traces. For further details on the high-level characteristics of the traces, as well as in-depth analysis of the packet loss and delay measurements, see [14].

3. Existing approaches to modelling loss patterns

Numerous attempts have been made to model the packet loss patterns seen on the network. These can be roughly categorised into end-to-end packet loss models and link-layer specific packet loss models. We are interested in end-to-end packet loss models, since it is end-to-end performance that is observed by applications. We do not further consider link-layer specific models for two reasons: (1) their effects are subsumed into the end-to-end model of the packet loss patterns and (2) most of the work on link-layer specific models has considered wireless links, since those have the most complex loss patterns. We are not aware of widely available link-layer specific models for ADSL and cable modem links that could be incorporated into an end-to-end model, and even if such were available, they would only address part of the end-to-end path. We rather consider end-to-end packet loss models trained based on the measurement data outlined in Section 2.

Commonly used end-to-end models of packet loss are based on either simple multi-state Markov models, or on more complex hidden Markov models. Perhaps the most straightforward of the simple Markov models is the two-state Markov chain, proposed by Gilbert [3], also referred

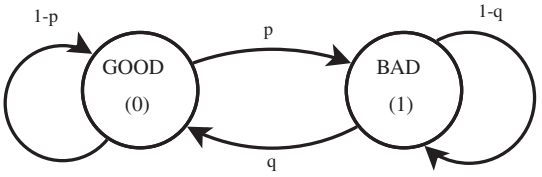

Fig. 1. Simple Gilbert Model (SGM).

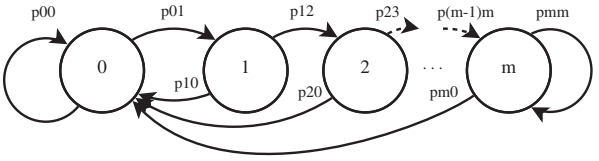

Fig. 2. Extended Gilbert Model (EGM).

to as the **Simple Gilbert Model** (SGM) [15]. In the SGM, the state of the model directly represents Z_i, as shown in Fig. 1 (i.e., the BAD state always produces packet losses, while the GOOD state never does). Transitions between the states are per-packet, and independent of the previous states. The parameters of the model are p, the probability of losing a packet, given the previous one was received; and q, the probability of receiving a packet, given that the previous one was lost. These parameters can be expressed as a matrix, as in Eq. (1):

$$\mathbf{SGM} = \begin{bmatrix} (1-p) & p \\ q & (1-q) \end{bmatrix} \tag{1}$$

These parameters are estimated by counting the number of transitions between states, as discussed in [16].

The SGM has been heavily used for evaluating application performance, especially after early work on academic networks found it to be suitable for describing the loss processes seen in these networks [6]. For example, recent evaluations of network video streaming systems [4] and video quality estimation tools [17] have used SGM models to generate their loss processes. Analytical evaluations of forward error correction (FEC) performance have also used the SGM as the basis for their work (e.g., [18,5]).

One of the drawbacks of the SGM is that there are only two states and the model is memoryless so it does not capture bursty behaviour in packet loss. To address this, a new model was proposed [7,16] where the number of states to represent losses is extended from one (as in the SGM) to m, as illustrated in Fig. 2. We refer to this as the **Extended Gilbert Model** (EGM). This extended model allows us to describe loss bursts of up to m packets, with each state representing a loss burst of a particular length. That is, state j (where $j < m$) represents a loss burst of j packets, while state m represents a loss burst of *at least* m packets. Received packets are represented by state 0, and the run-lengths of received packets are not specifically captured by the model. Considering all of the measured packet traces in our dataset [10], the 99^{th} percentile loss burst length is four packets as shown in Fig. 3. Accordingly, we consider extended Gilbert models with $m = 4$ in this paper.

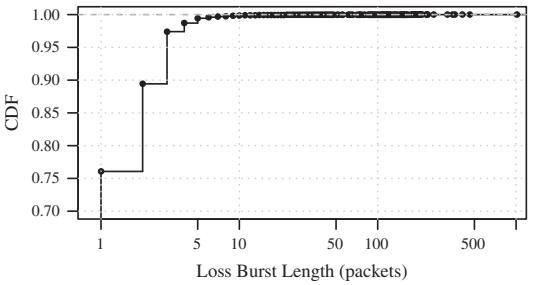

Fig. 3. Cumulative distribution of observed loss burst lengths.

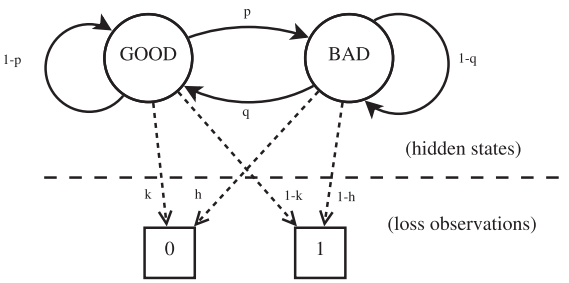

Fig. 4. Two-state Hidden Markov Model (2HMM).

The parameters of an EGM with $m = 4$ can be expressed as the matrix shown in Eq. (2):

$$\mathbf{EGM} = \begin{bmatrix} p_{00} & p_{01} & 0 & 0 & 0 \\ p_{10} & 0 & p_{12} & 0 & 0 \\ p_{20} & 0 & 0 & p_{23} & 0 \\ p_{30} & 0 & 0 & 0 & p_{34} \\ p_{40} & 0 & 0 & 0 & p_{44} \end{bmatrix} \tag{2}$$

Finally, an extension to the Gilbert model was proposed by Elliott [19], where *both* states produce errors with different probabilities, as shown in Fig. 4. The motivation behind this approach is that one of the states (GOOD) produces losses with a low probability, corresponding to isolated loss events, whereas the other state (BAD) produces losses with a higher probability, corresponding to packet loss bursts. One way to implement this model is as a two-state **Hidden Markov Model** (HMM), where only the observations of packet loss can be seen, while the actual state (GOOD or BAD) is hidden. The parameters of the HMM can be expressed as matrices **HMM A**, which defines transition probabilities between the hidden states, and **HMM B**, which defines the probability of packet loss in each hidden state, as in Eqs. (3) and (4):

$$\mathbf{HMM\ A} = \begin{bmatrix} (1-p) & p \\ q & (1-q) \end{bmatrix} \tag{3}$$

$$\mathbf{HMM\ B} = \begin{bmatrix} (1-k) & (1-h) \end{bmatrix} \tag{4}$$

In HMMs, the transition probabilities between hidden states and the loss probabilities for each hidden state are estimated from the observed data. This process is more complex than for the SGM and EGM, and involves the use

of expectation–maximisation algorithms as described in [20]. An HMM can be extended to contain more than two states. This will improve its accuracy, although in practice the complexity of the estimation process limits the number of states, since the estimation time increases quadratically with the number of states in the HMM. In this work, we will focus on two- and three-state HMMs, since these are computationally feasible to calculate, and since the states can be associated with a physical interpretation with different states representing different causes of network congestion (we discuss the physical interpretation of the states in Section 5.1).

HMMs were first employed for modelling packet loss in [8], where HMMs of varying numbers of states were applied to active measurement traces, finding that in the majority of cases, two- and three-state HMMs were sufficient. The HMMs were used to estimate the transitions between hidden states for these traces, and showed that greater resolution can be obtained using HMMs than the simpler SGM and EGM. Further work has used HMMs to model the behaviour of both packet loss and delay in Internet measurement traces between academic sites [21,9], and to predict network conditions and apply these to dynamic adaptation of FEC parameters [22].

4. Limitations of existing packet loss models

In this section we evaluate the performance of the SGM, EGM, and two- and three-state HMM (2HMM and 3HMM) packet loss models outlined in Section 3 against the measured loss traces described in Section 2. The model parameters for each of the models studied in this paper, as trained on each of the packet loss traces described in [10], can be found at http://martin-ellis.net/research/loss-modelling.

4.1. Trace classification and example loss patterns

An initial manual examination of the packet loss traces showed that they can be divided into three groups: those with zero or very low packet loss, those with non-bursty loss, and those with bursty loss. Separating out traces with very low levels of loss allows further analysis to be more significant, focusing on bursty or non-bursty loss patterns that are not meaningful when there are only occasional packet losses. For those traces visually identified to be very low loss, over 90% had 15 or fewer lost packets, so the threshold for a trace to be considered very low loss was set to 15 packets lost. A total of 2890 traces are categorised as very low loss. Of these 1679 are loss free, while the remaining 1211 have between 1 and 15 lost packets.

Among the traces with higher loss, the distinction between bursty and non-bursty loss was made by visual examination of the trace sequences. This finds 486 non-bursty traces and 433 traces with bursty packet loss. Representative samples of non-bursty and bursty packet loss behaviour are shown in Figs. 5 and 7 respectively. The bars indicate whether packets were lost (black) or received (grey). The top panel of each figure ("raw data") shows the measured loss sequence, while the lower four panels show sample synthetic loss sequences generated using the SGM, EGM, 2HMM, and 3HMM packet loss models with parameters estimated from the trace, as will be described in Section 4.2. The SGM, EGM, 2HMM, and 3HMM parameters estimated from the non-bursty and bursty traces are shown in Figs. 6 and 8, respectively. Figs. 9 and 10 show packet loss burstiness metrics to illustrate the differences between the bursty and non-bursty trace data. These figures show that the traces classified here as bursty indeed have a higher fraction of packet loss bursts than the non-bursty traces, using the metrics for bursty packet loss introduced in [23] (using $G_{\min} = 16$ as recommended) and [24] respectively.

