UNDERSTANDING THE PERFORMANCE OF INTERNET VIDEO OVER RESIDENTIAL NETWORKS

MARTIN ELLIS

SUBMITTED IN FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

SCHOOL OF COMPUTING SCIENCE

COLLEGE OF SCIENCE AND ENGINEERING

UNIVERSITY OF GLASGOW

OCTOBER 2012

Abstract

Video streaming applications are now commonplace among home Internet users, who typically access the Internet using DSL or Cable technologies. However, the effect of these technologies on video performance, in terms of degradations in video quality, is not well understood. To enable continued deployment of applications with improved quality of experience for home users, it is essential to understand the nature of network impairments and develop means to overcome them.

In this dissertation, I demonstrate the type of network conditions experienced by Internet video traffic, by presenting a new dataset of the packet level performance of real-time streaming to residential Internet users. Then, I use these packet level traces to evaluate the performance of commonly used models for packet loss simulation, and finding the models to be insufficient, present a new type of model that more accurately captures the loss behaviour. Finally, to demonstrate how a better understanding of the network can improve video quality in a real application scenario, I evaluate the performance of forward error correction schemes for Internet video using the measurements. I show that performance can be poor, devise a new metric to predict performance of error recovery from the characteristics of the input, and validate that the new packet loss model allows more realistic simulations.

For the effective deployment of Internet video systems to users of residential access networks, a firm understanding of these networks is required. This dissertation provides insights into the packet level characteristics that can be expected from such networks, and techniques to realistically simulate their behaviour, promoting development of future video applications.

Acknowledgements

I would like to thank my supervisors, Colin Perkins and Dimitrios Pezaros, for their guidance and support throughout this project. Thanks are also due to Joe Sventek, for secondary supervision in the early stages of the project, and useful advice and feedback throughout the process.

I would also like to acknowledge the Cisco University Research Program Fund, a corporate advised fund of Silicon Valley Community Foundation, and the Engineering and Physical Sciences Research Council, who funded this work.

For useful advice and discussions at various points in this work, I would like to thank Jörg Ott, Ali Begen, Theodore Kypraios, and Maurizio Filippone.

Special thanks are also due to my fellow students, for their constant encouragement and all the good times we had over the years. In alphabetical order: Paul Harvey, Paul Jakma, Alexandros Koliousis, Ross McIlroy, Helen Powell, and Stephen Strowes.

Finally, I would like to thank my family for their encouragement and support throughout my life, and especially thank Sarah Haig for her endless patience and words of support. Thank you for helping me through this process, and keeping me smiling even in the darkest days – it would not have been possible without you.

Contents

List of Figures

List of Tables

Glossary

Application Layer Forward Error Correction (AL-FEC) : Forward error correction, applied at the application layer; often used in real-time applications to provide resilience to packet loss.

Asymmetric Digital Subscriber Line (ADSL) : A variant of DSL where the upstream and downstream capacities are not equal; typically, the downstream capacity is greater than the upstream.

Cable Modem Termination System (CMTS) : A network device used to connect the Cable modems located at customers' premises (via the Cable access link) to the network of the Cable operator, and to interconnect with the rest of the Internet.

Content Distribution Network (CDN) : A distributed system of caching proxy servers used to deliver Internet content, usually containing servers located throughout the Internet, to provide efficient and fast access to content.

Data Over Cable Service Interface Specification (DOCSIS) : A technology for providing Internet service to existing Cable television networks, which is commonly used to provide Internet access to residential users.

Digital Subscriber Line (DSL) : A technology that allows digital data to be transferred over traditional telephone lines, which is commonly used to provide Internet access to residential users.

Digital Video Broadcasting (DVB) : A set of standards relating to digital television, and the organisation that maintains those standards.

DSL Access Multiplexer (DSLAM) : A network device connected via telephone lines to the DSL modems located at customers' premises, used to connect customers to the network of the Internet service provider.

Extended Gilbert Model (EGM) : A statistical model for packet loss, which directly models packet loss bursts up to a given length.

Forward Error Correction (FEC) : A technique for reducing errors in information sent over unreliable channels, by adding redundant information that can be used to correct errors in the original stream.

Hidden Markov Model (HMM) : A statistical model where the process being modelled is described by "hidden" states (which cannot be directly observed), each of which generate visible observations. The acronyms **2HMM** and **3HMM** are used here to refer to HMMs with two and three hidden states, respectively.

Hypertext Transfer Protocol (HTTP) : An application-layer protocol used for the transfer of data in the world wide web, including text, audio, and video.

Internet Control Message Protocol (ICMP) : A network-layer protocol used for transmitting control messages, such as error codes and diagnostic messages, used in tools such as ping and traceroute.

Internet Engineering Task Force (IETF) : An organisation that develops protocols and standards for the Internet.

Internet Protocol (IP) : A network-layer protocol used to connect networks together to form an internet that can deliver packets to hosts, irrespective of what network they use to connect.

Internet Protocol Television (IPTV) : A technology for delivering multimedia services over managed IP networks, typically with a required level of quality.

Interquartile Range (IQR) : A statistic describing the dispersion of a set of data, defined as the difference between the upper and lower quartiles of the data.

Kolmogorov-Smirnov (K-S) Distance : A statistic that describes the distance between two cumulative distribution functions; either between two sets of data, or between a set

of data and a particular probability distribution. The Kolmogorov-Smirnov test is a hypothesis test that tests whether two sets of data are drawn from the same distribution, or whether a single data set is consistent with a given probability distribution.

Multicast : A term that refers to the delivery of data to multiple destinations using a single transmission from the source.

Packet Loss Rate : A statistic describing the probability of packet losses; the packet loss rate for a measurement trace is defined here as the number of packets lost divided by the number of packets transmitted.

Quality of Experience (QoE) : A term that refers to the quality experienced by the end-user when, for example, watching an IPTV service. In contrast to QoS, QoE is a subjective measure, since it relates to human perception.

Quality of Service (QoS) : A term that refers to objective measures of quality in networks, such as packet loss, throughput, latency, and jitter. More generally, QoS refers to efforts to engineer networks to provide a given level of service, as defined by some objective measure.

Real-time Transport Protocol (RTP) : An application-layer protocol designed for the transport of multimedia data across the Internet (e.g., in telephony, video-conferencing, and video streaming).

Simple Gilbert Model (SGM) : A statistical model for packet loss that models errors using a Markov chain containing two states: GOOD, which never produces packet losses, and BAD, which always produces packet losses.

Transmission Control Protocol (TCP) : A transport-layer protocol that provides reliable transport in the Internet, ensuring that lost packets are retransmitted, and packets arriving at the receiver are in the correct order.

User Datagram Protocol (UDP) : A transport-layer protocol that provides unreliable transport in the Internet. UDP is often preferred to TCP for transport of real-time data, since TCP retransmissions can introduce unpredictable latency for real-time applications.

Video On-Demand (VoD) : A technology for delivering pre-recorded video content, such as catch-up TV, associated with other Internet video services like IPTV.

Chapter 1

Introduction

There has been wide deployment of Internet streaming video to residential users, both by Internet service providers (ISPs) as managed Internet Protocol Television (IPTV) services, and by content producers as "over-the-top" streaming video services. However, the performance of video streaming on end-to-end paths which include residential broadband links can be poor, and is often unpredictable. For example, when a home user watches a video streaming service, the video quality they experience can be highly variable. Relatively little is known about the performance characteristics of residential broadband networks, compared with the numerous measurement studies that have been performed on academic and backbone networks. Moreover, evaluation studies of new video streaming systems typically use models for packet loss developed for these academic and backbone networks, which remain untested on residential networks. Therefore, this work seeks to understand the characteristics of these types of networks and their effect on Internet video streaming, and to provide more accurate simulation models that better reflect the actual performance. Understanding the packet level loss and delay behaviour of these networks is essential to do this.

Currently, a number of Markov-chain models are widely used for modelling Internet packet loss. They are also widely used for packet loss simulation, where sequences of losses are generated randomly using the models and applied to the evaluation of multimedia systems, reducing the need for full-scale deployment of the systems on the Internet. These models were found to be suitable in describing the loss characteristics of academic networks, and have since been used in numerous performance evaluation studies for multimedia applications, such as error recovery mechanisms and video quality monitoring systems. However, the models have not been validated in terms of their accuracy in expressing the packet loss

characteristics of residential broadband networks (i.e., ADSL and Cable networks). Since the majority of home users use ADSL or Cable to access the Internet and deliver streaming video, to ensure that evaluations of application performance are realistic, it is important to validate the accuracy of existing models in representing the packet loss characteristics seen by residential Internet users.

1.1 Thesis Statement

I assert that packet loss simulation for streaming over residential networks is inaccurate because existing models do not capture the bursty nature of packet loss on these networks. To demonstrate this assertion, I will show the inaccuracy of existing Markov chain models under bursty packet loss by testing their goodness-of-fit against real packet loss data. Then, to overcome the limitations of existing models, I will develop a new model that more accurately models bursty packet loss.

As a first step towards demonstrating this assertion, I will capture the packet loss and delay characteristics of residential networks by performing new measurements. Then, using these measurements I will:

- Evaluate existing models for simulating packet loss, and show that these models can be inaccurate since they generate synthetic sequences that have different properties to the real data. This is important since when the models are used for simulation, the results obtained will not reflect reality. Moreover, I will explain what types of network performance cannot be accurately represented by existing models.

- Develop a new, more accurate model for packet loss simulation that explicitly models changes in packet loss and delay behaviour. This new model will demonstrate that by better understanding the network performance, simulation accuracy can be improved.

- Evaluate the performance of forward error correction (FEC) schemes for Internet video under real packet loss conditions, and show that more realistic simulation of FEC performance is possible by using the new packet loss model. This will demonstrate that the new model can be applied to a realistic application, and show the benefits of doing so.

1.2 Contributions

The contributions of this work are as follows:

- A new dataset of performance measurements from RTP streaming of high bit-rate video-like traffic to residential ADSL and Cable users. Unlike previous measurement studies that have either focused on performance within the core or ISP networks, or only obtained summary statistics of performance from residential links, the measurements presented here give in-depth insight into packet level performance. This allows more detailed analysis and, importantly, evaluation of packet loss models.

- Analysis of these new measurements, in terms of the packet loss, delay, and end-to-end capacity observed over time, at different sending rates, and between different link types. From this analysis, it is clear that there are no obvious high-level differences between packet loss and delay on ADSL and Cable links, with links of both types showing a wide range of behaviour.

- A comparison of existing widely used packet loss models using this dataset, showing that neither the widely used Gilbert models nor Hidden Markov Models can capture the extent of packet loss behaviour present in the measurements. In particular, the correlated, bursty loss patterns seen in some traces cannot be captured by the existing models.

- A new hierarchical model for packet loss, which takes into account both loss and delay data to better understand the state of the network, and uses this to more accurately model the packet loss conditions revealed by the measurements. The loss patterns obtained by using this new model for simulation are shown to more closely match the real trace data than when the existing models are used.

- An evaluation study of application-layer FEC performance, using the packet loss measurements to study the behaviour of three FEC schemes under loss conditions measured in a realistic video streaming scenario. The accuracy of packet loss models are compared within this scenario, showing that when the new two-level model is used to simulate packet loss, FEC performance is closer to the real data than when existing models are used (since existing models over-estimate the possibility of FEC recovery).

1.3 Publications

The work in this dissertation has been presented in the following papers:

- M. Ellis and C. Perkins. *Packet Loss Characteristics of IPTV-like Traffic on Residential Links*. In *CCNC 2010: Proceedings of the 7th Annual IEEE Consumer Communications & Networking Conference, Workshop on Emerging Internet Video Technologies*, Las Vegas, NV, USA, January 2010. [60]

- M. Ellis, C. Perkins, and D. P. Pezaros. *End-to-End and Network-Internal Measurements of Real-Time Traffic to Residential Users*. In *MMSys '11: Proceedings of the 2nd Annual ACM SIGMM Conference on Multimedia Systems*, San Jose, CA, USA, February 2011. [61]

- M. Ellis, D. P. Pezaros, and C. Perkins. *Performance Analysis of AL-FEC for RTP-based Streaming Video Traffic to Residential Users*. In *PV 2012: Proceedings of the 19th International Packet Video Workshop*, Munich, Germany, May 2012. [63]

- M. Ellis, D.P. Pezaros, T. Kypraios, and C. Perkins. *Modelling Packet Loss in RTP-based Streaming Video for Residential Users*. In *LCN 2012: Proceedings of the 37th Annual IEEE Conference on Local Computer Networks*, Clearwater, FL, USA, October 2012. [62]

1.4 Outline

The remainder of this dissertation is structured as follows.

Chapter 2 describes how Internet video systems are constructed, the technologies used, and the problems that exist when using these systems over residential access networks. The chapter discusses different approaches to implementing Internet video systems, and the mechanisms used to ensure video quality.

Chapter 3 presents previous work on measuring video performance, and Internet measurement in general. Then, it documents the reasons for and process of collecting measurement data for use in analysing the packet level performance characteristics of real-time streaming over end-to-end Internet paths including residential broadband links.

Finally, it describes the dataset collected in this work and used throughout this dissertation.

Chapter 4 presents analysis of the packet level characteristics of the dataset collected in Chapter 3, discussing the packet loss and delay behaviour, as well as the end-to-end capacity estimates obtained from packet-pair measurements. This analysis compares the differences between the different access technologies and other effects like time-of-day, and investigates whether network congestion is present.

Chapter 5 applies Markov models for packet loss, including the classical Gilbert model and Hidden Markov Models. This chapter evaluates the accuracy of these models for capturing packet loss on residential broadband networks, and explains the reasons for the inaccuracies when the packet loss is bursty.

Chapter 6 shows that bursty packet loss behaviour can be more accurately modelled by explicitly taking into account the transitions between the underlying states of the network. A two-level model that uses packet loss and delay information to classify network state is presented, and shown to improve modelling accuracy.

Chapter 7 applies the findings of the preceding chapters, by firstly evaluating performance of FEC schemes under the measured packet loss characteristics, and secondly by comparing the performance of the FEC schemes using different packet loss models, showing that the two-level model of Chapter 6 improves on previous models.

Chapter 8 provides a summary of the contributions and findings of this dissertation, and explores avenues for future work.

Chapter 2

Architecture of Internet Video Systems

Internet video services are now ubiquitous, having seen widespread deployment across the world in a number of forms. These include large-scale commercial offerings of traditional telecoms and Cable operators (e.g., AT&T U-verse, Telefónica Imagenio), to video on-demand and catch up TV services from ISPs (e.g., BT Vision), content providers (e.g., BBC iPlayer, Hulu), and peer-to-peer video services (e.g., PPLive). With millions of home users regularly using these services, it is clear that content delivery over IP networks is now a key component of the broadcasting world. However, given the inherent limitations of the best-effort IP infrastructure comprising the Internet (including the DSL or Cable last-mile into the user's home), ensuring delivery of acceptable quality video for this diverse range of services is a non-trivial task. To address this, techniques have been developed for various Internet video architectures to provide acceptable levels of service to users (known as *quality of experience*).

In this chapter, I describe the different architectures currently used to provide Internet video services, including managed IPTV, over-the-top video, and peer-to-peer video. I discuss the difficulties faced by these applications in terms of network impairments, and outline some of the techniques used to overcome them. Finally, I present the need for measurement of streaming video traffic on residential networks, and suggest how this should be conducted.

This chapter is structured as follows. Section 2.1 presents the various system designs typically used to deliver video over IP. Section 2.2 presents a model of the network over which these applications are used, describing the problems it introduces, and the effect these are likely to have. Section 2.3 describes the techniques used to overcome these limitations and monitor video quality. Section 2.4 explains why measurement of residential broadband

networks is needed to evaluate the performance of these techniques (and to help in the development of new ones), and provides a summary of the chapter.

2.1 System Designs

A number of different approaches have been taken to the design of Internet video systems, depending on the intended audience and type of content being delivered. Each of these approaches have their own benefits and drawbacks, in terms of user-perceived quality, infrastructure costs, bandwidth usage, and other considerations. In this section, I present three such approaches to delivering video content to home users over the Internet. Section 2.1.1 describes the managed IPTV systems typically deployed by ISPs, followed by Section 2.1.2, which discusses unmanaged services that run "over-the-top" of the ISP networks (i.e., using best-effort data services). Section 2.1.3 gives an overview of peer-to-peer based video services, followed by a summary in Section 2.1.4.

2.1.1 Managed IPTV

One example of Internet video is IPTV, a broad term that is used here to refer specifically to a managed service operated by ISPs. IPTV is defined by the ITU as: "Multimedia services such as television/video/audio/text/graphics/data delivered over IP-based networks managed to support the required level of QoS/QoE, security, interactivity and reliability" [92]. In practice, IPTV is a service provided by ISPs, telecoms companies and cable operators to replace or compete with existing broadcast television services (i.e., cable or satellite TV).

Since IPTV services are completely managed by the ISP, they typically operate using intra-domain source-specific multicast [22] for live TV services, with the IPTV source being the single sender at the root of the multicast tree, the set-top boxes as the receivers, and TV channels corresponding to multicast groups. When users want to watch a particular channel, the set-top box joins the appropriate multicast group to obtain the content. Video on-demand (VoD) content is also available, and is provided over unicast streams from video caches located within the network (as close to the receivers as possible). The IPTV traffic is often prioritised above other traffic on the network (i.e., using DiffServ [153]) to prevent video packets being queued behind bulk data traffic.

The multimedia data is typically MPEG compressed video and audio [85, 89, 74], carried within MPEG transport stream (MPEG-TS) packets [86]. These packets are carried within UDP packets for transport over the IP network. Some IPTV systems also use the Real-time Transport Protocol (RTP) [184], which provides mechanisms for the synchronisation of received data, feedback reporting via the RTP Control Protocol (RTCP), and support for retransmission requests [178]. However, not all IPTV systems use RTP, with some operators preferring to encapsulate the MPEG-TS packets directly into UDP [137, 200].

This approach to Internet video is a fully-featured system, with large infrastructure requirements. To ensure excellent video quality is delivered, each part of the distribution network needs to be optimised [38, 65], including the backbone network [32, 51] and access networks [210, 211]. Other aspects including channel-change time and failure recovery also need to be considered (these will be discussed in depth in Section 2.3). Although these requirements increase the cost of deploying an IPTV system for the operator, IPTV deployment is still increasing, with telecoms operators entering the market for TV, and existing Cable operators seeking to upgrade their existing networks to IPTV. By deploying IPTV, the operators can expand their services, offering more content with improved user experience (e.g., interactivity), while converging their networks into completely IP-based infrastructure to reduce costs.

2.1.2 "Over-The-Top" Video

An alternative approach to managed IPTV is simply to send streams "over-the-top" of best-effort IP services. This allows third parties or content providers themselves to offer services to end users, independently of the ISPs. In recent years, this type of streaming has been widely used to offer catch-up TV services, video on-demand, and (in some cases) live streaming. Much recent work has focused on developing TCP-based video streaming [213, 31], despite the unpredictability in timing caused by TCP's retransmission behaviour. The perceived advantages of using TCP transport are that TCP provides built-in congestion control, and may provide easier traversal of network address translation devices, making end-to-end communication with home users easier.

Many of the over-the-top video streaming applications use HTTP to transport the media, allowing the content to be easily watched using a web browser. Another advantage of this approach is that existing HTTP caching infrastructure and content distribution networks

can be used (rather than requiring additional infrastructure dedicated to video distribution). Examples of such services include YouTube, BBC iPlayer, and Hulu. This commercial activity has stimulated research in the area, with studies into how HTTP streaming might be improved by using parallel connections [118], better rate adaptation algorithms [117], and whether better performance can be achieved with rate adaptation being done at the client or server.

Efforts to make HTTP video streaming more adaptive to changes in network performance, and allow heterogeneous receivers to view the same content, have come together as Dynamic Adaptive Streaming over HTTP (DASH) [202], which is in the process of standardisation [87]. This is an architecture for streaming where the video content is split up into small files (chunks) corresponding to a particular time period in the stream, which are recorded in a manifest file describing the order of the chunks. To allow bit-rate adaptation, each chunk is usually encoded at a number of bit-rates (e.g., low, medium, and high quality). The clients request chunks as they play the video, and monitor the throughput of the downloads; if throughput drops, they can adapt their behaviour by requesting a lower quality version of the future chunks, reducing the bandwidth requirement, and allowing playback to continue.

Adaptive streaming players have become quite popular recently, and are used in services like Netflix, as well as in Apple's HTTP Live Streaming (HLS) [156]. There has also been interest in industry in using DASH-like services to compete with Cable TV services, rather than deploying a managed IPTV system, due to the lower infrastructure costs of DASH systems. However, there is still active research on how these players should estimate bit-rate and adapt their chunk requests [6, 116, 139], the effect on performance when multiple players compete for bandwidth [5], and how peer-to-peer technology might be incorporated into adaptive streaming [122]. A dataset of DASH content, including descriptions and chunk encodings, has also been recently published [121], and is likely to facilitate further research into these systems.

Recently, a measurement study of a range of over-the-top streaming services showed that the user experience from these systems is "far from perfect" [127], with many sessions experiencing stalls in playback. Moreover, the high start-up time (which is often over five seconds) means that it cannot yet compete with the fast channel-change provided by IPTV systems.

2.1.3 Peer-to-Peer Video

Various examples of peer-to-peer (P2P) based video systems exist (e.g., PPLive, PPStreams, SOPCast, and TVAnts, which were analysed in [7, 192]), distributing content between peers using overlay networks in an over-the-top fashion. These systems have been widely used to distribute video on-demand (i.e., non-live content) [80], but have also been used for live content such as sporting events [192, 193]. Other work has looked at how IPTV-like functionality, such as "time-shifting" (i.e., moving forwards or backwards to different parts of the video) can be implemented in peer-to-peer video systems [79]. The success of commercial peer-to-peer video systems such as Voddler [10] show that adequate performance (in terms of user experience) can be achieved using peer-to-peer video.

A related topic is the use of peer-to-peer technology to improve the performance of IPTV systems. Cooperation between ISPs and peer-to-peer applications is discussed in [222], which presents P4P: a "provider portal for applications", a technique to make routing information available to applications to improve efficiency of overlays. Without this information, it is easy to construct a peer-to-peer overlay that actually degrades performance, rather than enhancing it. It is stressed in [35] that the use of peer-to-peer technology within IPTV systems must take careful note of the physical network infrastructure; an ill-designed P2P system might involve multiple transmissions over backbone links, causing increased congestion and poor performance. This problem has been identified by the Application-Layer Traffic Optimization (ALTO) working group of the IETF [185], and work is ongoing to develop a service that will allow peer-to-peer applications to develop overlays with "better-than-random" peer selection.

A VoD system using P2P to assist delivery was proposed in [96]; this aims to make use of the potentially large amount of local storage in set-top boxes to enhance users' quality of experience (QoE). This system takes advantage of the locality of content access by viewers, allowing set-top boxes to serve content to their peers. The paper observes that non-popular content will likely still have to be served from video servers (since there are no local peers holding the content). However, since the demand for this is, as stated, comparatively low, the additional load on infrastructure will likely not be too great. Using a similar approach, [34] discusses how a "rewind function" can be provided for live TV content to users joining a channel, allowing them to view a programme from the beginning by receiving the content they are missing from local peers. Since the watching preferences of users exhibit locality

(i.e., nearby users are likely to be viewing the same content), a large portion of the bandwidth required for "rewinding" can be served by peers within the user's DSLAM. Indeed, it is claimed that "with the assistance of a P2P system, the video server load dramatically reduces to 5%" [34]. Other work has looked at how peers can be used to provide error repair for lost packets in IPTV [124], showing that the load on the retransmission servers (which would otherwise be used to recover lost packets at the set-top boxes) can be reduced by using peer-assisted repair. This approach can also exploit locality of preference, allowing the scheme to scale well with an increasing number of set-top boxes.

It should be noted that there are issues present (aside from the technical ones) in discussing localised P2P IPTV. Specifically, the privacy of users is potentially at risk, given that information on their TV viewing patterns will be accessible on the network (and potentially to their neighbours). Legal and ethical considerations in this regard are definitely worth taking into account, and it is possible that reservations on the part of ISPs might actually limit the deployment of this kind of technology.

2.1.4 Summary

In this section, I have discussed three approaches to implementing Internet video; managed IPTV, HTTP-based over-the-top streaming, and peer-to-peer video. Each of these approaches has advantages and disadvantages. Managed IPTV delivers high quality video, but needs a large investment in dedicated infrastructure. HTTP-based over-the-top streaming is more cost effective since existing web caching infrastructure can be used, but does not yet deliver the same high quality user experience, and often relies on clients storing large buffers (which increase the start-up and channel-change time, and limit its applicability to broadcasting live TV). Peer-to-peer approaches reduce the infrastructure costs, but can have potentially poor performance due to inefficiencies in the overlay networks used to distribute the content.

For the remainder of this work, I will focus on high quality RTP-based video, as the best of the available services, in terms of user experience. Although RTP streaming is widely used in intra-domain IPTV, the effect of the residential broadband networks on its performance have not been widely studied. Understanding these effects is important for future deployment of high quality inter-domain RTP streaming (this is likely to use unicast, since inter-domain multicast is not available at the moment). Since RTP streaming traffic is transported using UDP, its behaviour can be more closely associated with network performance

(unlike TCP, where mechanisms like retransmissions and congestion control can obscure the effect of the network on streaming performance). Moreover, since RTP streaming has similar traffic characteristics to other applications such as high quality video conferencing and telepresence, the results obtained from studying RTP streaming should be more general. Note however that studying the impact of access networks on HTTP streaming is an interesting prospect for future work, and insights gained in this study of RTP streaming provide a useful starting point for such an effort.

2.2 A Model for RTP-Based IPTV Systems

In this section, I describe the network model that will be referred to throughout this dissertation. This description will explain the different components of the network that end-to-end streaming video traffic will encounter, and highlights the main sources of impairments that might be present. The main elements of the network are shown in Figure 2.1. These include the video source, which might be located within the ISP network (for intra-domain systems) or on content producer or distribution networks (for inter-domain systems), the Internet and ISP networks, the ISP edge router, access link, and video receiver (e.g., set-top box).

The video source sends traffic towards the receivers from either within the ISP network, the network of a content provider, or from a content distribution network (CDN). The architecture of these core networks for IPTV systems have been discussed in survey papers including [221, 52]. The networks are typically built using IP multicast or multiprotocol label switching (MPLS), running over fibre or high-speed Ethernet links. At this part of the network, redundancy is important to ensure that it is possible to recover from link failures (this will discussed further in Section 2.3.1). Measurement studies of similar commercial ISP backbone networks have shown that these tend to be quite reliable in terms of packet loss [95], suggesting that typical traffic engineering techniques are effective, provided that the network is provisioned appropriately for the levels of traffic present.

At the edge of the ISP network, access routers will connect the core network to the access links of residential customers. The most common residential broadband Internet access technologies are Digital Subscriber Line (DSL) and Cable, which together account for 86% of OECD Broadband Subscriptions [154]. DSL is extensively used to provide IPTV service, as discussed in [210]. In its most common form, Asymmetric DSL (ADSL), a

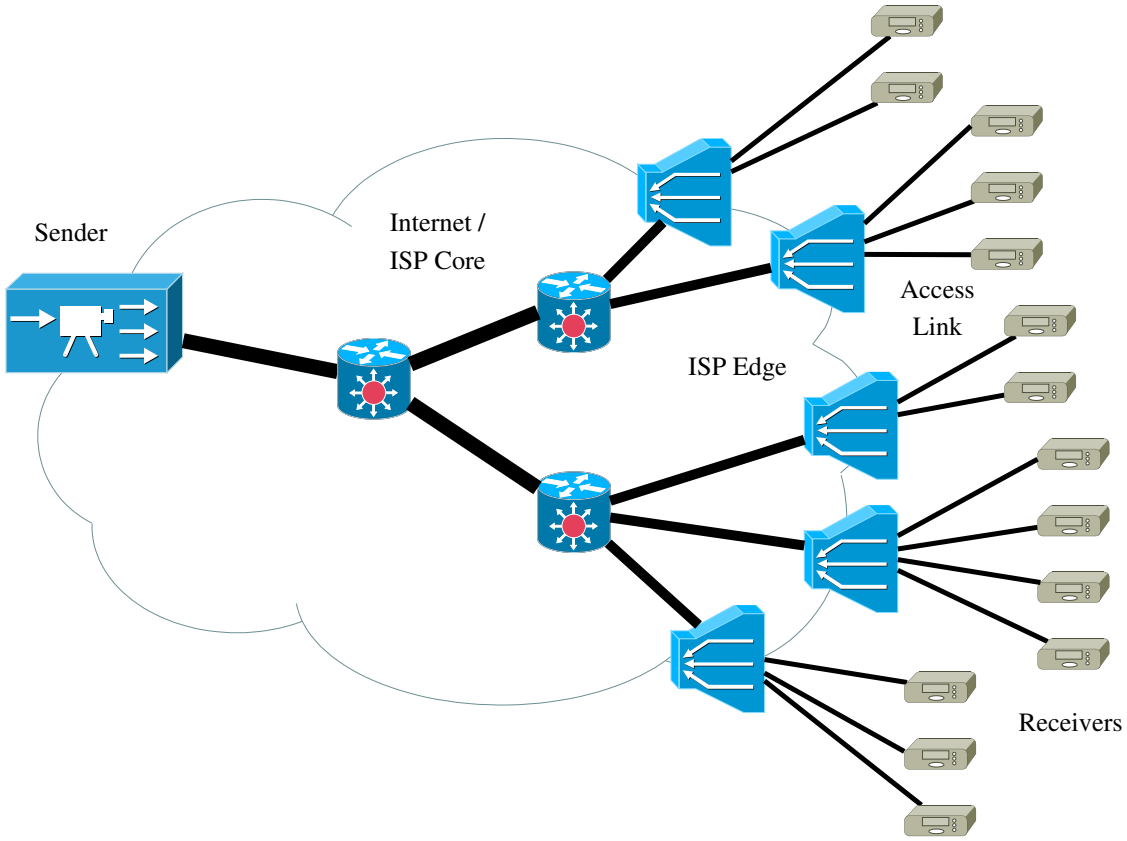


Figure 2.1: End-to-End IPTV Network Architecture: The video sender is the root of a source-specific multicast tree, transmitting TV content for different channels in separate multicast groups, which the receivers join to receive the content.

downstream bandwidth up to 24Mb/s is available (using the ADSL2+ standard), while upstream speeds are far lower. It is worth noting that the mostly asymmetric nature of DSL upstream/downstream speeds suits the IPTV model, since domestic receivers are likely to be consumers of content, rather than producers. Very high bit-rate DSL (VDSL) technology offers even higher speeds (up to 50Mb/s downstream); deploying this may allow the provision of even wider IPTV offerings.

Traditional cable TV networks (using hybrid fibre coax) can clearly deliver broadcast quality television, as well as IP data using DOCSIS [91, 90]. IPTV service is a natural progression for these networks [211]. The main difference between Cable and DSL (from the perspective of IP data transport) is that while DSL provides a point-to-point link between the DSL modem and the DSL access multiplexer (DSLAM) at the ISP premises, this is not the case for Cable. Instead, all the cable modems connected to the same Cable Modem

Termination System (CMTS) share the same medium, with downstream traffic being shared in time slices, and the cable modems requesting access to transmit upstream. This difference has important implications for the timing of packets, since the time scheduling may lead to packets being delayed until the next available slot, rather than being transmitted immediately.

Another access technology likely to make an impact on the IPTV industry is optical fibre, since fibre-to-the-premises/curb/home (FTT*x*) can provide high bandwidth directly to the consumer. Again, these high-bandwidth deployments will allow further deployment of high-definition TV and other bandwidth-intensive services. Some incarnations of FTT*x*, such as fibre-to-the-neighbourhood, or fibre-to-the-curb, involve deploying fibre-optic links to within a certain distance (less than 1km) of the customer's premises, and using another technology (e.g., coax or Ethernet) to bridge the final distance. Such a technique can reduce the cost of a fibre-based system, while retaining some of the speed.

Failures and anomalous behaviour at any point of the path between sender and receiver can lead to impairments in the video quality. These include:

1) Link failures causing route changes, which will manifest themselves as large bursts of packet loss. Although this could happen anywhere (including the access link), it will only be recoverable within the core network, using techniques like fast re-route [52].

2) Congestion within network core, the effect of the video flow interacting with all the other flows being passed through large routers with a high degree of statistical multiplexing. The video flow will often make up a small fraction of the overall traffic, so the congestion may not be noticeable in the queueing delay experienced by the flow [166].

3) Congestion at the access router, the effect of the video flow interacting with the other flows passing through the ISP edge router (i.e., DSLAM for DSL, or CMTS for Cable). Here, the video flow makes up a much larger proportion of the link bandwidth, and is likely to have a larger effect on the queues than in the core. Congestion here is more likely to be visible in the end-to-end queueing delay.

4) Line noise; this is mostly expected on the access link, since these are generally believed to be have poorer performance than backbone links, which have shown to have very low packet loss rates [95].

5) Traffic shaping; mostly expected on access link, since such conditions are common features of residential broadband contracts.

Any of the failures described above can occur on the end-to-end video path. Therefore, it is important that for a particular network (where IPTV will be deployed), the probability of these failures can be well estimated, and their effect on video quality is understood. The effect of link failures is quite well-known, and there are established approaches to dealing with these (as Section 2.3.1 will discuss). Similarly, the effect of traffic shaping is also well-known, since it is introduced by network policy. However, the effects of congestion and noise are less well-known, and distinguishing between them is non-trivial. Understanding these is important for determining the state of the network and its effect on video performance.

This section has presented a model of the network that is used to transport end-to-end Internet video traffic, describing the technologies used and the types of problems that might occur to degrade the quality of experience of end-users. Since these impairments can be expected to appear on most networks, Internet video systems need to be prepared to cope with them. Section 2.3 describes existing techniques to cope with problems at various points in the network, and later chapters will investigate how prevalent these effects are in real measurements.

2.3 Techniques for Ensuring Video Quality

The network impairments described in the previous section will have a negative impact on user experience, from visual artefacts disrupting picture quality (due to packet losses) to complete frame freezes when links fail. Therefore, operators and developers of video systems build mechanisms to cope with the problems introduced by the network.

In this section, I describe some of these techniques. Section 2.3.1 describes techniques to deal with link failures in the core networks, providing redundancy that can take effect before the customer experiences quality degradation. Section 2.3.2 discusses approaches to repairing packet loss, including both proactive (FEC) and reactive (retransmission) techniques. Section 2.3.3 examines techniques to provide consistently fast channel changes in video systems, since this is one of the most important elements of user-perceived quality. Section 2.3.4 discusses monitoring of video quality in video systems, including mechanisms used in commercial IPTV services. Section 2.3.5 gives a summary of these techniques.

2.3.1 Preventing Network Outages

Lots of work has been done on designing IPTV architectures to reduce the effects of network outages [32, 105, 51, 221, 38, 52]. Two common mechanisms for dealing with network outages, *reconvergence* and *link-layer fast reroute (FRR)*, are discussed in [51, 52]. Reconvergence involves waiting for the routing protocols to recover from a link failure. That is, allowing the routing tables to be recomputed to account for the lost link, then rebuilding the multicast distribution tree. This approach can be time-consuming, with outages lasting ten seconds or more "not uncommon" [51]. Clearly, delays of this magnitude are unacceptable for multimedia delivery, and other mechanisms are required to provide acceptable delay. Fast reroute involves providing a backup path (ideally independent from the primary path) that can carry data should the primary path fail, reducing the impact of single link failures. Improvements to the FRR approach, providing resilience to multiple link failures by using the backup path to forward traffic during primary link failures, while simultaneously initiating reconvergence, are proposed in [51]. In this way, the primary path recovers using the normal reconvergence mechanisms of the routing protocols, but the traffic does not suffer during the time this process takes (since it is forwarded using the backup).

In another study, various architecture alternatives (different technologies and topologies, as well as the role of multicast) are compared in terms of infrastructure costs [32]. In their comparison of unicast versus multicast, clear cost benefits of using multicast are highlighted, assuming a scenario where multicast can be deployed (e.g., within a single ISP). Various alternatives in IP backbone design are also compared, including the use of an existing IP network to carry IPTV, the use of a dedicated overlay on top of the existing network, or the construction of an entirely new IP network. While construction of a new network is clearly more expensive, the advantages of traffic isolation gained by use of a dedicated network (i.e., routers and links), or a dedicated overlay (dedicated links, shared routers) may be worth the expense, especially when excellent video quality is so important for acceptable user experience. This illustrates the trade-off between cost and service capability that must be considered in the design of IPTV deployments. It is also noted that the design of the network infrastructure is very closely related to the applications that use it. For example, broadcast TV requires high priority QoS to ensure interactivity, while VoD systems do not require the same real-time guarantees.

An important consideration when designing backbone networks is to consider the de-

mand for bandwidth that will be placed on them. Bandwidth demand for IPTV services is discussed in the context of VoD in [194], where a cost model is developed to determine the optimum positioning of VoD content to ensure bandwidth levels within metropolitan area networks remain at acceptable levels. By caching some VoD content closer to the edge of the network, bandwidth on the backbone links can be reduced. The study also compares the additional storage cost of caching with the bandwidth costs of having all content stored at the regional video hub offices (VHOs), looking for the optimum level of caching. Note that since VoD is typically transported over unicast (rather than services replacing traditional TV, which use multicast), it requires more bandwidth. The bandwidth demand of channel surfing is studied in [195], noting that the peak demand during a commercial break is twice the steady state multicast demand. Note that this effect is likely to be amplified by the fact that many channels typically take their breaks around the same time, suggesting that the capacity of the network should be provisioned for these "worst-case" periods. As Section 2.3.3 will discuss, the use of rapid channel-change increases the bandwidth requirements still further.

Capacity management is also discussed in [105], alongside other topics in IPTV service assurance. A trade-off between the number of channels that are served by a single multicast group is outlined. Fewer channels per group means higher bandwidth overhead, but can provide finer-grained control. Automated management of IPTV networks is also discussed in [105], including quality monitoring systems that can alert the network operator to issues before the customers report problems. Other work on video quality monitoring is discussed in Section 2.3.4.

This section has summarised the techniques that are used to provide resilience against link failures. These techniques have now become quite established, and the various options were compared in a recent survey article [65].

2.3.2 Repairing Packet Loss

The techniques described in Section 2.3.1 are used to deal with link failures occurring in the core networks. However, the loss of individual packets (which can occur at any point on the end-to-end path) must still be managed so that receivers can correctly decode the video content and play this to the end users. In this section I discuss two approaches for error recovery, forward error correction (FEC) and retransmission, and show how these approaches have been applied to improving Internet video performance. Techniques for packet loss

recovery have been described in surveys, including [47, 65]. The costs and benefits of both approaches are discussed, including the extra capacity needed to provide redundancy, the delays introduced, and the likely effect on user quality, as well as practical considerations.

The use of FEC involves sending redundant information alongside the video data, so that in the case of packets carrying video being lost, some of the (successfully received) repair packets can be used to reconstruct the original data. For Internet video, application-layer FEC is often used; this means that the FEC protection is applied at the level of application-layer payloads (e.g., the contents of RTP packets), and the recovery is determined by which packets arrive successfully.

Numerous FEC schemes have been proposed and applied to Internet video. The FEC Framework (FECFrame) working group of the IETF has defined a number of these schemes for use in multimedia applications. These include simple parity codes [196, 15], where the packets are arranged into grids (either one- or two-dimensional), and rows and columns of FEC packets are calculated by applying exclusive-OR to the data packets in the columns and rows, respectively. More complex schemes exist, such as Reed-Solomon codes [177, 119], which guarantee recovery if any k packets are received (out of n packets sent, where there are k data packets and $n - k$ repair packets). Another FEC scheme is Raptor codes, originally described by Shokrollahi [186], and described for use in IPTV in [129]. This approach implements the idea of a digital fountain, where the FEC encoder can produce an endless stream of repair packets for a given set of video data packets. This is an attractive approach, and has recently been standardised by the IETF [214]. Raptor codes are also part of a hybrid FEC scheme standardised by DVB [64], alongside 2D parity codes. One potential drawback of Raptor codes, however, is that they are patented, which may limit their use. In response to the proprietary nature of many FEC techniques, the OpenFEC project (http://openfec.org) has developed alternative open-source schemes that aim to match the performance of schemes such as Raptor codes. These include implementations of Reed-Solomon codes [119], and LDPC-Staircase codes [180], which were evaluated in a lab setting in [134].

Retransmission-based approaches to error recovery allow receivers to report which packets were lost, and ask for these to be resent (e.g., using RTCP feedback [155] and RTP retransmissions [178]). Obviously, each time a retransmission is requested, there will be a delay (equal to the round-trip time between receiver and retransmission server). As discussed

in [47, 65] it is possible to engineer the system so that this is not a problem (e.g., by placing retransmission servers close to the receivers).

2.3.3 Reducing Channel Change Time

Another important aspect of the user experience in video systems is the time taken to change channels. It is important to remember that user expectations for channel change time are typically based on their experiences of traditional broadcast television, where changing channels ("zapping") is very fast (almost instant for analogue systems), and fairly predictable.

However, unlike broadcast TV, where switching channels requires simply changing to a different input signal, multicast IPTV requires a more complicated process. As soon as the channel change request is initiated, the set-top box signals a request to change multicast groups. Once this is completed, and the new multicast stream is being received, the stream must be decoded using the appropriate MPEG decoding algorithm, as explained in [74]. Detailed analyses of the different elements of channel change times are given in [17, 187]. Essentially, when joining a multicast video stream, certain key data needs to be obtained before the video can be decoded and displayed. Once the receiver starts receiving the multicast stream, it needs to wait until it receives all the key information before it can decode the video.

The problem for user experience is that the time taken is larger than for traditional TV, and more importantly, the time is not consistent (it might be sometimes fast, sometimes slow). Experiments into subjective assessment of video quality were described in [114], aiming to determine a mapping between channel "zapping" time and the Mean Opinion Score metric (MOS), as defined in [88]. This study suggests that channel change time should be no longer than 430ms.

Efforts to improve channel change times in IPTV are discussed in [17, 18]. Their approach involves sending a rapid (faster than normal) burst over unicast while the multicast join is taking place. This allows the receiver to start decoding and displaying the video before the multicast stream is available. This approach has been standardised by the IETF as "Rapid Acquisition of Multicast Sessions" [212]. Since this technique involves using caches close to the receivers to provide bursts of video content, it is integrated with the retransmission servers that are used to provide recovery from packet losses, as discussed in [17]. A

case study of the performance of this solution on an operational IPTV network is described in [138], where the challenges of deploying this mechanism in the real world are explained.

2.3.4 Monitoring Video Quality

As the previous sections have discussed, ensuring acceptable video quality is a key challenge for Internet video systems, and is vital to maintain customer satisfaction. However, when using the techniques described earlier, it is also important to monitor the network to understand the impairments that receivers have to cope with, and the quality experienced by end users.

Some approaches that have been proposed include monitoring of RTP-based IPTV systems using the RTCP reports [19], and possibly exploiting the tree-based structure of IPTV multicast to use network tomography techniques to isolate faults [20]. Another study that describes a quality monitoring system used within a commercial IPTV network is described in [130]. In this system, data from various sources, including low-level network statistics, video quality reports, and customer service tickets are correlated to give insight into the overall network performance, and isolate the location of problems. Similar proposals have also been made for mobile networks [43] and for networks with both voice over IP and IPTV traffic [4].

Other work has proposed monitoring the user-perceived video quality, rather than just the network-level effects. A system to measure the user-perceived quality, taking into account the distortions in quality caused by losing particular packets, is discussed in [206]. This system, which was designed to operate in real-time, aims to provide fairly lightweight monitoring without having to inspect the video packets in depth (i.e., since the overhead of deep inspection would make the system unscalable). A summary of standardisation efforts for video quality metrics is described in [218]. While some progress has been made on objective assessment of video quality (i.e., understanding the quality only by looking at measured network conditions such as packet loss and delay), this remains an area of active research [39].

2.3.5 Summary

In this section, I have discussed techniques used to improve, maintain, and monitor quality in Internet video systems. Within the core networks, the effect of link failures can be

reduced by employing redundant links with fast reroute and other traffic engineering approaches, and careful design of the networks and planning the capacity usage can ensure the networks are able to cope with demand. At the ISP edge, and on the access networks, where there tends to be more problems due to packet loss, loss recovery techniques like FEC and retransmission are applied. The application of these recovery techniques also needs to be carefully planned, since both can have significant overhead (FEC has a constant bandwidth overhead, while retransmission has a cost in terms of both bandwidth and delay, but only when repairs are needed). The same infrastructure used for retransmissions can also be used to provide rapid channel change, addressing one of the most important requirements for good IPTV service. Monitoring the quality of the service is also important for ensuring customer satisfaction in commercial IPTV services. Quality monitoring systems work both in terms of network-level metrics, identifying the sources of network failures, as well as high-level metrics, understanding the absolute video quality as perceived by customers. Awareness of both the low-level network performance and the user-perceived quality (and the correlation between these) is critical to providing excellent IPTV user experience.

2.4 Discussion & Summary

In this chapter I have discussed the increasing popularity of Internet video systems, and described the different approaches taken to implementing these systems. I have shown a model describing the end-to-end path between sender and receiver, and discussed the various sources of failures along this path. I have also discussed the mechanisms used to cope with these failures, including redundancy and rerouting in core networks, and packet loss recovery techniques like FEC and retransmissions to deal with end-to-end packet losses (which may be due to noise on the access links, or congestion elsewhere in the network, if it is not provisioned appropriately).

While all these techniques are understood (particularly by the operators themselves), there is very little public measurement data available on the performance of these networks, and on the impact of network-level impairments on video quality. Moreover, the development of new loss recovery techniques for streaming video is limited by this lack of data, and although simulation models are often used to evaluate performance (e.g., for FEC, as Chapter 7 will discuss), their accuracy has not been widely tested. Developers of new video

systems and error recovery techniques need to understand the performance characteristics of the networks that will carry their applications and be able to realistically simulate these characteristics, so that they can test and optimise the applications. Since there is little network performance data already available, it is clear that to understand the performance of the video systems, networks, and error recovery techniques described in this chapter, it is necessary to take measurements of streaming performance, as Chapter 3 will discuss in more detail.

Chapter 3

Measuring Packet-Level Characteristics of Streaming to Residential Users

Despite the widespread deployment of Internet video and IPTV, little published performance data currently exists for streaming high quality Internet video to residential users. Data on streaming performance will provide valuable input to the design of new protocols and applications, such as congestion control and error correction schemes, and sizing playout buffers in video receivers. Since no suitable data are already available, I have collected a set of measurements, which give insight into the network effects experienced by streaming video traffic received by residential Internet users. Analysis of these measurements will provide insight into the effect of the network on streaming video applications, develop more accurate network simulation models, and therefore provide the means to improve Internet video performance for residential users. An anonymised version of the dataset, including full packet traces, can be downloaded at http://martin-ellis.net/research/datasets.

In this chapter, I introduce my dataset of the packet level characteristics of end-to-end synthetic video traffic. The traffic was transmitted using UDP over the open Internet from a well-connected server to residential hosts connected via a number of ISPs, using both ADSL and cable modem connections, in the UK and Finland. First, I present background on the existing work on measurement of video streaming, performance measurements of packet loss and delay on the Internet (including residential broadband networks), and techniques to estimate link capacities. Then, I outline the approach I have taken to measuring streaming performance, describe the traffic characteristics and the links measured, and explain the formats of the measurement traces themselves (including post-processing steps).

This chapter is structured as follows. Background on measurements of streaming video systems, and more general work on Internet measurement, is presented in Section 3.1. The rationale and approach taken to conducting the measurements is presented in Section 3.2. The trace formats used are described in Section 3.3. Section 3.4 describes the post-processing applied to the raw data for later analysis. Section 3.5 summarises the chapter.

3.1 Background

In this section, I describe previous work on Internet measurements. Specifically, I focus on measurements of Internet video and IPTV systems in Section 3.1.1, measurements of packet loss and delay in the Internet in Section 3.1.2, and work on estimating capacity and available bandwidth on the Internet in Section 3.1.3. Section 3.1.4 gives a summary of the section.

3.1.1 Internet Video Measurement

A number of studies have looked at the performance of video streaming over access networks (such as dial-up, DSL, and Cable). The authors of [128] study streaming of low bit-rate videos to dial-up users across the United States, investigating the packet loss, delay, and reordering behaviour of a large number of short (ten minute) measurement sessions. This study found that packet loss was typically quite low, with 38% of traces seeing no loss, and 75% of traces showing less than 0.3% loss. Moreover, their analysis of loss run-lengths (i.e., the number of packets lost consecutively) shows that a large number of loss runs consist of a single packet. Further analysis showed that the distribution of loss run-lengths is *heavy-tailed*, suggesting that there are a large number of short loss runs, and a small number of very large ones (this concept will be discussed in more detail in Chapter 4). This study also examined the round-trip time (RTT) calculated between sender and receiver, again finding evidence of heavy-tailed distributions, with most RTTs being less than one second, but with a minority of very large RTTs (more than 30 seconds). More recently, video streaming performance from home users towards a well-connected server was investigated in [176]. In this study, the senders were connected to home networks over 802.11 home wireless routers, which were in turn connected to the Internet via DSL and Cable access links. Their results of packet loss, which look at the wireless and wired segments of the path separately, show that wired loss run-lengths are quite short (as in [128]), although not dominated by single

packet losses. The losses on the wireless segment, on the other hand, show much longer loss runs, with 90% longer than five packets. The clear differences between wired and wireless performance re-iterate the importance of isolating the effects of wireless networks, which tend to have less predictable performance than wired links.

Since the introduction of IPTV services, there have also been studies that measure their performance. These include passive measurements of IPTV network performance taken within the network [84, 137], passive measurements taken by receivers [219, 220], and active measurements conducted within the networks [13]. The studies that measure performance within the network tend to conclude that performance is very good, with few distortions (e.g., [84] states that no packet loss was encountered throughout their measurement period). However, these findings may be due to the observation points in the measurements (i.e., [84, 137] measure at the at the ISP edge, rather than at subscriber premises), or the access technology used (the network measured in [13] uses an Ethernet last-mile, which might be expected to provide better performance than either DSL or Cable). In contrast, studies of IPTV services using DSL and Cable, measured at the customer premises (e.g., [220, 219, 200]), tend to show that performance is quite good, although not perfect. These measurements show that there will be periods where the video quality experienced by end-users will drop, and will require appropriate recovery mechanisms.

Another area of Internet video that has been widely measured is peer-to-peer (P2P) based video streaming, as discussed in Section 2.1.3. Numerous studies of P2P video streaming systems have been conducted. One of the largest of these systems, PPLive, was extensively studied in [80]. Some of the requirements for a successful P2P video streaming system identified by this study include the need for rapid channel change times to maintain user satisfaction (as discussed in Chapter 2). They also identify the importance of having peers with a high enough data rate to maintain the system. In the typical user environment of upload-restricted residential links, this implies some fixed infrastructure may be required to fill the gap (note that this complements the idea described in [3], where the performance of managed IPTV systems is improved by working with P2P systems). The performance of PPLive is studied alongside another P2P streaming video system, SOPCast, in [7]. This study highlights some of the drawbacks in using peer-to-peer distribution for video, such as lack of fairness (where peers with higher upload capacity end up contributing the most to the system), and inefficiencies (where a sub-optimal overlay network is created, leading

to increased latency and bandwidth use). A similar comparison study, of four P2P video streaming systems (PPLive, PPStream, SOPCast, and TVAnts) is described in [192]. The study discusses the transport protocols used (SOPCast uses mainly UDP, while the others are TCP-dominated), as well as the varying overheads and video download policies of each system. An interesting point about this study is that it measures video streams of a live event (i.e., world cup football matches). This is much closer to broadcast TV than many other studies that have been conducted in this field. A later study by the same authors [193] compared the performance of P2P live streaming in both France and Japan, noting that live streaming creates additional challenges for finding appropriate peers (i.e., peers in Europe may have particular content, but cannot share it with peers in Asia because the latency incurred in transferring it means it would arrive too late to be useful).

Some measurement studies have also been conducted for over-the-top video streaming services. The majority of these have looked at YouTube, aiming to understand user behaviour and system organisation (e.g., [33], [1]). Other studies have looked at traffic measurements too, such as [230], which passively measured all YouTube traffic on a university network, and [169], which monitored traffic from YouTube and DailyMotion (another video sharing site) within an ISP network. The results of [169] are particularly interesting, since they discuss per-flow measurements of round-trip time, transmission rate, and packet loss. Their packet loss results are differentiated between the access link and the rest of the network, and show that most flows experience less than 1% loss on the access link, but higher loss elsewhere. The higher loss seen on the backbone suggest that over-the-top video flows experience congestion, with 60–80% of flows experiencing more than 1% packet loss. A recent study of Apple's HTTP Live Streaming (HLS) in a commercial mobile Internet TV system [123] showed that performance was reasonably good, although 4.5% of streams experienced stuttering (i.e., buffer under-runs), which is known to be one of the most important factors for user satisfaction [50]. Moreover, the start-up times for HLS were found to be fairly high (60% of video playbacks were longer than five seconds). Start-up time and time spent buffering are also discussed two wider studies of a large number of video content providers. In [50], the authors found that the most important metric for video quality was the "rebuffering ratio" (i.e., time spent buffering), looking at the short-term and long-term effects on user engagement. A later study with new data from the same authors [127] suggests there are still significant challenges in delivering acceptable quality in over-the-top streaming, with 40%

of videos spending at least 1% of the time buffering, with start-up times being more than five seconds in 23% of cases. Their conclusions are that further network coordination is needed, to allow the video clients to make more informed decisions about adaptation (i.e., which bit-rates to choose, or whether to change the CDN from which the content is accessed).

3.1.2 Internet Loss/Delay Measurement

Packet loss and delay are two of the most important characteristics of the network from the perspective of real-time applications like streaming video. Since receivers running these applications require a steady stream of packets, packet losses and large variations in delays are likely to be disruptive to application performance. Therefore, understanding the loss and delay characteristics of the target network is important before deploying applications. These metrics are not only important for streaming video applications, since they also affect other Internet applications. As this section will discuss, they can be measured independently of video streaming systems.

Measurements of Internet performance have been conducted since the early days of the ARPANET, with experiments studying packet sizes, utilisation, loss rates, and delay [107]. Later, as early multimedia applications such as audio streaming and conferencing started to be deployed, a number of studies investigated the performance of the Internet at that time when carrying such traffic [25, 27, 26]. Other work focused specifically on measuring network performance, with the intention of understanding problems in applications such as audio streaming [223, 141, 224] and TCP performance [159, 12]. Measurement and analysis results have also been presented in order to better understand the performance of the Internet, in terms of packet loss [28], delay [146, 83, 29], and routing performance [158].

A number of projects have been conducted to measure the Internet at a larger scale than these previous studies. These include the National Internet Measurement Infrastructure (NIMI) [165, 163], a distributed system designed to facilitate network measurements, which was used by various academic sites to perform distributed measurements, similar to those using the earlier network probe daemon [160]. Another project looking into real-time passive measurement of commercial Internet backbone traffic was carried out by Sprint Labs. This effort resulted in a number of interesting studies giving insight into the behaviour of commercial networks (in contrast to the prior work, which typically focused on academic networks). These include studies of traffic loads and packet sizes [67], delay within a single

router [157], delay between routers [37], out-of-sequence (i.e., loss and reordered) packets [95], and network failures [133].

Although measurement studies of backbone networks are useful, they do not necessarily represent the network conditions faced by typical end-users of streaming video applications (i.e., home users). A number of measurement studies of residential broadband networks have also been conducted, which give some insight into this. The TCP traffic of around 1300 ADSL users was passively measured over a 24-hour period in [188]. The results describe the user activity as quite low, and highly skewed in terms of volume and duration (i.e., while many users use very little of their bandwidth, others are "heavy hitters", consuming far more and using peer-to-peer file sharing applications more extensively). This study notes that link utilisation across the range of clients (including the heavy hitters) is typically far less than capacity. Their explanation is that individual users limit the bandwidth available to their peer-to-peer file sharing applications, preventing large levels of bandwidth usage. A later study conducted on a different ISP [131] found different behaviour, with HTTP traffic, rather than peer-to-peer, being the most common. 25% of the HTTP traffic was found to be Flash video (i.e., YouTube and similar web video) traffic, and this paper may provide some evidence of a shift from P2P downloading of videos to streaming, a trend confirmed in [168]. Another important finding in this study is that the delay experienced by DSL hosts is dominated by the delay on the access link. This is important, since it has implications for the design of applications that require low delay, such as streaming video.

An active measurement approach was used to do large-scale uncooperative measurement of residential broadband users in [49]. In this work, streams of probe packets were sent to ranges of IP addresses, allowing measurements of bandwidth and delay to be calculated. This study found that there are large differences between the performance of different ISPs, reflecting the different policies employed on their networks. Other interesting findings were that there are quite large latencies on access links (possibly due to large buffers at access routers [72]), and that packet loss rates were usually quite low. Another active measurement project, which uses cooperative measurements, is Netalyzr [115]. This allows users to access a website, and perform a number of tests that estimate the performance of the connection to the server. This study has been able to collect a large amount of information on the performance of different ISPs, including IPv6 availability, bandwidth and latency, packet loss and reordering, as well as performance of applications such as HTTP. Another wide-scale

measurement study of access network performance, supported by the US Federal Communications Commission [204] combined wide-scale measurements for *breadth* (involving over 4000 residential subscribers across 16 ISPs), and augmented these with more detailed measurements of 16 subscribers on 3 ISPs to give *depth*. This study found that the last-mile (i.e., the DSL or Cable access link) dominates the delay, with DSL links showing higher delays. This is somewhat surprising, since Cable is a shared access medium, but the increased delay seen on DSL lines is likely due to the use of interleaving for protection from physical layer noise.

In [101], passive measurements of home users' Internet performance were augmented by a questionnaire in which the participants gave feedback on their perception of network performance, which was then correlated with the low-level measurement data [102]. By recording the user's subjective opinion of network performance, the effect of network impairments on user experience can be better understood, and might lead to the development of better objective metrics for network performance.

As levels of interest in Internet measurement have increased, attempts have been made to document and standardise best practices across the discipline [162, 42]. The IP Performance Metrics (IPPM) working group of the IETF have produced number of documents pertaining to particular metrics of Internet performance. RFC 2330 [164] specifies a framework for the other documents, outlining and defining key concepts related to Internet measurement, and highlighting important issues (for example, the complexities of dealing with time in measurements, sampling and statistics, and dealing with errors). Documents dealing with various metrics have been developed by the IPPM group, including packet loss (RFC 2680 [9]), loss patterns (RFC 3357 [113]), loss burstiness (RFC 6534 [57]), one-way delay (RFC 2679 [8]), packet delay variation (RFC 3393 [48]), and packet reordering (RFC 4737 [145]). Other work on assessing measurement accuracy [197, 198] aims to improve the performance of packet loss measurements by using a probing strategy that more effectively samples the loss process (specifically, using probes with geometrically distributed inter-packet intervals, rather than Poisson distributed intervals as previous work has suggested). Similarly, other work has focused on calibrating and improving the accuracy of one-way delay measurement [161, 46], which is prone to difficulties due to issues with lack of clock synchronisation between the machines sending and receiving measurement traffic. A widely used solution, allowing estimation and removal of clock skew in one-way delay measurements, was pro-

posed in [142], and later improved upon in [106]. A range of issues affecting measurement accuracy, and best practices to adopt are discussed in detail in [42].

3.1.3 Bandwidth Estimation Techniques

In many Internet applications, it is important to know the bandwidth available between a sender and receiver. For example, in a streaming video application, the sender may want to know how fast to send the stream, so as not to cause congestion for the receiver. However, this is rarely known beforehand and can vary over time, so a number of techniques have been developed to estimate the bandwidth of a particular Internet path.

Two related yet distinct metrics associated with the bandwidth on a path are *capacity* and *available bandwidth*. The capacity of a path is the minimum transmission rate of all the links in the path (i.e., the maximum rate that can be transmitted end-to-end). The available bandwidth at a particular time is the minimum spare capacity available of all the links in the path (i.e., the maximum rate that can be transmitted at that time, given the other traffic using the links on the path). Precise definitions of these metrics are given in [54].

A large number of methods, techniques, and tools have been proposed to measure both capacity and available bandwidth. Surveys of these techniques include [173, 93, 41, 75]. End-to-end capacity estimation techniques tend to focus on the timing dispersion of packets sent in back-to-back sequences, either in *packet-pairs*, longer *packet trains*, or variants such as *trains of packet pairs* [136], as discussed in depth in [173]. These work by comparing the difference between sender timestamps between groups of packets (which should be close to zero, since they are sent back-to-back) with the difference in receiver timestamps. The dispersion in timestamp differences is caused by queueing at the bottleneck link of the path. The capacity of this link can be estimated by dividing the packet size (the number of bytes to be transferred) by the dispersion (the time taken to transfer those bytes). A detailed explanation of this process, and the advantages of using shorter trains (i.e., just packet-pairs) are discussed in [53, 54]. In short, packet-pairs are preferable since they are less sensitive to the cross-traffic present on Internet paths. Another technique augments the timing dispersion by also studying the changes in delay between probe packets [104].

A range of tools for estimating available bandwidth have been presented, including [203, 179, 40, 76]. Surveys and comparisons of the effectiveness of these techniques include [173, 41, 75]. An interesting use of available bandwidth estimation is to choose paths in overlay-

based video streaming using available bandwidth estimates, as discussed in [94]. Although path selection may not be an option for residential users connected by ADSL or Cable, this approach might allow applications to avoid congestion elsewhere in the network, or to adapt other aspects of the video stream (e.g., reduce to a lower quality) based on feedback about the available bandwidth.

3.1.4 Summary

In this section, I have looked at measurement studies of video streaming applications, more general Internet measurement studies considering packet loss and delay, and techniques to estimate bandwidth on Internet paths. These techniques provide useful insight for their particular applications, but have not been applied together in understanding the performance of streaming video applications as experienced by residential Internet users. Furthermore, although some of the measurement studies have provided access to their datasets, these are generally quite limited (e.g., reporting only summary statistics of measurement traces, rather than full packet traces). In the remainder of this chapter, I describe my approach to conducting a measurement study of real-time streaming to residential users, and describe the format of the data that is available to the research community.

3.2 Methodology

To measure performance of real-time streaming to residential users, I use an active measurement approach, sending RTP traffic [184] (containing video-like payloads) over UDP/IP. This gives precise control of packet size and timing, allowing generation of traffic patterns that match commonly used video formats (standard- and high-definition MPEG video [81]). To perform the measurements, I have used a dedicated platform that can be deployed into residential premises. This platform is built using Soekris net5501 single-board computers running FreeBSD 7 with a custom measurement application. These devices are low-power, easily transported, and can be connected to a home network with zero configuration. This provides an environment with known timing behaviour to reduce the variability in performance of home computers with differing configurations and running a variety of other applications.

I focus primarily on end-to-end packet loss and delay performance, since this is what applications experience, and what drives user perception of the video quality. The measurements also include hop-by-hop probing, using low-rate TTL-limited packets to solicit ICMP responses from intermediate routers, to attempt to give some insight into the location of loss events, and how timing disruptions evolve across a network path. Finally, one-way packet-pair probing is also used to estimate the capacity of the network path. Packet pair has well-known limitations [54], but because of its ease of implementation, it has been widely deployed in some commercial streaming systems. With these results, it is possible to explore the accuracy of the technique on paths where the edge link capacity is known.

I describe two datasets, *dataset-A* and *dataset-B*, collected between July 2009 and September 2010. The same general methodology was used for each, although specific details evolved over time. Tables 3.1 and 3.2 show the residential links hosting receivers, the rates measured, trace schedules, and durations. Link *adsl5* is the same physical link as *adsl1*, but was upgraded by the ISP during the course of the study; the others are distinct links. All the links are located in the UK, with the exception of *finadsl0* and *fincable0*, which are located in Finland. The server is a well-connected machine at the University of Glasgow. In total, around 146 million packets were sent in *dataset-A* within $\sim$2300 traces; in *dataset-B*, around 96 million packets were sent within $\sim$1600 traces.

The measurement traffic is constant bit rate RTP/UDP flows where the RTP sequence number and logical timestamp are augmented with accurate transmission timestamps. Transmission and reception times are logged at the receiver for later analysis. Sender and receiver clocks are synchronised using NTP, allowing measurement of one-way delay variation, but not accurate one-way delay, as discussed in Section 3.4.

Most of the participants hosting the measurement devices have monthly-limited or time-of-day-capped bandwidth usage quotas imposed by their ISPs. Extreme connection throttling (to a few kb/s) and excess use fees are possible on exceeding the quota. While this was not considered before the collection of *dataset-A*, in *dataset-B* the bandwidth consumption of the traces was limited to around 2GB per day for each link, a value that avoids exceeding the volunteers' quotas. Given the video rates being simulated, the total bandwidth consumed per day B_{day} may be calculated as $B_{day} = N \times T \times (B_{1Mb/s} + B_{2Mb/s} + B_{5Mb/s})$, where N is the number of traces per rate per day, and T is trace length.

To give a snapshot of activity at each time, and capture the variation over different times

Dates	Link	Rate (Mb/s)	Time				Trace Length (minutes)
2009/06/27-	*adsl1*	1	Hourly at :50				1
2009/07/18	8Mb/s	2	03:15	10:15	15:15	20:15	10
		4	05:15	12:15	17:15	22:15	10
		6	05:35	12:35	17:35	22:35	10
2009/07/07-	*adsl2*	1	Hourly at :30				1
2009/07/13	2Mb/s	2	04:15	11:15	16:15	21:15	10
2009/06/27-	*cable1*	1	Hourly at :30				1
2009/07/04	2Mb/s	2	04:15	11:15	16:15	21:15	10
2009/07/16-	*cable2*	1	Hourly at :05				1
2009/07/22	10Mb/s	2	04:10	11:10	16:10	21:10	5
		4	04:20	11:20	16:20	21:20	5
		6	04:30	11:30	16:30	21:30	5
		8.5	04:40	11:40	16:40	21:40	5
2009/09/12-	*adsl1*	1	Hourly at :50				1
2009/09/18	8Mb/s	2	03:12	10:12	15:12	20:12	5
		4	03:20	10:20	15:20	20:20	5
		6	03:32	10:32	15:32	20:32	5
2009/09/12-	*adsl3*	1	Hourly at :05				1
2009/09/18	2Mb/s	2	04:12	11:12	16:12	21:12	5
2009/09/22-	*adsl4*	1	Hourly at :05				1
2009/09/28	8Mb/s	2	04:12	11:12	16:12	21:12	5
		4	04:20	11:20	16:20	21:20	5
		6	04:32	11:32	16:32	21:32	5
2009/10/07-	*adsl5*	1	Hourly at :50				1
2009/10/13	24Mb/s	2	03:12	10:12	15:12	20:12	5
		4	03:20	10:20	15:20	20:20	5
		6	03:32	10:32	15:32	20:32	5
2009/10/07-	*adsl6*	1	Hourly at :05				1
2009/10/13	8Mb/s	2	04:12	11:12	16:12	21:12	5
		4	04:20	11:20	16:20	21:20	5
		6	04:32	11:32	16:32	21:32	5

Table 3.1: Measurement Schedule (*dataset-A*)

Dates	Link	Rate (Mb/s)	Time	Trace Length (minutes)
2010/04/25-2010/05/01	*adsl5* 24Mb/s	1	(02,05,08,11,14,17,20,23) at :22	4
		2	(02,05,08,11,14,17,20,23) at :28	4
		5	(02,05,08,11,14,17,20,23) at :34	4
	adsl6 8Mb/s	1	(02,05,08,11,14,17,20,23) at :40	4
		2	(02,05,08,11,14,17,20,23) at :46	4
		5	(02,05,08,11,14,17,20,23) at :52	4
2010/05/13-2010/05/19	*finadsl0* 8Mb/s	1	(02,05,08,11,14,17,20,23) at :04	4
		2	(02,05,08,11,14,17,20,23) at :10	4
		5	(02,05,08,11,14,17,20,23) at :16	4
	cable2 10Mb/s	1	(02,05,08,11,14,17,20,23) at :22	4
		2	(02,05,08,11,14,17,20,23) at :28	4
		5	(02,05,08,11,14,17,20,23) at :34	4
2010/05/25-2010/05/31	*cable3* 20Mb/s	1	(02,05,08,11,14,17,20,23) at :22	4
		2	(02,05,08,11,14,17,20,23) at :28	4
		5	(02,05,08,11,14,17,20,23) at :34	4
2010/06/12-2010/06/18	*fincable0* 5Mb/s	1	(02,05,08,11,14,17,20,23) at :04	4
		2	(02,05,08,11,14,17,20,23) at :10	4
		5	(02,05,08,11,14,17,20,23) at :16	4
	cable4 20Mb/s	1	(02,05,08,11,14,17,20,23) at :22	4
		2	(02,05,08,11,14,17,20,23) at :28	4
		5	(02,05,08,11,14,17,20,23) at :34	4
	cable5 20Mb/s	1	(02,05,08,11,14,17,20,23) at :40	4
		2	(02,05,08,11,14,17,20,23) at :46	4
		5	(02,05,08,11,14,17,20,23) at :52	4
2010/08/01-2010/08/07	*adsl4* 8Mb/s	1	(02,05,08,11,14,17,20,23) at :22	4
		2	(02,05,08,11,14,17,20,23) at :28	4
		5	(02,05,08,11,14,17,20,23) at :34	4
2010/08/28-2010/09/04	*adsl7* 8Mb/s	1	(02,05,08,11,14,17,20,23) at :22	4
		2	(02,05,08,11,14,17,20,23) at :28	4
		5	(02,05,08,11,14,17,20,23) at :34	4

Table 3.2: Measurement Schedule (*dataset-B*)

Rate (Mb/s)	Size (bytes)	Spacing (ms)
1	1316	10
2	1316	5
4	1128	2
5	1316	2
6	752	1
8.5	1128	1

Table 3.3: Sending Rates, Packet Sizes and Spacings

of day, N was chosen to be 8; this allows T to be as long as 240 seconds. The eight traces per-day capture enough of the diurnal variation seen in the short, hourly traces used previously, and their increased length gives better insight into the packet delay distributions and the variations in loss and delay within a trace.

Both datasets used a range of transmission rates, chosen to be representative of both standard-definition and high-definition video. Due to limited scheduling granularity in the measurement system, different packet sizes were required to achieve certain transmission rates (see Table 3.3). In *dataset-A* the chosen rates cover the full range of bandwidth of the links; *dataset-B* used a more limited set of rates, matching common MPEG-TS packetisation rates, that were achievable with fixed packet size. This gives less coverage of the extremes of link capacity, but removes the influence of packet size on the results. Similarly, trace lengths are also standardised in *dataset-B*.

TTL-limited hop-by-hop probes and packet pair measurements were taken as part of *dataset-B*; *dataset-A* is end-to-end only. Logs of which packets were sent with reduced TTL were kept by the sender, along with records of the timing of the corresponding ICMP responses. The TTL-limited packets were sent at a rate of once per second to each of the responsive routers on the path (determined by probing each of the routers on the path before starting the measurement). This low rate was chosen to avoid overloading routers, and to ensure that only one ICMP response was outstanding at any time, to ease matching of response packets to probes.

The packet-pairs were sent every ten seconds, by generating two packets back-to-back, then leaving a gap of twice the usual interval before the next packet to maintain the average

sending rate. The server logs the timestamps when both packets were sent, as well as the logical RTP timestamps of each of the packets. These are combined with the arrival timestamps to estimate the path capacity [54].

3.3 Trace Formats

The datasets are arranged hierarchically, with a directory for each link, and within these, a directory for each rate. The log files are found within these "rate" directories. Reception log files are named according to the time at which they were captured (e.g., `20100501-0222.log` was captured on May 1st 2010, at 02:22). For each trace, another file shows the anonymised output of a traceroute from the receiver to the sender, taken at the end of the trace. These are named according to the time of capture, with suffix `.rs.traceroute` (e.g., `20100501-0222.rs.traceroute`).

In *dataset-B*, the sender also generates log files, named similarly based on the start time of the trace. The file extension represents the type of file (either `.path`, `.pathprobes`, `.packetpairs`, or `.icmp`). Additionally, *dataset-B* includes traceroutes from sender to receiver, stored with file extension `.sr.traceroute`.

The format of the packet trace files captured at the receiver and present in both datasets is shown in Figure 3.1a. Each line begins with the capture timestamp (all timestamps measure seconds since 1970). The first line is a header line. The following `(rtp...)` lines report capture of each RTP packet, giving the decimal values of the RTP header fields [184] with a 1MHz RTP timestamp clock. The `sender_ts` fields is the transmission time inserted by the sender.

Figure 3.1b shows the format of the additional trace files present in *dataset-B* relating to packet-pair and hop-by-hop probing. In particular:

- Before the start of the trace, the sender sends five RTP packets to each hop in turn, checking for multiple IPs per hop, logging the IP addresses of the responses, and timing out if no response is received after one second. Files with the `.path` extension show this mapping from TTLs to (anonymised) router IP addresses; this is used to match the received ICMP messages to the correct TTL-limited packets, as discussed in Section 3.4.2.

```
dataset-B/adsl5/cbr1.0/20100501-0222.log:
  1272676920.206448 (airmtrecv (version 4.0.1) (build r1000))
  1272676921.297392 (rtp (v 2) (p 0) (x 1) (cc 0) (m 0) (pt 1 unknown) (seq 44954) (ts 0) (ssrc 475832294) (sender_ts 1272676921.276892))
  1272676921.307410 (rtp (v 2) (p 0) (x 1) (cc 0) (m 0) (pt 1 unknown) (seq 44955) (ts 10000) (ssrc 475832294) (sender_ts 1272676921.287114))
  ...
```

(a) Format of Receiver Trace Files

```
dataset-B/adsl5/cbr1.0/20100501-0222.path:
  Hop   1 :   glasgowuni-3    glasgowuni-3    glasgowuni-3    glasgowuni-3    glasgowuni-3
  Hop   2 :   glasgowuni-4    glasgowuni-4    glasgowuni-4    glasgowuni-4    glasgowuni-4
  ...

dataset-B/adsl5/cbr1.0/20100501-0222.icmp:
  icmp  recv_ts 1272676921.340018 icmp_src glasgowuni-3
  icmp  recv_ts 1272676921.408699 icmp_src glasgowuni-4
  ...

dataset-B/adsl5/cbr1.0/20100501-0222.pathprobes:
  pathprobe   send_ts  1272676921.337370 rtp_ts  60000 ttl 1
  pathprobe   send_ts  1272676921.407785 rtp_ts  130000 ttl 2
  ...

dataset-B/adsl5/cbr1.0/20100501-0222.packetpairs:
  packetpair  send_ts1 1272676932.024460 rtp_ts1 10690000 send_ts2 1272676932.024508 rtp_ts2 10700000
  packetpair  send_ts1 1272676942.783357 rtp_ts1 21390000 send_ts2 1272676942.783403 rtp_ts2 21400000
  ...
```

(b) Format of Sender Trace Files (*dataset-B* only)

```
dataset-B-proc/adsl4/cbr1/20100801-0222.qdelay:
  0.009468 0.00241679
  0.021026 0.00401279
  ...

dataset-B-proc/adsl4/cbr1/20100801-0222.pathprobe_rtt
  1 1280625722.579040 1280625722.580453 0.00141287
  1 1280625723.704560 1280625723.708295 0.00373507
  ...

dataset-B-proc/adsl4/cbr1/20100801-0222.packetpair_dispersion:
  10.747899 0.001209 0.000040 8.303
  21.496965 0.000985 0.000044 10.192
  ...
```

(c) Format of Processed Trace Files

Figure 3.1: Trace Formats

- Files with the `.icmp` extension contain a line for each of the ICMP messages received by the sender within the trace. The 3rd field shows the receive timestamp (seconds since 1970). The 5th field shows the anonymised address of the router that generated the ICMP packet.

- Files with the `.pathprobes` extension contain a line for each of the TTL-limited packets sent within the trace. The 3rd field shows the timestamp (seconds since 1970) just before the TTL-limited packet was sent. The 5th field shows the RTP timestamp ([184], Section 5.1). The 7th field shows the TTL with which the packet was sent. When processing the receiver log file, this log file is consulted to make sure the TTL-limited packets (which stop at the designated router rather than reaching the receiver) are not counted as lost. It is also processed to calculate per-hop loss rates and round-trip times for TTL-limited probes.

- Files with the `.packetpairs` extension contain a line for each of the packet-pairs sent within the trace. The 3rd and 5th fields show the sender and RTP timestamps of the first packet in the pair, and the 7th and 9th fields show the sender and RTP timestamps of the second packet in the pair.

To anonymise the trace files, they are passed through a script that replaces IP addresses and hostnames with a token; these have been selected to distinguish, but not identify, the ISPs. The home routers have been named according to the link ID to which they correspond.

3.4 Post-Processing

This section describes some of the post-processing applied to the traces to extract metrics of interest, including how clock skew is removed from the traces, how one-way delay is calculated, how the logs of TTL-limited packets and received ICMP messages are processed to produce round-trip times, and how the packet-pair measurements are used to estimate capacity. The processed data discussed in this section are also available in the dataset; each of the following sections describe the processing and file formats used. Figure 3.1c shows an example of these output files.