Returning to Fig. 5, we see that the synthetic packet loss sequences generated by the SGM, EGM, 2HMM, and 3HMM are comparable to the non-bursty packet loss in the measured trace. This indicates that these models are suitable for this type of loss behaviour, matching results from previous work [6,16,8]. Fig. 7 shows a sequence with bursty periods of loss in the raw data, with long runs of received packets between them. In this case, however, it is apparent that the SGM, EGM, 2HMM, and 3HMM generate synthetic packet loss sequences that differ significantly from the measured trace.

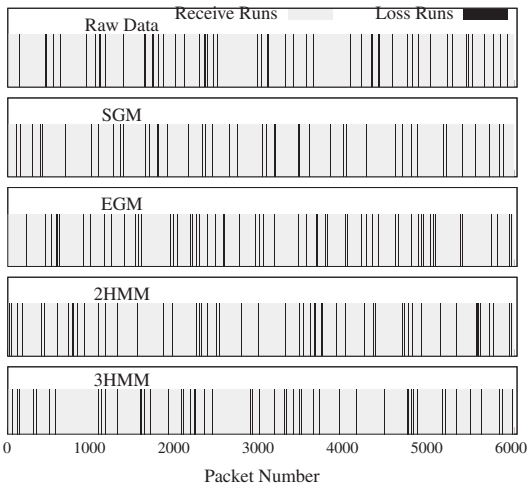

Fig. 5. Example non-bursty trace: original/synthetic loss sequences.

$$\mathbf{SGM} = \begin{bmatrix} 0.9904 & 0.0096 \\ 0.9661 & 0.0339 \end{bmatrix}$$

$$\mathbf{EGM} = \begin{bmatrix} 0.9904 & 0.0096 & 0 & 0 & 0 \\ 0.9649 & 0 & 0.0351 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{2HMM\ A} = \begin{bmatrix} 0.677 & 0.323 \\ 0.109 & 0.891 \end{bmatrix}$$

$$\mathbf{2HMM\ B} = \begin{bmatrix} 0.0327 & 0.0021 \end{bmatrix}$$

$$\mathbf{3HMM\ A} = \begin{bmatrix} 0.5455 & 0.1807 & 0.2738 \\ 0.0661 & 0.7275 & 0.2065 \\ 0.0476 & 0.0984 & 0.8540 \end{bmatrix}$$

$$\mathbf{3HMM\ B} = \begin{bmatrix} 0.0652 & 0.0067 & 0.0017 \end{bmatrix}$$

Fig. 6. Example non-bursty trace: model parameters.

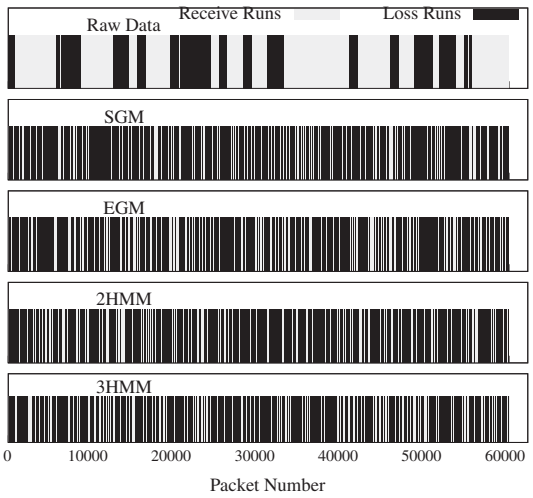

Fig. 7. Example bursty trace: original/synthetic loss sequences.

$$\textbf{SGM} = \begin{bmatrix} 0.9883 & 0.0117 \\ 0.9106 & 0.0894 \end{bmatrix}$$

$$\textbf{EGM} = \begin{bmatrix} 0.9883 & 0.0117 & 0 & 0 & 0 \\ 0.9062 & 0 & 0.0938 & 0 & 0 \\ 0.9538 & 0 & 0 & 0.0462 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\textbf{2HMM A} = \begin{bmatrix} 0.703 & 0.297 \\ 0.080 & 0.920 \end{bmatrix}$$

$$\textbf{2HMM B} = \begin{bmatrix} 0.0596 & 0.0000 \end{bmatrix}$$

$$\textbf{3HMM A} = \begin{bmatrix} 0.6652 & 0.1400 & 0.1948 \\ 0.0562 & 0.7283 & 0.2155 \\ 0.0343 & 0.0946 & 0.8711 \end{bmatrix}$$

$$\textbf{3HMM B} = \begin{bmatrix} 0.1128 & 0.0011 & 0.0001 \end{bmatrix}$$

Fig. 8. Example bursty trace: SGM, EGM, and HMM parameters.

There is no clear link between the burstiness of traces and experimental variables such as time of day. As discussed in [10], some of the measured links showed evidence of time-of-day variation in loss behaviour, although others do not. Moreover, since there are variations in behaviour within some of the traces, we choose to perform further analysis at a per-trace level.

4.2. Evaluation using parametric bootstrap

Figs. 5 and 7 give a clear visual representation of the goodness-of-fit of the models to the original data, but do not provide a quantitative measure of accuracy. To quantify model accuracy, we need a procedure to test how well the original data is captured by the model. That is, we need a numeric measure of the goodness-of-fit for the models.

To do this, we adapt a commonly used statistical technique known as *bootstrapping*. This proceeds as follows. We first estimate model parameters from each trace, using the process described in [16] for the SGM and EGM, and the *R* package *hmm.discnp* [25] for the 2HMM and 3HMM. Then, we use these parameters to generate a large number

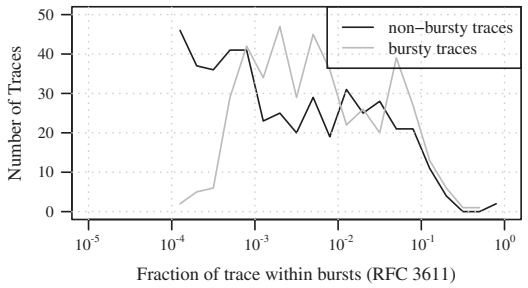

Fig. 9. Distribution of trace burstiness, using metric from [23].

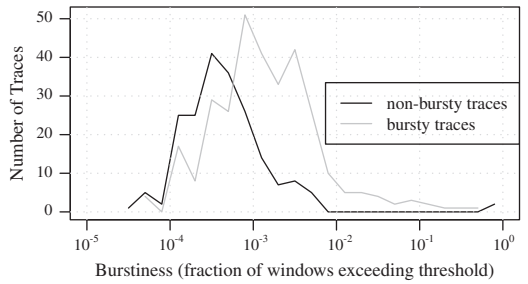

Fig. 10. Distribution of trace burstiness, using metric from [24].

of synthetic sequences from each model (in our case, we use 1000 sequences, a typical choice). We then calculate a set of statistics, S, relating to each generated sequence, as described in Section 4.3. For each of these statistics S_k, the values obtained from calculating the S_k on each of the synthetic sequences are used to produce a distribution, S_k^{synth}, which is compared to S_k^{raw} (the value of S_k obtained from the raw data). This is a similar approach to [26], which tested the "curvature" of distributions when testing for long-tailed behaviour in Internet traffic. Since this technique generates new sequences using the parameterised models, we refer to it as *parametric bootstrap*, in contrast to traditional bootstrap techniques which involve resampling from within the existing data [27].

To assess the model's goodness-of-fit, we test the null hypothesis that the observed value of S_k^{raw} is a typical draw from the distribution S_k^{synth}. If the null hypothesis is not rejected, this suggests that the model offers a good fit to the data, since the realisations of the fitted model are similar to the observed data (always in terms of the summary statistic S_k). We test this hypothesis by calculating a central 95% confidence interval and checking if S_k^{raw} falls into that interval. This is equivalent to a hypothesis test at significance level 5% where the probability of rejecting the null hypothesis given that the null hypothesis is true is 5% [28]. Although setting 5% is a typical choice for statistical hypothesis testing, a larger significance level which leads to a narrower interval does not alter our results considerably. Goodness of fit results are presented in Section 4.4.

4.3. Choice of statistics for parametric bootstrap

The parametric bootstrap aims to measure the accuracy by which the models describe the observed packet loss

processes, and must therefore be based on packet loss metrics. We calculate two metrics of packet loss: the mean packet loss fraction and the run-length distributions of lost and received packets (loss run-lengths are defined as the number of packets lost in a row, while receive run-lengths are defined as the number of packets received in a row; analysing both these metrics together is necessary, since they give insight into packet loss correlation). The former measures the total level of packet loss experienced in each trace, while the latter describes the loss patterns in the traces. These metrics form the base from which we calculate the statistics needed for parametric bootstrap.

The set of statistics, S, calculated from our raw and synthetic sequences and used for parametric bootstrap is therefore: the mean packet loss rate over the whole trace; the 5th, 25th, 50th, 75th, and 95th percentiles of the receive run-length distribution; the mean, median, and maximum loss run-lengths; and the number of receive runs in each order of magnitude (i.e., number of runs of order $10^0, 10^1, 10^2, 10^3, 10^4, 10^5$ packets).

These statistics include the loss and receive run-length distributions to assess whether the loss patterns of the raw data are matched by the models. They cover a large range of the receive run-length distribution, along with the key points of the loss run-length distribution (which is less variable). Since the receive run-lengths can range from very short (i.e., single packets) to tens of thousands of packets within a single trace, it is important that the models can capture this range. To test this, we count the number of receive run-lengths in each order of magnitude. Statistics related to the run-length distributions have been chosen, since in previous work we found the run-lengths to be a key factor in explaining the effect of packet loss on real-time applications [24].

We use these empirical metrics to describe the run-length behaviour in the traces, rather than attempting to fit probability distributions to the loss and receive run-length observations, because they can be more fairly compared between different sequences. Indeed, it is not clear which distribution is appropriate, or even whether it is possible to describe the empirical run-length distributions with any single well-known probability distribution. For example, when the loss run-lengths of two different sequences are characterised by different distributions (e.g., one showing Poisson-distributed loss run-lengths, and the other showing geometrically distributed loss run-lengths), it is unclear how they should be compared. This is especially true since loss run-lengths might be characterised by a mixture of such distributions. Another reason against fitting the run-lengths to a distribution is that the degree of fit would also have to be taken into consideration, alongside the estimated distribution parameters. For the purposes of parametric bootstrap, we feel that using empirical metrics which can be simply calculated and compared between sequences is most appropriate.

4.4. Results of parametric bootstrap

Goodness-of-fit results from applying the parametric bootstrap technique with 1000 synthetic sequences per model to the SGM, EGM, 2HMM, and 3HMM models are shown in Fig. 11a for the 486 non-bursty traces, and Fig. 11b for the 433 bursty traces in the dataset. These figures show, for each statistic S_k (y-axis), the number of traces where the model had good fit to the packet traces, in terms of S_k. Visually, for each statistic, bars that extend further right mean that the corresponding model offers a good fit to that statistic in more traces. We see that:

- All models capture the mean loss rate, offering a good fit for both the bursty and non-bursty traces.
- In terms of the receive run-lengths, the SGM, EGM, 2HMM, and 3HMM all perform poorly, i.e., offer a good fit to a relatively small number of traces, for the bursty traces, and only a little better for the non-bursty traces (recall from the example in Fig. 7 that bursty loss was not well captured by these models). Interestingly, there are differences between the models for different types of trace. The SGM and EGM appear to capture the receive run-lengths better in the non-bursty traces, while the 2HMM and 3HMM are slightly better for short receive run-lengths in the bursty traces. Since the 2HMM and 3HMM aim to capture the changes in loss states, they perform slightly better in bursty traces. This extra complexity does not help when the loss pattern is not bursty.
- The patterns in the loss run-length results are similar for both non-bursty and bursty traces (with better performance on non-bursty traces, as expected). The SGM and EGM perform particularly well, capturing mean and median of the loss run-lengths for most traces. The 2HMM and 3HMM capture the mean loss run-length less well, although they do capture the median (since most loss bursts consist of a single packet, the median loss run-length is likely to be 1; even in bursty periods, individual loss runs are short, but fall close together). Although loss run-length statistics are well-captured, this does not mean the trace is accurately modelled (since receive run-length statistics are not well-captured by the models), and can lead to loss patterns that are quite different to the raw data, as seen in Fig. 7.

To summarise, the parametric bootstrap results show that although the SGM, EGM, 2HMM, and 3HMM perform adequately on certain types of traces and in terms of some metrics, they do not accurately capture the loss patterns of the most bursty traces (as illustrated by the example in Fig. 7). Although bursty packet loss does not occur in all traces, over 10% of traces were classified as bursty. The implications of this loss behaviour for the performance of streaming video are too important to be ignored, since bursty packet loss severely degrades the performance of the FEC schemes needed to ensure acceptable video quality at residential receivers [24].

To understand the real-world performance of such schemes, and more generally, to understand end-to-end video quality, it is important to ensure that the packet loss simulation used in testing is accurate. The results shown in this section have illustrated that the existing packet loss models studied here are inaccurate in the presence of bursty packet loss (such as that seen on residential ADSL and

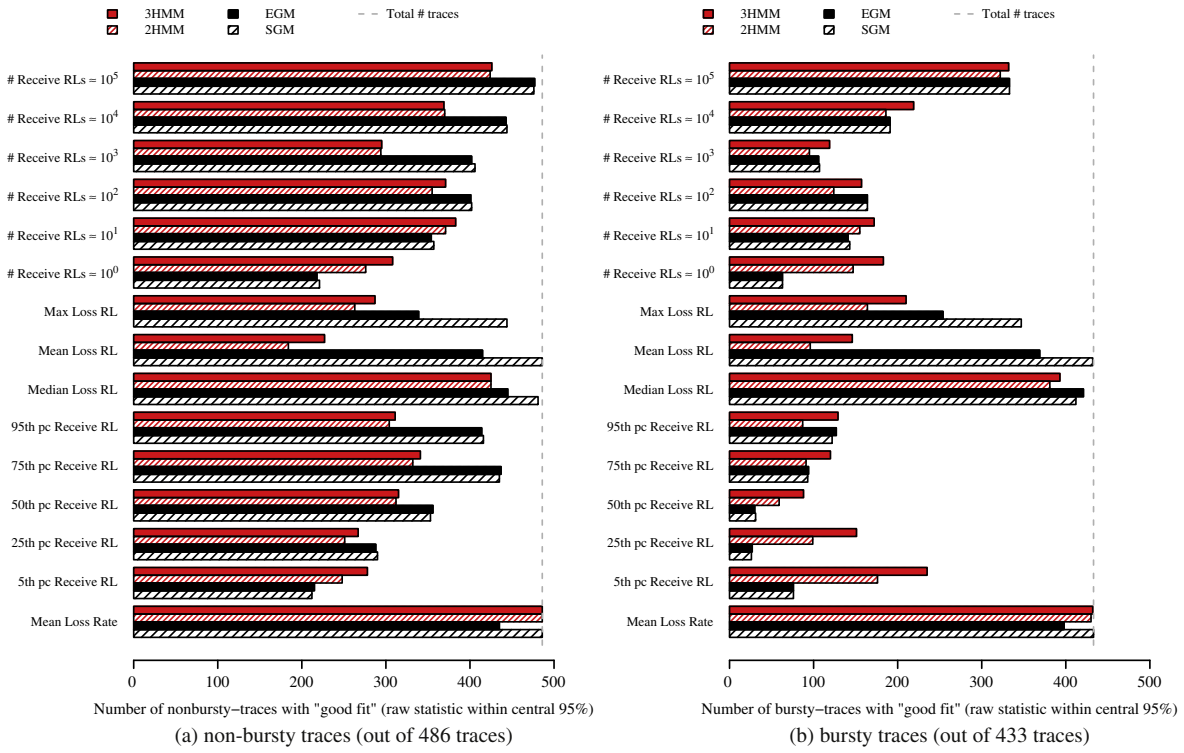

Fig. 11. Parametric bootstrap results (loss models).

Cable networks). Therefore, an alternative modelling approach that is more capable of modelling bursty packet loss is needed to better understand FEC performance and the overall video quality experienced by residential Internet users.

5. A new two-level model for packet loss

Section 4 showed that the existing models cannot accurately model the bursty packet loss behaviour seen in some of our traces. To improve the models, we begin with the observation that periods of bursty loss are often associated with network congestion in environments using drop-tail queues, and that the location of those queues within the network will influence the degree of statistical multiplexing of flows, and hence the impact of loss bursts on individual flows. Both the nature and location of the loss event is important.

We hence propose a new *two-level* model that explicitly considers the congestion state of the network, and the location of congestion within the network, to more accurately model bursty loss. This model is similar to the layered model outlined in [29], which divides traces into separate sub-traces based on loss rate. However, our model differs in the following ways. First, we explicitly consider the bursty nature of packet loss, rather than just the overall loss rate. Second, we incorporate additional delay information captured in the measurement data [10] to give insight into the causes of loss. Finally, the outer states of our model were chosen specifically to model the location of congestion within the network, rather than being purely data-driven. Our new model uses the combination of packet loss and delay information to obtain a more accurate picture of the network conditions. Understanding the relationship between delay and loss is important because packet delay gives insight into the likely location of congestion within the network, potentially allowing us to distinguish losses due to congestion and queue overflow in the core network, from losses caused by other factors (e.g., electrical noise on the last mile link).

In our two-level model, the outer states represent the network state (i.e., congested or uncongested) and the location of the congestion, while the inner states capture the packet loss process. Although it is possible that several of these conditions might occur at the same time (e.g., both edge and core congestion), we focus on the *dominant* loss mode, allowing us to represent the current network conditions as one of the outer states of the model. For example, given that a single flow makes up a relatively higher fraction of the traffic at the network edge than it would in the core, congestion at the edge is likely to have a more dominant effect than congestion elsewhere.