3.4.1 Skew Removal / One-way Delay

Conceptually, one-way delay is obtained by simply subtracting send timestamp from receive timestamp. This approach assumes that both clocks are running at the same constant rate, and have zero relative offset. In reality, these assumptions are typically not true, and therefore some external clock synchronisation mechanism is required (as considered in [46]). Although the clients and server are synchronised using NTP [140], the synchronisation is not perfect, and their clocks are still subject to an unknown relative offset β (the difference between the values of the clocks), and relative skew α (the ratio of the rates of the clocks).

End-to-end delays are made up of propagation (fixed), serialisation and queueing (variable) components. The true end-to-end delay of a packet i, d_i (which includes all three components), is the difference between the sender and receiver timestamps (t_i^s and t_i^r) calculated with perfect knowledge of the relative clock offset and skew between sender and receiver. However, since only the measured timestamps $\tilde{t_i^r}$ and $\tilde{t_i^s}$ are available, and the offset and skew are unknown, the *measured* end-to-end delay $\tilde{d_i}$ must be used instead.

This $\tilde{d_i}$ is subject to the relative offset (β) and skew (α) between receiver and sender clocks. Since α and β are unknown, they need to estimated from the data; to do this, the approach proposed by Moon *et al.* [143] and implemented by Kohno *et al.* [110] has been applied. This uses a linear programming technique to generate estimates for the clock skew and offset, $\hat{\alpha}$ and $\hat{\beta}$. Using these estimates, skew can be corrected as shown in Equation 3.1, producing the *corrected* end-to-end delay $\hat{d_i}$ as an approximation of d_i:

$$\hat{d_i} = \tilde{d_i} - (\hat{\alpha} - 1)\tilde{t_i^s} + \hat{\beta} \tag{3.1}$$

Assuming the minimum observed delay $\hat{d}_{min}$ corresponds to a packet that experienced minimal queueing delays at the routers along the path, the variation of other packets above $\hat{d}_{min}$ can be seen as a measure of the extent of queueing these packets experienced. $\hat{d}_{min}$ can be subtracted from the other $\hat{d_i}$ values to approximate queueing delay:

$$DQ_i = \hat{d_i} - \hat{d}_{min} \tag{3.2}$$

The output of this process is logged in files with the `.qdelay` extension, as shown in Figure 3.1c. The first shows the relative arrival time (in seconds, since the start of the trace); the second shows DQ_i, calculated as shown in Equation 3.2.

3.4.2 Matching ICMP Responses

The timestamp and target hop of each TTL-limited probe are obtained from files with the `.pathprobes` extension. The timestamp of each received ICMP message is obtained from the corresponding `.icmp` file, and the `.path` file is consulted to identify the hop number of the sending router. Using these, each probe is matched to its ICMP response by checking the timestamps of messages received from the router being probed. Since the probes are spaced at one second intervals (larger than the highest observed RTT), the ICMP message following a probe is counted as its response, and the RTT is calculated from the send and receive timestamps. Losses are identified as cases where a sent probe is not followed by an ICMP response.

The output of this process is logged in files with the `.pathprobe_rtt` extension, as shown in Figure 3.1c. Each line in this file represents a sent probe and ICMP response. The 1st field shows the hop number, and the 2nd and 3rd fields show the send and receive timestamps, respectively. The 4th field contains the RTT for this probe.

Initial analysis of these results suggests that the variability of the RTT measurements obtained using the TTL-limited probes is high. For example, the variance in the RTT measurements from some routers on the paths show higher variation than the overall end-to-end delay for that path. This may be due to the variation in ICMP processing times at the routers (which is done in software, rather than the hardware "fast-path" for normal traffic), or might be due to traffic shaping of ICMP responses by the routers. Due to this high variability, these "end-to-middle" RTT results are not studied any further in this dissertation. However, since the TTL-limited probe and ICMP response traces are available in the published dataset, the rationale for taking the measurements and their trace formats are included in this chapter for completeness.

3.4.3 Calculating Capacity with Packet-Pairs

As described in [54], the estimate of capacity, $\widehat{C}$ is obtained by dividing packet size L (in this case, 1316 bytes) by the arrival dispersion between the packets in the pair, δ.

$$\widehat{C} = \frac{L}{\delta} \tag{3.3}$$

Note that this estimate assumes that there is no network cross-traffic, and therefore there

will be some error in this estimate. Also, since the dispersion δ depends only on the arrival timestamps (which are local to the receiver), there are no issues relating to clock skew here.

The output of the calculation described in Equation 3.3 is contained in files with the `.packetpair_dispersion` extension, as shown in Figure 3.1c. The 1st field shows the arrival time of the second packet in the pair (in seconds since the start of the trace). The 2nd and 3rd fields show the dispersion in seconds between the arrival and departure times, respectively. The 4th field shows the capacity estimate from this pair, in Mb/s.

3.5 Discussion & Summary

In this chapter, I have introduced new measurements of RTP-based streaming to residential Internet users (the first such dataset publicly available). This improves upon previous Internet measurement work by integrating performance measurement of real-time application traffic to residential ADSL and Cable links. I have explained my measurement approach, which uses a dedicated measurement platform to send and receive RTP streams with the characteristics of video streams, and outlined the format of the trace data that are available at http://martin-ellis.net/research/datasets.

With these measurements, it is now possible to study the effect of the packet loss and delay conditions of residential networks on real-time video streaming. This allows more realistic simulation work and evaluation of application performance. In Chapter 4, I analyse the high-level characteristics of the packet loss, delay, and capacity measurements, to understand how streaming traffic is affected by residential ADSL and Cable networks. The packet loss traces will be examined in more depth in Chapter 5, and the accuracy of existing models used for packet loss simulation will be tested. In Chapter 6, the delay and loss measurements will be combined to give a more complete view of the network behaviour (particular in terms of network state and congestion), and give a more accurate model for packet loss. The packet loss measurements are used again in Chapter 7 to show the performance of forward error correction (FEC), an important element of streaming video performance.

Chapter 4

Analysis of Packet-Level Characteristics

The measurements presented in Chapter 3 contain data on the performance of inter-domain RTP streaming from the perspective of residential users, giving insight into the performance of streaming video applications. Using these measurements, the typical performance of streaming video running over ADSL and Cable links can be understood. Three important metrics for the performance of these applications are packet loss (since this determines the quantity of data that successfully arrives at the receiver), delay (since real-time applications require a relatively predictable packet arrival rate so that the video can be smoothly decoded and played out), and capacity (since this determines the rate at which video can be transmitted, and therefore its quality).

In this chapter, I present a high-level analysis of the packet level characteristics of the measurement traces presented in Chapter 3, to describe the loss and delay characteristics of residential networks. I investigate patterns in performance due to time-of-day, sending rates, and link type, and examine the relationship between packet loss and queueing delay. I also discuss the performance of packet-pair measurements in estimating the capacity of ADSL and Cable links, and investigate the feasibility of using packet-pairs to identify whether residential receivers are connected to either ADSL or Cable links.

This chapter is structured as follows. Section 4.1 describes the packet loss characteristics seen in the measurement traces, focusing on the effect of time-of-day, sending rate, and link type. This section also gives a high-level overview of the loss and receive run-length distributions seen in the measurements, and investigates the relationship between loss and queueing delay. Section 4.2 presents results of queueing delay, including variation over time-of-day and different sending rates, and studies the variability present in the queueing delay distri-

butions of the traces. Section 4.3 describes the results of the packet-pair capacity estimation in the measurements of Chapter 3, explaining why these are inaccurate for Cable links, and outlining a possible technique to use this to classify ADSL and Cable links. Section 4.4 gives a summary of the chapter.

4.1 Packet Loss

This section discusses the packet loss characteristics seen in the measurements of residential links. Understanding packet loss is important since it is a key factor in the quality perceived by the users of streaming video systems. For example, techniques to improve performance such as FEC need to be tuned to work with the particular loss patterns and characteristics of the networks to which they are applied. Section 4.1.1 describes loss rates seen in the measurements, comparing variation between different links (and access types), sending rate, and time of day. Section 4.1.2 analyses the burstiness of packet loss in more detail, examining the typical lengths of loss bursts and gaps between losses. Section 4.1.3 looks at how the sources of different losses might be identified, allowing the classification of losses as being either "congestive" or "non-congestive". Section 4.1.4 summarises the findings of analysing the packet loss measurements.

4.1.1 Comparing Loss Rates

In this section, the loss rates of the different links, and time-of-day variation in loss rates are examined. Previous studies of residential Internet performance have found diurnal variation in packet loss [49]; this section seeks to see if such behaviour is also present in *dataset-A* and *dataset-B*. Figures 4.1 and 4.2 show the packet loss over times of day, for the links in *dataset-A* and *dataset-B*, respectively (the measurement schedules were described in Section 3.2). Differences between the links are obvious, with some links (e.g., *adsl3* and *adsl7*) showing time-of-day variation, and others showing very little (with either occasional spikes of packet loss, or low loss rates throughout). For link *adsl7*, the time-of-day variation in loss rates is more pronounced at higher sending rates (i.e., at higher sending rates, more loss is observed). This suggests there is congestion within the ISP network, since loss rates increase when the network is more heavily loaded.

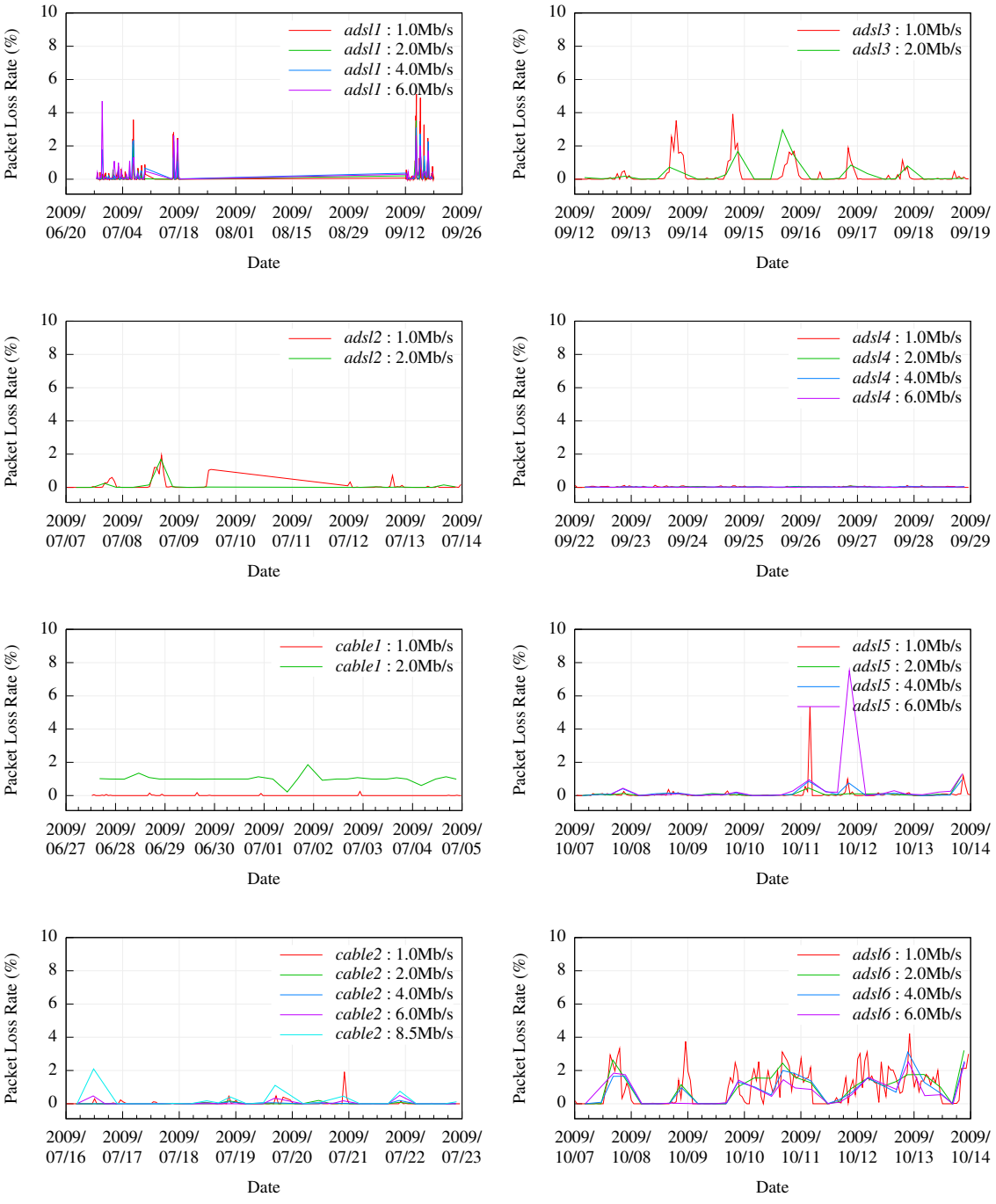


Figure 4.1: Loss Rate Time-Series (*dataset-A*)

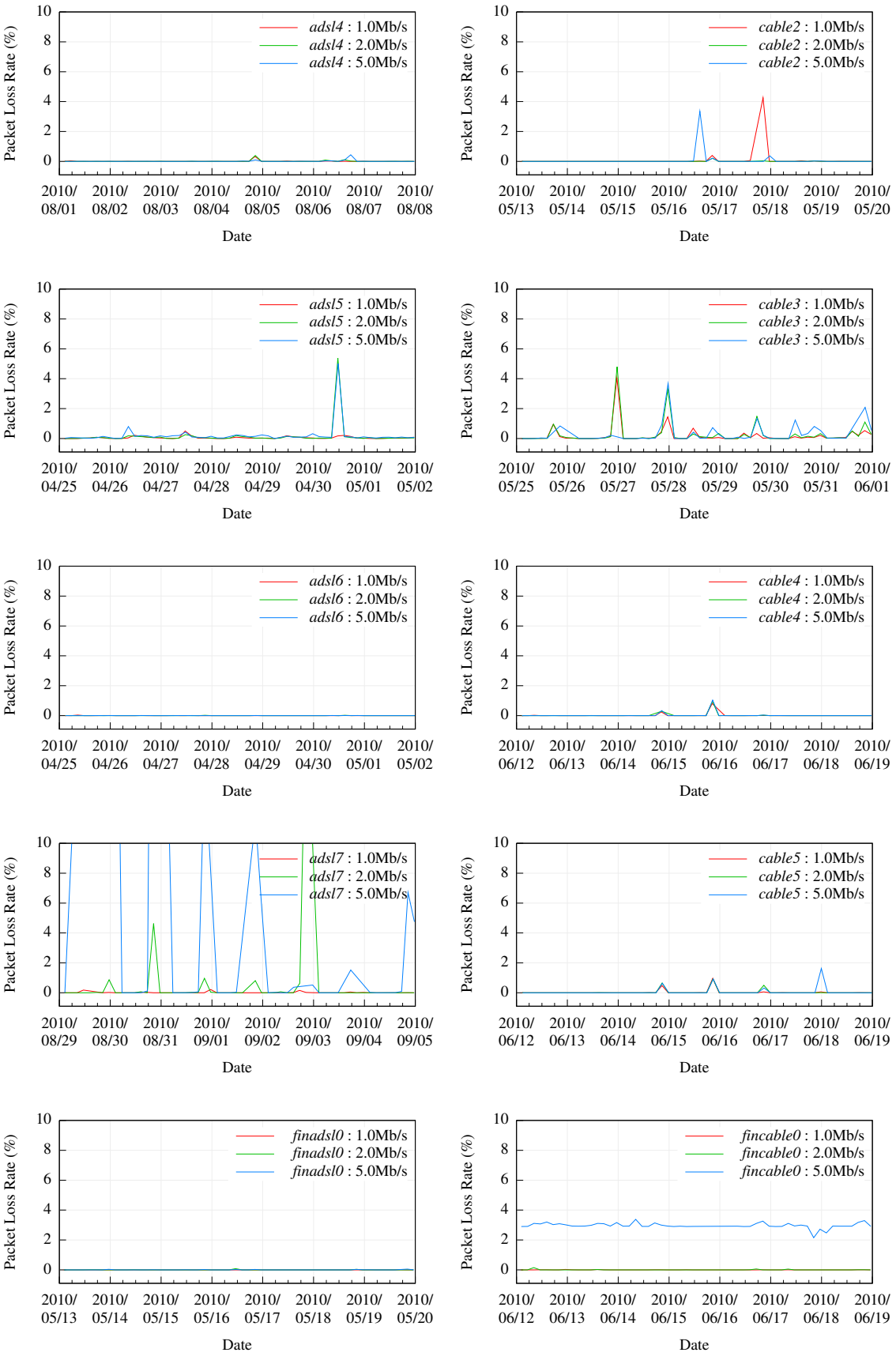


Figure 4.2: Loss Rate Time-Series (*dataset-B*)

Some links show time-of-day variation in packet loss, while others appear fairly stable throughout the measurement period. Interestingly, link *adsl6* shows this variation in *dataset-A*, but did not in *dataset-B*, showing that the performance of the links can also change over long timescales.

Another pattern in the loss rates can be seen for links *cable1* at 2Mb/s and *fincable0* at 5Mb/s. In both these cases, there is a mostly consistent non-zero loss rate throughout the entire measurement period. These loss events occur periodically within the traces, suggesting that the packet loss is due to traffic shaping being employed by the ISPs to enforce the bandwidth allocations in the customers' subscriptions. Since the sending rates of these streams (2Mb/s and 5Mb/s) are close to the subscription bandwidths reported in Tables 3.1 and 3.2, it appears the measurement streams are *slightly* exceeding the allowance, causing packet loss. Since these losses are artificial, and are caused by ISP policy rather than network performance, the affected traces (*cable1* at 2Mb/s, and *fincable0* at 5Mb/s) are excluded from further study. However, it is worth noting that this type of behaviour is present, and applications such as over-the-top video streaming should take this into account, possibly providing mechanisms to adapt to such long-term limitations in network performance.

4.1.2 Loss Burstiness

This section examines the "burstiness" of packet loss (i.e., looking at the lengths of loss bursts, and the frequency of their occurrence). Understanding this behaviour is important since the effect of packet losses on the performance of streaming video and FEC is related to packet loss burstiness [70, 103].

Figures 4.3 and 4.4 show the loss run-length distributions of each of the links in *dataset-A* and *dataset-B*. These figures show the complementary cumulative distribution function (CCDF), showing the fraction of loss bursts longer than that of the x value. This is commonly used in analysing network data, since it allows the plot to be easily understood with a log-scale to show the differences between small values.

Most links show a relationship between the sending rate and the frequency of longer loss run-lengths. This is likely due to the fact that at higher rates, more packets are being sent within a fixed period of time (i.e., the inter-packet spacing is smaller), so that more packets are affected by periods of loss in the network. This might be due to periods of link noise or periods of network congestion that are present for a fixed length of time. This can be seen

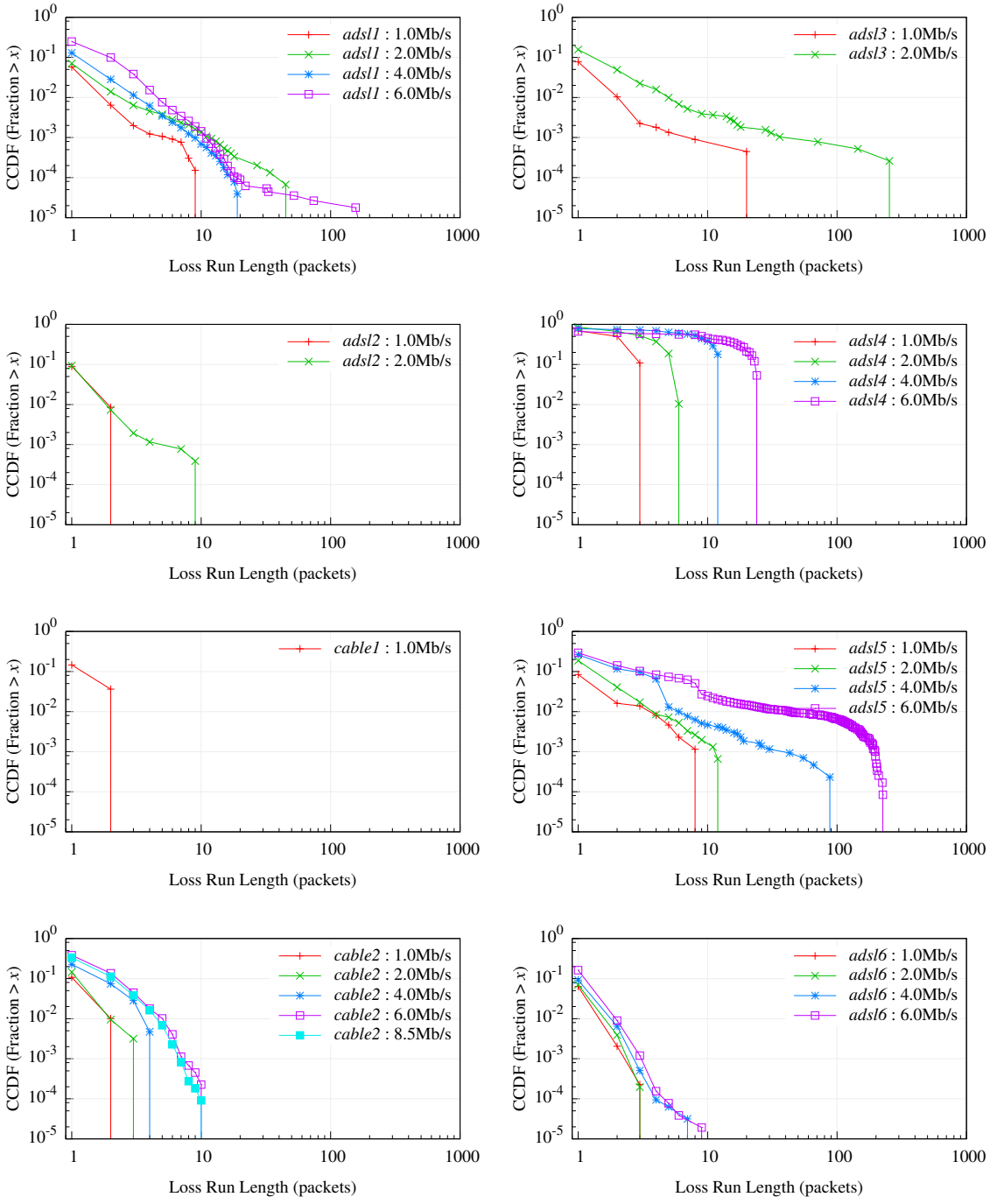


Figure 4.3: Aggregate Loss Run-Length Distributions (*dataset-A*)

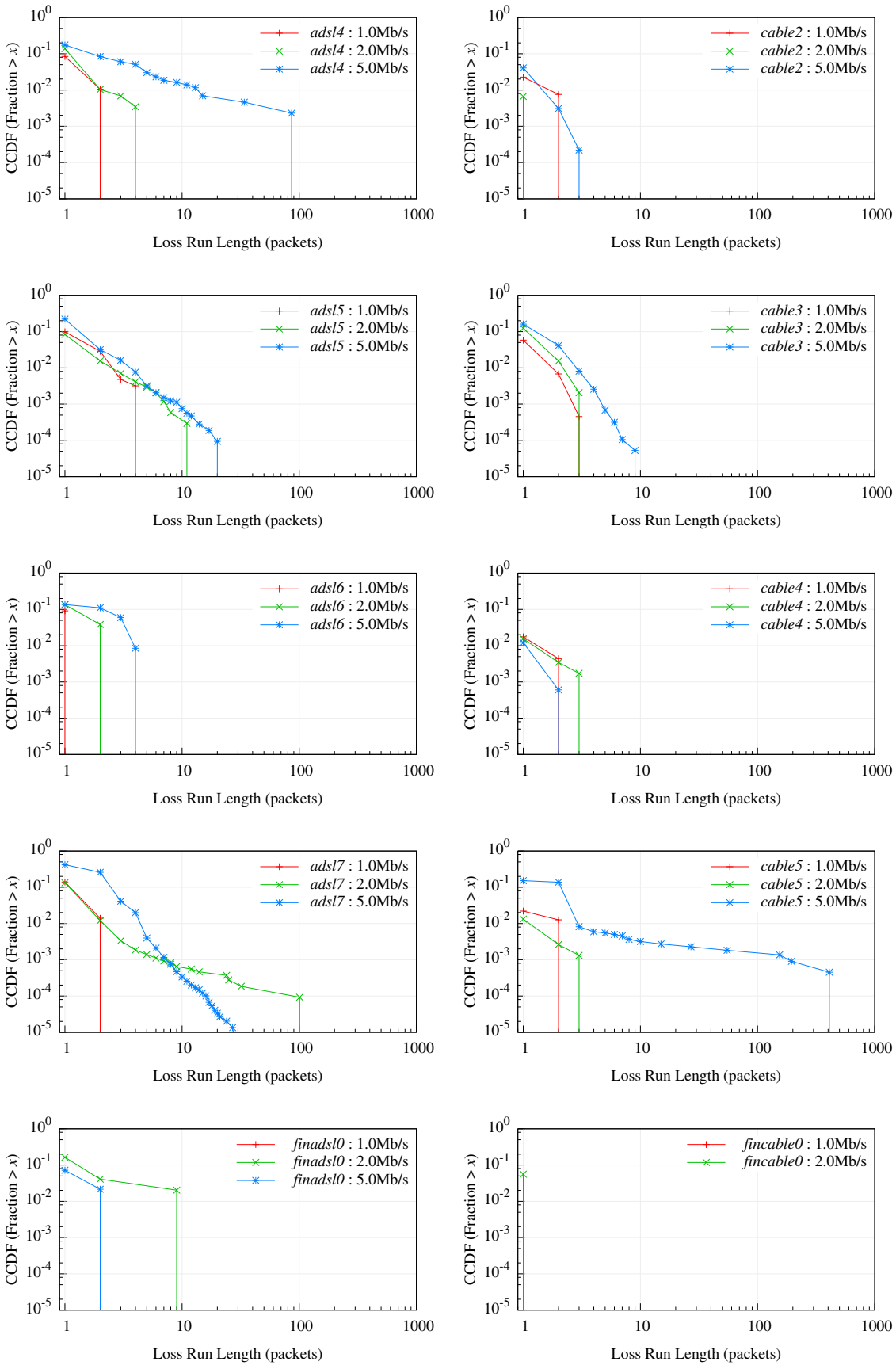


Figure 4.4: Aggregate Loss Run-Length Distributions (*dataset-B*)

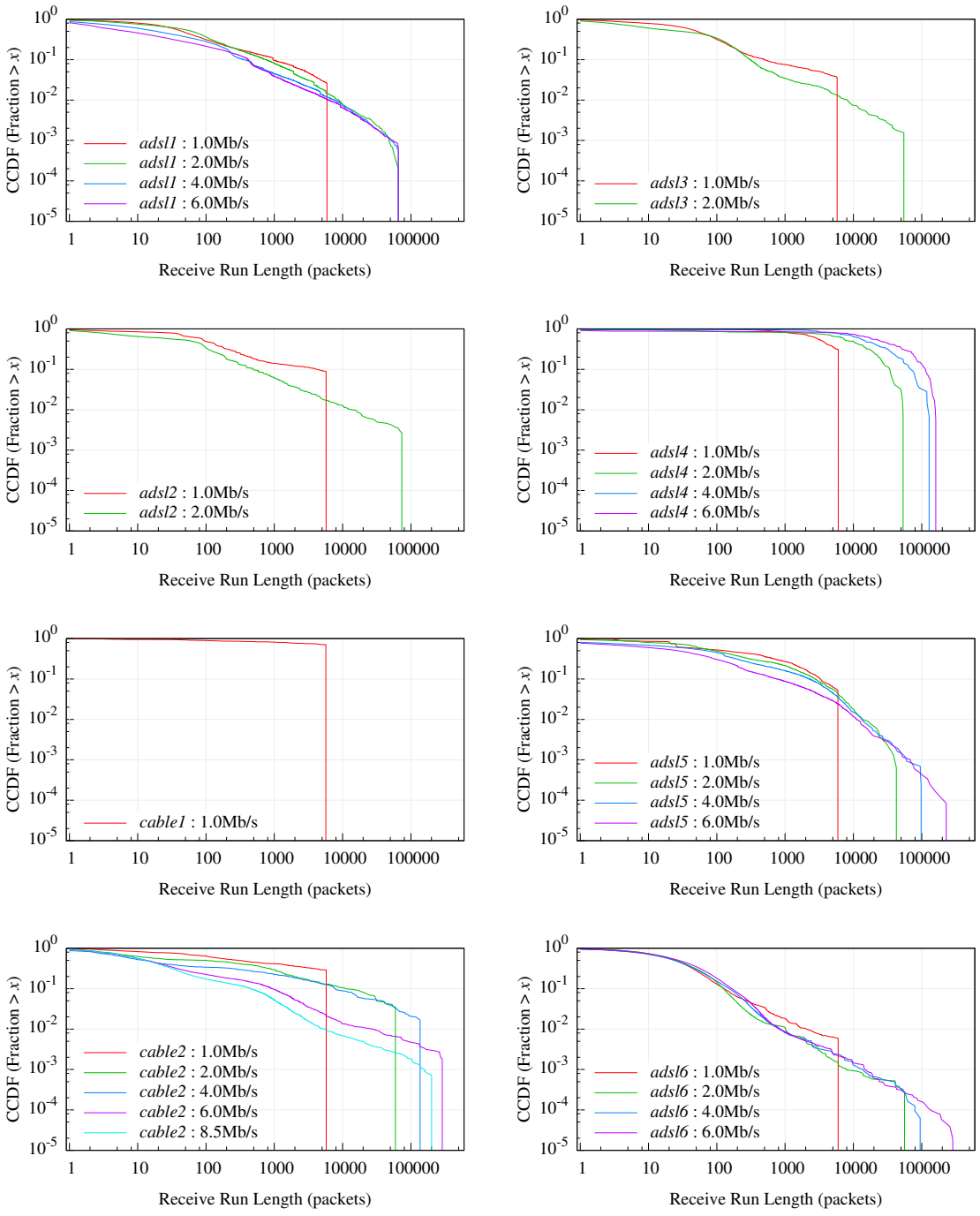


Figure 4.5: Aggregate Receive Run-Length Distributions (*dataset-A*)

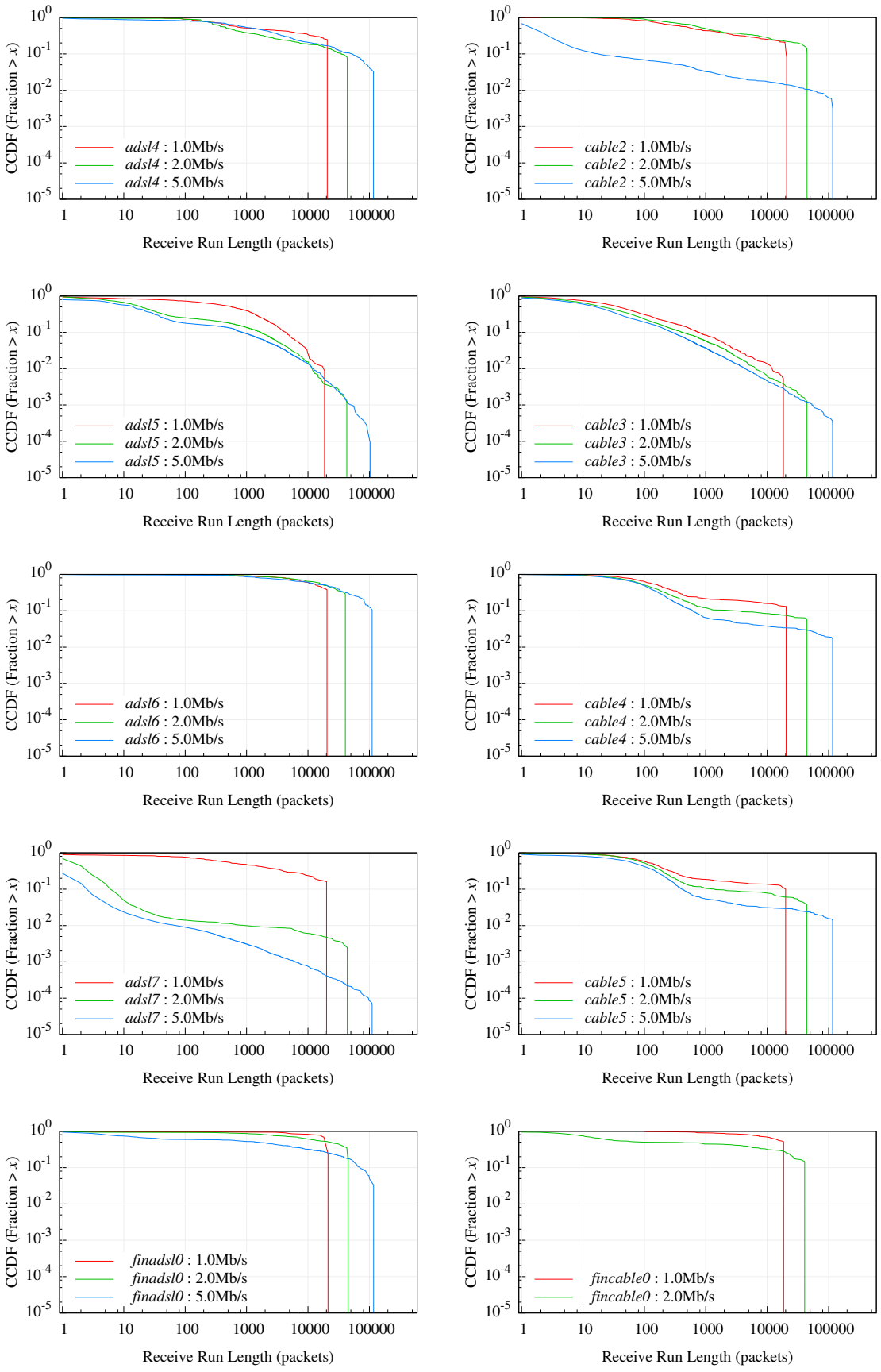


Figure 4.6: Aggregate Receive Run-Length Distributions (*dataset-B*)

by looking at the loss run-length distributions for *cable2* at 6Mb/s and 8.5Mb/s in Figure 4.3. Recall from Table 3.3 that both these sending rates have a 1ms inter-packet spacing, and only vary in their packet size. In Figure 4.3, their loss run-length distributions almost overlap, while the lower sending rates (with larger inter-packet spacing) are disjoint.

In Figures 4.3 and 4.4, the distributions show that most loss bursts are short, and in almost all cases, single packet loss bursts are the most common, by a large margin. This result is consistent with prior Internet loss measurements [30, 224, 28, 128]. Some of these studies suggest that loss burst distributions follow a *heavy-tailed* distribution, with the possibility of extremely large values [128]. Considering the tail behaviour of a variable or dataset essentially means examining how its probability density function decays as it approaches zero. *Light-tailed* distributions are said to decay at least as rapidly as the exponential distribution (e.g., the normal distribution). On the other hand, *subexponential* distributions decay less rapidly than the exponential. This class of distributions include heavy-tailed distributions, which have a distinctive power-law shape. A random variable X follows a heavy-tailed distribution if it satisfies:

$$P[X > x] \sim kx^{-\alpha} \text{ as } x \to \infty$$

When the complementary cumulative distribution function of X is plotted on log-log axes (an LLCD plot), it will appear as a straight line with gradient $-\alpha$ (in this context, α is often called the *tail index*). For further detail on the study of heavy-tailed behaviour in Internet traffic, see [42, 2].

In [128], this technique is used as evidence that the packet loss bursts follow a heavy-tailed distribution. However, since these the results in Figures 4.3 and 4.4 show distributions aggregated from a number of traces taken at different times, it is somewhat misleading to discuss the run-length distributions in this way. Instead, these figures show the relative proportions of different lengths of loss bursts throughout the datasets. Since the traces exhibit a widely varying range of loss behaviours (as Chapter 5 will explain in detail), I do not believe it makes sense to describe the loss run-length distributions seen across the whole dataset as being heavy-tailed. The obvious non-stationarity seen at different times of day (and even within individual traces) shows that a number of different processes at responsible for the loss behaviour.

Figures 4.5 and 4.6 show receive run-length distributions across the same links. These show a wide variation in receive run-lengths (i.e., loss-free runs), from single packets, up to

runs of tens of thousands. The high frequency of single packet receive runs highlights that there are periods where packets are received between bursts of lost packets. The extremely long receive runs show cases where all or most of the trace was received with no loss. There are traces where both extremes exist, with very short receive runs between loss bursts, and very long receive runs. The presence of these differing types of behaviour has important implications for video performance and tuning, as well as developing packet loss models, as will be discussed in more detail in Chapter 5.

The loss run-length distributions in Figures 4.3 and 4.4 show that while most loss bursts are short, there are rare cases of very large loss bursts over 100 packets. For most links, larger maximum loss burst lengths are seen at higher sending rates. Figures 4.5 and 4.6 show that there is a wide range in receive run-lengths. Many of the receive runs contain fewer than ten packets, indicating that there are periods where losses occur close together, separated by just a few received packets. This finding shows that although most loss bursts are short, it does not necessarily mean that most *lossy periods* are short; there may instead be multiple loss bursts, separated by a few received packets (the implications of this will be discussed in detail in later chapters).

4.1.3 Congestive and Non-Congestive Loss

As discussed in Section 2.2, packet losses may be due to a number of sources, such as lower-layer errors (e.g., noise on at the physical layer causing checksum failures), or packets being discarded at routers due to congestion or network policy (i.e., traffic shaping). A characteristic of packet loss due to congestion is that the previous packets will often experience larger queueing delay as the queue at the congested router fills. In this section, I show examples of the relationship between queueing delay and packet loss, and using a simple classification, give a breakdown of how often packets are lost during congested periods.

Figure 4.7 shows an example of congestive packet loss, where the queueing delay DQ sharply increases from a steady period of around 0.02s to more than 0.2s. After this, the DQ eventually falls (alongside packet losses) before returning to normal. These packet losses are likely to be caused by the router discarding packets with a drop-tail policy.