In this two-level approach, the trace being used to train the model is split up into fixed duration windows, and the outer state for each window is explicitly chosen by applying a simple classification algorithm to the loss and delay data (described in Section 5.1). The window duration was one second, chosen to balance the need to capture the dynamics of the system while including enough data

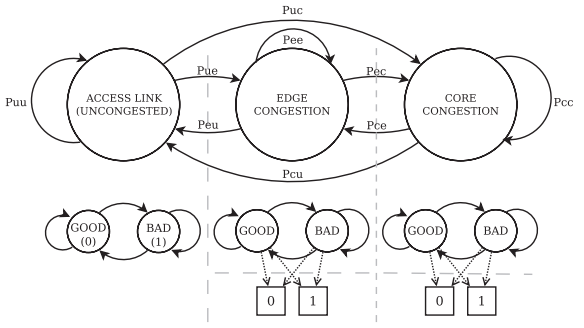

Fig. 12. Two-level model, SGM/2HMM/2HMM configuration.

points within a single window to train the inner model; investigating the effects of different window durations is for future work. Self-transitions between the outer states are possible, allowing the model to capture the situation where conditions stay the same for an extended period of time. The transitions between outer states are modelled as a Markov chain, and the transition probabilities are estimated by counting the number of transitions between windows of each outer state. The parameters of this outer model, therefore, can be represented as a transition probability matrix, $\mathbf{A}_{\mathbf{outer}}$:

$$\mathbf{A}_{\mathbf{outer}} = \begin{bmatrix} p_{uu} & p_{ue} & p_{uc} \\ p_{eu} & p_{ee} & p_{ec} \\ p_{cu} & p_{ce} & p_{cc} \end{bmatrix} \tag{5}$$

More formally, the probability of moving from outer state α to outer state β is calculated using the number of transitions from α to β, $n_{\alpha\beta}$, and the total number of transitions from α, n_α, as in Eq. (6) (following [30]):

$$\mathbf{A}_{\mathbf{outer}}[\alpha, \beta] = \frac{n_{\alpha\beta}}{n_\alpha} \tag{6}$$

The observation sequence Z_i from all windows classified as being in a particular outer state is used to calculate the parameters of the inner packet loss model for that outer state, as discussed in Section 5.2.

5.1. Classification schemes to identify outer states

Fig. 12 illustrates an example of the two-level model, with three outer states (corresponding to different conditions on the underlying network), and inner models to capture packet loss within each outer state. The outer states for the model correspond to the different sources of packet loss along the end-to-end path that our measurement data traversed:

- *access link noise*, where issues with the physical layer cause IP-layer packet loss;
- *edge congestion*, where queue overflows at the ISP edge (i.e., DSLAM or CMTS) cause packet loss; and
- *core congestion*, where queue overflows occur at routers within the core IP networks.

In terms of the physical effects seen in the data, access link noise will cause low levels of uncongested loss regardless of the levels of delay. Edge congestion will cause higher levels of loss, which will be associated with higher delay since the building queues at edge routers will noticeably increase queueing delay DQ due to the low levels of statistical multiplexing expected on edge links. Core congestion will also cause higher loss, but without the noticeable effect in DQ since higher statistical multiplexing at core routers means that the effects of queueing on DQ will be less obvious at the receiver. These intuitive insights into the effects of congestion on packet loss and delay form the basis of the classification schemes presented here.

We consider two classification schemes for the outer states. One is based on fixed thresholds for loss and delay (the *ld* classifier), while the other looks for increasing trends in queueing delay before loss (the *ldbl* classifier). Both classifiers consider fixed one-second windows of trace data, and in each window examine the number of losses (N), number of loss bursts (M), and median DQ ($\widetilde{DQ}$). Other schemes for classification are possible, and due to the two-level nature of the model, could be used in place of the classifiers we consider.

The *ld* classifier is described in Algorithm 1. It uses thresholds for N and M to identify periods of high loss, which indicate congestion, and another threshold for $\widetilde{DQ}$ to distinguish between core and edge congestion. The loss thresholds ($N > 2$ or $M > 2$ indicating congestion) are chosen based on the assumption, backed up by analysis of our dataset, that non-congestive loss is unlikely to create more than two separate loss events with a one second window, and that a loss burst of longer than two packets is likely to be due to congestion. The $\widetilde{DQ}$ threshold (5 ms) is also based on examination of the trace data; traces with non-bursty loss typically exhibit $DQ < 5$ ms. These thresholds are based on the dataset in [10], and while we expect them to be valid for residential ADSL and Cable links of the type we have measured, some calibration may be needed before they can be applied to other sorts of network.

Algorithm 1. Loss/Delay (*ld*) classifier

```
if (state = "uncongested") then
  if (N > 2) or (M > 2) then        #"high loss"
    if (DQ~ > 5 ms) then            #"elevated DQ"
      state ← "edge congestion"
    else
      state ← "core congestion"
    end if
  end if
else
  if (N ⩽ 2) and (M ⩽ 2) and (DQ~ ⩽ 5 ms) then
    state ← "uncongested"
  end if
end if
```

The *ldbl* classifier is described in Algorithm 2. It is a generalisation of the *ld* classifier, that no longer uses a fixed $\widetilde{DQ}$ threshold to distinguish between edge and core congestion. Instead, when congestion is identified (as before, when $N > 2$ or $M > 2$), *ldbl* looks at the median DQ from

packets immediately before the loss event ($\widetilde{DQ}_{BL}$), and compares this against $\widetilde{DQ}$. If $\widetilde{DQ}_{BL}$ is more than twice $\widetilde{DQ}$ (i.e., delays preceding losses are elevated above the average), then the window is classified as edge congestion; otherwise, the window is classified as core congestion. In the core congestion state, DQ does not give insight into congestion, so if N and M fall below their thresholds, the state returns to uncongested. However, in the edge congestion state, DQ is important; so, an extra check is needed to test for the slow fall in delay observed when the congested queue empties (i.e., at an ISP-edge router). Once the losses from congestion have stopped and $\widetilde{DQ}$ has fallen within 10% of the $\widetilde{DQ}$ in the last uncongested state ($\widetilde{DQ}_{UC}$), the state switches back to uncongested. To do this, in Algorithm 2, we test $\widetilde{DQ} \leqslant k\widetilde{DQ}_{UC}$, where $k = 1.1$. This tolerance of 10% is chosen to prevent the model remaining in a congested state for longer than necessary, simply because the delay has not fallen to precisely the pre-congestion level. A final addition to the *ldbl* classifier is to test for cases when DQ increases *before* any losses are observed (e.g., to detect the increase in queue lengths at the edge router). If a sudden increase in DQ is detected within a window (defined as 25% of the difference between maximum and minimum DQ for the whole trace), the window is classified as showing edge congestion. This threshold was chosen based on examples of such sudden increases in the dataset.

Algorithm 2. Loss/Delay Before Loss (*ldbl*) classifier

```
if (state = "uncongested") then
  if (N > 2) or (M > 2) then      #"high loss"
    if (DQ̃_BL > 2DQ̃) then         #"elevated DQ"
      state ← "edge congestion"
    else
      state ← "core congestion"
    end if
      DQ̃_UC ← DQ̃
  end if
else if (state = "edge congestion") then
  if (N ⩽ 2) and (M ⩽ 2) and (DQ̃ ⩽ kDQ̃_UC) then
    state ← "uncongested"
  end if
else if (state = "core congestion") then
  if (N ⩽ 2) and (M ⩽ 2) then
    state ← "uncongested"
  end if
end if
```

The thresholds chosen for the classifiers are appropriate for this dataset. Applying these classification algorithms to other network scenarios may require a different choice of thresholds. When choosing these thresholds, it is important to select values that can be used to distinguish between different network states. For example, understanding typical rates of uncongested packet loss are necessary when choosing thresholds for N and M. Further work might develop ways to self-configure the threshold values, by looking at the patterns in the loss and delay time-series. The requirement of such techniques is that they must be able to identify between the different outer states of the model.

5.2. Parameter choice for inner states

The outer states of the model correspond to different network conditions (core congestion, edge congestion, and access link noise), modelled as a three-state Markov chain. Within each outer state, an inner packet loss model is used to model the observed loss processes.

The inner model can be either a SGM or a 2HMM, as described in Section 3. The two configurations used for the inner models are SGM/SGM/SGM, where packet loss is modelled by an SGM in each outer state, and SGM/2HMM/2HMM, where the outer states potentially subject to congestion are modelled by a 2HMM. Since Section 4 showed that non-bursty loss is well-modelled by the SGM, it is suitable for uncongested loss due to access link noise. The SGM/2HMM/2HMM configuration tests whether using HMMs *within* the pre-classified states also improves performance. Configurations using the EGM or 3HMM are also possible, but based on the results in Section 4 we believe they would offer little benefit, while being significantly more complex.

6. Performance of the two-level model

In Fig. 13, we present sample results using the two-level model on the example bursty packet loss trace shown in Fig. 7. The top panel shows the measured trace data, while the lower four panels show synthetic traces generated using different configurations of the two-level model: we use each of the *ld* and *ldbl* classifiers for the outer states with both the SGM/SGM/SGM and SGM/2HMM/2HMM inner models.