With this, a simple method to gain insight into the types of losses seen on different links and at different sending rates is to compare the queueing delay experienced before each loss event to the average queueing delay for that trace, and estimate whether that loss was

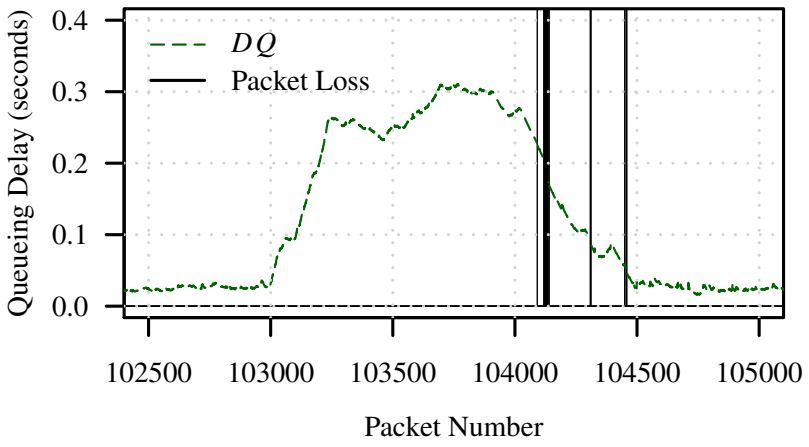


Figure 4.7: Example of Congestive Packet Loss

congestive (i.e., caused by congestion) or *non-congestive*. Figures 4.8 and 4.9 show the results of classifying each loss event in *dataset-A* and *dataset-B*, respectively. For each loss event, if the DQ of the previous packet received before the loss is greater than twice the median DQ for the trace, the loss is classified as congestive; otherwise, it is considered to be non-congestive. The figures show differences between the links, but again do not show a clear distinction between ADSL and Cable, suggesting that more in-depth analysis is needed.

4.1.4 Summary

This section has illustrated some of the high-level features in the packet loss measurements. From the measurements analysed here, it is clear that there are differences between links, with some showing time-of-day variation and others not, a range of shapes of loss and receive run-length distributions, and differing degrees of congestive and non-congestive packet loss. However, there does not appear to be a clear distinction between ADSL and Cable, with different links of the same type behaving differently. Although Cable links appear to have generally lower packet loss, link *cable3* appears to be more lossy than the others, and while most Cable links experienced no loss bursts longer than ten packets, *cable3* showed some loss bursts of more than 100 packets). For this reason, in further analysis I will treat each link separately, rather than aggregating links of the same type.

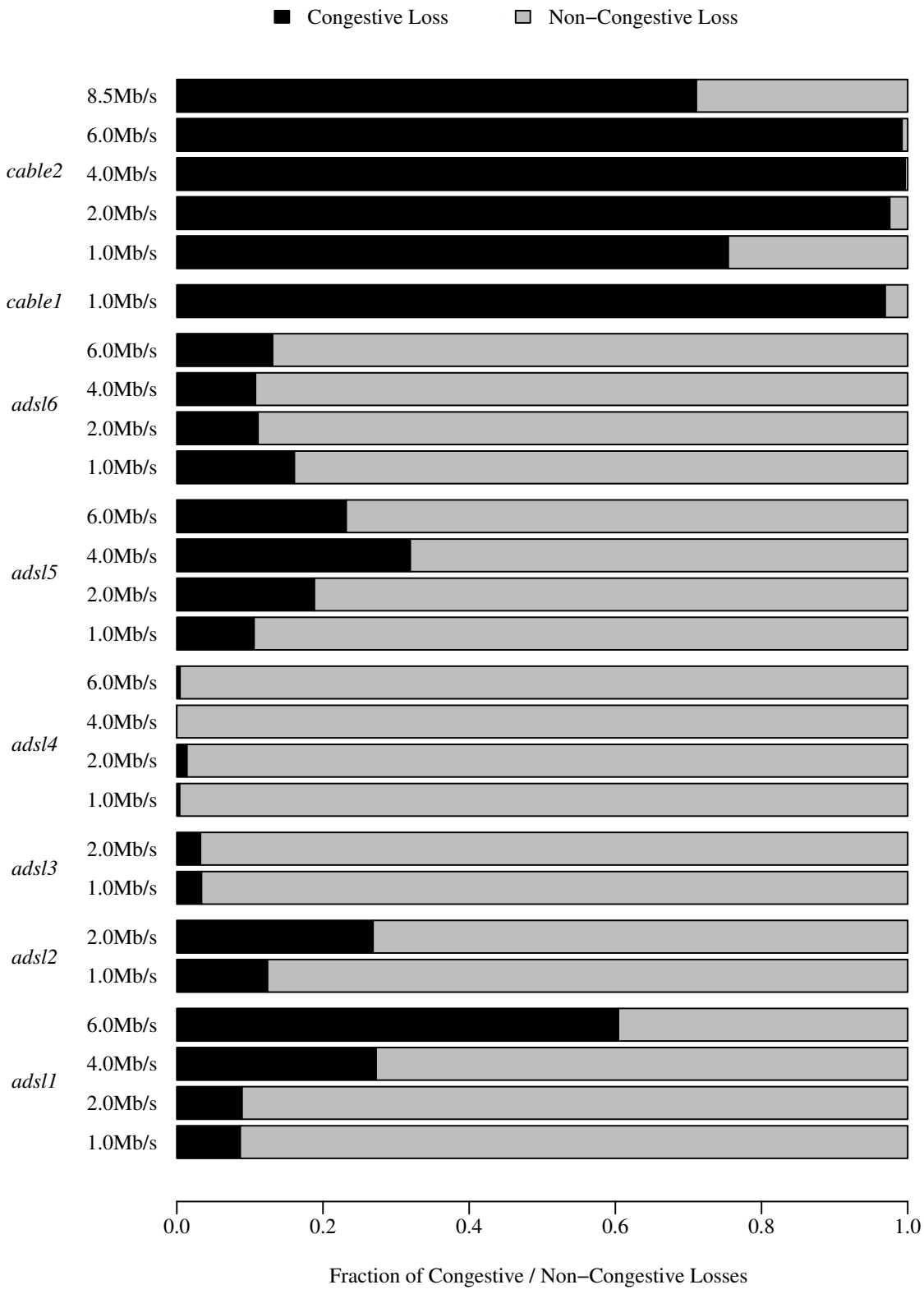


Figure 4.8: Congestive / Non-Congestive Packet Loss (*dataset-A*)

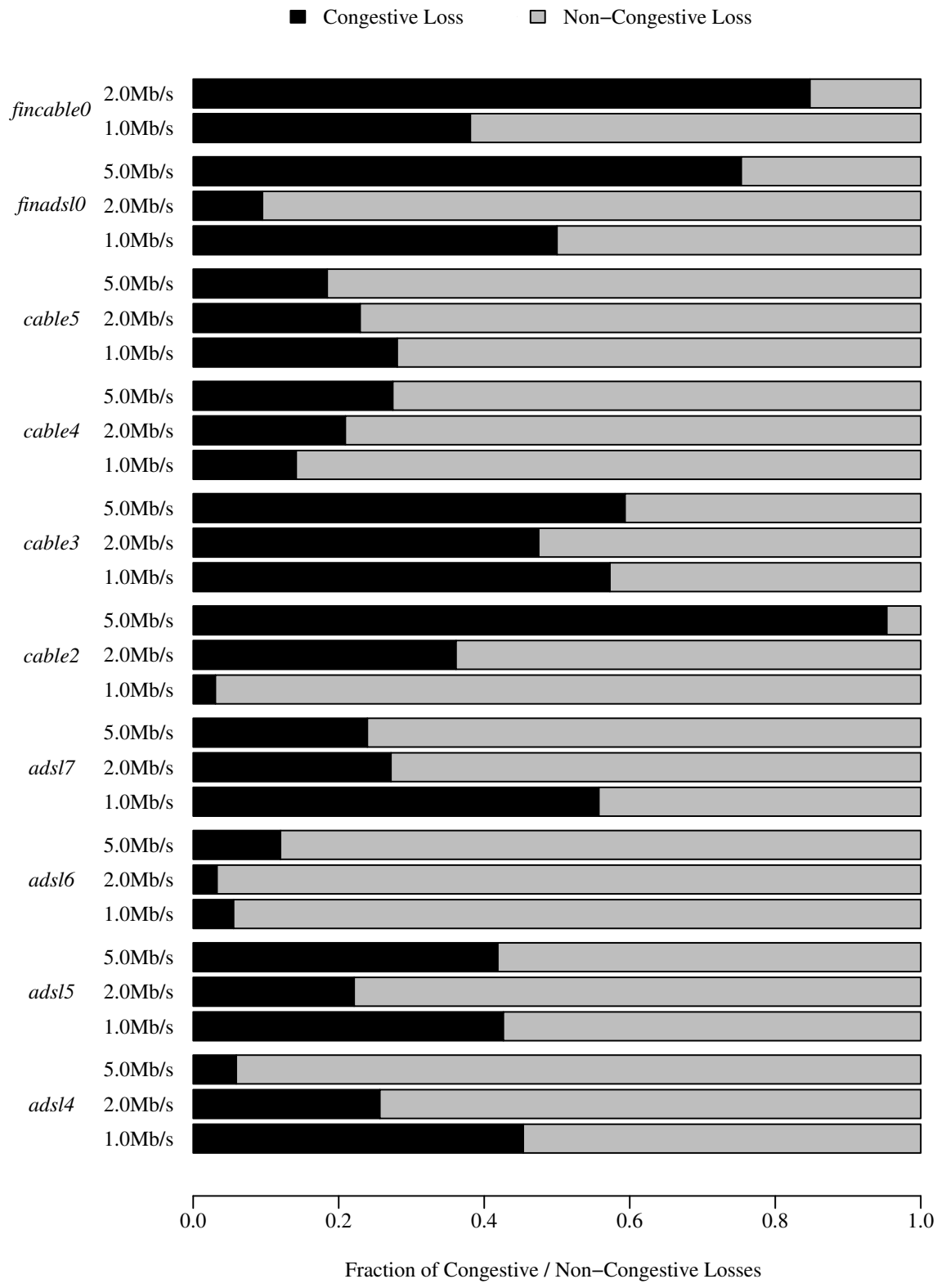


Figure 4.9: Congestive / Non-Congestive Packet Loss (*dataset-B*)

4.2 Queueing Delay

This section describes the characteristics of queueing delay (DQ) in the measurement data. Understanding queueing delay is relevant since this is an important indicator of network performance (i.e., whether there are signs of congestion), and also because video applications tend to be sensitive to variability in delay. First, I examine the effect of time-of-day on queueing delay in Section 4.2.1, and then describe the distributions of queueing delay in Section 4.2.2, both for individual traces, and aggregated over all the traces for each link.

4.2.1 Effect of Time-of-Day

Figures 4.10 and 4.11 show time-series of queueing delay over time for *dataset-A* and *dataset-B*. These show, for each trace, the median and inter-quartile range (IQR) of the delays, to give a sense of the average and variation in DQ. Median and IQR are used instead of other statistics (e.g., mean and standard deviation), since they are more robust in cases of skewed (or non-symmetric) distributions (Section 4.2.2 will discuss how the delay distributions are skewed).

As with the loss time-series seen earlier in Figures 4.1 and 4.2, there are clear differences between the links, with some showing evidence of time-of-day variation (e.g., *cable3*), and others showing consistent values of delay (e.g., *cable5*). Links *adsl7* and *cable3* also showed time-of-day variation in packet loss, as seen in Figure 4.2, with the periods of high packet loss matching those with higher queueing delay. However, link *adsl3*, which showed time-of-day variation in packet loss, does not show the same variation in queueing delay.

Another interesting observation is that there are differences in some of the links measured in both *dataset-A* and *dataset-B*, between the first and second measurement periods. For example, in *dataset-A*, link *adsl4* shows very stable DQ behaviour (the most stable of all the links in *dataset-A*), while in *dataset-B*, it has changed to be more variable both within traces (i.e., higher IQR per-trace) and between traces (i.e., showing some time-of-day variation). Similarly, link *adsl6* has changed, going from the most variable in *dataset-A*, to among the most stable in *dataset-B*. This illustrates that the performance of links can change (e.g., due to changes in the ISP network and customer demand), and that although application-layer performance may be good at one measurement point, it can degrade (or, conversely,

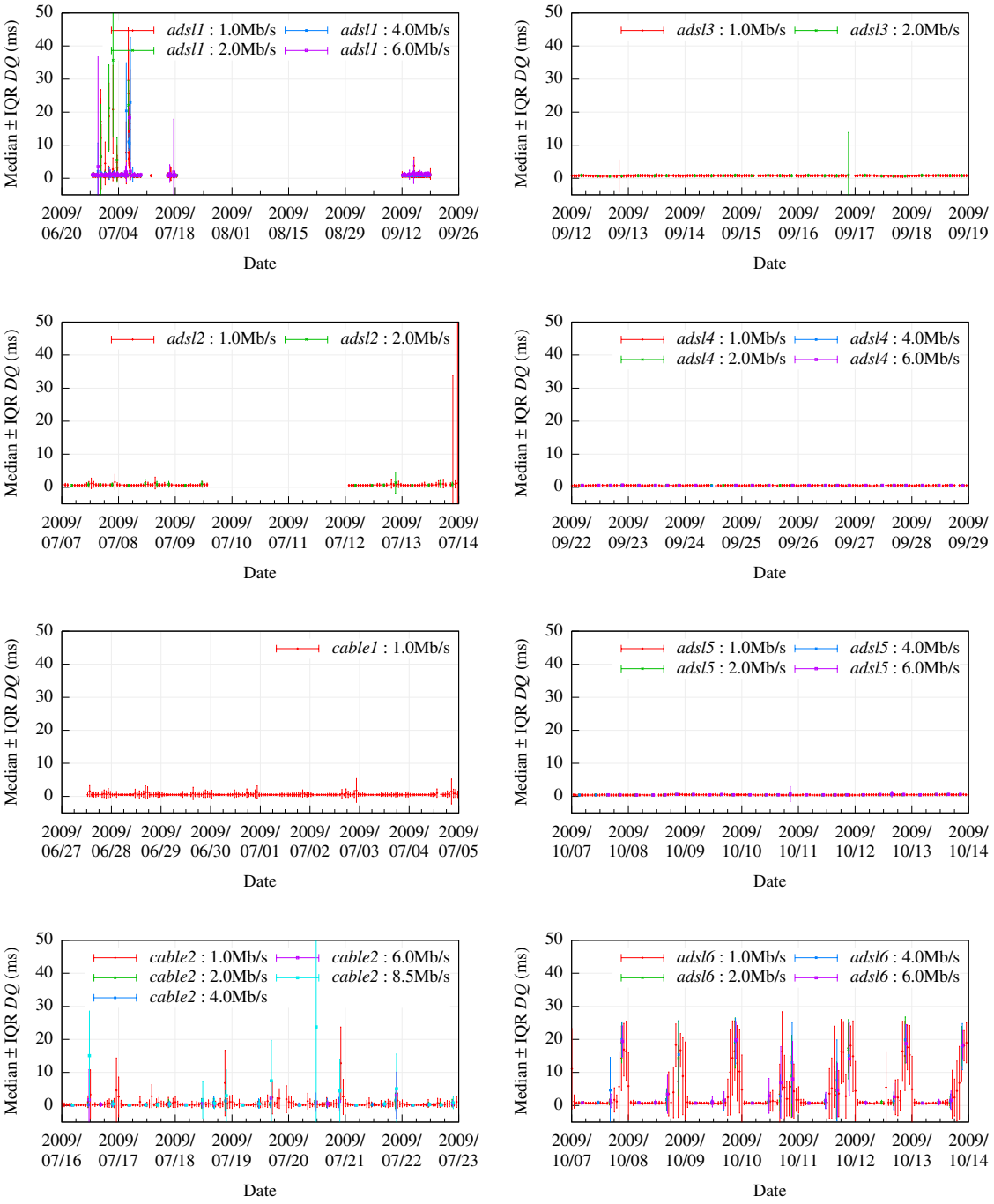


Figure 4.10: Queueing Delay Time-Series (*dataset-A*)

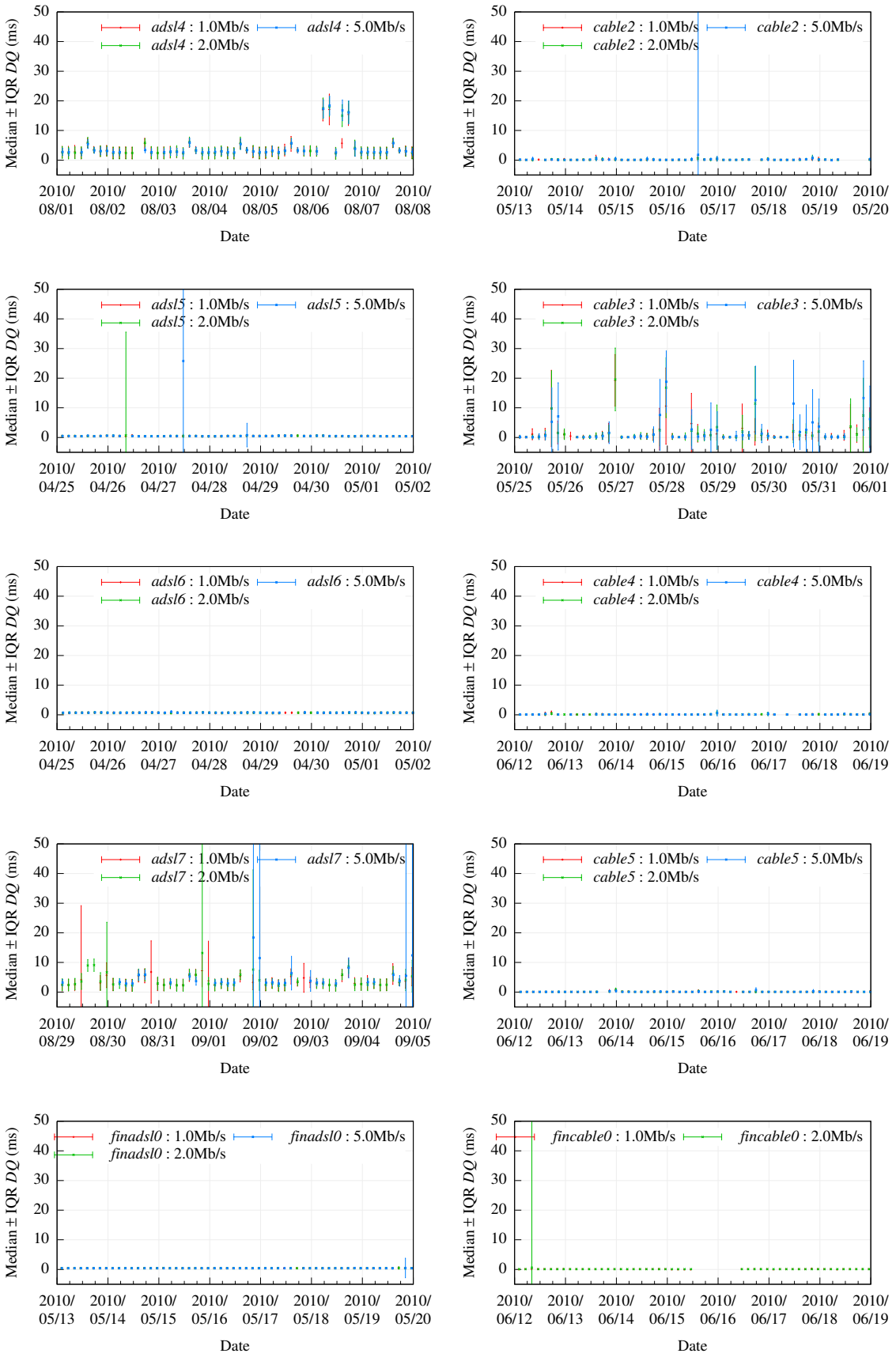


Figure 4.11: Queueing Delay Time-Series (*dataset-B*)

improve) over time. Therefore, being able to monitor performance in deployed systems is vital to ensure they can be adjusted to cope with changes in network performance.

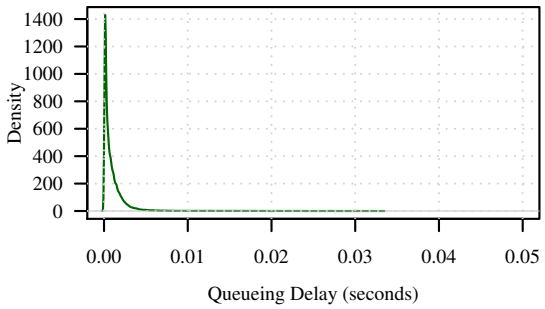
4.2.2 Queueing Delay Distributions

The variable queueing delay behaviour seen for some links in Figures 4.10 and 4.11 suggests that understanding the queueing delay characteristics may not be straightforward, and that looking at the whole DQ distribution (rather than just summary statistics) may give important insight. Understanding the whole range of delay characteristics is important, since video streaming applications rely on understanding the delay characteristics of the network so they can tune their own behaviour (e.g., determining the appropriate size for de-jitter buffers at video receivers). Therefore, this section studies the distributions of DQ, to understand how variable the delay behaviour is within particular traces, and to compare this between traces (over time and between different links).

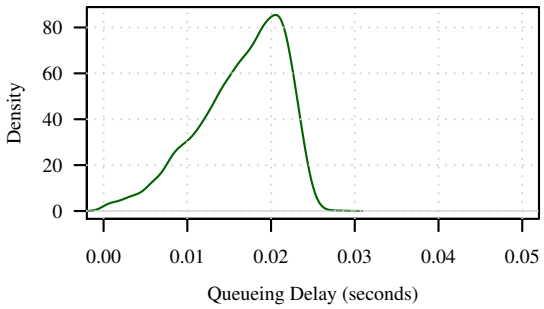
Investigations into the queueing delay distributions show that many have a Gamma-like shape, with strong left peak, and a right-skewed tail; this corresponds to most delays being close to a baseline value, with peaks appearing from time to time (e.g., Figures 4.12a and 4.12c). Some other traces also have a less skewed distribution (sometimes appearing more Normal-like), corresponding to queueing delay fluctuations around a central point (e.g., Figure 4.12b), while others have a completely different shape that is hard to define (e.g., Figure 4.12d). These different "classes" of distribution are consistent with previous studies of delay [83, 29], where a large fraction of measured traces showed Gamma-like distributions for queueing delay.

The inter-quartile range (IQR) of each trace is examined to identify traces with high variability, such as the one in Figure 4.12d. The distribution of IQRs across all the traces is shown in Figure 4.13. The DQ IQR is quite low for the majority of traces, although there is a long tail in the aggregate IQR distribution containing traces with higher IQRs. These traces will show a large degree of variability in DQ, and are therefore worth examining in more detail. To identify the traces with more variability, a threshold of 5ms is chosen using the distributions in Figure 4.13 as a guide.

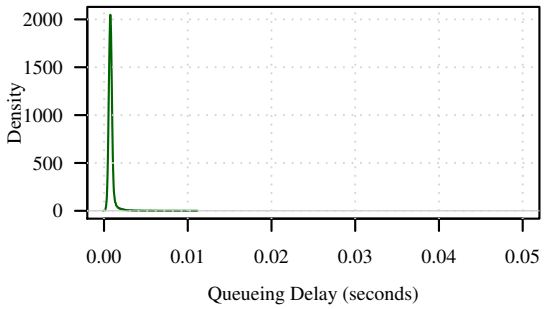
For these traces with higher IQR, some show a Gamma-like shape, while others show a quite different shape (including more "symmetric" distributions, "bimodal" distributions, and others). The traces with Gamma-like shape (i.e., strong left peak, and right tail) in the



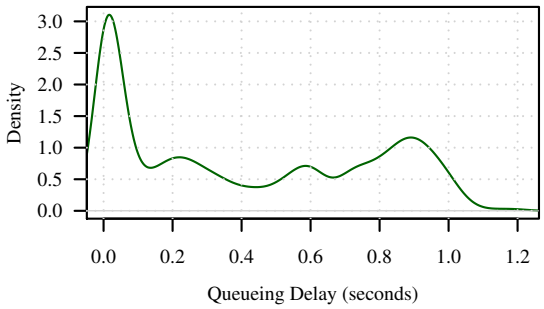
(a) "Gamma-like" shape



(b) "Normal-like" shape

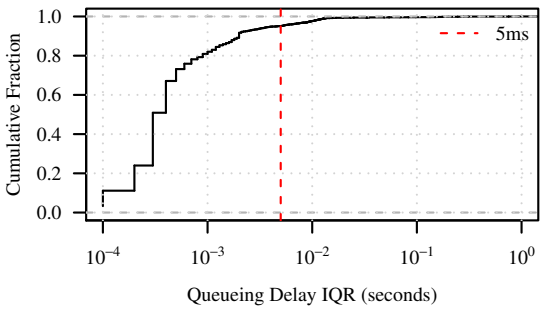


(c) "Gamma-like" shape

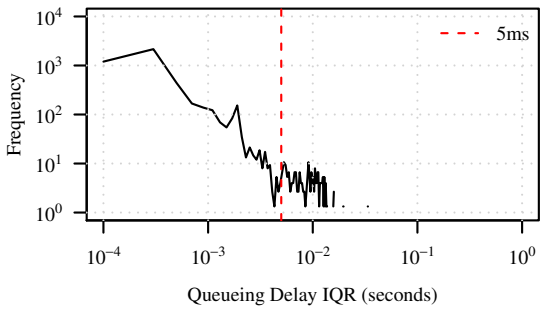


(d) High variation in DQ (note change in scale). The important point here is not the multiple modes in the distribution (which may be misleading due to the smoothing in the kernel density estimation), but rather its high variability.

Figure 4.12: Example Queueing Delay Distributions



(a) DQ IQR cumulative distribution



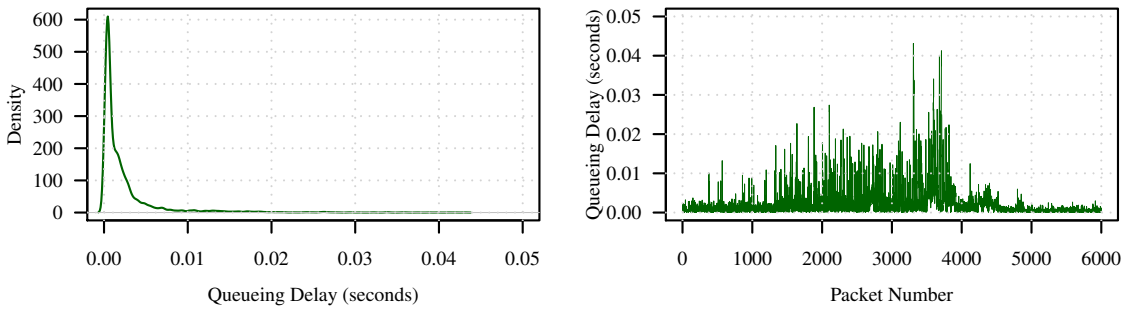
(b) DQ IQR histogram

Figure 4.13: Distribution of DQ Interquartile Ranges

delay distribution show traces with most delays close to zero, with occasional spikes, as shown in Figures 4.14a and 4.14b. These spikes sometimes coincide with loss periods, with losses often (although not always) occurring during periods of increased delay. A few of the traces show "mode switches" between less variable and more variable delay (typically associated with more bursty, higher packet loss). This queueing behaviour is probably present when queues along the end-to-end path are not full, and occasional spikes indicate the queue filling (and losses occur when the queues overflow). All the time-series with more symmetric shapes in the delay distribution (e.g., the example in Figures 4.14c and 4.14d) have a more variable delay, but there is not an obvious relationship to higher packet loss. Although there is more variability, the values are oscillating around a middle value, rather than spiking up from a lower value (as with the traces with Gamma-like distributions). In these traces, packet losses tend to be spread out throughout, rather than grouped into bursts. This suggests that the queues in the network are quite full, and losses (frequently) occur due to queue overflows. The traces do not show long periods of low delay since the queues do not drain. The traces with delay distributions appearing to be bimodal tend to be fairly similar to the symmetric ones, as well as having periods of low loss like those with the left-peaked distribution. This group is somewhere between the left-peaked and symmetric traces, since the traces are exposed to both types of behaviour (periods of unloaded queueing *and* almost full queues). Figures 4.14e and 4.14f show an example of this behaviour. Finally, the traces with "other" distributions seem to be the most interesting, although these are quite rare. These traces show examples of delay "sawtooths", with delay spiking up to some value, then some period of loss, followed by a drop in delay. There are also examples of clear mode switches between low loss, low delay to high loss, high delay, as shown in Figures 4.14g and 4.14h. Many of the traces with highest loss rates appear in this group.

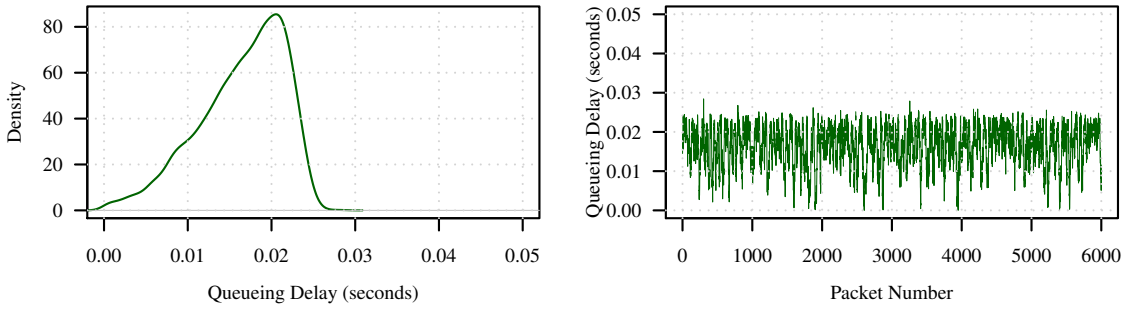
Among the traces with lower IQR (which make up most of the traces), the vast majority appear to have a Gamma-like shape similar to the example in Figure 4.12a. This is consistent with the queues in the network being unloaded most of the time, with occasional spikes in DQ due to cross-traffic. These findings are indicative of the dataset as a whole, in that much of the time conditions are good, while occasionally poor performance is encountered (either with periods of packet loss, spikes in delay, or both).

The distributions obtained by aggregating the queueing delays observed for each link and rate are shown in Figures 4.15 and 4.16. These are shown as CCDFs (as in [128]), to



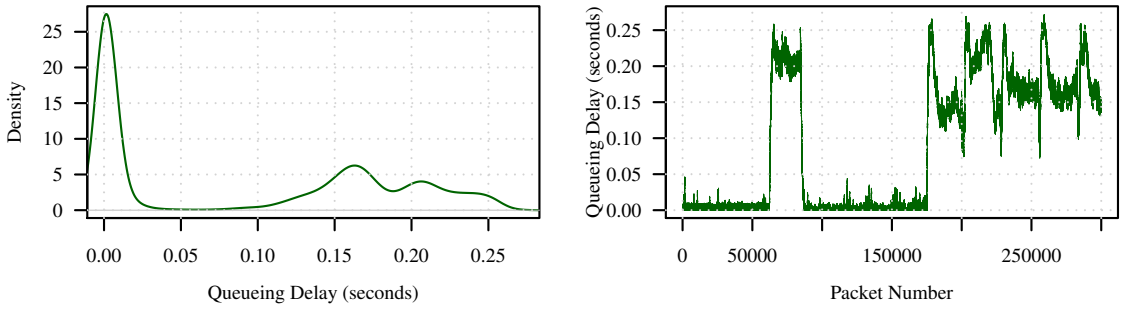
(a) "left-peaked" distribution

(b) DQ time-series



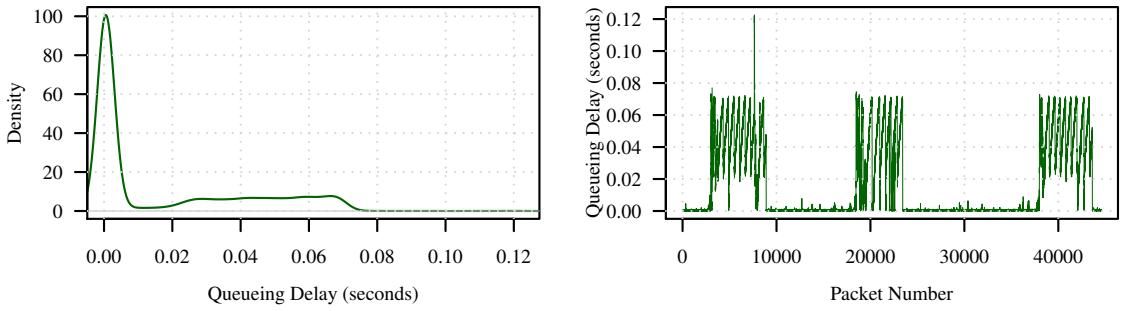
(c) "symmetric" distribution

(d) DQ time-series



(e) "bimodal" distribution

(f) DQ time-series



(g) "other"

(h) DQ time-series

Figure 4.14: Variability in Queueing Delay Distributions (note non-constant scale)

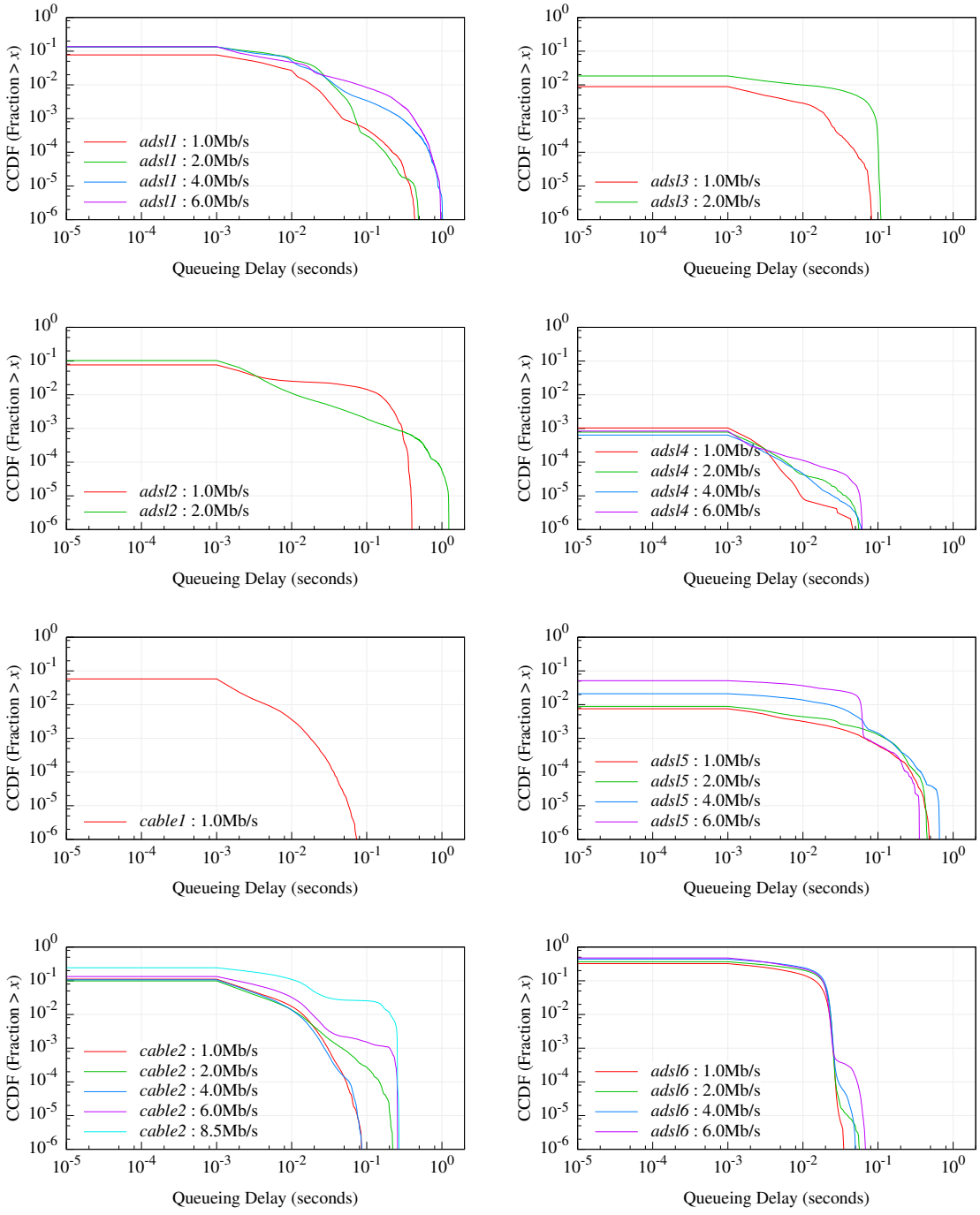


Figure 4.15: Aggregate Queueing Delay Distributions (*dataset-A*)

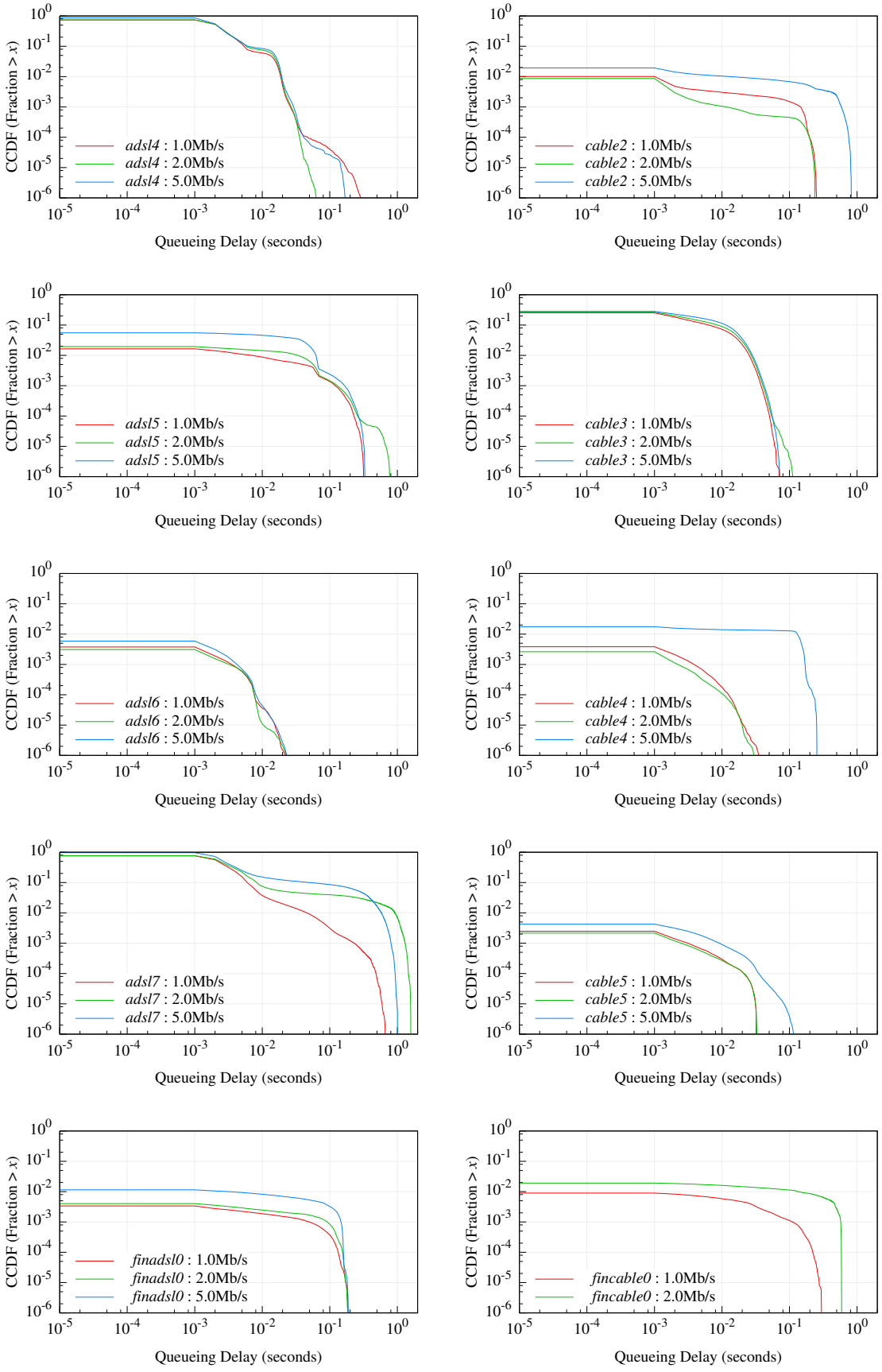


Figure 4.16: Aggregate Queueing Delay Distributions (*dataset-B*)

see the types of distribution (and whether these are heavy-tailed). The "straight-line on log-log axes" often cited as being indicating heavy-tailed behaviour is not present for the delay distributions shown here. However, since the individual delay distributions seen in Figures 4.12 and 4.14 show very different behaviour, the aggregate distributions just show the range of this variation (for particular links and sending rates). As with packet loss it is clear that the links have different behaviour (the distribution shapes are very different), and that it is not possible to group the links together as either ADSL or Cable links. Furthermore, it is again clear that the performance of some of the links changed between *dataset-A* and *dataset-B*, with the shape of the DQ distributions for *adsl4*, *adsl6*, and *cable2* changing between the datasets (probably due to upgrades or configuration changes by the ISPs). These results suggest that since the distributions change over time (both with time-of-day variation, and differences year-by-year), looking at the aggregate distributions is less useful than studying the in-depth performance of the traces.