As the lower panels in Fig. 13 show, the sequences generated by the new models match the bursty nature of the measured data more closely than those from the previous models shown in Fig. 7. We observe that the models show periods of bursty loss, separated by long runs of successfully received packets. This differs from the previous models, which generated losses throughout the trace. It is clear that the receive run-length distributions in bursty traces are more accurately captured by the two-level model than by the simple models.

The model parameters for this example trace are shown in Figs. 14 and 15 for the two-level models using the SGM/SGM/SGM inner model, and Figs. 16 and 17 for the two-level models using the SGM/2HMM/2HMM inner model. Observe that in this particular trace, the model switches between the uncongested and core congestion states; neither algorithm has classified any window as showing edge congestion, since the median queueing delay does not exceed the threshold of either classifier. It is also worth noting that for this trace, both classifiers agree on the classification of the outer states, and yield the same transition probability matrix for the outer states. However, this is not

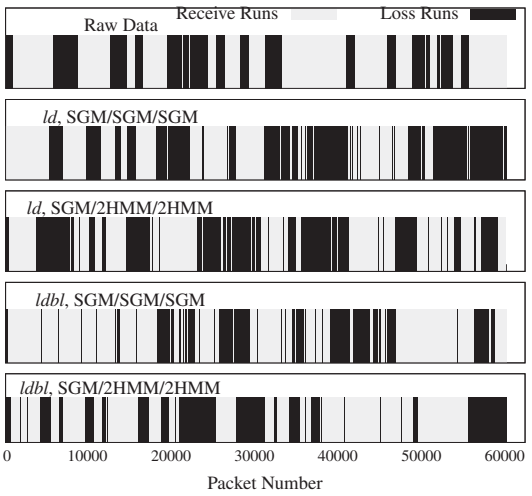

Fig. 13. Example bursty trace (two-level loss/delay models).

$$\text{Outer States } \mathbf{TPM} = \begin{bmatrix} 0.9263 & 0 & 0.0632 \\ 0 & 0 & 0 \\ 0.1182 & 0 & 0.8818 \end{bmatrix}$$

$$\text{Uncongested } \mathbf{SGM} = \begin{bmatrix} 0.999526 & 0.000474 \\ 1 & 0 \end{bmatrix}$$

$$\text{Edge congestion } \mathbf{SGM} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\text{Core congestion } \mathbf{SGM} = \begin{bmatrix} 0.9682 & 0.0318 \\ 0.9071 & 0.0929 \end{bmatrix}$$

Fig. 14. Example bursty trace: *ld*, SGM/SGM/SGM.

the case for all traces, which is why different classification algorithms have different overall performance.

The performance of the different configurations of the two-level model is shown in Fig. 18a and b, which show the results of applying parametric bootstrap with 1000 synthetic sequences to the 486 non-bursty and 433 bursty traces identified earlier. These demonstrate the improved modelling accuracy shown in Fig. 13 over all the traces. As before, the performance of the models on the non-bursty traces is better than for the bursty traces. However, the two-level models also show improved performance in terms of all the metrics, for both non-bursty and bursty traces, with more traces offering a good fit for these models than the previous models. For comparison, examine the number of traces where the model offers a good fit for the existing loss models in Fig. 11 with those in Fig. 18.

Both the *ld* and *ldbl* classifiers appear to have similar performance. However, the choice of inner model configurations has a large impact on the performance of the two-level model. The SGM/SGM/SGM configuration improves slightly on the single-level SGM, EGM, and HMMs using only loss data, although the percentiles of the receive run-length distribution are still not well-modelled in the majority of traces. However, the two-level models using

$$\text{Outer States } \mathbf{TPM} = \begin{bmatrix} 0.9263 & 0 & 0.0632 \\ 0 & 0 & 0 \\ 0.1182 & 0 & 0.8818 \end{bmatrix}$$

$$\text{Uncongested } \mathbf{SGM} = \begin{bmatrix} 0.999526 & 0.000474 \\ 1 & 0 \end{bmatrix}$$

$$\text{Edge congestion } \mathbf{SGM} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\text{Core congestion } \mathbf{SGM} = \begin{bmatrix} 0.9682 & 0.0318 \\ 0.9071 & 0.0929 \end{bmatrix}$$

Fig. 15. Example bursty trace: *ldbl*, SGM/SGM/SGM.

$$\text{Outer States } \mathbf{TPM} = \begin{bmatrix} 0.9263 & 0 & 0.0632 \\ 0 & 0 & 0 \\ 0.1182 & 0 & 0.8818 \end{bmatrix}$$

$$\text{Uncongested } \mathbf{SGM} = \begin{bmatrix} 0.999526 & 0.000474 \\ 1 & 0 \end{bmatrix}$$

$$\text{Edge congestion } \mathbf{2HMM\ A} = \begin{bmatrix} NA & NA \\ NA & NA \end{bmatrix}$$

$$\text{Edge congestion } \mathbf{2HMM\ B} = \begin{bmatrix} NA & NA \end{bmatrix}$$

$$\text{Core congestion } \mathbf{2HMM\ A} = \begin{bmatrix} 0.674 & 0.326 \\ 0.112 & 0.888 \end{bmatrix}$$

$$\text{Core congestion } \mathbf{2HMM\ B} = \begin{bmatrix} 0.1057 & 0.0092 \end{bmatrix}$$

Fig. 16. Example bursty trace: *ld*, SGM/2HMM/2HMM.

$$\text{Outer States } \mathbf{TPM} = \begin{bmatrix} 0.9263 & 0 & 0.0632 \\ 0 & 0 & 0 \\ 0.1182 & 0 & 0.8818 \end{bmatrix}$$

$$\text{Uncongested } \mathbf{SGM} = \begin{bmatrix} 0.999526 & 0.000474 \\ 1 & 0 \end{bmatrix}$$

$$\text{Edge congestion } \mathbf{2HMM\ A} = \begin{bmatrix} NA & NA \\ NA & NA \end{bmatrix}$$

$$\text{Edge congestion } \mathbf{2HMM\ B} = \begin{bmatrix} NA & NA \end{bmatrix}$$

$$\text{Core congestion } \mathbf{2HMM\ A} = \begin{bmatrix} 0.674 & 0.326 \\ 0.112 & 0.888 \end{bmatrix}$$

$$\text{Core congestion } \mathbf{2HMM\ B} = \begin{bmatrix} 0.1057 & 0.0092 \end{bmatrix}$$

Fig. 17. Example bursty trace: *ldbl*, SGM/2HMM/2HMM.

the SGM/2HMM/2HMM configuration show much improved performance over the previous models, with the majority of traces being well-modelled in terms of every statistic of interest (and many showing over 75% of traces as well-modelled). This is because they more accurately capture the different states in packet loss, and use the most appropriate model for each (i.e., SGM for uncongested, and 2HMM for congested periods).

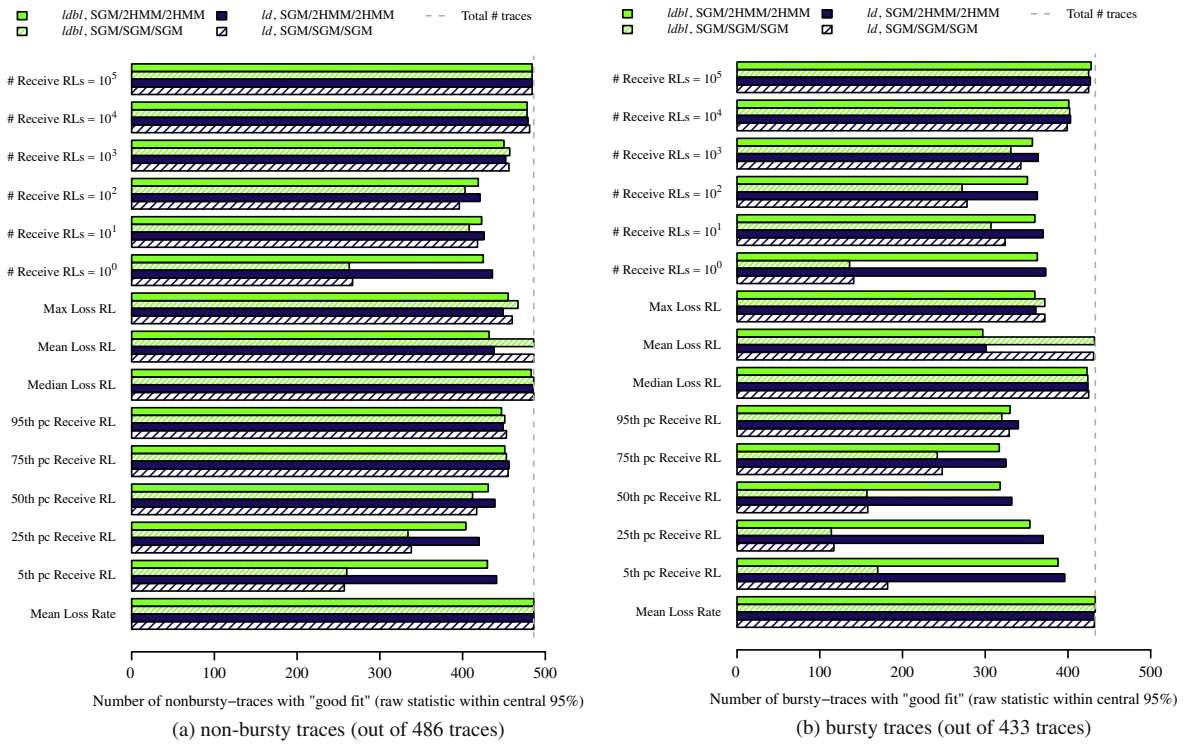

Fig. 18. Parametric bootstrap results (two-level loss/delay models).