4.2.3 Summary

This section has discussed the queueing delay behaviour present in the measurement data presented in Chapter 3. As with packet loss, some obvious patterns like time-of-day variation were present, although not on all links. Also like packet loss, it appears that there are no obvious differences in performance due to the link type, but that the performance of the individual links can vary widely. Furthermore, examination of the queueing delay distributions at the trace level (i.e., plotting the DQ distribution from each trace separately, as seen in Figures 4.12 and 4.14) shows that the performance can vary even within traces. Therefore, further work should focus on understanding the variability within traces and the underlying behaviour of the network, such as congestion state, as discussed in Section 4.1.3.

4.3 Analysis of Packet-Pair Measurements

This section discusses the results of estimating the capacity of the end-to-end path between the sender and residential receiver using the packet-pair technique, as outlined in Section 3.2. I show the results obtained from the estimates, discuss their accuracy, and highlight interesting differences between the performance of the estimation from the ADSL and Cable links. Some further work investigating these differences is also presented.

4.3.1 Initial Packet-Pair Results

As discussed in Section 3.4.3, the arrival times of the two packets send back-to-back can be used to calculate an estimate of the capacity, as discussed in detail in [54]. The estimates from each link and rate were calculated using Equation 3.3. Figure 4.17 shows the capacity estimates obtained from each link. There are differences between some of the ADSL and Cable links, with the estimates obtained from the ADSL being fairly close to what they are expected to be, based on the speeds reported by the participants (between 8Mb/s and 24Mb/s), while the estimates for Cable are much higher, between 30Mb/s and 50Mb/s. This illustrates a clear difference between ADSL and Cable links. For the ADSL links, the packet pair capacity estimates match the values stated by the participants in Table 3.2 (except *adsl5* and *adsl6*), while the Cable links are not close at all. Further examination of the *adsl5* and *adsl6* links, looking at the downstream ADSL line rate as reported by the ADSL modem, shows that the packet-pair estimates shown in Figure 4.17 are actually fairly accurate. For *adsl5*, Figure 4.17 shows a strong mode between 5 and 10Mb/s; this is much lower than the 24Mb/s quoted in Table 3.2, but is close to the 6Mb/s reported by the modem. Although the service is marketed by the ISP as "up to 24Mb/s", lower rates are actually achieved in practice (e.g., due to noise on the line, or distance from the exchange). For *adsl6*, the modes in the packet-pair capacity estimate distribution are around 15-20Mb/s, which is close to 15Mb/s, the rate reported by the modem. However, although the packet-pair estimates appear quite accurate for ADSL links, the estimates obtained from Cable links are not.

One possible explanation for this is that since Cable uses a time-shared downstream channel, the estimates of 50Mb/s are due to both packets in the pair passing through in a single time slice. Therefore, although lower rates are achievable over the long-term (according to the fraction of time allocated to the user's subscription), the packet-pair estimates suggest that capacity is higher. In this sense, what is being estimated is not the capacity of the user's link, but rather the overall capacity of the shared downstream channel. This explanation is consistent with the findings of [120], which observed similar behaviour when sending packet pairs over Cable links. Since ADSL does not use this kind of time-shared access, this feature is not present for the ADSL links, and so in that case, the packet-pair estimates are more accurate.

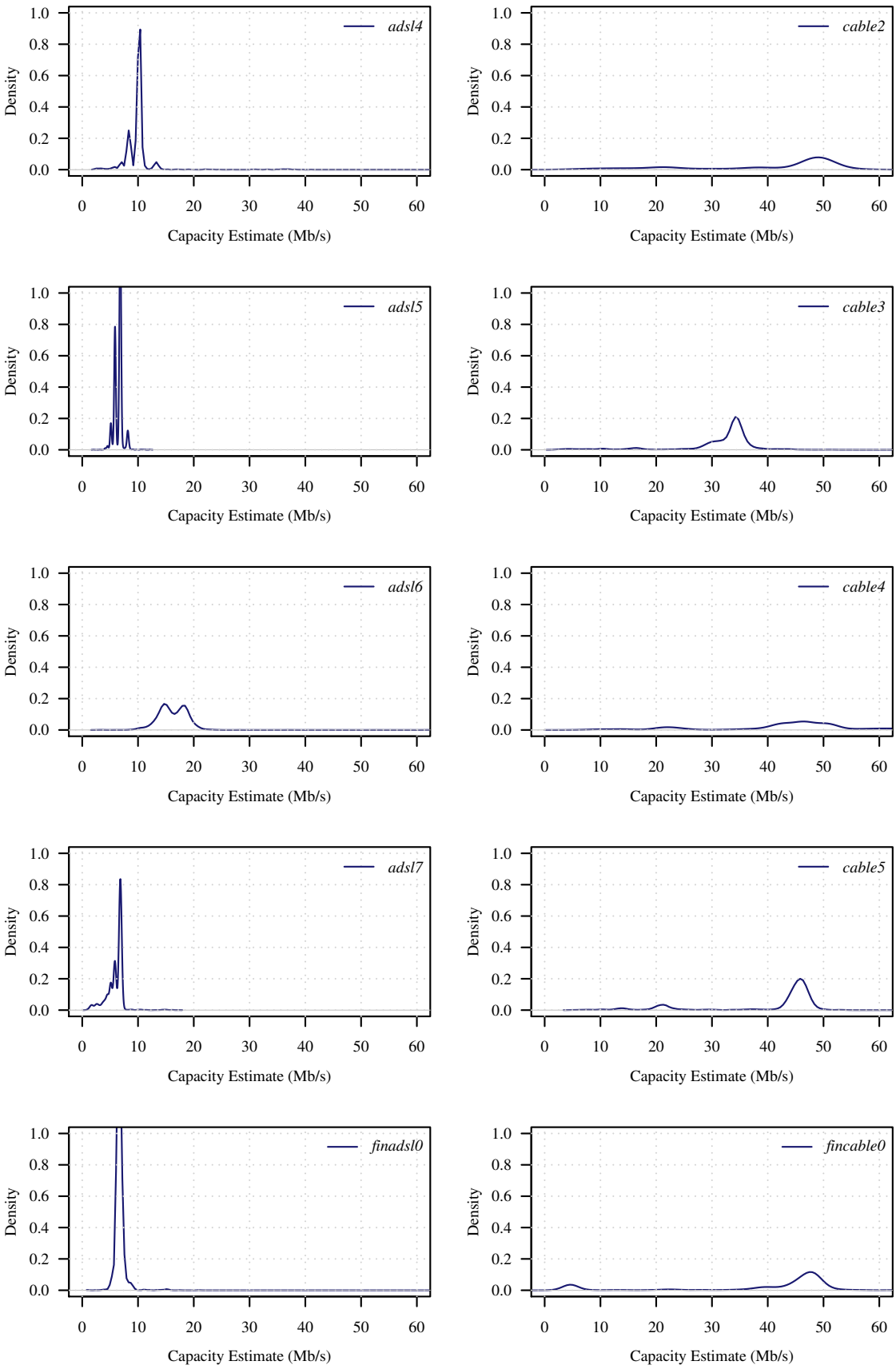
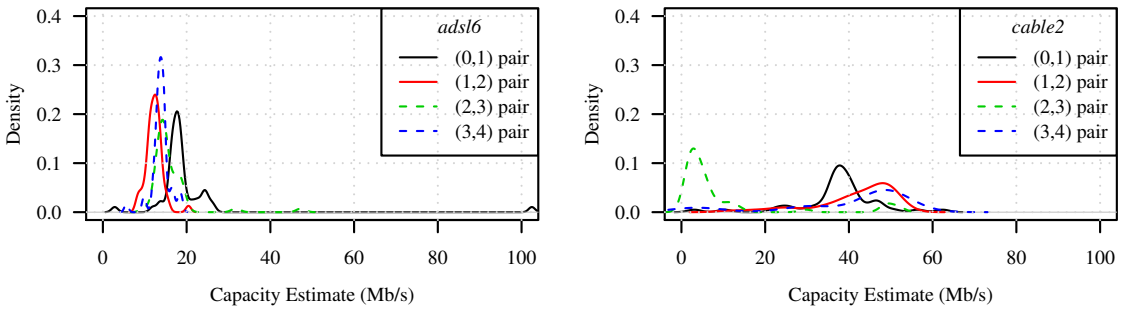


Figure 4.17: Packet-Pair Capacity Estimate Distributions (*dataset-B*)



(a) *adsl6* (b) *cable2*

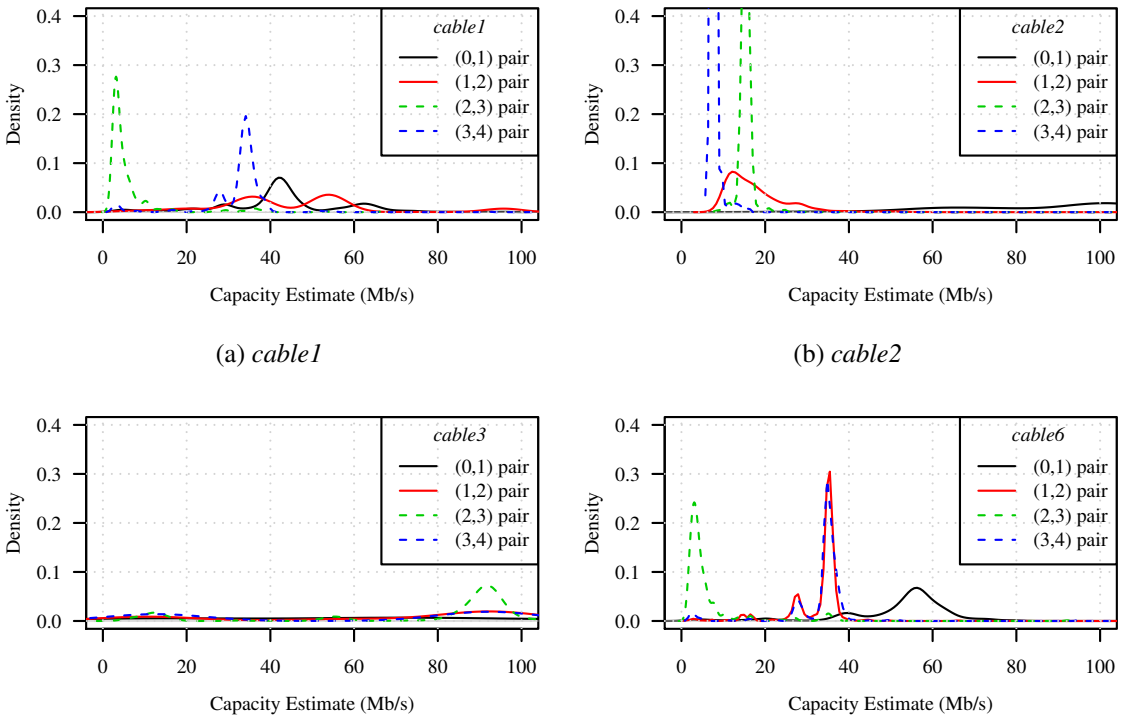
Figure 4.18: Capacity Estimate Distributions from Packet Trains (initial results)

4.3.2 Additional Measurements using Packet Trains

To validate the hypothesis that the erroneous capacity estimates obtained from Cable links are due to the downstream link scheduling in Cable, further experiments using *packet trains* (i.e., longer runs of back-to-back packets [54]) were conducted. The aim of these experiments was to show that, due to the lower-layer link scheduling on the Cable links, some packet-pairs pass through on a single time-slice at a higher instantaneous transmission rate, while others are separated between two time slices (separated on arrival by a large spacing).

In an initial study, conducted in Autumn 2010, packet trains of five 1316-byte packets were sent to one ADSL and one Cable link, from the same university-based host as in Chapter 3. The trains were separated by one second (so that their results are independent), and the packets within the train were sent as close together as possible. For each packet, the send and receive timestamps were recorded. Using this data, four pairs were identified in each train, from packet IDs (0,1), (1,2), (2,3), and (3,4), and capacity estimates were derived from each pair. Note that the per-pair capacity estimates are used, rather than the estimates obtained from the timestamps at the start and end of the whole train, since that approach exposes the timing to more variability due to cross-traffic [54].

These initial results are shown in Figure 4.18. For the ADSL link (Figure 4.18a), the results are as before, with the capacity estimates from different pairs falling around the same range. However, for the Cable link (Figure 4.18b), the capacity estimates depend on which packet-pair they are calculated from. The estimates obtained from the (2,3) pair are far lower than those from the other pairs, which show the higher-than-expected capacity seen in Figure 4.17. This supports the theory that the higher capacity estimates obtained from Cable links



(a) *cable1* (b) *cable2*

(c) *cable3* (d) *cable6*

Figure 4.19: Capacity Estimate Distributions from Packet Trains (further results)

in the original *dataset-B* packet-pair measurements are due to both packets in the pair being transmitted in a single lower-layer time slice. The lower estimates obtained from the (2,3) pair correspond to the case where the pair is split between two time slices.

Since these preliminary experiments implied a difference between the performance of capacity estimation using packet trains, it is possible that these might be used as part of a technique to identify access network types (e.g., as in [217], but using capacity estimates instead of inter-arrival times). Having an accurate technique to uncooperatively estimate access network type would be useful for over-the-top streaming video (e.g., allowing video parameters to be tweaked based on the access network type).

To investigate the feasibility of using packet trains as a link classification technique, some further measurements were collected, to determine whether the phenomenon in Figure 4.18b holds for Cable links in general. In these subsequent experiments (taken in Spring 2011), trains of packets were sent from the same university-based host as in Chapter 3 to four Cable receivers. The results of these experiments are shown in Figure 4.19. The results for two receivers, *cable1* and *cable6*, appear similar to before, with the (2,3) pair showing

much lower capacity estimates than the rest. However, the estimates from the other receivers (*cable2* and *cable3*) are different, no longer showing clear differences between the (2,3) pair and the others. Note that since the initial experiments also included receiver *cable2*, it is clear that something about the link changed during the measurement periods (such as an upgrade to the infrastructure, or a change in configuration). These results suggest that there is some promise to using packet trains for link identification, but that further work is required to investigate its feasibility, and the relationship between the Cable network configuration (i.e., sizes of lower-layer time slices) and the accuracy of the packet trains.

4.3.3 Summary

This section has discussed the results of estimating the capacity of end-to-end Internet paths traversed by residential broadband traffic. I have shown that for ADSL links, the packet-pair approach is fairly accurate, giving results quite close to the true capacity of the ADSL line. I have also confirmed previous findings about the inaccuracy of packet-pair based capacity estimates for Cable lines, and showed using additional packet train measurements that the downstream link scheduling policy used in Cable is responsible for this inaccuracy. An interesting avenue for future work would be for a sender to exploit this difference in accuracy to determine the type of access link (i.e., ADSL or Cable) being used by an arbitrary receiver. Initial investigations have showed that this might be possible, but that more work is needed, particularly in understanding the effect of Cable network configuration on this technique.

4.4 Discussion & Summary

In this chapter, I have studied the high-level characteristics of the measurement data introduced in Chapter 3, looking at results of packet loss, queueing delay, and capacity estimation. I have shown that for packet loss and queueing delay, there is a large amount of variation present between different links, on different traces for the same link, and in some cases even within the same trace. Some links show time-of-day variation in loss rates and queueing delay (as seen in other studies), but others do not. Similarly, the different sending rates used in the measurements affect loss rates, loss run-length distributions, and queueing delay for some of the links. An important issue is that these differences between rates tend not to be

related to the access technology (i.e., ADSL and Cable links do not always behave the same). As such, further analysis will treat each link separately.

One exception to this finding is the results of packet-pair based capacity estimation, where there is a clear difference between ADSL and Cable links. The results of this estimation are interesting, since the capacity estimates obtained for ADSL links are reasonably accurate, while the estimates obtained for Cable tend to be over-estimates with large variance. The reason for the inaccuracy on Cable links is due to the underlying mechanism used to control access to the shared physical medium. Although this may prevent estimation of link capacity using packet-pairs, it may be possible to exploit this inaccuracy in capacity estimation to classify the access network type as either ADSL or Cable, although further work is needed to investigate this.

The packet loss and delay behaviour observed in the measurements, particularly the congestive loss discussed in Section 4.1.3, suggest a difference between these measurements of end-to-end paths towards residential users and previous measurements of backbone networks. Previous work on measuring the performance of backbone networks suggested that there was low loss and not much variability in delay, indicating that "there was little congestion along the end–end path" [95]. However, the results presented in this chapter have shown evidence of congestive packet loss (Figure 4.7), as well as highly variable queueing delay (Figure 4.12d). Therefore, at least in some cases, the conditions experienced by residential users are different from those in backbone networks.

The large variation between the links and between traces of the links themselves indicate that for the following chapters, working on a per-trace basis is the most appropriate. Since these chapters will look at how to model, simulate, and evaluate application performance, it makes sense to focus on understanding the characteristics of a particular link *at a particular time*, rather than trying to consider the performance of the links as a whole. Once the behaviour during the relatively short periods of time seen in the measured traces is understood, conclusions can be drawn about the behaviour of the links in general.

Chapter 5

Evaluating Markov Models for Packet Loss

Packet loss in residential broadband networks can disrupt end-user experience for many real-time network applications used in the home, such as Internet video and conferencing, and has highly variable behaviour, as described in Chapter 4. Using statistical models to capture the behaviour of these packet loss processes allows analytic and simulation studies to evaluate the performance of new applications and services, without the need for full-scale deployments. Moreover, by using models, network performance can be represented using only a set of model parameters, rather than relying on full packet traces (which may require a large amount of memory or storage capacity), or summary statistics, which tend to give a limited perspective.

Many studies have used Markov-chain models such as the classical Gilbert model to simulate packet loss processes; however, the accuracy of these models for characterising packet loss on residential broadband networks remains untested. Since the technologies, configuration, and usage of residential networks differs from those of academic or enterprise networks, different loss characteristics can be seen, as discussed in Chapter 4.

In this chapter, I apply a number of well-known Markov models to the measurements of packet loss described in Chapter 3, to evaluate the accuracy of these models in capturing real-world packet loss behaviour as seen by residential broadband users. The evaluation process consists of two parts. First, I evaluate model performance by generating a small number of synthetic sequences from the model for each trace. I compare these to the original trace sequence, looking at metrics important to the application, and visualise the packet loss

sequences themselves, for some representative example traces. Then, I use a more general approach, generating a larger number of synthetic sequences from each trace, and apply a goodness-of-fit test to validate whether each particular model captures a range of statistics of the packet loss traces. The results show that while in many cases these well-known models capture the measured loss conditions, there are also periods where distinctly different behaviour is present, which cannot be captured using these models. I explain why the loss behaviour in the traces varies, and why the models are unable to capture the range of behaviour in the loss patterns. Chapter 6 will incorporate packet delays into the modelling, but this chapter will focus on loss alone. This allows the introduction of the modelling techniques on the simpler case, and establishes how much can be understood about the network using existing models that use only loss observations. Also, since there is a larger body of existing work on loss modelling, although not for residential broadband networks, this provides a more suitable starting point.

This chapter is structured as follows. Section 5.1 describes the Markov models that will be evaluated in this chapter. Section 5.2 details the first part of the approach I will use to evaluate models, including performance and comparison metrics. Section 5.3 presents the results of the first part of the evaluation, showing how each model captures the performance metrics described in Section 5.2. Section 5.4 presents the second part of the evaluation, examining the overall performance of the models across all the traces, by generating a large number of synthetic sequences in a process called *parametric bootstrap*. Section 5.5 explains why the models perform as they do, and suggests how they can be improved. Section 5.6 discusses alternative approaches to modelling, and how these techniques might be applied to the data to better understand network performance. Section 5.7 gives a summary of the chapter.

5.1 Models for Packet Loss

This section describes the Markov models that will be evaluated in this chapter. For these models, the parameters and parameter estimation process are described in detail, and their benefits and drawbacks are compared.

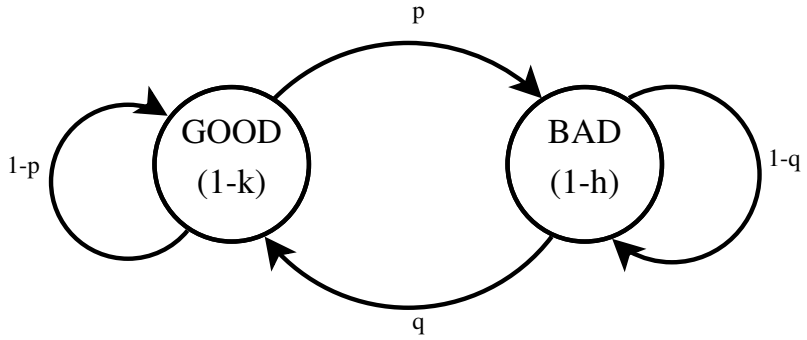


Figure 5.1: Gilbert-Elliott Model

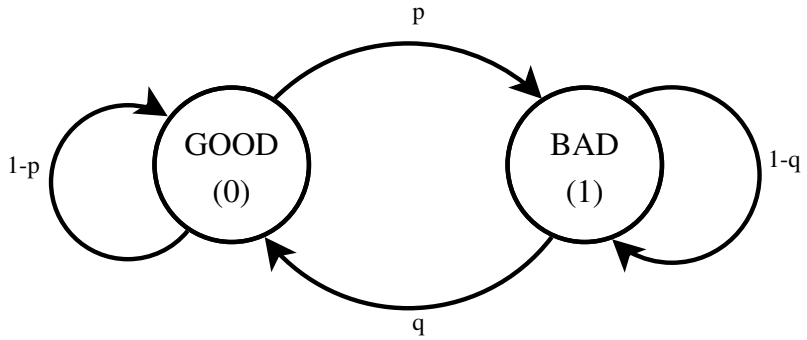


Figure 5.2: Simple Gilbert Model

5.1.1 Gilbert Models

The simplest of the Markov models considered is a two-state Markov chain. The model first proposed by Gilbert [73] and later expanded by Elliott [59] contains two states (GOOD and BAD). As shown in Figure 5.1, in the Gilbert-Elliott model the GOOD state produces errors with probability $(1 - k)$, and the BAD state produces errors with probability $(1 - h)$.

This can be simplified by setting $h = 0$, and $k = 1$, such that the GOOD state never produces losses, while the BAD state always does, as shown in Figure 5.2. Hereafter, this will be referred to as the Simple Gilbert Model (SGM). In this model, the observation sequence Z_i (i.e., a "bitmap" of packet loss) directly represents the state of the model.

The parameters of the SGM are p, the probability of losing a packet, given that the previous packet was received (i.e., the "first loss" probability), and q, the probability of receiving a packet, given that the previous packet was lost. To estimate these parameters, it suffices to count the number of transitions between states n_{ij} and the number of received and lost packets (n_0 and n_1, respectively), as described in [224]:

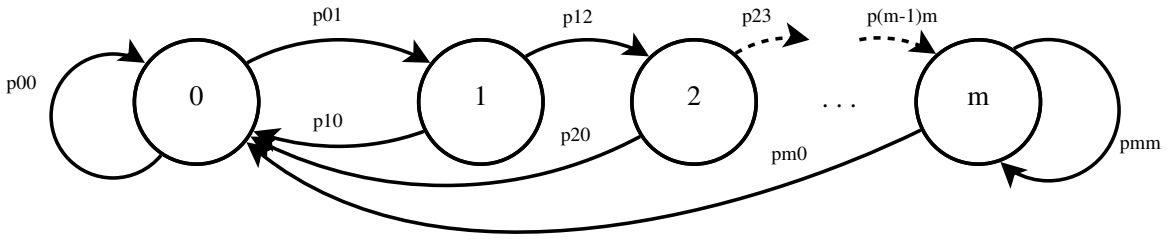


Figure 5.3: Extended Gilbert Model

$$\hat{p} = \frac{n_{01}}{n_0}, \quad \hat{q} = \frac{n_{10}}{n_1} \tag{5.1}$$

The SGM has been heavily used for evaluating application performance, especially after early work on academic networks found it to be reasonably suitable for describing loss processes [24, 224]. Recent evaluations of network video streaming systems [208] and video quality estimation tools [206] have used SGM models to simulate packet losses. A number of analytical evaluations of FEC performance have also used the SGM as the basis for their work [70, 71, 97, 225, 103]. Outside the multimedia area, the SGM has been used for simulation of packet loss when evaluating network diagnosis and inference tools [229, 152, 199].

One of the drawbacks of the Simple Gilbert Model is that since there are only two states, it does not capture longer bursts of packet loss. To address this, a new model, which will be referred to as the Extended Gilbert Model (EGM) was proposed [183, 100]. In the EGM, the number of states to capture loss is extended from one (as in the SGM) to m, as illustrated in Figure 5.3. This allows the model to describe loss bursts of up to m packets, with each state representing a loss burst of a particular length. That is, state $i, i < m$ represents a loss burst of i packets, while state m represents a loss burst of *at least* m packets. Received packets are represented by state 0, and run-lengths of received packets are not captured by the model.

The parameters of the EGM are, as for the SGM, the probabilities of observing a lost or received packet, for each of the states. To estimate these parameters, the number of occurrences of run-lengths, o_k (where k is the loss burst length), is counted. The probability of moving from state $(k-1)$ to state k, $p_{(k-1)(k)}$ is then (as shown in [183]):

$$p_{(k-1)(k)} = \frac{\sum_{n=k}^{\infty} o_n}{\sum_{n=k-1}^{\infty} o_n} \tag{5.2}$$

In the special case of p_{mm}, all $o_n, n \geq m$, are counted to measure all loss run lengths of *at least* m packets. In this analysis, $m = 5$ was chosen since more than 99% of the loss bursts

in the data consist of 5 packets or fewer, as shown in the loss burstiness results presented in Section 4.1.2. In their study of the EGM, Jiang & Schulzrinne [100] suggest that using the EGM to capture loss burst lengths alongside the inter-loss period lengths metric proposed by the IETF IPPM working group [113] to capture receive run-lengths is more effective than the SGM, showing results using the same traces used to evaluate the SGM in [224].

Although Gilbert models have received some criticism [226], they are still widely used (as discussed at the start of this section), since they are simple to compute and understand. Therefore, I believe that it is appropriate to study the performance of Gilbert models, to learn just how suitable they are for describing packet loss from paths including residential broadband networks.

5.1.2 Hidden Markov Models

The extension to the original Gilbert model proposed by Elliott [59] allows both states to produce errors, and with different probabilities. 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 with a Hidden Markov Model (HMM), where only the observations of packet loss can be seen, and the actual state (GOOD or BAD) is hidden. This allows correlations in the sequence of packet loss observations Z_i to be better captured, since in an HMM, the observation process being modelled is no longer assumed to be Markov (i.e., memoryless), as is the case for the SGM. That is, the states are hidden (just as the underlying state of the network is hidden from the observer at an end-point), and transitions between states are inferred from the observations of packet loss.

The two-state HMM contains states to capture two of the different types of loss that can be expected; random, uncorrelated losses (possibly due to noise on the access link), and more bursty, correlated losses (possibly due to congestion). The two-state case is introduced first, partly because this seems an appropriate starting point before adding more states, but also because this is how the Gilbert-Elliott model is structured. Figure 5.4 illustrates a two-state HMM, with the parameters of the Gilbert-Elliott model (seen previously in Figure 5.1).

As Figure 5.4 shows, the parameters of such a two-state HMM are p, q, h, and k (i.e., the same parameters of the Gilbert-Elliott model). These are the components of the transition

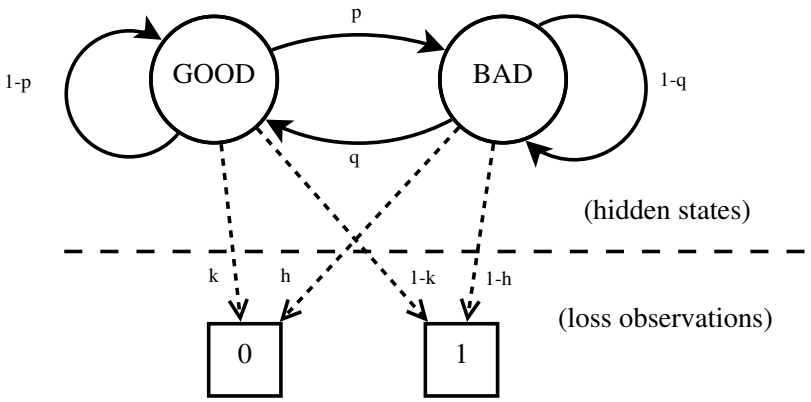


Figure 5.4: Gilbert-Elliott Model as a Two-State Hidden Markov Model

probability matrix of the hidden states, $\mathbf{A}$, and the state-dependent observation probabilities, $\mathbf{B}$ (i.e., the probability of observing a lost packet in each of the hidden states). In terms of the parameters of the Gilbert-Elliott model:

$$\mathbf{A} = \begin{bmatrix} (1-p) & p \\ q & (1-q) \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} (1-h) & (1-k) \end{bmatrix} \tag{5.3}$$

When using HMMs, the transition probabilities between hidden states are estimated from the observed data, although this is more complex than for the Gilbert models. This estimation is commonly done using the Baum-Welch algorithm, an Expectation-Maximisation (EM) algorithm that aims to converge on the parameters with the maximum likelihood. This works by iteratively calculating the likelihood of the observation sequence, given parameters $\mathbf{A}$ and $\mathbf{B}$ (the "expectation" step), and then maximising this expectation (the "maximisation" step). This process repeats until either convergence is reached, or a specified maximum number of iterations is performed. Note that in some cases, the EM process can fail to converge, or can converge on a "local maximum" in the likelihood function that may not be the global maximum. Further detail on HMMs, parameter estimation, and the EM algorithm can be found in [175, 23], and a survey of HMM applications can be found in [45].

The approach described here can lead to issues with the parameter estimation, and may be a problem for the accuracy of the models, since only a single set of parameters is being estimated using EM (this is a particular problem since the estimation process may not converge to a global maximum for the parameters). If more traces were available from each time period, this would allow multiple sets of parameters to be estimated; if similar parameters are estimated from multiple traces, this would give more confidence that the parameters are accu-

rate. This issue should possibly be taken into account in future measurement work, allowing for more measurement traces collected around the same time, to enhance confidence in the results. Collecting a larger number of shorter traces at each time period will allow estimation of multiple parameter sets, possibly improving the accuracy of modelling techniques. However, it is important to remember that this needs to be carefully balanced to ensure that the traces themselves have enough observations to capture the events of interest in the measurements (e.g., switches between different congestion states). Finally, these measurements will also need to take into account the constraints on the measurement methodology (particularly bandwidth usage limitations) as discussed in Chapter 3, and choose an appropriate trade-off between the bandwidth usage, the insight into time-series behaviour, and the usefulness of the traces for modelling

To do the parameter estimation, the R [174] package *hmm.discnp* [209] is used. This provides functions to calculate $\mathbf{A}$ and $\mathbf{B}$, as well as to simulate synthetic data from a given HMM, which can be compared to the original data, as Section 5.2 will discuss. Using *hmm.discnp*, the raw loss time-series (i.e., Z_i) is passed into the function to estimate the parameters of the HMM, which also uses this time-series to derive initial estimates for $\mathbf{A}$ and $\mathbf{B}$ (these initial estimates are used since no single value for $\mathbf{A}$ and $\mathbf{B}$ will be appropriate over all the traces). Once the HMM parameters have been estimated, these can then be passed into a further function to simulate the transitions in the HMM (i.e., switching between hidden states, and generating the synthetic loss sequence).

The HMM can be extended to contain more states, which will improve its accuracy, although in practice the complexity of the estimation process limits the number of states. In this work, I will focus on two- and three-state HMMs. In reference to the underlying network conditions, the states of two-state HMM are expected to correspond to quiet loss periods (only isolated packet losses due to electrical noise, which is expected to only occur on access links), and bursty loss (where losses are due to buffer overflows and congestion). These correspond to the GOOD and BAD states originally discussed by Gilbert [73] and Elliott [59]. By adding another state, the three-state HMM should offer more resolution, with the aim of being more precise about the underlying loss process. Thinking further about where loss can occur, as discussed earlier in Section 2.2, distinguishing between congestion-based losses within the core networks and at the ISP access routers implies that a three-state HMM may be appropriate. While it is unclear whether the parameter estimation process of the

HMMs will identify states in this way, the number of states is chosen with these distinctions in mind. So, the two-state HMM has states corresponding to loss due to electrical noise on the access link, or loss due to congestion, and the three-state HMM has states corresponding to loss due to electrical noise on the access link, loss due to congestion in core networks, or loss due to congestion at ISP access routers.

HMMs were first employed for modelling packet loss by Salamatian & Vaton [181], 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 simpler Gilbert models. Wei *et al.* [215] and Salvo-Rossi *et al.* [181] used HMMs to study the behaviour of both loss and delay within their measurement traces. Silveira & de Souza e Silva [190] show results of using HMMs for prediction of network conditions, and apply these to dynamic adaptation of FEC parameters.

5.1.3 Models to Compare

For the remainder of this chapter, I will study four models in detail; the Simple Gilbert Model (SGM), Extended Gilbert Model (EGM), and two- and three-state Hidden Markov Models (2HMM and 3HMM). The SGM is important to evaluate since it is so widely used in analytical and simulation studies. Assessing its performance in capturing the loss patterns perceived by residential streaming video customers is important for future studies of such systems. Similarly, since the EGM was proposed to improve upon the SGM specifically in Internet streaming media [100], I will study its performance (and particularly its improvement over the SGM).

Since HMMs aim to model the "hidden states" of the network, they seem appropriate for modelling packet loss in residential streaming video, since the residential end-points have no knowledge of the state of the network, and the performance they see is dictated by the "dominant" loss characteristic of the network. Moreover, previous work has shown HMMs to be suitable for capturing the loss characteristics of backbone networks [181, 182], so in this chapter I study their performance using the packet loss measurements from Chapter 3. The choice of two- and three-state HMMs corresponds the level of detail that the model perceives; either "congested" or "uncongested" (2HMM), or looking at "uncongested loss due to noise", "congestion in core network", or "congestion at access routers" (3HMM).

5.2 Evaluating Models

To evaluate the performance of the models, it is necessary to understand how closely they capture particular features of the packet loss sequences. Therefore, Section 5.2.1 defines performance metrics that will be calculated for the original data, and then obtained from the model. These metrics will then be compared using measures of distance, which will be described in 5.2.2, with their "closeness" representing how well the model fits. The process to assess the model is as follows:

1) apply modelling process, and derive model parameters

2) generate synthetic data using the model parameters

3) calculate performance metrics for the original data

4) calculate performance metrics for the synthetic data

5) compare metrics between original and synthetic data

This involves deriving model parameters from the input trace data, then using the models to simulate three synthetic sequences that are each compared to the original data. Three sequences, rather than one, are generated: 1) to obtain a measure of the variability of the sequences produced by the model, and; 2) to ensure that when loss rates are low, sequences with some losses are available for comparison (since the synthetic sequences might otherwise contain zero losses). For these purposes, three sequences are sufficient, and reduce the time taken to apply the modelling process over all the traces. However, a more intensive process where a larger number of sequences is calculated will be discussed in Section 5.4.

Generating synthetic data from the models is more attractive than alternative approaches (e.g., numerically deriving performance metrics from the model parameters), since it produces actual sequences. This allows easy comparison between the original data and synthetic data generated from a number of different models, using any number of possible performance metrics, as well as allowing side-by-side visualisation of the original and synthetic data.

5.2.1 Performance Metrics

The most fundamental and easy to understand performance metric for packet loss is the *mean loss rate*, the average rate of packet losses. Therefore, any model that seeks to capture loss

behaviour must be able to accurately express the mean loss rate. For a sequence of packets, where Z_i represents whether packet i was lost or not ($Z_i = 0$ if packet received, $Z_i = 1$ if packet lost), the mean loss rate is $\bar{Z}$.

However, since the loss process is bursty, the mean loss rate is not sufficient to fully describe the loss characteristics of the traces. For example, consider two sequences of length 10000 packets, which both have a loss rate of 1%. The first sequence loses roughly one packet in every 100; the second has a long period with no loss, then a short period with 100 losses in short succession, followed by another long loss-free period. The difference between these two loss sequences is important, since their effect on video quality are likely to be very different. In the first case, a simple FEC scheme will easily cope with the loss, ensuring the video is uninterrupted, while in the second case, the FEC is likely to be overwhelmed, leading to a degradation in video quality.

Looking at the *distribution of run-lengths* (both runs of lost packets, X_i, and runs of received packets, Y_i) gives more insight into the loss behaviour by giving information about the burstiness of the loss process. A second metric, therefore, is the *complementary cumulative distribution function* (CCDF) of the loss run-lengths and the received run-lengths. CCDFs are commonly used to visualise distributions of packet run-lengths (e.g., [128, 28]), since they show on the y-axis the probability of seeing a run-length *greater than* the run-length on the x-axis:

$$CCDF_{RL} = Pr[RL > x] \tag{5.4}$$

Previous work [224, 28, 60] has shown evidence of burstiness and correlation between packet losses. Therefore, another feature of interest is the *autocorrelation of the loss time-series* Z_i. This gives a measure of the dependence in packet loss (i.e., if a packet is lost, future packet losses are more likely).