To quantify the improvement obtained from the new models, we use a metric showing the performance of each model $\mathcal{M}$ relative to the SGM. The metric of model improvement is calculated for each S_k as in Eq. (7):

$$\mathcal{I}_{S_k}(\mathcal{M}) = \frac{\#\textit{traces with good fit for } S_k(\mathcal{M})}{\#\textit{traces with good fit for } S_k(SGM)} \tag{7}$$

Model improvement averaged over all S_k, denoted by $\overline{\mathcal{I}}(\mathcal{M})$, is calculated as in Eq. (8), and shown in Fig. 19:

$$\overline{\mathcal{I}}(\mathcal{M}) = \frac{\sum_{i=1}^{n} \mathcal{I}_{S_k}(\mathcal{M})}{n} \tag{8}$$

The results shown in Fig. 19 demonstrate the difference between non-bursty and bursty traces, showing that while only a slight improvement is obtained by using the two-level model for the non-bursty traces, the improvement is much larger for bursty traces. The two-level models (SGM/SGM/SGM configuration) produce a good fit in roughly twice as many bursty traces as the SGM (on average). With the SGM/2HMM/2HMM configuration, almost four times as many bursty traces are well-modelled compared to the SGM.

These results show that the two-level delay/loss models (using the SGM/2HMM/2HMM configuration) are more accurate than the previous single-level SGM, EGM, or HMMs. For the traces previously identified as non-bursty, the two-level models with the SGM/2HMM/2HMM configuration are suitable in almost all cases. In terms of the bursty traces, most of which were poorly modelled by the previous models, the two-level models again show a clear improvement, as illustrated in the example of Fig. 13.

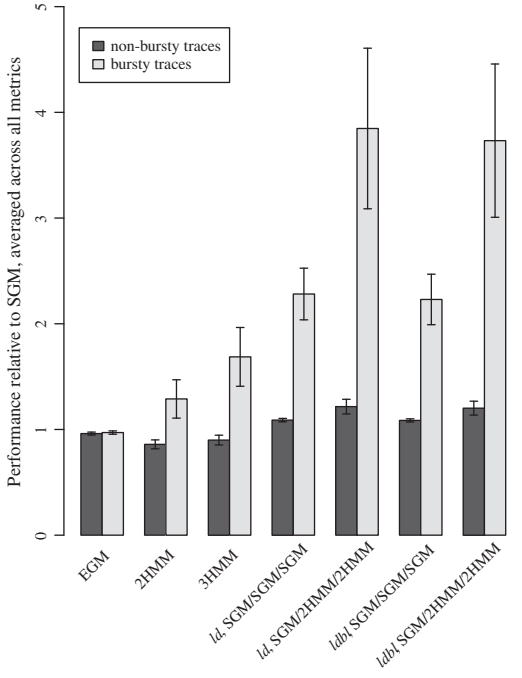

Fig. 19. Performance improvement of new models, relative to SGM.

Looking at the percentiles of the receive run-length distribution (which the previous models failed to model accurately), the importance of the choice of inner model configuration is made clear. Two-level models with the

SGM/SGM/SGM configuration perform better than the single-level SGM and EGM models for all the receive run-length percentiles, but not as well as the HMMs for the 5th and 25th percentiles. However, by using the SGM/2HMM/2HMM configuration, the performance is better than the single-level SGM, EGM, and HMMs, for all the statistics. In terms of the computational complexity and estimation time, we note that since the two-level model treats the different outer state separately, parameter estimation is often faster than when using the HMMs. This is due to the simpler SGM estimation procedure used for the uncongested portions of the trace; the more time consuming HMM estimation process is only applied to the portions of the trace classified as showing congestion.

7. Application to modelling FEC effectiveness

Forward error correction (FEC) can be used to maintain video quality in IPTV applications [31]. These typically use FEC at the application level, adding redundant packets to the media stream that can be used to repair packet loss. The effectiveness of FEC is highly dependent on the packet loss characteristics of the network. Therefore, it is important to tune the FEC parameters to suit the network conditions to maximise the FEC recovery rate while minimising the bandwidth overhead. With ongoing commercial deployment of multicast IPTV service to residential Internet users, it is becoming important to understand how to tune FEC parameters to suit such services. In particular, it is important to understand how the loss patterns of ADSL and cable access links differ from more widely studied backbone network links, and how this impacts media quality and user experience.

Section 6 showed that the two-level model is more accurate in modelling packet loss statistics than previous models. In this section, the two-level model is applied to model the performance of FEC, and the accuracy of this model for FEC performance is compared to the single-level SGM and uniform random loss, using input packet loss sequences generated by these models. This is important since FEC performance can be evaluated using packet loss models, rather than requiring full measurement traces (provided that the models induce similar FEC performance to the raw data). Since the SGM is already widely used for this purpose (e.g., [18,32,33,5,4]), this section demonstrates that the two-level model is more suitable when simulating the packet loss patterns of residential broadband networks.

7.1. Comparing models to raw data

To compare performance of the models when evaluating FEC effectiveness, we follow the same experimental procedure as described in [24], where simulations of the OpenFEC library were conducted using the packet loss traces described in Section 4. Here, we compare the performance of the best performing FEC scheme in OpenFEC, Low-Density Parity Check (LDPC) codes [34], using two different parameter settings. As in [24] we evaluate FEC recovery using the performance evaluation tool of OpenFEC, eperftool. Our aim is to compare the improved accuracy of the two-level model, when predicting FEC performance, compared to previous approaches (e.g., using uniform random packet loss, or generating loss sequences using the SGM).

To demonstrate the improved accuracy of the two-level models using FEC as an example application, the FEC performance results obtained from these synthetic sequences are compared to the results obtained from the original raw data. For each trace, this gives FEC performance measured from:

- the original raw trace data, showing measured packet loss;
- a synthetic trace subject to uniform random loss, with a loss probability matching the average loss probability seen in the raw trace data;
- a synthetic sequence calculated using the SGM with model parameters derived from the raw trace data; and
- a synthetic sequence calculated using the *ld*, SGM/2HMM/2HMM model with parameters derived from the raw trace data.

The FEC performance from each of the models is compared to the performance calculated on the raw data, to assess the accuracy of the models, in terms of simulating FEC repair performance.

7.2. Improved accuracy of two-level models

Fig. 20 compares the performance of FEC obtained from the loss traces (as discussed earlier) against FEC performance results using synthetic data from (1) uniform random packet loss; (2) packet loss generated by the SGM model; and (3) packet loss generated by the *ld*, SGM/2HMM/2HMM model. The metric for performance is *FEC effectiveness* (i.e., number of source packets repaired divided by number of source packets lost). This is used instead of residual loss rate, as it better illustrates performance differences between the models.

From each raw data trace, ten synthetic sequences were generated (within eperftool for uniform random loss model, using the approach described in Section 4 for the single-level SGM, and the approach described in Section 5 for the two-level *ld*, SGM/2HMM/2HMM model). Each point on the scatter plots in Fig. 20 represents a measurement trace, and shows the FEC effectiveness obtained from the raw data (x-axis), and the mean of the FEC effectiveness obtained from the 10 synthetic sequences. This demonstrates the typical FEC performance that would be obtained from simulation using these models. If the models perfectly matched the raw data, the points would fall on a 45° diagonal line. Deviations from this line show the extent to which the FEC performance obtained from the models is different to that obtained from the raw data.

The left column of Fig. 20 shows the results from uniform random loss, the middle column shows results from the SGM model, and the column on the right shows results from the *ld*, SGM/2HMM/2HMM model. The top row shows results from the LDPC $(k = 67, r = 33)$ scheme, and the bottom row shows results from LDPC $(k = 80, r = 20)$.