Yajnik *et al.* [224] present a metric for comparing the extent of correlation in the loss traces. The *correlation timescale*, c, described as follows: "the minimum lag, in terms of time, such that Z_i is uncorrelated at all lags, d", where $d \geq c$ [224]. Since this metric has been used in previous work to compare correlations, and is relatively simple to calculate, this section will use it as a means of comparing raw and synthetic data, by examining the differences between the correlation timescales in original data, and those in the synthetic data. To calculate the correlation timescale, I look at the autocorrelation values for the observation se-

ries (either original or synthetic), starting with the maximum lag seen. If the autocorrelation $\rho(h)$ at lag h is significant, then the correlation timescale is at least h packets. Note that this differs slightly from the definition given above; the calculation of the autocorrelation (using the standard `acf` function in R [174]) requires a maximum lag, which means a lower bound on the correlation timescale is obtained.

To test whether $\rho(h)$ is significant, the procedure in [224] is followed to test that the observation sequence Z_i is independent and identically distributed (i.e., not correlated). The null hypothesis H_0 that Z_i is uncorrelated implies that the sample autocorrelations of Z_i, $\rho(h)$ follows a normal distribution with mean 0 and variance $1/n$. If this is true, around 95% of $\rho(h)$ will fall in the range $[\frac{-1.96}{\sqrt{n}}, \frac{1.96}{\sqrt{n}}]$. Therefore, if $|\rho(h)| > \frac{1.96}{\sqrt{n}}$, then H_0 is rejected with significance level 0.05, implying that significant correlation is present at lag h.

5.2.2 Comparing Metrics

To compare the mean loss rates for the original and synthetic traces ($\bar{Z}_{raw}$ and $\bar{Z}_{synth}$, respectively), I use two distance measures. The *mean loss ratio* R_{ML} captures the proportional difference in loss rates, and the *mean loss difference* Δ_{ML} captures the absolute difference:

$$R_{ML} = \frac{\bar{Z}_{synth}}{\bar{Z}_{raw}} \tag{5.5}$$

$$\Delta_{ML} = \bar{Z}_{synth} - \bar{Z}_{raw} \tag{5.6}$$

There are cases where the loss rate is low (e.g., one packet lost out of ten thousand); however, if a model generates two lost packets instead, the R_{ML} value will be 2, suggesting the model is producing 100% higher loss rates than in the original data, which although true, is somewhat misleading. By using the absolute and relative loss rates together (Δ_{ML} and R_{ML}, respectively), a clearer picture can be obtained.

For the run-length distributions, however, it is not so simple. The CCDFs can be plotted and compared visually, which gives a good way to compare the performance of a model on a particular trace. However, applying such visual checks to all traces will be quite cumbersome for a large dataset, and liable to inaccuracy. A more numeric approach is to use a distance measure between the run-length distributions themselves, such as the Kolmogorov-Smirnov (K-S) D statistic [201]:

$$D = \sup_x |F_{raw}(x) - F_{synth}(x)| \tag{5.7}$$

where $\sup S$ is the supremum of set S, and $F_{raw}(x)$ and $F_{synth}(x)$ represent the empirical distribution function of run-lengths from the original (raw) data and synthetic data, respectively. This comparison measure is appropriate, since it is non-parametric (i.e., makes no assumptions about the underlying distribution of the data). The D statistic has previously been used to compare distributions of network data [77]. Although it is associated with a hypothesis test (the Kolmogorov-Smirnov test), some problems exist with directly applying the results of this test. Haddadi points out that "it is misleading to use this test to indicate if two distributions are similar, as it is highly sensitive to large sample sizes" [77], and instead just uses the D metric as a measure of relative closeness. Therefore, the results of the Kolmogorov-Smirnov test, and associated p-value (which determines whether to reject the null hypothesis that the samples come from the same distribution), should be used with caution. For the evaluation of run-length distributions, I will calculate the K-S distances and p-values, but also visualise the synthetic data generated by the models (as Section 5.2.3 will discuss) to verify the K-S distance results.

To test how well the models capture the correlation in the traces, the difference between the correlation timescale of each synthetic sequence c_{raw} and that of the original trace c_{synth} is calculated:

$$\Delta_c = c_{raw} - c_{synth} \tag{5.8}$$

Using the difference in correlation timescales, Δ_c, the extent to which the models capture the correlation in the original traces can be assessed. When there is correlation in the original trace that is not captured in the synthetic data, Δ_c will be negative.

5.2.3 Visualising Model Results

In addition to the aggregate metrics already discussed, it is important to be able to visualise exactly what synthetic sequences look like, on a per-trace level. Having this greater level of detail is important to assess whether the aggregate metrics (e.g., the K-S D distance between run-length distributions) are missing any details, since using aggregate statistics inevitably

loses information (by collapsing all the detail from an entire trace into a single value). Therefore, a visualisation of the model output will be used to complement the aggregate metrics.

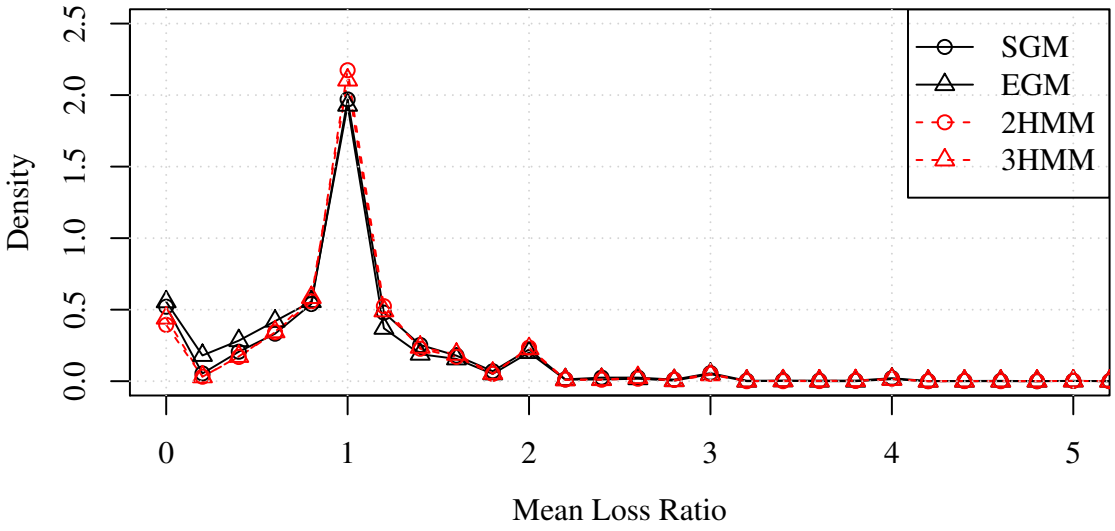
The visualisation used shows loss and receive run-lengths from the packet loss sequences, visualised as a plot where the black vertical bars represent loss runs, and grey vertical bars represent receive runs, with the thickness of the bar representing the run-length. This provides a fairly compact representation of the packet loss observation series Z_i generated by the models, which can be displayed alongside their corresponding original packet loss traces for ease of comparison.

5.2.4 Summary

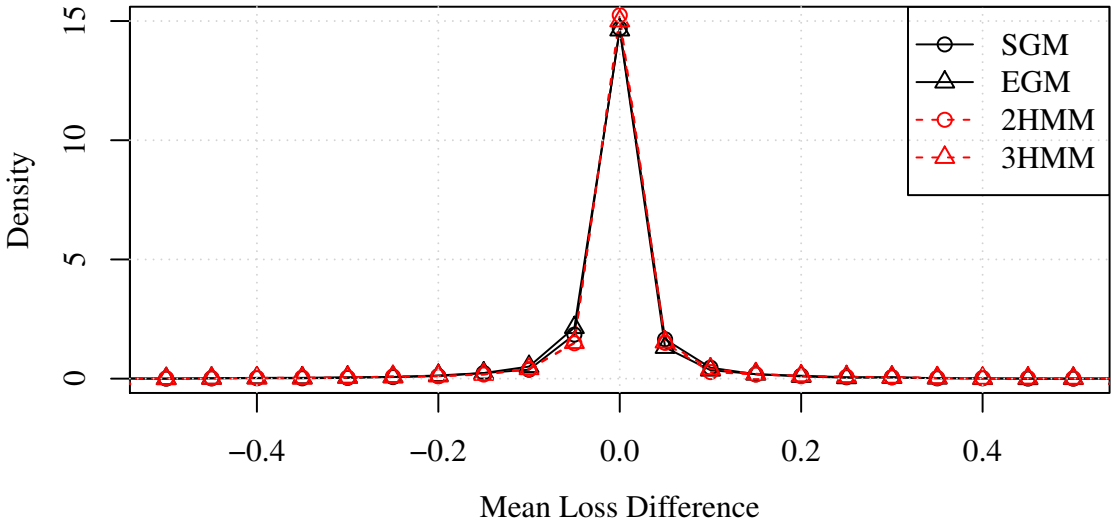
This section has presented the process for evaluating models for packet loss, and described performance metrics (mean loss rate, loss and receive run-lengths, and autocorrelation) that the models need to capture in the raw data. It has also introduced "distance measures" that will be used to compare the differences between the original and synthetic (model-generated) data at a per-trace level. The mean loss ratio R_{ML} and mean loss difference Δ_{ML} for comparing mean loss rates, the Kolmogorov-Smirnov D statistic for comparing run-length distributions, and Δ_c, the difference in correlation timescales, for comparing correlation in the loss time-series. Using these distance measures to look at model performance overall, and per-trace visualisation on example traces to show in-depth model performance, the following sections will report the extent to which the various models match the data.

5.3 Model Performance Results

This section presents results of applying the models described in Section 5.1 to the loss data described in Chapter 3. Initially, the results of each of the metrics across all traces are presented, according to the process described in Section 5.2.1. Then, a number of example traces are visualised in detail, to explain the differences in the loss patterns, and how these affect the accuracy of the models.



(a) Mean Loss Ratios R_{ML}



(b) Mean Loss Differences Δ_{ML}

Figure 5.5: Distributions of Mean Loss Ratios and Differences. Synthetic sequences are generated from each trace, and R_{ML} and Δ_{ML} from each of these is calculated.

5.3.1 Mean Loss Rate

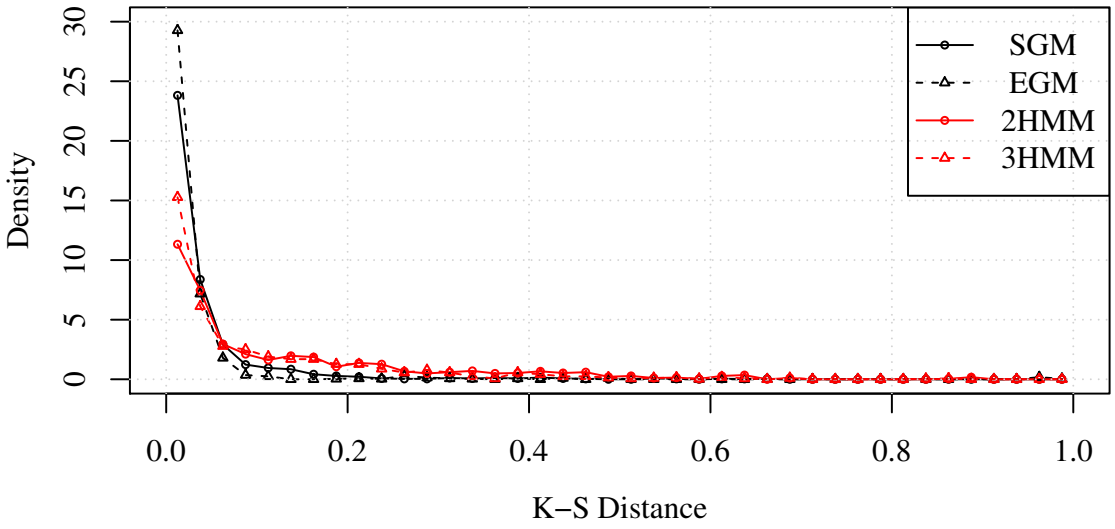
Figure 5.5 shows how closely each of the models capture the mean loss rates in the traces, by aggregating the results of the R_{ML} and Δ_{ML} metrics discussed in Section 5.2.1. These figures show the aggregate performance over all traces. For each of the three synthetic traces generated by the model (as discussed in Section 5.2), R_{ML} and Δ_{ML} are calculated, and this is repeated for all traces to get a set of values. Figures 5.5a and 5.5b show the distribution of this set of values for each model.

Figure 5.5a shows a strong peak around 1, showing that many of the synthetic sequences match the loss rate very closely. The peak around zero shows cases where the synthetic sequence generated no losses, while there were losses in the original trace. However, as Figure 5.5b shows, the absolute difference in loss rates is not large, mostly within 0.1%. These results suggest that all models perform well on this metric, with no clear distinction between their performance.

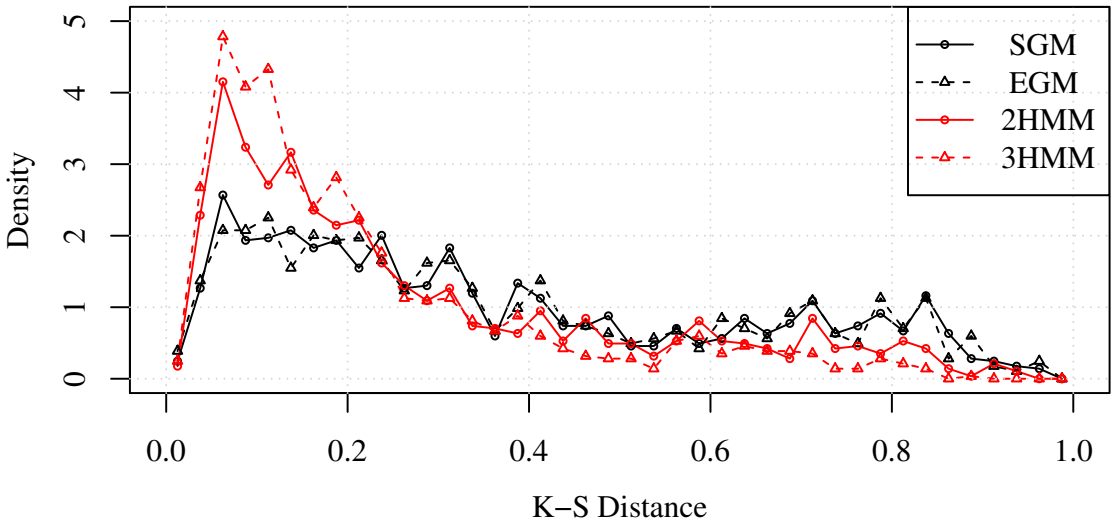
5.3.2 Run-Length Distributions

Figure 5.6 shows the K-S distances between the run-length distributions of the original traces and the synthetic data, for loss run-lengths (Figure 5.6a) and receive run-lengths (Figure 5.6b). Note that only those traces where at least 100 run-lengths are present are shown, since applying the Kolmogorov-Smirnov test to distributions with only a few values might lead to inaccurate conclusions. Since a threshold is necessary, a value of 100 runs was chosen after examining the data. Table 5.1 shows the percentage of trace runs (including all three synthetic sequences per trace) that are well-modelled, according to the Kolmogorov-Smirnov hypothesis test (as discussed in Section 5.2.1).

Figure 5.6a shows that for loss run-lengths, the Gilbert models perform well, with the EGM being the most accurate. The HMMs, however, show poorer performance, with a less pronounced peak around 0, and a longer tail. This means that for the HMMs, fewer traces have a K-S distance close to zero, showing more traces in the 0.1–0.3 range. This is also reflected by looking at the results of the K-S hypothesis tests in Table 5.1 (although recall that these results have some caveats, as discussed in Section 5.2.1). For the SGM and EGM, the percentage of traces being "well-modelled" (i.e., where the null hypothesis of the original and synthetic model-generated loss run-length distributions being drawn from the



(a) Loss Run-Lengths



(b) Receive Run-Lengths

Figure 5.6: Distributions of K-S Distances (D) Between Run-Lengths. Synthetic sequences are generated from each trace, and D is calculated for those traces with at least 100 run-lengths.

Model	Percentage of traces	
	Loss Run-Lengths	**Receive Run-Lengths**
SGM	91.0%	20.2%
EGM	96.0%	19.6%
2HMM	55.2%	24.0%
3HMM	63.9%	27.5%

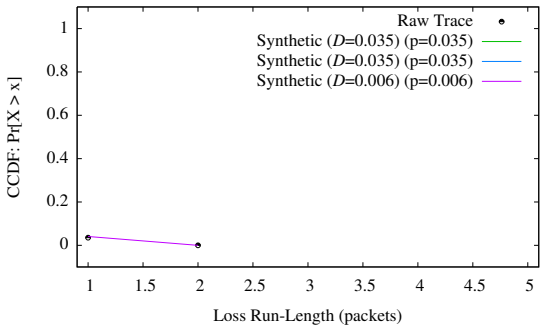
Table 5.1: K-S Hypothesis Test Results: percentage of traces where null-hypothesis of "same distribution" cannot be rejected at a significance level of 0.05.

same distribution could not be rejected) is 91% (SGM) and 96% (EGM) of traces. For the HMMs, these numbers are much lower; 55% (2HMM) and 64% (3HMM).

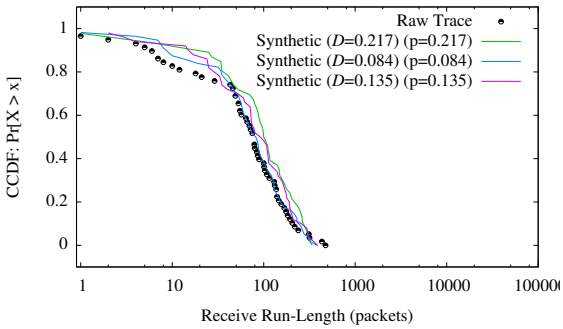
Figure 5.6b shows the distribution of K-S distances for receive run-length distributions. In this case, the HMMs have higher proportion of traces with low K-S distances, showing better performance than the SGM and EGM. However, the percentage of "well-modelled" traces is lower for all models than was the case for loss run-lengths. The SGM and EGM models capture the receive run-length distributions for only $\sim$20% of traces, which improves slightly to 24% for the 2HMM, and 28% for the 3HMM.

The results for both the loss and receive run-length distributions show rather poor performance for all the models (with the exception of SGM/EGM performance in capturing loss run-lengths). In particular, the poor performance of the HMMs is somewhat surprising since, for example, the two-state HMM generalises the SGM, and might therefore be expected to give performance at least as good as the SGM. Looking further into the run-length distributions obtained from each of the models, it is clear that the Gilbert models do indeed capture the loss run-lengths more accurately than the HMMs. However, the SGM and EGM fail to capture the long receive runs present in some of the traces, particularly when there are many short receive runs (within bursty loss periods), as well as long receive runs within the same trace. Section 5.3.4 will demonstrate this further by showing examples of the original traces, comparing these against the sequences generated by the models.

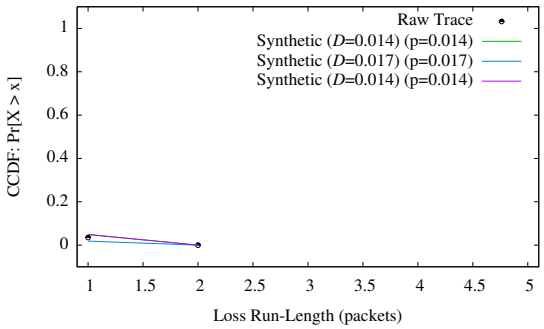
Figures 5.7 and 5.8 show CCDF plots for two example traces, showing the original data and each of the three synthetic sequences. In the first trace, losses occur frequently throughout, while in the second, there are "on/off" bursty loss and loss-free periods. Figure 5.8



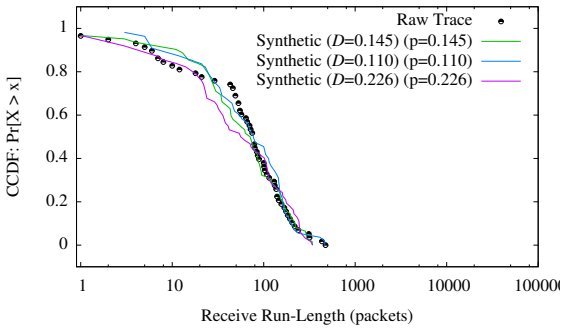
(a) SGM (loss)



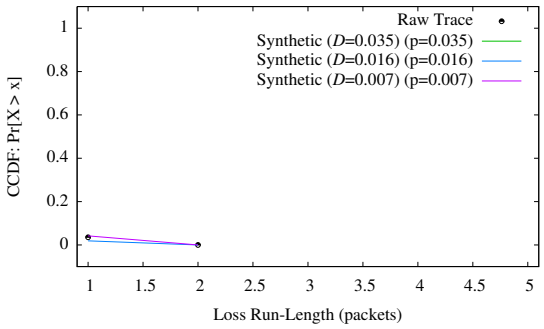
(b) SGM (receive)



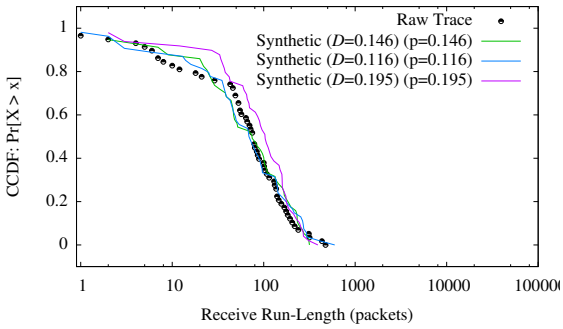
(c) EGM (loss)



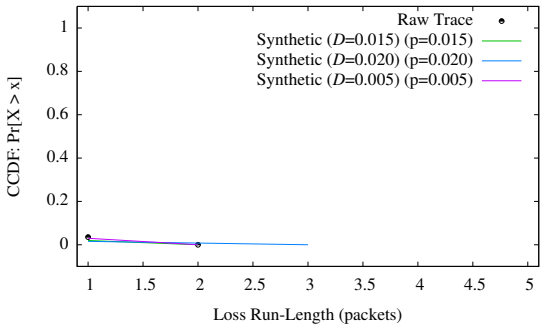
(d) EGM (receive)



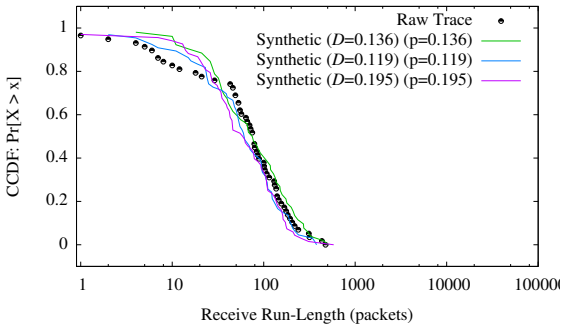
(e) 2HMM (loss)



(f) 2HMM (receive)

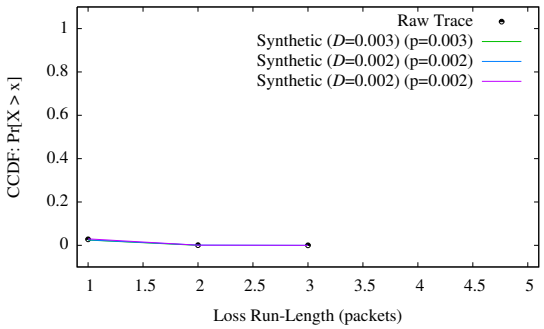


(g) 3HMM (loss)

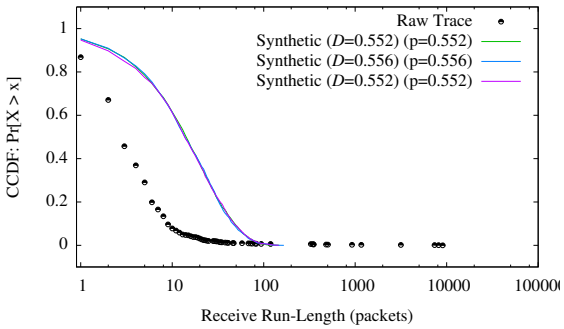


(h) 3HMM (receive)

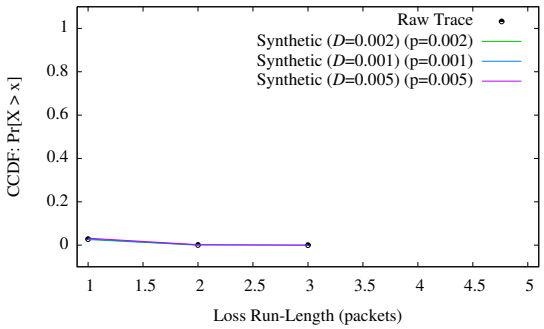
Figure 5.7: Example CCDF Distributions (“non-bursty” loss) (same trace as Figure 5.13b)



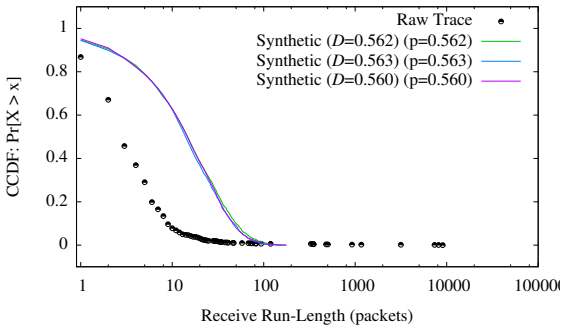
(a) SGM (loss)



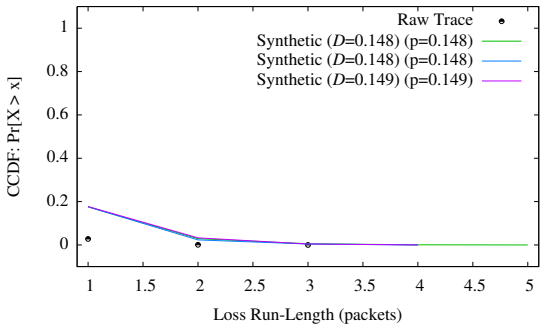
(b) SGM (receive)



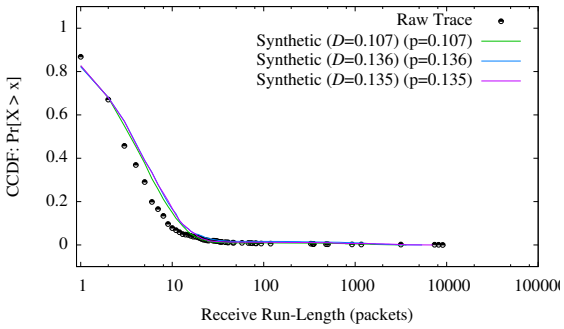
(c) EGM (loss)



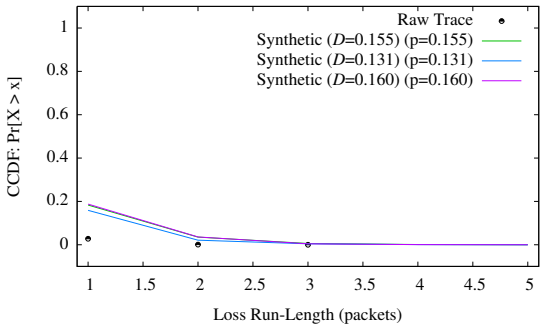
(d) EGM (receive)



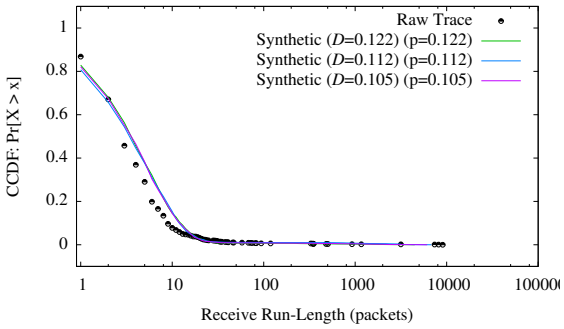
(e) 2HMM (loss)



(f) 2HMM (receive)



(g) 3HMM (loss)



(h) 3HMM (receive)

Figure 5.8: Example CCDF Distributions ("bursty" loss) (same trace as Figure 5.13d)

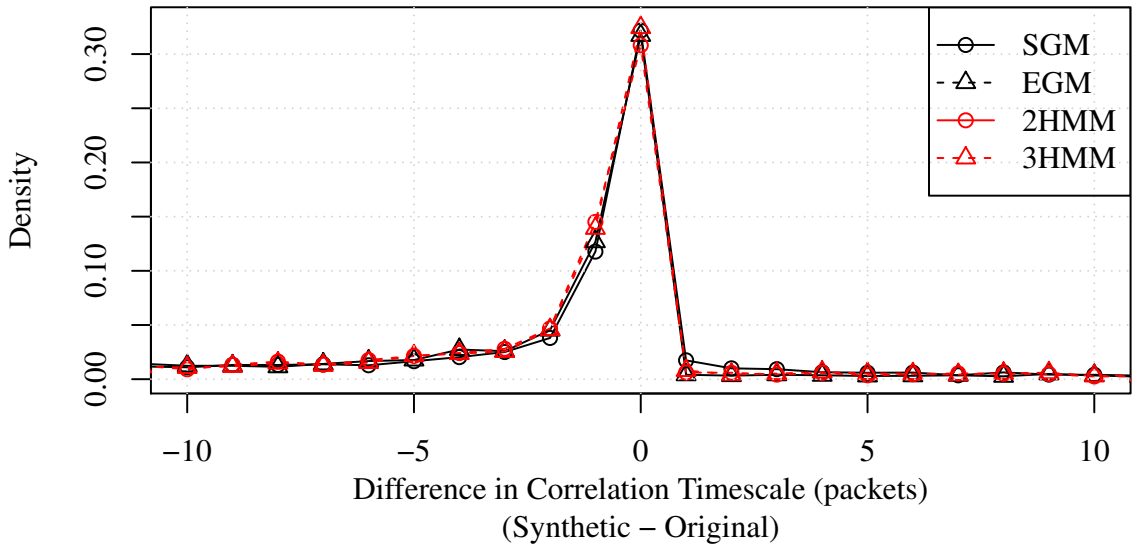
shows that the HMMs do capture the receive run-length distributions better when there are both "on/off" and loss-free periods, although they are not as good at capturing the loss run-lengths, underestimating the number of single packet losses. As Section 5.3.4 will discuss in more detail (by looking at the actual output of the models and comparing with the original data) it is clear that although the SGM and EGM capture the loss run-length distributions better, they do not actually generate similar traces. However, from examining the CCDFs alone, it is not clear why these differences exist between the models, implying that the run-length distributions alone (and measures of distance between such distributions, like the K-S distance) are insufficient to characterise the loss patterns in the trace.

This section has shown that the Gilbert models are better at capturing the loss run-length distributions, while the HMMs are better at capturing the receive run-length distributions. However, more importantly, it shows that the run-length distributions themselves are not a good indicator of loss behaviour within traces, since they cannot capture how well models capture the on/off periods within the trace (a feature that is very important to understand). Using the K-S D distance metric shows whether one model or another is "closer" to the original run-length distribution, but using the K-S test to actually classify a match in the distributions is not so useful. This is a reminder of the dangers of relying on a single metric, without reference to the data being compared, or the context.

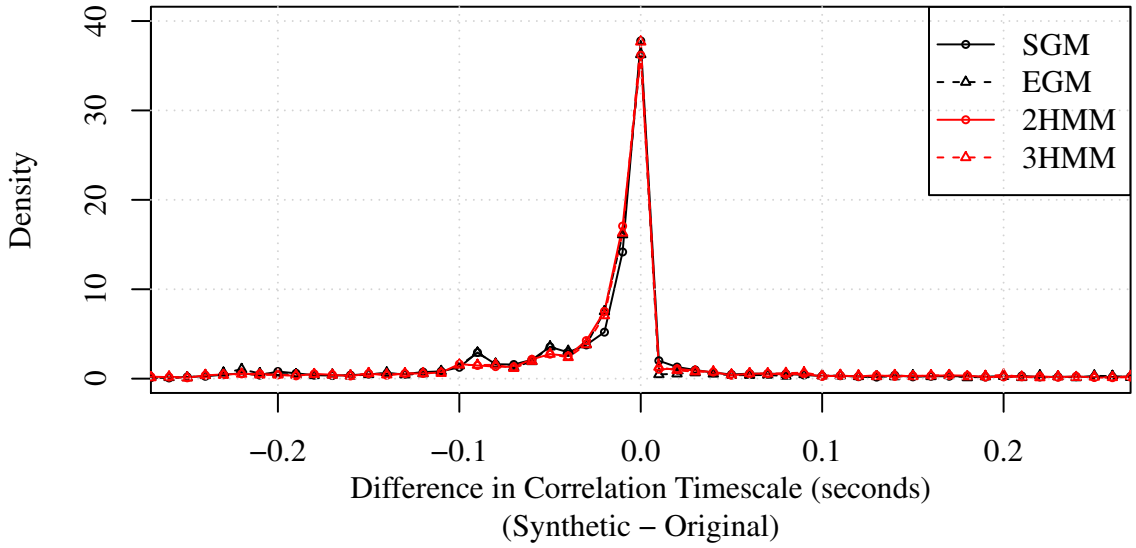
5.3.3 Loss Correlation

This section looks at how well the models capture the features of correlation in packet loss within the traces, focusing on the correlation timescales of the binary loss observations Z_i, as discussed in Section 5.2.2. Looking at the correlation timescales (c_{raw} and c_{synth}) shows how the different models capture the degree of correlation in the packet loss events (i.e., how dependent the probability of packet loss in the time-series is to losses earlier in the time-series).

Figure 5.9 shows the distribution of differences in correlation timescales Δ_c, across all traces, for each of the models. Here, positive values represent cases where the correlation timescale in the synthetic trace is larger than that in the original trace, while negative values show cases where the correlation timescale in the synthetic trace is smaller than in the raw data. This distinction is important, since the distribution is asymmetric. The left-tail of negative values contains cases where the models are not accurately capturing the size of the



(a) Δ_c in packets



(b) Δ_c in seconds

Figure 5.9: Distributions of Differences in Correlation Timescales (Δ_c). Synthetic sequences are generated from each trace, and Δ_c from each of these is calculated.

correlation timescale (i.e., the original data shows more temporal dependence between losses than the models capture).

The strong peak around $\Delta_c = 0$ represents the large number of traces for which there is no difference in the correlation timescales, meaning that the models accurately capture the extent of correlation in these traces. Since many of the traces have low levels of loss, and appear not to be subject to the bursty loss conditions associated with congestive loss, it might be that the traces where $\Delta_c = 0$ also have $c_{raw} = 1$ (i.e., the raw data is uncorrelated). To check this, Figures 5.10 and 5.11 show heatmaps of the original correlation timescale c_{raw} and correlation timescale difference Δ_c, with the intensity of the colour representing the frequency of each combination occurring. The clustering around $(c_{raw} = 1, \Delta_c = 0)$ shows a high number of traces where the models capture low degrees of correlation in the original data. The much brighter colour in the $(c_{raw} = 1, \Delta_c = 0)$ bin shows the extremely high proportion of traces in this category, implying that the strong peaks around $\Delta_c = 0$ in Figure 5.9 show the models capturing the *lack of correlation* in the original data.

The diagonal line present in all the sub-plots of Figures 5.10 and 5.11 corresponds to the left tail of the distributions in Figure 5.9, showing that there are traces for which the extent of correlation (the correlation timescale) is not captured by the models. Figure 5.9a shows that in many cases, $\Delta_c = -1$ packets (i.e., the original trace showed correlation up to a lag of h packets, while in the model-generated trace it was up to $h - 1$). Figure 5.10 shows that these cases mostly occur when c_{raw} is low.

These figures suggest that there is little difference in the performance of the different models, with all models performing well when there is little correlation in the original sequence, and tending to not capture the extent of correlation when it exists. Figure 5.10 shows that the HMMs have less variation in the Δ_c values they generate, giving more consistent performance, although still not capturing the extent of correlation in the original traces.

Figure 5.12 gives further insight, showing that the peak in Δ_c comes from traces with 15 losses or fewer (Figure 5.12a also showing a strong peak around 0), with the left tail coming from traces with higher loss (the distribution of Δ_c also showing a strong left tail). The rationale for using "15 or fewer losses" to distinguish lossy traces will be discussed in Section 5.3.4. Note that the HMMs show smaller Δ_c values for the higher loss traces, indicating that the correlation is being captured somewhat better by these models, although only slightly.

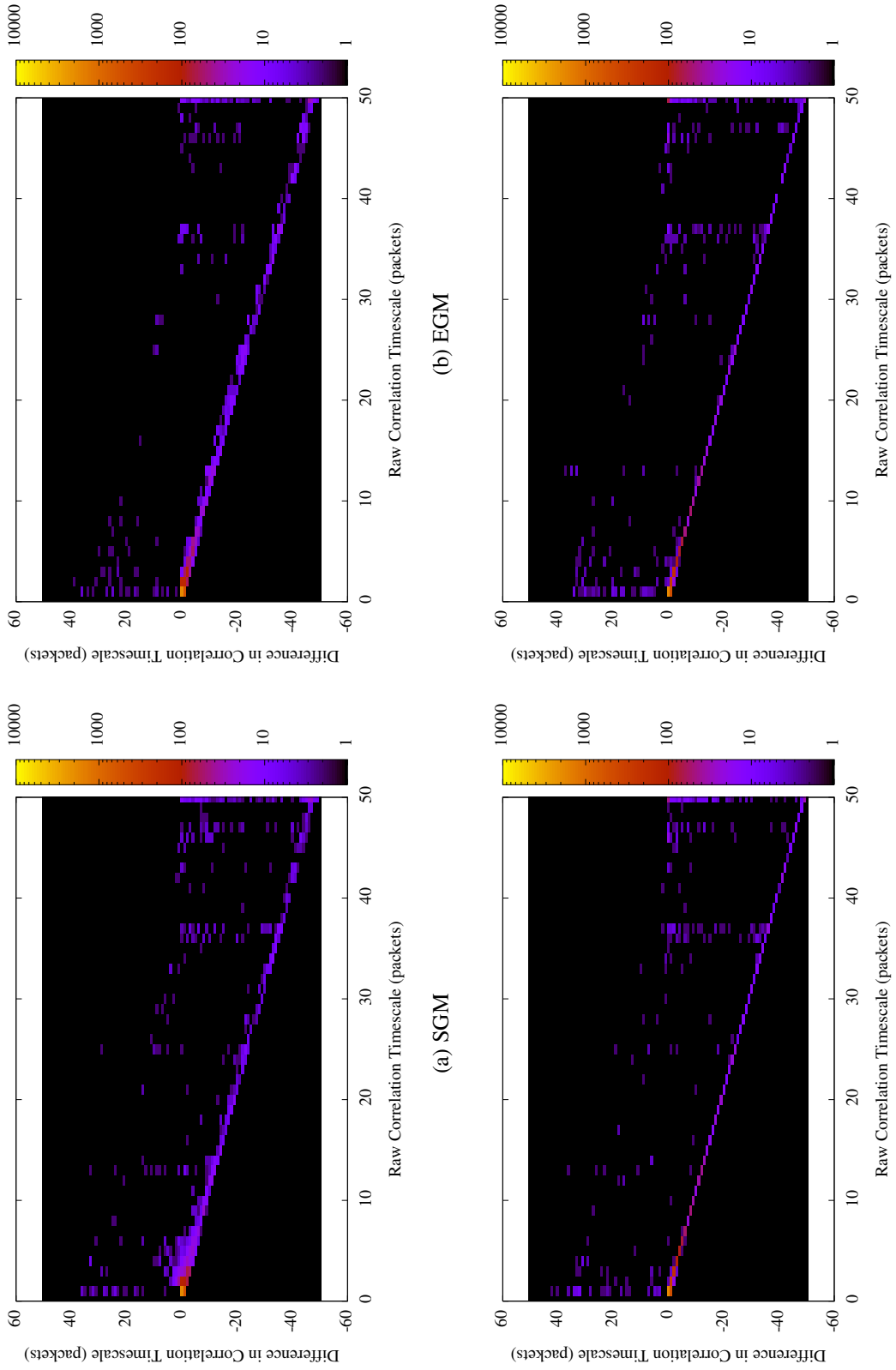
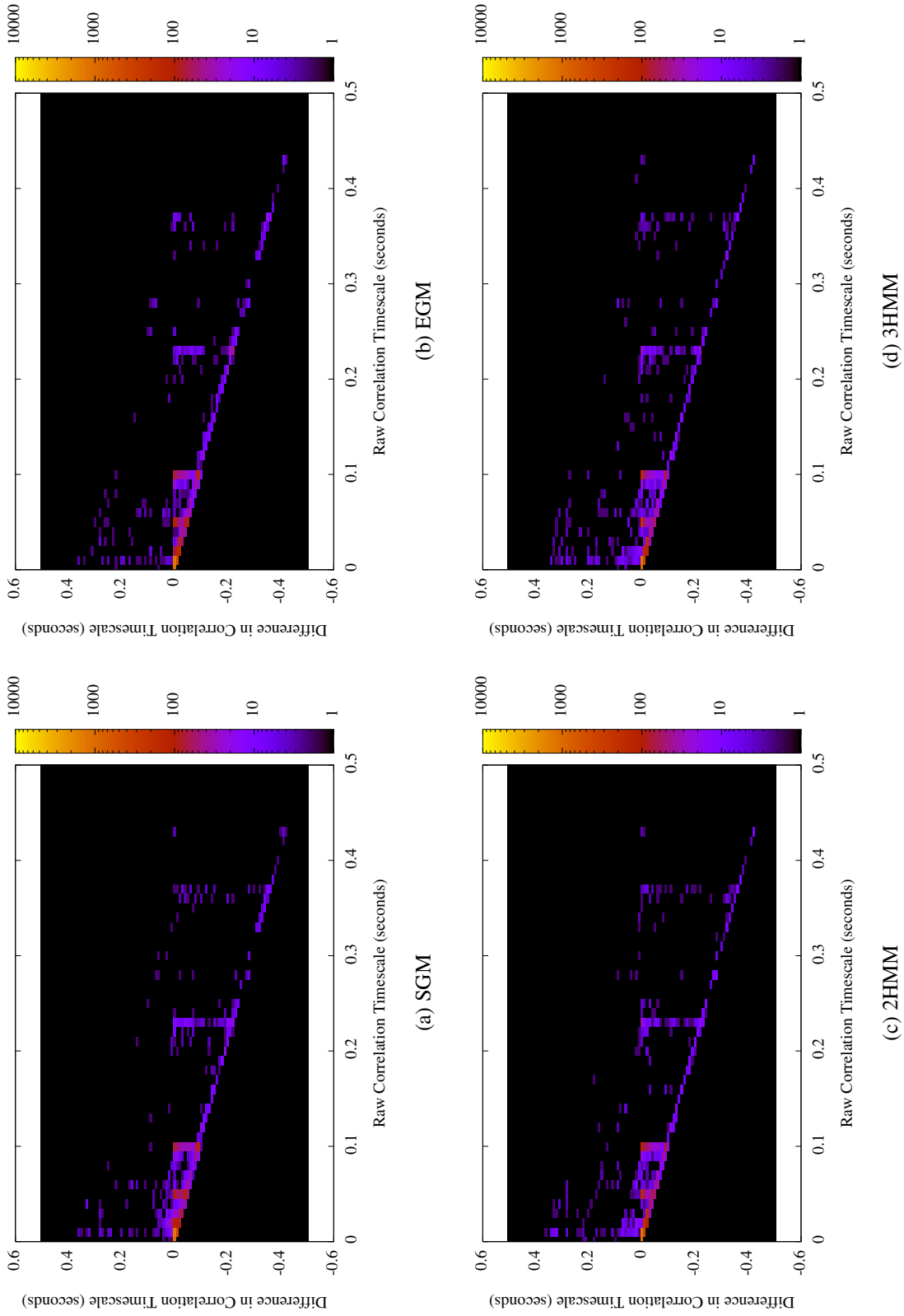


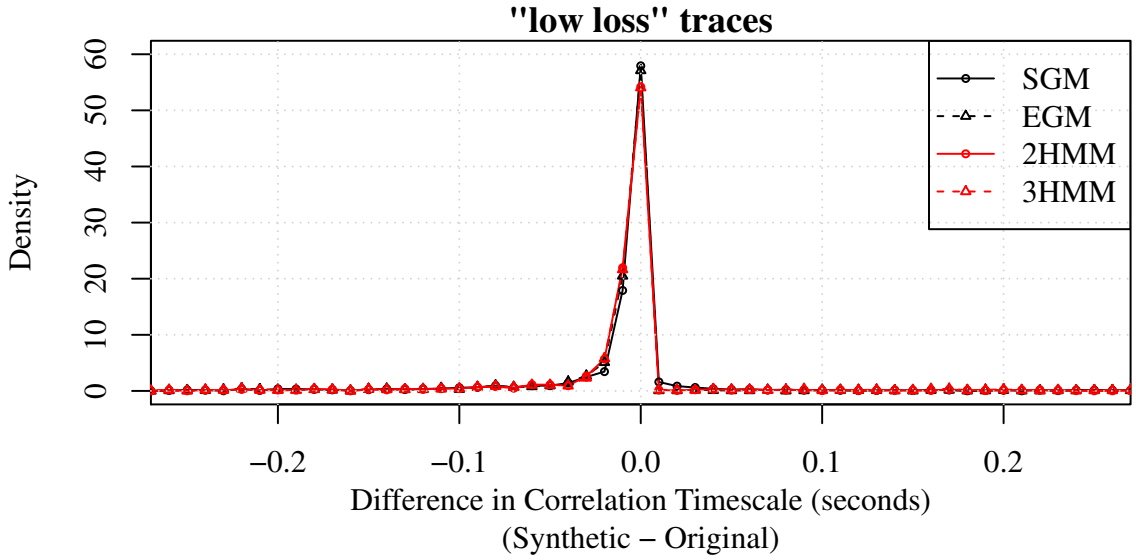
Figure 5.10: Original Correlation Timescales vs. Differences in Correlation Timescales (packets)



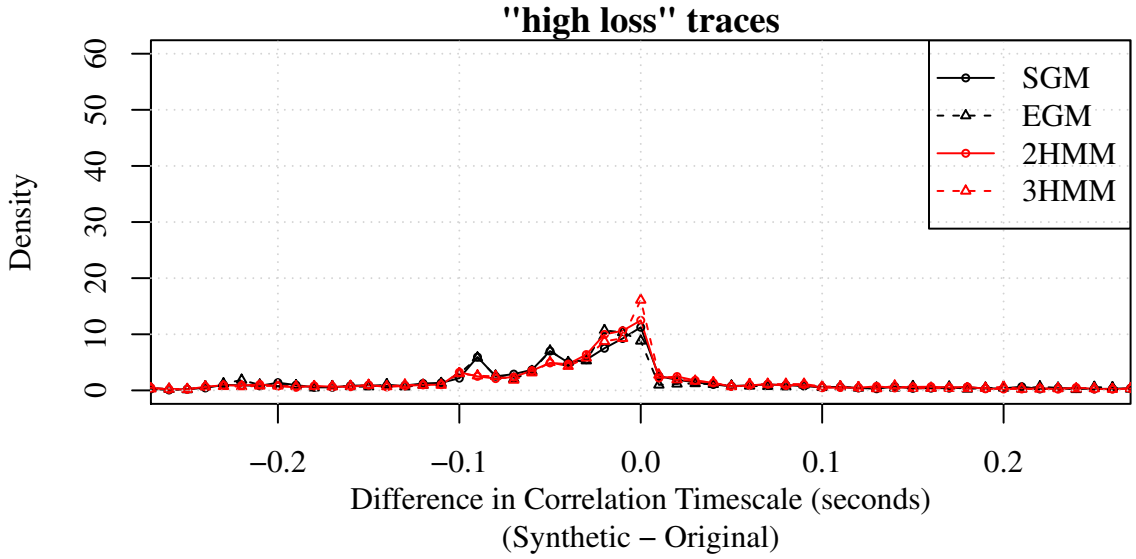
(a) SGM (b) EGM

(c) 2HMM (d) 3HMM

Figure 5.11: Original Correlation Timescales vs. Differences in Correlation Timescales (seconds)



(a) Traces with $\leq$ 15 losses



(b) Traces with $>$ 15 losses

Figure 5.12: Differences in Correlation Timescales for Traces with Low/High Loss

These results show that the correlation timescale in many traces is captured well (particularly when loss rates are low), but that there are a number of traces where the synthetic sequences generated by the models do not exhibit correlation at the same timescales. It can be argued that the more "interesting" traces are those showing cases of higher loss in the network, since these are when FEC performance and video quality are most likely to be degraded. On this basis, the poorer performance of the models in capturing correlation in lossy traces is disappointing.

5.3.4 Visualising Model Fit

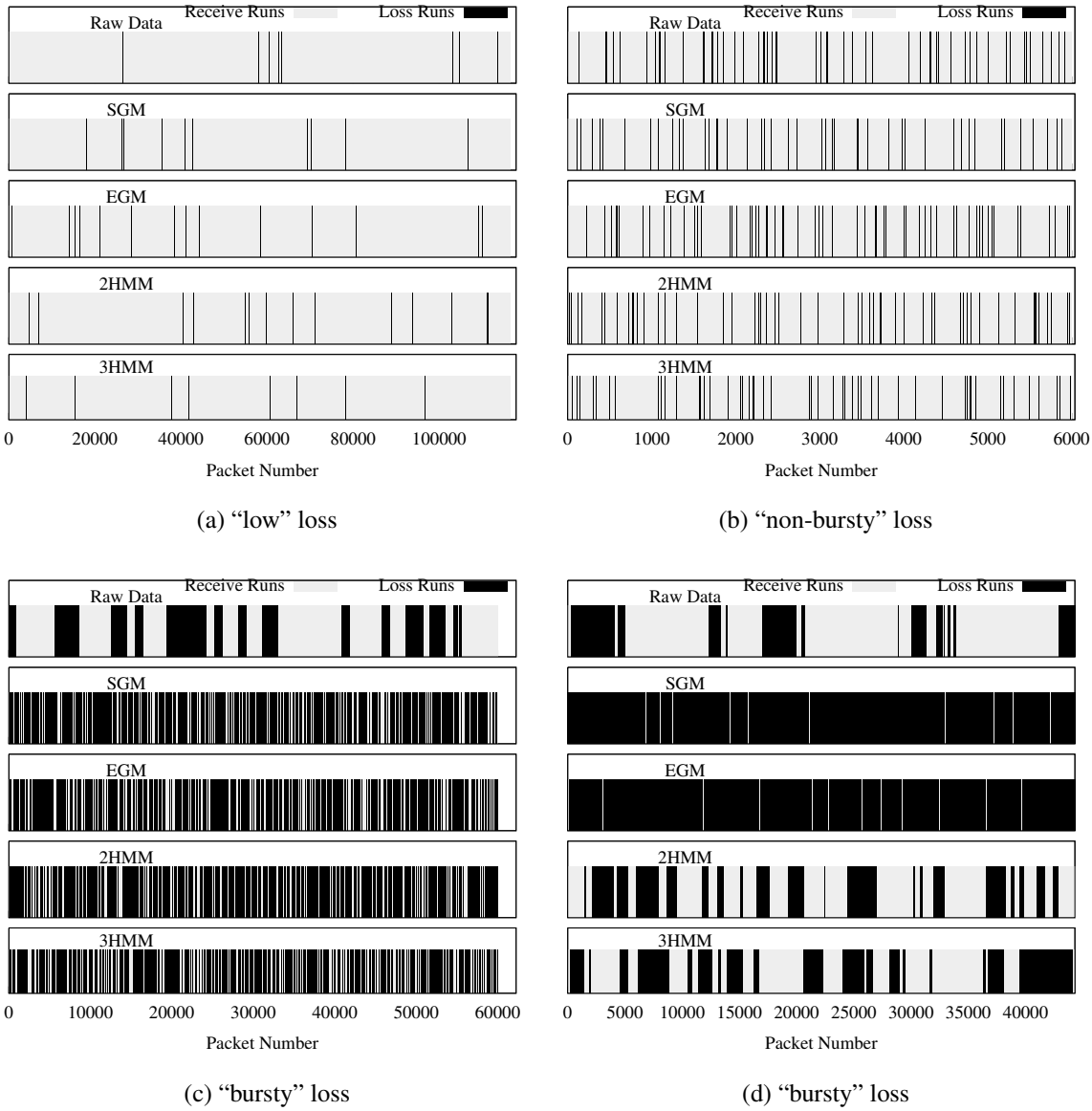
The preceding sections showed aggregate results of performance metrics across all traces (or in the case of K-S distance metric, those traces with a sufficient number of run-lengths), representing each synthetic sequence as a single point. However, as already discussed, it is also beneficial to examine in detail (i.e., visualise) the synthetic packet loss sequences generated by the models, alongside the corresponding original sequence. Choosing a subset of the traces to show as examples requires some division into categories. This section discusses how these categories were chosen, before actually examining the example traces in detail.

Initial manual examination of the loss traces showed they can be divided into three broad groups, those with: i) zero or "low" packet loss; ii) "non-bursty" loss; or iii) "bursty" loss. Separating traces with low levels of loss allows further analysis to be more meaningful, focusing on bursty or non-bursty loss patterns, which does not make sense when there are only sporadic packet losses. For the traces visually identified to have few loss events, over 90% had 15 or fewer lost packets, so the threshold was set at ≤ 15 losses. Among the traces with higher loss, the distinction between bursty and non-bursty loss was made by examining the sequence to see whether the loss is spread out in the trace (as in the top panel of Figure 5.13b), or confined in bursts separated by longer runs of received packets (as in the top panel of Figure 5.13c). Table 5.2 presents the number of traces within each of these categories.

This section looks in more depth at the output of the models, and how they compare to the original data. Figure 5.13 shows examples of original traces alongside synthetic sequences generated by each of the models. These figures show the Z_i sequences, with the x-axis representing packet number (i.e., time), with grey and black regions representing receive and loss run-lengths, respectively. The top panel of each sub-figure shows the original trace, while the four below show a synthetic trace generated by each model.

Loss Type	Number of Traces	Percentage of Traces
no losses (zero loss)	1679	44.1%
1 — 15 losses (“low” loss)	1211	31.8%
$>$ 15 losses (“non-bursty” loss)	486	12.8%
$>$ 15 losses (“bursty” loss)	433	11.4%

Table 5.2: Percentage of Traces in Loss Categories



(a) “low” loss

(b) “non-bursty” loss

(c) “bursty” loss

(d) “bursty” loss

Figure 5.13: Example Loss Traces and Synthetic Sequences from SGM, EGM and HMMs

Figure 5.13a shows a trace with very few losses; here, each of the models generate sequences with similarly low loss rates. Figure 5.13b shows a trace with more frequent losses that are fairly evenly spaced. Again, each of the models generate synthetic trace data that appear similar. In contrast, Figure 5.13c shows a trace with a similar mean loss rate to Figure 5.13b, but where the losses occur in bursts. Note that for this trace, none of the models appear capable of capturing the correlated loss patterns, generating completely different sequences with frequently occurring loss. Figure 5.13d shows another trace with correlated packet loss, but where the loss rate is higher, and where the HMMs perform differently than in Figure 5.13c. The SGM and EGM perform similarly to before, generating sequences with frequent losses throughout (and failing to capture the correlation pattern). The HMMs do perform a little better, generating some longer loss-free receive run-lengths as in the original trace, although still not completely capturing the clearly delineated "lossy" and "non-lossy" regions that are immediately visible in the original trace.

The important conclusion from these findings is that there are a non-trivial number of traces that exhibit loss behaviour that cannot be captured by the models evaluated here. Although there are a large number of traces where the models perform well (i.e., when loss rates are low, and conditions are "good"), the most challenging cases, when the loss processes are bursty, are not well modelled. In particular, the Simple Gilbert Model (which has formed the basis of the experimental evaluations of a number of systems) has been shown to be insufficient at capturing the bursty and correlated loss behaviour seen in the measured packet loss.

5.4 Testing Goodness-of-Fit using Parametric Bootstrap

The previous sections describe how the models perform, in terms of relevant performance metrics, and by showing examples of the sequences generated by each of the models. However, to more objectively test the accuracy of the models, a goodness-of-fit test that can be applied over the whole dataset and all the models is necessary. The results of this test should also be visualised together, for ease of comparison between different models. In this section, I present such a goodness-of-fit test, which involves generating a large number of synthetic sequences from the models, and comparing a range of statistics from the synthetic sequences to those from the original trace. Since this technique generates new sequences using the pa-

rameterised models, I refer to it as *parametric bootstrap*, in contrast to traditional bootstrap techniques that involve resampling from within the existing data [36].

5.4.1 Comparing Statistics using Parametric Bootstrap

To test the goodness-of-fit of a model, model parameters are estimated from each trace, using the process described in Section 5.1. Then, these parameters are used to simulate synthetic sequences, which are compared to the original data to assess goodness-of-fit. 1000 synthetic sequences per trace are generated from each model, and a set of statistics is calculated for each sequence. For each of these statistics S_i, the values obtained from calculating S_i on each of the synthetic sequences are then used to produce a distribution, S_i^{synth}, which is compared to S_i^{raw} (the value of S_i obtained from the raw data). This is a similar approach to [55], which compared the "curvature" of distributions when testing for long-tailed behaviour in Internet traffic.

To assess the model's goodness-of-fit, this procedure tests the null hypothesis that the observed value of S_i^{raw} is a typical draw from the distribution S_i^{synth}. If the null hypothesis is not rejected, then 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_i). This hypothesis is tested by calculating a central 95% confidence interval and checking if S_i^{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% [21]. Although setting 5% is a typical choice for statistical hypothesis testing, a larger significance level that leads to a narrower interval does not alter the results considerably. An example S_i^{synth} distribution, with S_i^{raw} and percentiles is shown in Figure 5.14. By applying this process for a range of statistics of interest, the performance of each model on a given trace can be obtained. Then, repeating this process for each of the models provides a way to numerically compare the performance of the models against each other; for the same trace, the performance of each model can be compared for each "statistic of interest".

The statistics of interest for this work include those already discussed in Section 5.2.1, as well as some others:

- Mean loss rate, $\bar{Z}$;
- Correlation timescale, c;

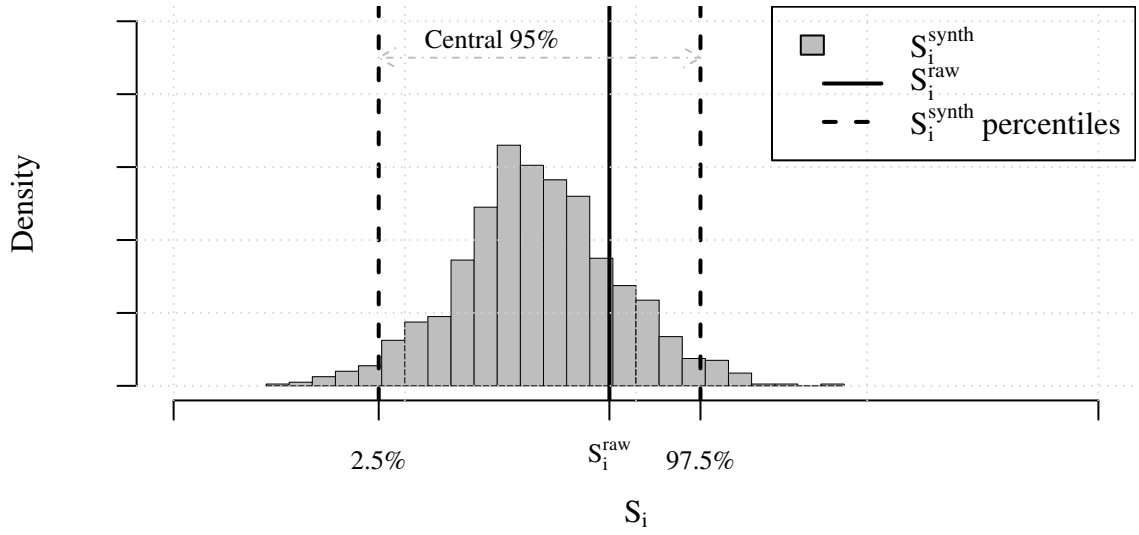


Figure 5.14: S_i^{synth} Distribution, with S_i^{raw} and Central 95%

- 5^{th}, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the receive run-length distribution;
- Mean, median, and max loss run-lengths;
- 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, and 10^5).

These include percentiles of the receive run-length distributions, and mean, median and max loss run-lengths, to assess whether the loss patterns of the raw data are matched by the models. The statistics 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, the number of receive run-lengths in each order of magnitude are counted.

5.4.2 Parametric Bootstrap Results

This section discusses the results of applying the parametric bootstrap technique, to validate through simulation how similar the models are to the original data. As discussed in Section 5.4.1, this involves, for each of the models, and for each particular trace, simulating a large number of synthetic sequences from that model using the parameters estimated from the trace. By calculating a range of statistics on each of the synthetic sequences, and comparing

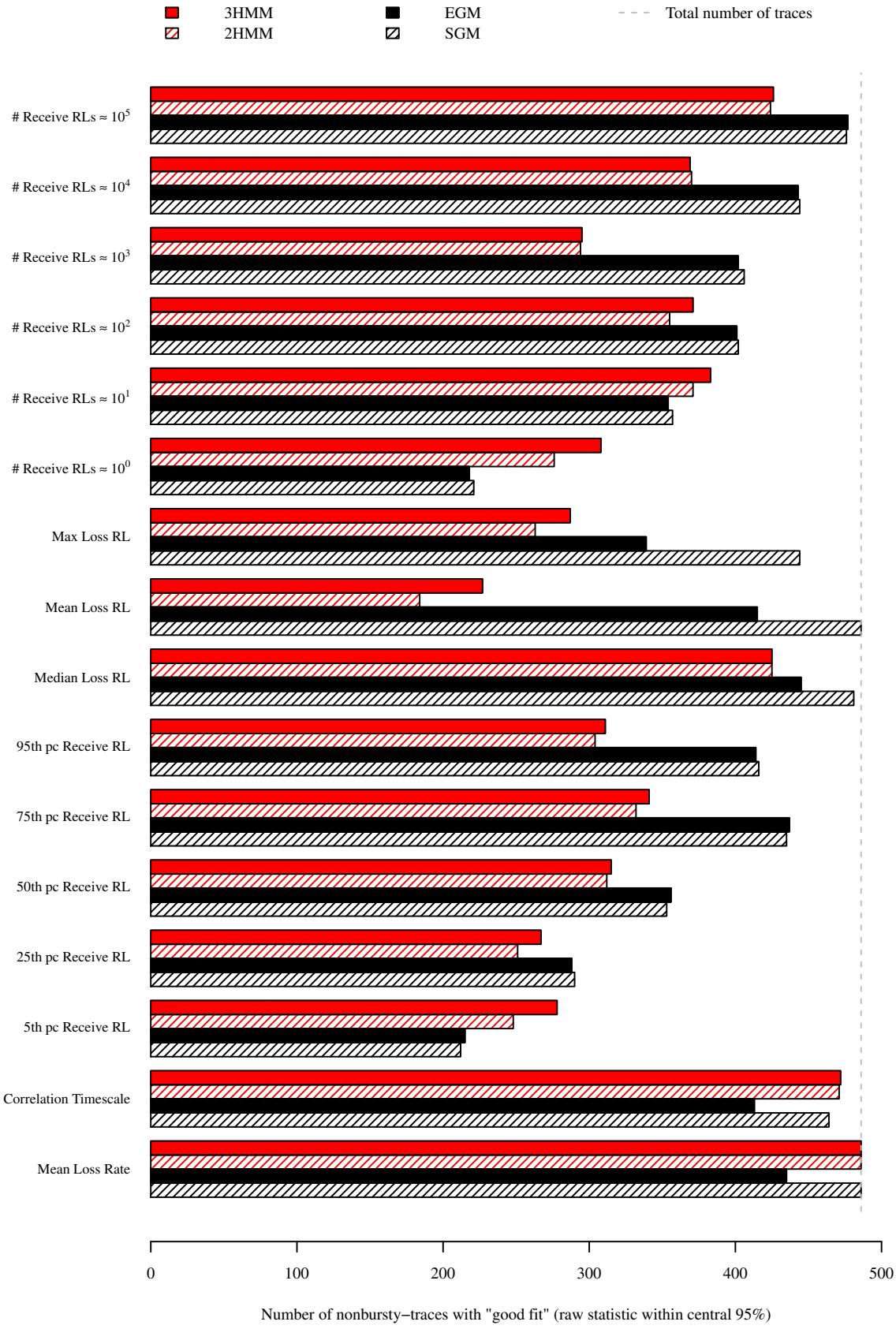


Figure 5.15: Parametric Bootstrap Results (“non-bursty” traces)

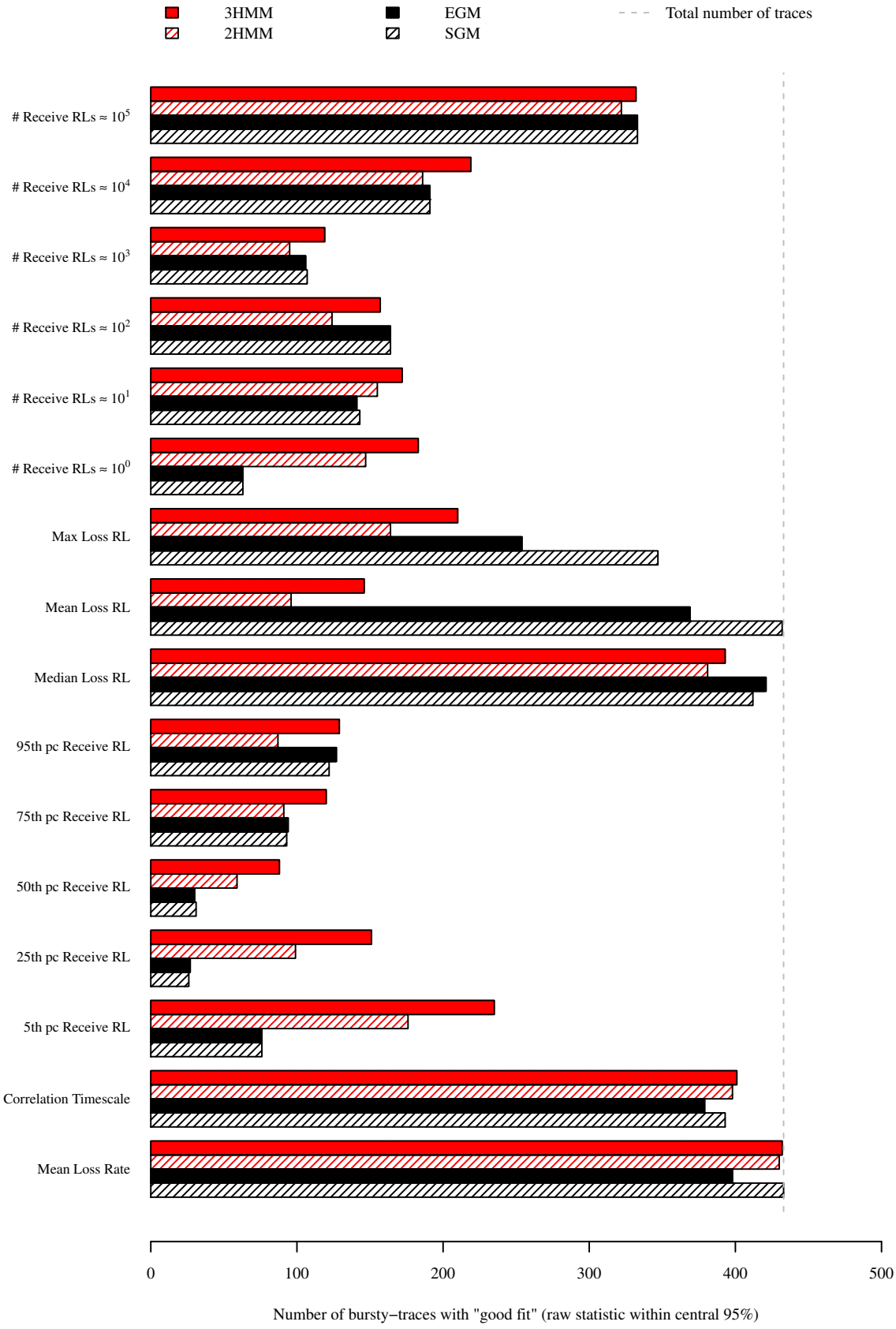


Figure 5.16: Parametric Bootstrap Results (“bursty” traces)

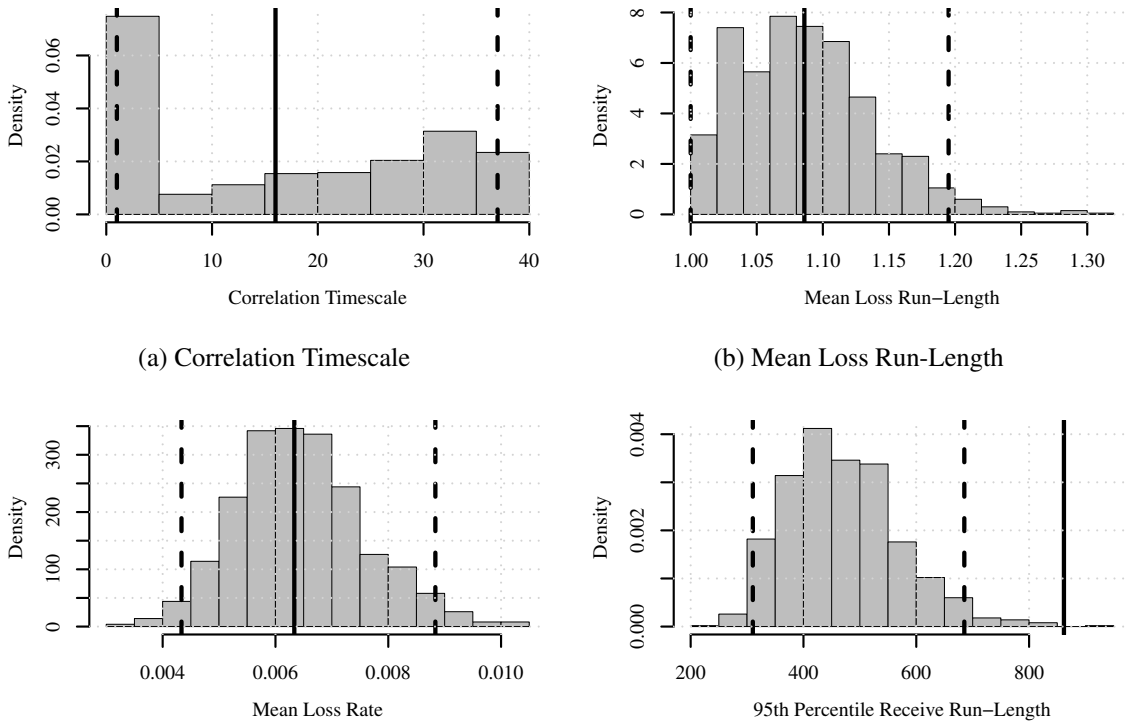
these to the same statistics calculated from the raw data, the "goodness-of-fit" of the model can be tested.

The results of Section 5.3.4 showed that the HMMs had better performance for traces classified as showing "bursty" packet loss; therefore, the parametric bootstrap is applied separately to the "non-bursty" and "bursty" traces (as described in Section 5.3.4, these are traces that show more than 15 losses). For each of the trace groups, the parametric bootstrap is applied to each trace, calculating 1000 synthetic sequences using the model parameters estimated from the trace.

Goodness-of-fit results from applying the parametric bootstrap technique to the SGM, EGM, 2HMM, and 3HMM models are shown in Figures 5.15 ("non-bursty" traces) and 5.16 ("bursty" traces). These figures show, for each statistic S_i (y-axis), the number of traces where the model had "good fit", in terms of S_i. Visually, longer bars mean that the model fits more traces. These figures show that all models appear to capture the mean loss rate, for both the "bursty" and "non-bursty" traces (as seen earlier in Section 5.3.1).

The correlation timescale metric also seems to be well-captured by all of the models, which is surprising since the Gilbert models do not really aim to capture correlation in the loss process. Closer examination of the correlation timescale bootstrap results explains the higher-than-expected fraction of traces identified as "well-modelled". As described in Section 5.4.1, the goodness-of-fit test assessing model fit tests whether S_i^{raw} falls within the central 95% of S_i^{synth}. However, this test relies on the S_i^{synth} distribution being *unimodal* (i.e., having its mass grouped together into one mode). The distribution of S_i^{synth} correlation timescale appears to be multimodal, with one mode around 0, and another at higher values (closer to 50). Examination of the S_i^{synth} distribution for the other metrics presented in Section 5.4.1 suggests that correlation timescale is the only one of the S_i statistics to show a multimodal distribution (see Figure 5.17 for a representative example; observe that only the correlation timescale metric is multimodal). Therefore, in this dissertation, the correlation timescale metric is excluded from further discussion of parametric bootstrap results. Note that future work might design a more sophisticated goodness-of-fit test, accounting for the multimodality of correlation timescale (e.g., using a similar approach to [66]), but this is beyond the scope of this work.

In terms of the receive run-lengths, both the SGM, EGM, and HMMs perform poorly for the "bursty" traces, and a little better for the "non-bursty" traces. Recall in the example



(a) Correlation Timescale

(b) Mean Loss Run-Length

(c) Mean Loss Rate

(d) 95^{th} Percentile Receive Run-Length

Figure 5.17: Examples of S_i^{synth} Distributions

in Figure 5.13 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 HMMs are slightly better for short receive run-lengths in the "bursty" traces. Since the HMMs aim to capture the changes in loss states, they perform slightly better in "bursty" traces, but this extra complexity does not help when the loss pattern is not bursty.

The patterns in the loss run-lengths 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 loss run-length for most traces. The HMMs capture the mean loss run-lengths 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 the loss run-length statistics are well-captured, this does not mean the trace is accurately modelled (since the 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 Figure 5.13.

To summarise, the parametric bootstrap results show that although the SGM, EGM, and HMMs perform adequately on certain types of traces and in terms of some metrics, the important point is that they do not accurately capture the loss patterns of the most bursty traces (as demonstrated in Figure 5.13c). Although bursty packet loss does not occur on every trace, over 10% of traces were classified as "bursty", and the implications for the performance of streaming video are too important to be ignored (e.g., bursty packet loss severely degrades the performance of FEC schemes [63]).

5.5 Understanding Model Performance

As Sections 5.3 and 5.4 showed, there are differences in performance between the different models, and for the same model when applied to different traces. This section will discuss the reasons for these differences in performance, and suggest improvements to allow accurate modelling of all the cases identified in Section 5.3.4.

The results presented in Section 5.3.4 explain the differences in results seen in Section 5.3.2 for the K-S distance between run-length distributions. Since the raw traces do not always have losses interspersed throughout (traces instead often show on/off periods of bursty loss and loss-free periods), the range of receive run-lengths within a trace can range across several orders of magnitude, from below 10 to above 10000. Since the Gilbert models do not capture the on/off behaviour of the loss process, they cannot generate long receive runs when the loss probability is high, since this probability dictates that the receive run will be cut short by a loss. However, this also explains why Gilbert models initially *appear* to perform better in capturing loss run-length distributions. During the bursty "on" loss periods, there are frequent oscillations between loss and receive runs, giving rise to loss run-length distributions similar to those generated by the Gilbert models. This can be viewed as a case of the traces capturing the models, rather than the models capturing the traces.

As discussed in Section 5.3.2, although the Gilbert models capture the loss run-length distributions more accurately (the EGM is particularly optimised to do this), they do not capture the overall loss patterns in the trace, since they do not capture the on/off periods of bursty loss and low loss, as shown in Figures 5.13c and 5.13d. Unlike the Gilbert models, the HMMs try to capture the transitions between the "hidden" states in loss behaviour. By doing this, they are able to give better performance in capturing receive run-length distributions

than the Gilbert models, although this seems to cause the loss run-lengths to be less well-captured, with the number of single packet losses being underestimated by the HMMs (as shown in Figure 5.8 on page 102).

In terms of the correlation results, there is not such a clear difference in the performance of the Gilbert models and HMMs. When loss rates are low, and the losses are uncorrelated, all models capture this behaviour well, as demonstrated by the strong peaks in Figures 5.9a and 5.9b (page 104). However, when there is correlation, this tends not to be captured very well by the models, as illustrated in the left tails in Figures 5.9a and 5.9b, and the diagonal line tailing off in Figures 5.10 and 5.11 (pages 106 and 107, respectively).

The reason for the poor performance of the Gilbert models when the loss is correlated is because such models do not consider correlation in the input, since the next "state" (i.e., whether the next packet will be received or lost) only depends on the current state, regardless of what is happening in the network. In other words, the Gilbert models see a received packet as a renewal, or "reset", where the network is assumed to be in a "good" state, and therefore cannot distinguish the short receive runs between loss bursts from longer periods of received packets where the network actually is in a good state. Section 5.3.4 illustrates this clearly, showing that the SGM and EGM *always* produce a pattern of "frequent" losses (according to the specified loss rate).