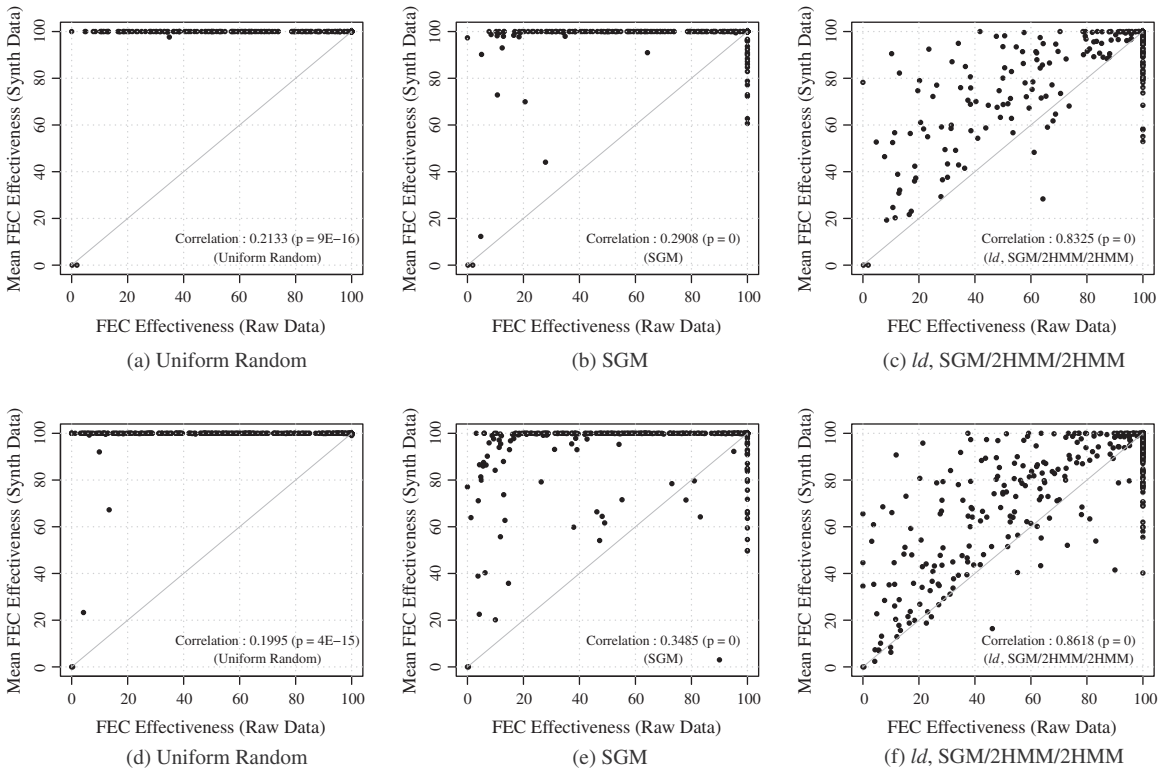

Fig. 20. FEC performance on raw and synthetic data (top row: LDPC ($k = 67, r = 33$); bottom row: LDPC ($k = 80, r = 20$)).

From Fig. 20a and d, it is clear that the FEC performance obtained by simulating uniform random loss is quite different from that in the real data. This is as we have discussed in [24], but is visible in the line present across the top of Fig. 20a and d, corresponding to the case where the synthetic sequence resulted in 100% FEC recovery, while the original raw data sequence was not fully recovered. Fig. 20b and e show that the performance of the SGM is only slightly better than using uniform random loss. The lines of points representing 100% recovery is also present for the SGM. In contrast, the *ld*, SGM/2HMM/2HMM model, as shown in Fig. 20c and f, tends to generate synthetic sequences that show FEC performance closer to the measured data, with the points clustering nearer to the 45° diagonal.

The vertical line around $x = 100$ on the plots for the SGM and *ld*, SGM/2HMM/2HMM models indicates cases where 100% recovery was achieved in the original trace, while the synthetic trace resulted in lower recovery. In these cases, the model appears to be somewhat pessimistic about the FEC performance. However, looking at the correlation between raw FEC effectiveness, and mean synthetic FEC effectiveness (displayed on each plot), it is clear that the two-level model is more accurate, with correlation of over 0.8. This is a strong improvement over the correlation of ∼0.2 (for uniform random loss) and ∼0.3 (for the SGM).

Compared to the uniform random loss model, and the widely used single-level SGM model), it is clear that the two-level model we propose gives more accurate, but not perfect, FEC performance results. This suggests that our two-level model is more suitable for simulation of FEC performance on video streaming over residential broadband networks.

8. Application to other network scenarios

To apply the two-level model presented in this paper to different network scenarios, with characteristics that differ significantly from the residential broadband links we studied, it is necessary to understand the conditions of the network to be modelled. There are two elements to this: analysing the system architecture to consider the possible sources of packet loss and the reasons for this loss (i.e., the outer states of the model), and collecting measurement traces of the network. Analysis of the traces (using packet loss and delay measurements, as discussed in Section 5, or other relevant metrics) can then be used to assist in developing a classification algorithm to identify the network state.

To apply the two-level model as presented in Section 5, with the same outer states, it would be sufficient to study the measurement traces for the new type of network, and develop a suitable classifier to determine the congestion state of the network from the measurements. This might be as simple as choosing appropriate thresholds to use with the algorithms presented in Section 5.1. Following this, the next step would be to estimate the model parameters from the trace data, and apply the same model evaluation procedure described in Section 4.4. Scripts to generate synthetic traces using all of the packet loss models studied in this paper are available at http://martin-ellis.net/research/loss-modelling. These synthetic traces

can then be used to evaluate the two-level model (in the configuration described in Section 5, or a different one), once appropriately tuned, to simulate network behaviour for performance evaluation of applications.

The two-level model described in this paper can be directly used to evaluate the behaviour of other applications on residential broadband networks. The parameter files and scripts needed to generate synthetic trace data for such networks have also been made available at http://martin-ellis.net/research/loss-modelling. Application of these parameters will allow simulation of typical loss characteristics of ADSL and cable networks of the type we measured, and will hence allow application performance on such networks to be evaluated.

It may be more difficult to directly simulate loss behaviour of other environments, such as WiFi or mobile networks, since the model parameters we calculate for residential broadband links will not match those environments. Nonetheless, the two-level model approach outlined here could be reused. Through analysis of the possible network conditions and empirical measurement, the two-level model framework presented in this paper could be used in future work to simulate both the longer-term changes in network behaviour through the outer states of the model, and the short-term, correlated packet loss behaviour using the inner models. In this case, different classification algorithms might be used, and could even consider using not just packet loss and delay data, but also available bandwidth, capacity, or jitter estimates. An interesting avenue for future work would be the application of online classification algorithms to learn the network conditions associated with different environments, and use these to improve the simulation of more diverse scenarios.

9. Related work

Related models have been proposed to model frame errors in wireless networks. These include a four-state Markov chain model, which includes separate states for long and short loss and receive run-lengths [35]. The thresholds for long and short are defined in terms of the physical-layer characteristics of the wireless channel, making this approach hard to directly apply to Internet losses. The Markov-based trace analysis (MTA) [36] technique uses a data-preconditioning approach, similar to our pre-classification of network state, aiming to classify frame error traces from GSM networks into lossy or loss-free sub-traces, then concatenating and modelling these separately.

In [37], the MTA approach and other Markov models were evaluated against a new alternative, the extended on/off model (modelling loss and receive runs with lengths derived from mixtures of geometric distributions). The results of [37] show that the extended on/off model improves modelling accuracy, although only a single trace was used throughout the evaluations making it unclear that the results hold in general. In another comparison study [38], the performance of the SGM, the four-state Markov-chain model, and the MTA technique are compared using error measurements from DVB-H networks. This latter study concluded that the good performance and relative simplicity of the four-state model makes it well-suited for simulation of DVB-H errors. However, since their approach relied on manually estimating model parameters, it is difficult to scale up for large-scale use.

Hidden *Semi-Markov* Models (HSMMs), were applied to validate estimates of Internet packet loss from PlanetLab measurements [39]. HSMMs extend HMMs, considering the packet loss process as an alternating on/off process (i.e., loss and receive runs), but allowing the run-lengths to be drawn from non-exponential distributions (unlike HMMs, where run-lengths are drawn from an exponential distribution). This approach is more general than the HMMs we consider, although the parameter estimation is more complex. However, future work could use an HSMM model within our two-level model, possibly further improving performance.

10. Conclusions and future work

In this paper, we evaluated the accuracy of commonly used packet loss models using traces measured from residential access networks, and found that they are insufficient to capture the bursty loss conditions present on such networks. We introduced a new two-level model that more accurately captures these loss conditions. The results of Section 6 show that our new model performs better than the previous widely-used models (SGM, EGM, 2HMM, and 3HMM) across all types of traces. Our two-level model uses a three-state outer Markov chain to model network state, with inner loss models to capture packet loss processes. Combining the simplicity of the SGM for non-congestive loss, and the extra power of HMMs for capturing bursty, congestive loss, the SGM/2HMM/2HMM inner model configuration has been shown to accurately model measured packet loss on a range of residential ADSL and cable links.

Accurate simulation of packet loss relies on understanding the end-to-end path the traffic is likely to take (e.g., across the Internet, over the ISP backbone, through an access router and finally across an ADSL or cable modem link). By measuring the end-to-end performance on a representative link, and using this to train the two-level packet loss model we have proposed, realistic simulation results can be obtained. This is demonstrated in Section 7, which compared the performance of FEC simulations to real data, showed that the two-level model we propose provides better accuracy than previous models.

Although this study has focused on the use of the two-level model in residential broadband networks, it is possible to apply it to other network types, as discussed in Section 8. An important element of this would be to understand the different sources of loss in the network to be modelled, and capture these in the outer states of the two-level model. For example, an 802.11 wireless link can be expected to have very different behaviour than an ADSL link, so the single outer state representing edge congestion in our model might need to be replaced with multiple states to model different link behaviours. This type of extension can be done within the framework of the two-level model we have defined.

An interesting area for future work would be to add online classification algorithms, to deduce the outer states dynamically according to observed data (and similarly to choose the correct combination of models for the inner states). Such an approach would require a significant new effort, including collection of a large quantity of additional trace data from multiple types of networks, development of new classification algorithms, and thorough validation of the technique. However, it would also represent a significant advance in improving the accuracy of network models.

Our results show that future work on simulating FEC performance on networks with bursty packet loss, such as the residential broadband we have measured, would benefit from using the two-level model rather than the single-level SGM. Another potential use for the new models is in adaptive FEC, where the receiver uses the classification and modelling mechanisms presented in Section 5 as input to a control loop which determines which FEC parameters should be used (e.g., similar to [40]). For example, the ratio of video data to FEC could be adjusted, so that a higher video bit-rate can be offered while network conditions are good, while more error resilience is available when conditions degrade. The issues involved in the design and optimisation of such a system are an interesting direction for future work.

Acknowledgements

This work was supported in part by a gift from The Cisco University Research Program Fund, a corporate advised fund of Silicon Valley Community Foundation, and by the UK Engineering and Physical Sciences Research Council (Grant Nos. EP/P50418X/1 and EP/L005255/1). Thanks to Maurizio Filippone for advice on the use of parametric bootstrap.

References

[1] Cisco, Visual Networking Index: Forecast and Methodology, 2012–2017, White Paper, 2013.
[2] T. Stockhammer, Dynamic adaptive streaming over HTTP – standards and design principles, in: Proceedings ACM MMSys, ACM, San Jose, CA, USA, 2011.
[3] E. Gilbert, Capacity of a burst-noise channel, Bell Syst. Tech. J. 39 (1960).
[4] P.U. Tournoux et al., On-the-fly erasure coding for real-time video applications, IEEE Trans. Multimedia 13 (2011).
[5] S.-R. Kang, D. Loguinov, Modeling best-effort and FEC streaming of scalable video in lossy network channels, IEEE/ACM Trans. Netw. 15 (2007).
[6] M. Yajnik, et al., Measurement and modelling of the temporal dependence in packet loss, in: Proc. IEEE Infocom, 1999.
[7] H. Sanneck, G. Carle, A framework model for packet loss metrics based on loss runlengths, in: Proc. SPIE/ACM MMCN, 2000.
[8] K. Salamatian, S. Vaton, Hidden Markov modeling for network communication channels, in: Proc. ACM SIGMETRICS, 2001.
[9] P. Salvo Rossi et al., Joint end-to-end loss-delay hidden Markov model for periodic UDP traffic over the internet, IEEE Trans. Signal Proc. 54 (2006).
[10] M. Ellis, et al., End-to-end and network-internal measurements of real-time traffic to residential users, in: Proc. ACM MMSys, 2011.
[11] M. Ellis, et al., Modelling packet loss in RTP-based streaming video for residential users, in: Proc. IEEE LCN, 2012.
[12] H. Schulzrinne, et al., RTP: A Transport Protocol for Real-Time Applications, IETF, 2003 (RFC 3550).
[13] S. B. Moon, et al., Estimation and removal of clock skew from network delay measurements, in: Proc. IEEE Infocom, 1999.
[14] M. Ellis, Understanding the Performance of Internet Video over Residential Networks, Ph.D. thesis, University of Glasgow, 2012.
[15] O. Hohlfeld, Statistical Error Model to Impair an H.264 Decoder, Master's thesis, Technische Universität Darmstadt, 2008.
[16] W. Jiang, H. Schulzrinne, Modeling of packet loss and delay and their effect on real-time multimedia service quality, in: Proc. NOSSDAV, 2000.
[17] S. Tao et al., Real-time monitoring of video quality in IP networks, IEEE/ACM Trans. Netw. 16 (2008).
[18] P. Frossard, FEC performance in multimedia streaming, IEEE Commun. Lett. 5 (2001).
[19] E.O. Elliott, Estimates of error rates for codes on burst-noise channels, Bell Syst. Tech. J. 42 (1963).
[20] L.R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE 77 (1989).
[21] W. Wei et al., Continuous-time hidden Markov models for network performance evaluation, Perform. Eval. 49 (2002).
[22] F. Silveira, E. de Souza e Silva, Predicting packet loss statistics with hidden Markov models, Perform. Eval. Rev. 35 (2007).
[23] T. Friedman, R. Caceres, A. Clark, RTP Control Protocol Extended Reports (RTCP XR), IETF, 2003 (RFC 3611).
[24] M. Ellis, et al., Performance analysis of AL-FEC for RTP-based streaming video traffic to residential users, in: Proc. Intl. Packet Video Workshop, 2012.
[25] R. Turner, L. Liu, hmm.discnp: hidden Markov models with discrete non-parametric observation distributions, 2012 (R package v. 0.1-8).
[26] A.B. Downey, Evidence for long-tailed distributions in the internet, in: Proc. ACM IMW, 2001.
[27] M.R. Chernick, Bootstrap Methods: A Practitioner's Guide, John Wiley & Sons, 1999.
[28] J.S. Bendat, A.G. Piersol, Random Data: Analysis and Measurement Procedures, third ed., John Wiley & Sons, 2000.
[29] S. Tao, R. Guérin, On-line estimation of internet path performance: an application perspective, in: Proc. IEEE Infocom, 2004.
[30] W. Zucchini, I.L. MacDonald, Hidden Markov Models for Time Series: An Introduction Using R, Chapman & Hall, CRC, 2009.
[31] M. Luby, T. Stockhammer, M. Watson, Application layer FEC in IPTV services, IEEE Commun. Mag. 46 (2008) 94–101.
[32] P. Frossard, O. Verscheure, Joint source/FEC rate selection for quality-optimal MPEG-2 video delivery, IEEE Trans. Image Process. 10 (2001).
[33] C. Neumann, et al., Impacts of packet scheduling and packet loss distribution on FEC performances: observations and recommendations, in: Proc. ACM CoNEXT, 2005.
[34] V. Roca, et al., Low Density Parity Check (LDPC) Staircase and Triangle Forward Error Correction (FEC) Schemes, IETF, 2008 (RFC 5170).
[35] Y. Yu, S. L. Miller, A four-state Markov frame error model for the wireless physical layer, in: Proc. IEEE WCNC, 2007.
[36] A. Konrad et al., A Markov-based channel model algorithm for wireless networks, Wirel. Netw. 9 (2003).
[37] P. Ji et al., Modeling frame-level errors in GSM wireless channels, Perform. Eval. 55 (2004).
[38] J. Poikonen, J. Paavola, Error models for the transport stream packet channel in the DVB-H link layer, in: Proc. IEEE ICC, 2006.
[39] H. X. Nguyen, M. Roughan, Rigorous statistical analysis of internet loss measurements, in: Proc. ACM SIGMETRICS, 2010.
[40] F. Silveira Filho, et al., Adaptive forward error correction for interactive streaming over the internet, in: Proc. IEEE Globecom, 2006.

Martin Ellis is currently a software engineer at Skype, having completed a Ph.D. in Computing Science at the University of Glasgow in 2012. His research interests include network performance evaluation and real-time application performance, particularly understanding the effect of network degradations on user quality of experience.

Dimitrios Pezaros is Lecturer (Assistant Professor) at the School of Computing Science, University of Glasgow. His research is currently focusing on dynamic resource allocation for Data Centre networks, traffic classification, and measurement-based network control through Software-Defined Networking. He has attracted funding from a number of agencies (e.g., the UK Engineering and Physical Sciences Research Council – EPSRC, the London Mathematical Society) and the industry to work on the above areas and to build programmable network testbeds. Previously, he has worked as a postdoctoral and senior research associate on a number of EPSRC and EU-funded projects, in the areas of performance measurement and evaluation, network management, cross-layer optimisation, QoS analysis and modelling, and network resilience. He holds a B.Sc. (2000) and a Ph.D. (2005) in Computer Science from Lancaster University, and has been a doctoral fellow of Agilent Technologies (2000–2004). Dimitrios is a member of the IEEE and the ACM, and a fellow of the HEA.

Theodore Kypraios is currently a Lecturer in Statistics at the University of Nottingham. His background and expertise lie in both frequentist and Bayesian statistical inference. However he has (very) broad research interests which range all the way from modelling, inference, model choice and model assessment to the analyis of high-dimenstional complex real data. Most of my currrent interests lie in the development and application of computational techniques for statistical inference for partially observed stochastic processes. His work so far can be broadly split into two main areas: (i) the development of efficient computational algorithms such as Sequential and Markov Chain Monte Carlo (SMC & MCMC) respectively as well as Approximate Bayesian Computation (ABC) for missing data problems and (ii) the application of these methods to answer important scientific questions in different areas, such as in infectious disease modelling, biology, neuroscience and internet traffic modelling.

Colin Perkins is a Senior Lecturer at the School of Computing Science, University of Glasgow. His research interests include real-time networked multimedia transport protocols, network measurements, and next-generation Internet routing, and he has published more than 40 papers in these areas.

He has co-chaired the IETF Audio/Video Transport and Multiparty Multimedia Session Control Working Groups, and has published 20 Internet RFCs on topics relating to multimedia transport. He has been led projects funded by DARPA, the NSF, and various industry sources. He holds a BEng (1992) and DPhil (1996) in Electronics from the University of York, and is a senior member of the IEEE, a member of the ACM and IET, and a Fellow of the Higher Education Academy.