The HMMs capture the correlation better in some cases (e.g., in Figure 5.13d), since the outcome of the next packet in the model depends not on whether the current packet was received or lost, but on which hidden state the model is in (i.e., the underlying state of the network). However, within each state, the choice of whether each packet is lost is made independently of each other, meaning that correlations *within* hidden states are not captured.

The different behaviour of the HMMs observed in Figures 5.13c and 5.13d show that although there are changes in "state" (i.e., oscillations between periods of bursty loss and infrequent loss), the models do not always recognise these. One possible reason for this is that the HMM parameter estimation algorithm fails to converge to the solution with maximum likelihood (meaning that the model is poorly trained). Table 5.3 shows the number of traces for which HMM parameter estimation converged, or failed to converge, in both the "bursty" and "non-bursty" trace groups. The low rates of convergence may partly explain why the HMMs perform more poorly than expected. Another possible reason for the HMMs not capturing the oscillations between different states of loss is that the distinction between

Model	Trace Group	Converged	Failed to Converge
2HMM	"bursty"	44 (10.2 %)	389 (89.9 %)
	"non-bursty"	56 (11.5 %)	430 (88.5 %)
3HMM	"bursty"	70 (16.2 %)	363 (83.8 %)
	"non-bursty"	51 (10.5 %)	435 (89.5 %)

Table 5.3: Convergence Rates in HMM Estimation

different states is not clear. Although looking at the Z_i sequences in Figure 5.13 makes this clearer, since the HMMs are trained with only the binary loss sequence Z_i, the patterns obvious in the plots may not be captured. A potential solution to this problem is to pre-process the traces to identify bursty periods; this will be discussed further in Section 5.7.

This section has discussed the results of Section 5.3 and explained why the models perform as they do. The results show that where loss rates are low or uncorrelated (i.e., the majority of cases), all models perform well. However, for the smaller (but still significant) number of traces showing bursty loss behaviour (and oscillations between different loss states), the Gilbert models cannot capture the loss patterns. Although the HMMs do perform better in some cases, they are still limited, and need to be enhanced to completely model the network characteristics.

5.6 Alternative Models

This section describes some alternative approaches to modelling packet loss, including variants and modifications to the Gilbert models, proposals for modelling errors in wireless networks, and techniques that extend the HMMs.

5.6.1 Variants of Gilbert Models

An early variant of the Gilbert model was proposed by Fritchman [69], a "partitioned" model with separate states for loss and receive run-lengths. This can be viewed as a more general version of the EGM, although as discussed in Section 5.1.1, the state-space of such a model will be very large when the receive run-lengths are long (as is the case for the loss traces modelled in this chapter).

As well as applying the SGM to early measurements of Internet packet loss, Yajnik *et al.* [224] examined the performance of higher-order Markov chains in capturing the measured loss processes, finding that in some traces the estimated order of the Markov chain was as high as 42. This means that to capture such loss processes, a Markov chain model will need a large number of parameters, which will become unwieldy to use in practice.

More recently, the accuracy of Gilbert models has been questioned. Yu *et al.* [226] found that the SGM underestimates FEC performance when compared to exact queueing analysis for a single-multiplexer queuing system, although finding that using more states for the receive run-lengths (i.e., the inverse of the EGM presented in [100]) performs better. Becerra-Yoma *et al.* [14] found that the run-length distributions can be better modelled using a Gamma distribution, and that a mixed Gilbert-Gamma model more accurately models their measurements of packet loss from academic backbone networks.

Haßlinger & Hohlfeld [78, 82] studied the accuracy of Gilbert models in capturing the second-order statistics (i.e., variability) of packet loss patterns over long time scales using active measurements of real-time UDP traffic captured at the backbone of Deutsche Telekom. They observed that by using Hidden Markov Models to implement a Gilbert-Elliott model, they could generate data matching the second-order characteristics of their traces.

5.6.2 Models Proposed for Wireless Loss

Related models have been proposed to model frame errors in wireless networks. These include a four-state Markov model, which includes separate states for "long" and "short" loss and receive run-lengths [135, 227]. In [227], this four-state Markov model is also expressed as a "two-state run-length model", where rather than having probabilities of moving between good and bad states, it is the *state durations* (i.e., the run-lengths themselves) that are modelled.

The Markov-based trace analysis (MTA) [111], and multiple-states MTA [112] techniques proposed by Konrad *et al.* use a "data-preconditioning" approach, aiming to classify a trace of GSM frame errors into sub-traces that are "lossy" or "loss-free", then concatenating and modelling these separately. The MTA approach, as well as other Markov models were evaluated by Ji *et al.* [99, 98] using the same trace as [111], finding that their proposed alternative, the "extended On/Off model" (which contains loss and receive run-lengths with lengths derived from mixtures of geometric distributions) is more suitable. However, al-

though these results are compelling, the use of a single error trace throughout the evaluations, casts a question as to how suitable these approaches are for large-scale use.

Poikonen & Paavola [171, 172] evaluate the performance of the SGM, four-state Markov model and the MTA technique with measurements of error performance in digital video broadcasting for hand-held terminals (DVB-H), finding that the four-state model and MTA perform well. They conclude that the relative simplicity of the four-state model makes it well-suited for simulation of DVB-H loss.

5.6.3 Extending HMMs

This section discusses extensions to the HMMs, to improve their performance on those traces where the two- and three-state HMMs were insufficient. These extensions include adding extra states, relaxing the assumptions of the model, adding extra parameters, and using packet delay information alongside the loss data.

The simplest approach to improving HMM performance might be to extend the HMMs described in Section 5.1.2 with extra states. Adding more states into the model should improve the fit, and more accurately describe the characteristics *of that particular trace*. However, the danger of this approach is that the model will become too specific, which will limit its usefulness to be applied to other situations, as well as disconnecting it from the intuition of how the model states relate to network conditions. The other drawback is the additional computational complexity of using extra states, which increases the time for parameter estimation [175].

Recently, Hidden Semi-Markov Models were applied to modelling packet loss in PlanetLab measurements by Nguyen & Roughan [150, 151], in order to validate estimates of Internet packet loss (rather than for packet loss simulation). This approach extends the HMM modelling technique, considering the packet loss process as an alternating on/off process (i.e., receive runs and loss runs), but allowing the run-lengths themselves to be drawn from non-exponential distributions (i.e., no longer having the Markov property). The *semi-Markov* property means that "state-transitions are Markov, even if the times between transitions are not" [151]. This approach gives promising results in packet loss estimation for the PlanetLab traces.

Another interesting option is to develop a new variant of the HMM, extending it to incorporate loss probabilities depending on whether the previous packet was lost (following the

approach of [108]). This approach is attractive, since it incorporates some extra information about the correlation of losses (no longer assuming that losses within each hidden state are independent). However, the autocorrelation results suggest that for the bursty traces, packet losses are dependent not only on the outcome of previous packet, but also on packets before that in the time-series; therefore, this approach may not be sufficient. More concretely, the congestion state of the network, which has proven to be hard to capture in the models, is more complex than just the fate of the preceding packet. An additional drawback is that applying this technique will require development of new parameter estimation techniques.

A different approach is to follow the approach of other studies using HMMs, and to incorporate both delay and loss information into the model [215, 182]. This is also an attractive option, since it matches well with the philosophy and motivation behind using HMMs (i.e., that the underlying state of the network is being captured by the models). Using delay, a better picture of the congestion state of the network can be seen, since the delay due to queueing is directly related to congestion. This approach is expected to give better performance, due to the far greater insight into network conditions that can be gathered using the delay data. Recall the discussion of loss/delay correlation from Section 4.1.3, and how this can be used to estimate the sources of loss events (e.g., spikes or high variability in delay indicate a congested network, implying that lost packets are due to queue overflows). Incorporating both delay and loss into the model, therefore, can be expected to more accurately show the state of the network, improving model performance.

5.7 Discussion & Summary

The results presented in this chapter show that there are cases where Gilbert models (i.e., the SGM and EGM) perform well, and accurately capture the loss conditions seen in some of the trace data. However, more importantly, there also exist traces where this is not the case. This means that the assumptions of a number of multimedia systems papers, which evaluate system performance using a Gilbert model for packet loss, are not valid when considering residential users.

Applying more complex Hidden Markov Models to the problem allows the loss patterns of some traces to be captured somewhat more accurately, while other traces remain unable to be accurately modelled. The 3HMM shows better performance than the 2HMM, suggesting

that higher-order HMMs might be able to capture the observed loss patterns. However, there is of course some danger here of "over-fitting", introducing models with ever more states, while making these models ever less realistic. For example, in the extreme case, a model might contain a large number of parameters, allowing it to memorise the input trace, and generate identical traces as output. Such a model is, of course, not useful in generalising the packet loss behaviour. An alternative approach to validating the accuracy of the models, which does not suffer from this problem, is to collect a second dataset for validation purposes (or only use a subset of the available data to train the models). In this way, the model parameters trained using one trace can be compared to another for validation, which will identify the over-fitting problems described above.

This presents a choice, whether to continue experimenting with more sophisticated models for loss, or to introduce the additional delay information into the model to gain further resolution into the problem. Using data on the delay experienced by packets, greater insight into the state of the network is available (i.e., since queueing delay is a proxy for queue occupancy throughout the end-to-end path). Using this insight also presents an opportunity to understand the sources of loss (i.e., due to noise or congestion, and to isolate where in the network the losses occur), as in the initial work presented in [60].

Since delay information is already available from the traces presented in Chapter 3, I choose to take this approach. Chapter 6 will discuss how the modelling work presented in this chapter can be extended to include delay, and to enable better modelling of network conditions.

Chapter 6

Improving Model Performance using Packet Delay Data

The preceding chapter evaluated the performance of a number of previously proposed models for Internet packet loss, using the measurements of streaming video-like traffic to residential users presented in Chapter 3. The results of Chapter 5 showed that these previous models were sufficient some of the time, but that a significant fraction of the measured traces showed packet loss patterns that could not be captured by the existing Gilbert models and HMMs.

In this chapter, I introduce a new type of model, which incorporates the extra delay information captured in the measurement data to obtain a more accurate picture of the network conditions. By understanding the network better (e.g., whether loss is due to access link noise, or congestion), the models can more accurately capture the loss process.

This chapter is structured as follows. Section 6.1 gives background on different approaches to incorporating delay and loss data into models. Section 6.2 describes the chosen approach of "pre-classifying" the state of the network, to improve the model performance, and introduces the concept of a two-level model, first identifying network state, and second modelling packet losses within each state. Section 6.3 describes the classification schemes developed in this work. Section 6.4 recalls the packet loss models introduced in Chapter 5, and how these are adapted to work within the new two-level model. Section 6.5 presents results of the new models, showing the improvement over previous approaches. Section 6.6 gives a summary of the chapter.

6.1 Modelling Delay and Loss

In this section, I will discuss existing approaches to modelling packet loss and delay, and the relevance of these approaches to improving the loss models of Chapter 5. In particular, I will look at: 1) techniques for modelling packet loss and delay, to characterise loss and delay patterns (and generate synthetic sequences); and 2) techniques to identify the underlying network state from the loss and delay observations (since this is so important in being able to model what is happening, and how streaming video applications react).

6.1.1 Directly Modelling Loss and Delay with HMMs

Following on directly from the work in Chapter 5, one approach is to extend the HMMs described in Section 5.1.2 to operate on observation sequences including both packet loss and delay. This approach has been used in prior work such as [215, 216], which use HMMs with "discretised" delay values (with a separate value for loss), and [182], which develops a joint variable for loss and delay that contains a discrete part to represent loss (where the variable $= -1$), and a continuous (Gamma-distributed) part to model the delays of received packets (where the variable takes a continuous value > 0).

Both [182] and [215] describe their approach for estimating the model parameters using Expectation-Maximisation techniques (i.e., a similar approach to the HMM learning procedure in Chapter 5). However, these are more complicated than the technique described in Section 5.1.2, since they involve a more complex observation sequence (a joint loss/delay variable, rather than a binary sequence). As such, the estimation procedures are likely to be more time-consuming. In [182], applying the HMM algorithm to a training sequence of 1000 observations took around 15 seconds (using a 600MHz Athlon processor). While it is unclear without implementing the technique what the running times would be on a modern host, it is clear that the estimation time is non-trivial, especially since the traces described in Chapter 3 range from 6000 to 600000 observations.

The direct modelling approaches discussed here will, in a similar fashion to the HMMs described in Chapter 5, identify states within each particular trace being modelled. However, as discussed earlier, using HMMs in this way means that the states are not particularly well-defined (i.e., it is not clear what they represent, in terms of the underlying network conditions), and there is not a "global" sense of what each state represents (e.g., the same

subset of observations might be classified differently in different traces, depending on the rest of the trace). For example, [182] discusses what the states automatically identified by the HMM algorithm (for different numbers of hidden states) might correspond to, in terms of loss and delay. However, between different traces (especially given the range of network behaviour seen in the traces of Chapter 3), these states are likely to have quite different meanings. Therefore, I feel it is more appropriate to seek an approach that more explicitly takes into consideration the conditions on the underlying network in identifying states within the observation sequence.

6.1.2 Identifying Network State using Loss and Delay

A slightly different approach is to not model the observations of loss and delay directly, but rather to understand the network state at the time of observation. Understanding the underlying network state is important for applications, for example in making choices about adapting their performance (e.g., for streaming video applications, adjusting the transmission rate or switching FEC parameters).

Identifying the sources of packet loss has been widely studied in prior work on improving TCP performance over wireless links, where being able to distinguish between non-congestion-related wireless losses and losses due to congestion at the IP-layer is important to maintain acceptable TCP throughput [126]. This work focused on modelling the RTTs measured from *loss pairs* [125] using an HMM, estimating the most probable states for the observation sequences, then examining the distributions of RTTs within each hidden state to identify congestion-based or wireless losses. This idea has also been integrated into TCP congestion control algorithms, to improve TCP throughput performance on wireless networks [11].

Online estimation of path performance was considered in [207], which looks at loss measurements from probe streams sent across the Internet (using Internet2 and Cogent), estimating the parameters of an HMM for packet loss, and predicting loss rates and loss burst distributions for a window into the future. An interesting aspect of this work is that it uses a "layered" model (described in [45] as a *Hierarchical Hidden Markov Model*, HHMM). This involves looking at the loss observations over two timescales (giving two "layers"), with the outer timescales representing windows of time, into which the observations are divided. Within each of these windows, the loss observations are divided into periods corresponding

to their loss rate: "no loss" (0%), "minor loss" (0–0.5%), "tolerable loss" (0.5–1%), "serious loss" (1–5%), "very serious loss" (5–10%), "unacceptable loss" ($> 10\%$). An HMM is trained within each window, and the loss conditions within that window are used to predict performance in the next.

Other work has looked at identifying current network characteristics using observations of loss and predicting loss conditions for the short-term future, with the goal of adapting FEC parameters [56], and other real-time applications [191, 189]. The approach of [191] and [189] is also a hierarchical HMM, with the "inner" states being represented as an SGM, capturing the short-term dynamics of loss, while the outer (hidden) states capture longer-term shifts.

6.1.3 Summary

This section has looked at some of the options for incorporating the available observations of packet delay, to improve the performance of the loss models of Chapter 5. Although the HMMs provided the best performance in Chapter 5, an important drawback is that the states identified by the HMM are determined purely according to the training data, and may not necessarily reflect the underlying network states.

The alternative approach presented in Section 6.1.2, to identify the packet level conditions associated with network conditions (such as congestion in different parts of the network), and use these to pre-classify the states, is more appealing since the states in the data are those of interest. Another advantage of this approach over a purely data-driven model like the HMM is that knowledge of current network state can also be used for other applications, such as adaptive FEC control, or playout buffer tuning.

6.2 Pre-Classifying Network State using Loss/Delay

The Hidden Markov Models used in Chapter 5 work by "learning" the states of the network, and predicting based on conditions in each state. In this section, I will explain the state of the network in the traces using the delay and loss characteristics throughout the trace, and describe how these relate to network conditions (e.g., congestion at different parts of the network).

6.2.1 Network States

As described in Chapter 2, the end-to-end path experienced by streaming video flows consumed by home users traverses a number of different "regions", and can be subject to various conditions. These include the "core" of the Internet, from the media source (which is either located on a content-distribution network, or within the ISP's network), towards the edge of the ISP network where residential customers are aggregated; the "edge" itself (i.e., the customer-facing routers at local exchanges and switching offices, to which customers are connected); and the "access link", which provides connectivity from the customer's premises to the DSL Access Multiplexer (DSLAM) for ADSL, or Cable Modem Termination System (CMTS) for Cable. In this section, I will discuss how network conditions will affect these different regions of the network, and how these are manifested in the end-to-end delay and loss observations.

Within the "core", it can be expected that there are many flows traversing the routers at any given time, with a high degree of statistical multiplexing. Therefore, amongst these flows, the flow containing the streaming video traffic being transmitted to the residential user makes up a relatively small proportion of the overall traffic. If congestion occurs, all flows will experience higher delays on average, and will have an increased probability of loss. However, the correlation between jitter (i.e., delay variation) and packet loss is limited, since the streaming video flow only makes up a small fraction of the overall traffic, and the variation is spread across all these flows [166].

At the ISP "edge", the streaming video flow can be expected to comprise a much larger proportion of the available bandwidth than in the "core", since the link capacities are lower. In this case, there is a lower degree of statistical multiplexing (since fewer flows are multiplexed at ISP edges than in the core). As in "core" congestion, average delays and loss rates will be higher in periods of congestion; however, there is likely to be a more noticeable effect on jitter, and a clearer correlation between periods of increased jitter and congestion-induced packet loss.

On the "access link", the delay is likely to be relatively stable, although some variation might be present if underlying error correction techniques are present (e.g., ADSL interleaving or FEC), which can cause delay variation during periods where losses are being corrected. This region of the network is also the most likely to see packet loss due to corruptions and checksum failures, since the physical infrastructure in these "last-mile" links (twisted-pair

telephone cable in ADSL, and coaxial cable for Cable) is typically older and harder to maintain [38, 16]. In contrast, studies of packet loss within backbone networks have shown that packet loss events are extremely rare. For example, in [95], measured loss rates between Amherst, Massachusetts, and Burlinghame, California (across an over-provisioned optical backbone) were rarely more than 0.1%. When isolated losses occur with no change in delay or jitter conditions, these packet losses are likely due to corruptions on the "access link", since they occur independently from congestion-induced losses elsewhere in the network.

This section has described the loss, delay, and jitter characteristics that can be expected from impairments in different regions of the end-to-end path between the video sender and receiver. Using these features (that can be measured at the receiver), a scheme to classify network state can be defined. Such a scheme will be outlined in the following sections.

6.2.2 Two-Level Hierarchical Models

Using the results of pre-classification, a two-level hierarchical model can be constructed, similar to that in [207]. In this two-level model, there is an "outer" level corresponding to the underlying network state (identified by a *classifier*), and an "inner" level determining whether packets are lost or received within these states (represented by a *packet loss model*).

So, using the network states presented in Section 6.2.1, the outer states of this hierarchical model will correspond to the following network conditions (as discussed earlier in Section 2.2):

1) *uncongested / access link noise*, where issues with the physical layer cause IP-layer packet loss;

2) *edge congestion*, where queue overflows at the ISP edge (i.e., DSLAM or CMTS) cause packet loss, and;

3) *core congestion*, where routers within the "core" networks overflow.

In terms of the physical effects seen in the data, access link noise (1) will cause low levels of uncongested loss, regardless of the levels of delay. Edge congestion (2) will cause higher levels of loss, which will be associated with higher delay (since the building edge queues will noticeably increase queueing delay DQ). Core congestion (3) will cause higher loss,

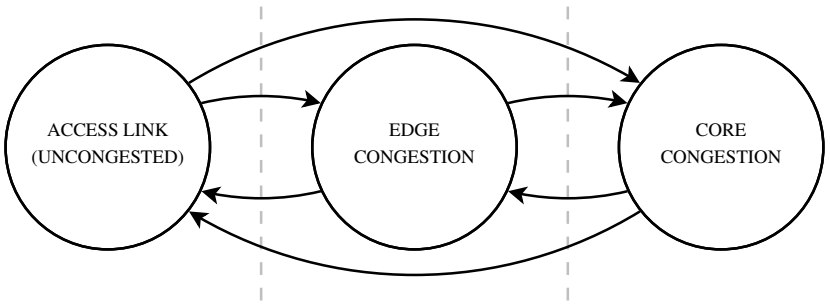


Figure 6.1: Outer states of a hierarchical model, with states for different network conditions

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 outer states, and the transitions between them, are shown in Figure 6.1. Within each of the outer states, a packet loss model (e.g., the models discussed in Chapter 5) is used to capture the packet loss process within each of the "outer" states. The two-level structure allows different classifiers and packet loss models to be used together. For example, an SGM model can be used for the "uncongested" outer state (since the results of Chapter 5 showed that the SGM works well in uncongested conditions), while a more complex model can be used for the "congested" outer states.

As Figure 6.1 shows, the model for the outer states is a three-state Markov chain. The parameters of this "outer" model, therefore, can be represented as a transition probability matrix, $\mathbf{A_{outer}}$:

$$\mathbf{A_{outer}} = \begin{bmatrix} p_{uncongested,uncongested} & p_{uncongested,edge} & p_{uncongested,core} \\ p_{edge,uncongested} & p_{edge,edge} & p_{edge,core} \\ p_{core,uncongested} & p_{core,edge} & p_{core,core} \end{bmatrix} \tag{6.1}$$

These parameters are estimated by counting the transitions between outer states (as determined by the classifier). The probability of moving from state i to state j is calculated using the number of transitions from i to j, n_{ij}, and total number of transitions from i, n_i, as described in [231]:

$$\mathbf{A_{outer}}[i,j] = \frac{n_{ij}}{n_i} \tag{6.2}$$

This section has described how the two-level model is constructed, containing an outer level representing the state of the network, and an inner level representing whether packets

are being lost or not. The transitions between these outer states are governed by a Markov chain, whose parameters are estimated by counting the transitions between states (as determined by the classifier). The two-level approach gives flexibility, allowing experimentation with the inner packet loss models (which can be different for each outer state), and outer state classifiers. The following sections introduce the classification schemes and corresponding inner packet loss models that are used in this dissertation.

6.3 Classification Schemes for Outer States

Two classification schemes have been used; one based on thresholds for loss and delay (*ld*), and another that looks for increasing trends in queueing delay before loss (*ldbl*). The schemes consider one-second windows of time, and in each window examine the number of losses (N), number of loss bursts (M), and median DQ ($\widetilde{DQ}$). Using this information about packet loss and delay, the classifier can identify the congestion state of the network within each period of time, and associate the window with the appropriate outer state.

6.3.1 Loss/Delay Threshold (*ld*) Classifier

The *ld* classifier, shown in Algorithm 6.1, uses thresholds for N and M to identify periods of high loss (which indicate congestion), and another threshold for $\widetilde{DQ}$ to identify between "core" and "edge" congestion. The choice of loss thresholds ($N > 2$ or $M > 2$ indicating congestion) is based on the assumption 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, which appears valid for this dataset. The $\widetilde{DQ}$ threshold (5ms) is also based on examination of the trace data; traces with "non-bursty" loss typically exhibit $DQ < 5$ms.

6.3.2 Loss / Delay Before Loss (*ldbl*) Classifier

The *ldbl* classifier, described in Algorithm 6.2, aims to generalise the *ld* classifier by no longer using a $\widetilde{DQ}$ threshold to distinguish between "edge" and "core" congestion. Instead, when congestion is identified (as before, when $N > 2$ or $M > 2$), the *ldbl* classifier looks at the median DQ from packets received before losses ($\widetilde{DQ}_{BL}$), and compares this against

Algorithm 6.1 Loss / Delay (*ld*) Classifier

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

$\widetilde{DQ}$. 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 doesn't 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 to 10% above $\widetilde{DQ}$ in the last "uncongested" state ($\widetilde{DQ}_{UC}$), the state switches back to "uncongested" (Algorithm 6.2 tests $\widetilde{DQ} \leq k\widetilde{DQ}_{UC}$, where $k = 1.1$). 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".

6.4 Packet Loss Models for Inner States

This section describes the "inner" packet loss models, which are used to actually describe the loss process within each of the outer states. Two of the models evaluated in Chapter 5 are

Algorithm 6.2 Loss / Delay Before Loss (*ldbl*) Classifier

if (state = "uncongested") **then**
  if ($N > 2$) or ($M > 2$) **then** # "high loss"
    if ($\widetilde{DQ}_{BL} > 2\widetilde{DQ}$) **then** # "elevated DQ"
      state $\leftarrow$ "edge congestion"
    else
      state $\leftarrow$ "core congestion"
    end if
    $\widetilde{DQ}_{UC} \leftarrow \widetilde{DQ}$
  end if
else if (state = "edge congestion") **then**
  if ($N \leq 2$) and ($M \leq 2$) and ($\widetilde{DQ} \leq k\widetilde{DQ}_{UC}$) **then**
    state $\leftarrow$ "uncongested"
  end if
else if (state = "core congestion") **then**
  if ($N \leq 2$) and ($M \leq 2$) **then**
    state $\leftarrow$ "uncongested"
  end if
end if

used here, the SGM and 2HMM. Using these allows evaluation of the two-level models using both simple (SGM) and more complex (2HMM) models within the outer states, to understand the improvement in accuracy made possible by using different models for different network conditions.

6.4.1 Simple Gilbert Model (SGM)

As mentioned earlier, the results of Chapter 5 showed that the SGM model worked well in low-loss conditions. Therefore, it seems appropriate to use this model to capture the packet loss within the "uncongested" state of the two-level model. The SGM is described in detail in Section 5.1.1; this section describes the differences in parameter estimation for use in a hierarchical model.

The process for estimating the SGM parameters within the two-level model is as follows:

1) go through binary loss sequences for the trace, noting the outer state determined by the classifier for each window;

2) in each window, count the number of packets lost and received (and transitions between loss and receive runs), and add to a count for the current outer state;

3) use the transition counts and number of lost/received packets in each outer state to estimate $\hat{p}$ and $\hat{q}$ for each outer state, using Equation 5.1.

The results of this process are estimates of $\hat{p}$ and $\hat{q}$ for each outer state that is modelled using the SGM. For example, estimates for $\hat{p}$ and $\hat{q}$ in the "uncongested" outer state, $\hat{p}_{uncongested}$ and $\hat{q}_{uncongested}$, are obtained.

6.4.2 Two-State Hidden Markov Model (2HMM)

The results of Chapter 5 showed that in traces where there is congestion, the SGM is not particularly effective, due to the inherent correlations between congestion-induced losses. Therefore, for the "congested" states, an HMM might be more appropriate, since it is somewhat more capable of capturing the correlations between losses. Since the performance of two- and three-state HMMs in Chapter 5 was not radically different, this chapter focuses on two-state HMMs.

In a similar fashion to the process for estimating SGM parameters in the hierarchical model described in Section 6.4.1, the parameters for the 2HMM in the hierarchical model are as follows:

1) go through the windows identified by the classifier;

2) for each window associated with the outer state being modelled, concatenate the binary loss sequence for the window, $Z_{i_{window}}$ onto a new sequence for the outer state, $Z_{i_{state}}$;

3) pass the sequence $Z_{i_{state}}$ into the HMM learning function from the *hmm.discnp* package [209], as described in Section 5.1.2.

For each of the outer states modelled using the 2HMM, the hidden state transition probability matrix, $\mathbf{A}$, and the hidden state loss probabilities, $\mathbf{B}$, are estimated as described in Section 5.1. For example, using the 2HMM for the "core congestion" outer state, $\mathbf{A_{core}}$ and $\mathbf{B_{core}}$ are obtained.

6.5 Two-Level Model Results

This section presents results of applying two-level models using the components described in Sections 6.3 and 6.4, showing how these perform in terms of the metrics defined earlier in Section 5.2. The models evaluated in this section use all of the classification schemes and packet loss models described earlier, combined into four configurations of the two-level model:

- *ld*, SGM/SGM/SGM; using the *ld* classifier, with SGMs in each of the outer states;
- *ld*, SGM/2HMM/2HMM; using the *ld* classifier, with SGMs in the "uncongested" outer state, and two-state HMMs in the "congested" outer states;
- *ldbl*, SGM/SGM/SGM; using the *ldbl* classifier, with SGMs in each of the outer states;
- *ldbl*, SGM/2HMM/2HMM; using the *ldbl* classifier, with SGMs in the "uncongested" outer state, and two-state HMMs in the "congested" outer states.

The approach is as follows: for each trace, classify the states using loss and delay information, according to Section 6.2, into "uncongested", "core congestion", and "edge congestion". Then, for each of these outer states, the parameters of the inner models for packet loss are estimated (as described in Section 5.1), leading to a separate set of parameters for each of the outer states. Using these separate per-state models, synthetic sequences can be generated according to the current outer loss/delay state, using the inner model for that state. This idea is illustrated in Figures 6.2 and 6.3.

The reason for choosing both SGMs and HMMs within the SGM/2HMM/2HMM configuration is to capture the loss burstiness present in congested outer states (i.e., although the delay level indicates there is congestion across a number of windows, this does not imply packet loss throughout the windows, since packet loss only occurs when queues overflow). Within the "uncongested" outer state, the SGM remains sufficient, since the results of Chapter 5 showed that non-bursty packet loss is captured sufficiently by the SGM.

Following the same approach as in Chapter 5, the parameters for each trace have been calculated, and three synthetic sequences generated and stored for each. From these, the mean loss rates and correlation timescales were obtained, and raw and synthetic RLEs generated. Since the K-S distance used in Chapter 5 was found to be of limited use, it is not

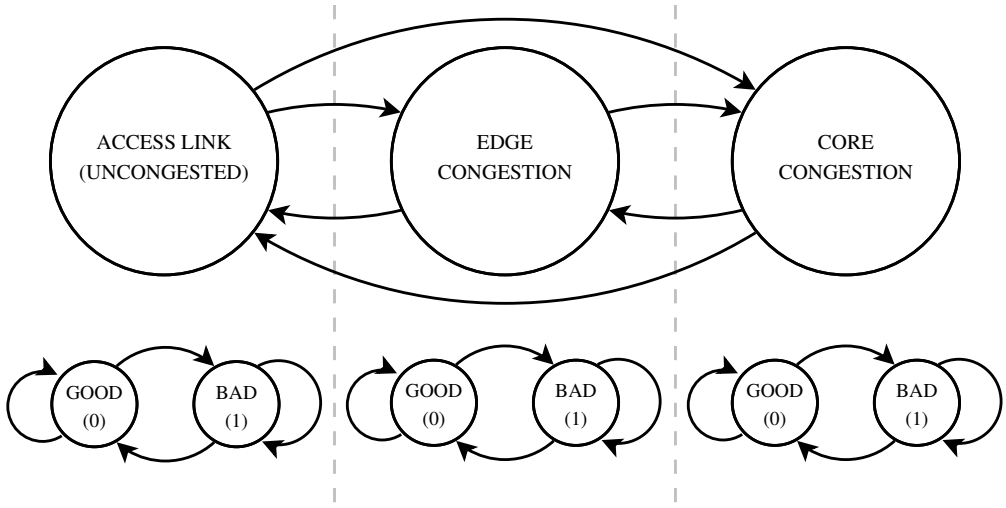


Figure 6.2: Two-Level Model with SGM in each outer state (SGM/SGM/SGM).

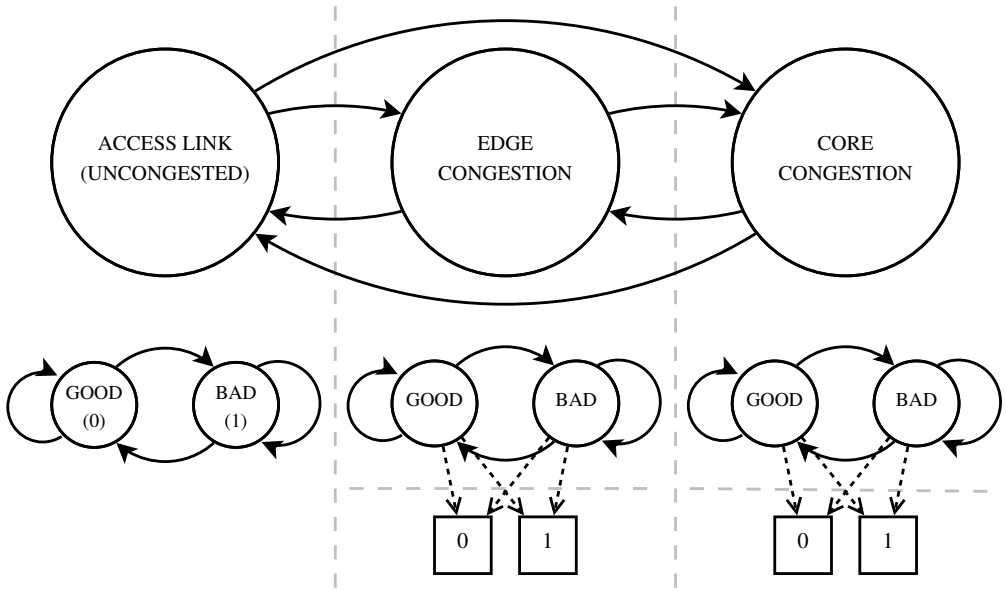


Figure 6.3: Two-Level Model with SGM in "uncongested" outer state, HMMs in "congested" outer states (SGM/2HMM/2HMM).

discussed here. Section 6.5.1 presents the aggregate results for mean loss rates and correlation timescales (R_{ML}, Δ_{ML}, and Δ_c), and Section 6.5.2 shows in-depth examples of the sequences generated by each of the two-level models. The parametric bootstrap process, described earlier in Section 5.4.1, has also been applied, and the results of this are presented in Section 6.5.3.

6.5.1 Aggregate Metrics

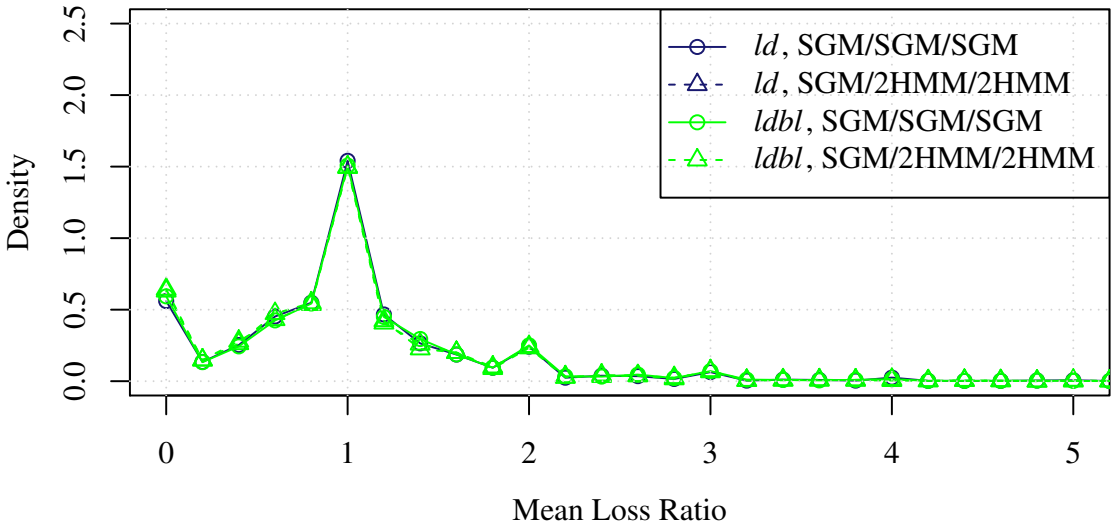
Figure 6.4 shows mean loss ratio R_{ML} and mean loss difference Δ_{ML}, for each of the synthetic sequences calculated from all the loss traces. Just as with the Gilbert models and HMMs using only packet loss (seen in Figure 5.5), all the models appear to perform similarly for these metrics. These results show that the loss rates are well-estimated by all the models in the majority of cases.

Figure 6.5 shows the differences in correlation timescale Δ_c (i.e., correlation timescale in synthetic trace minus correlation timescale in original trace), as shown previously for the loss models in Figure 5.9). The results from the two-level models are similar to those from Chapter 5, with the majority of the traces showing that the correlation timescale is well-captured by the model, but showing some traces where the correlation timescale is not captured.

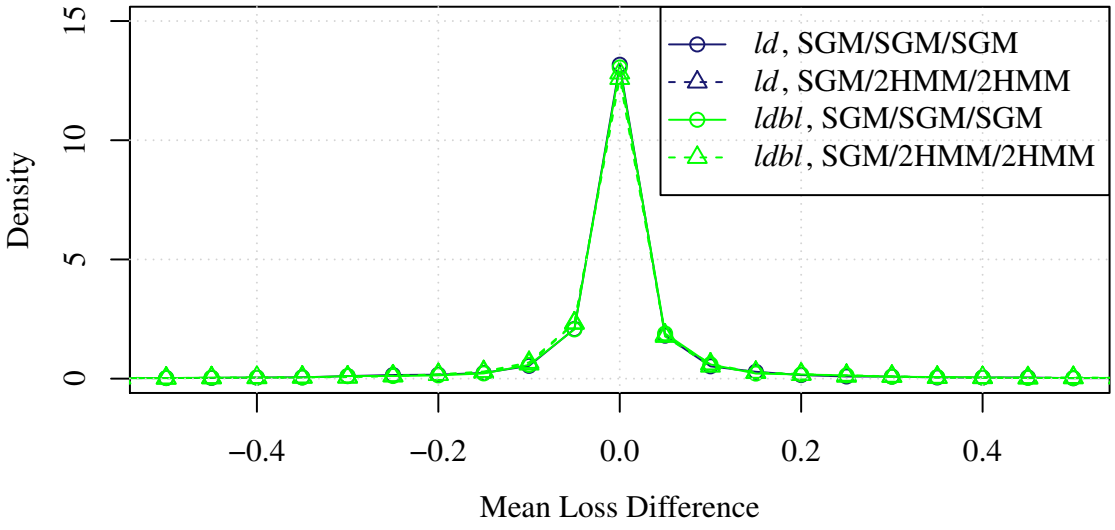
6.5.2 Visualising Example Traces

Figure 6.6 shows plots to visualise the synthetic sequences generated by the two-level models, alongside the corresponding raw traces. These are generated from the same example traces as those shown in Figure 5.13.

As before, the example traces showing low loss and non-bursty loss are equally well captured by all the models. However, as shown in Figure 6.6c, the two-level models perform much better at capturing bursty loss traces, with all of the two-level delay/loss models showing an improvement on the loss-only models (recall from Figure 5.13c that none of the Gilbert models or HMMs could adequately model this trace, due to the burstiness in the packet loss). Interestingly, Figure 6.6d shows that there are differences between the performance of the two-level models. For this trace, the *ld*, SGM/SGM/SGM model performs very poorly, as bad as the SGM model in Chapter 5, while the others are better. The

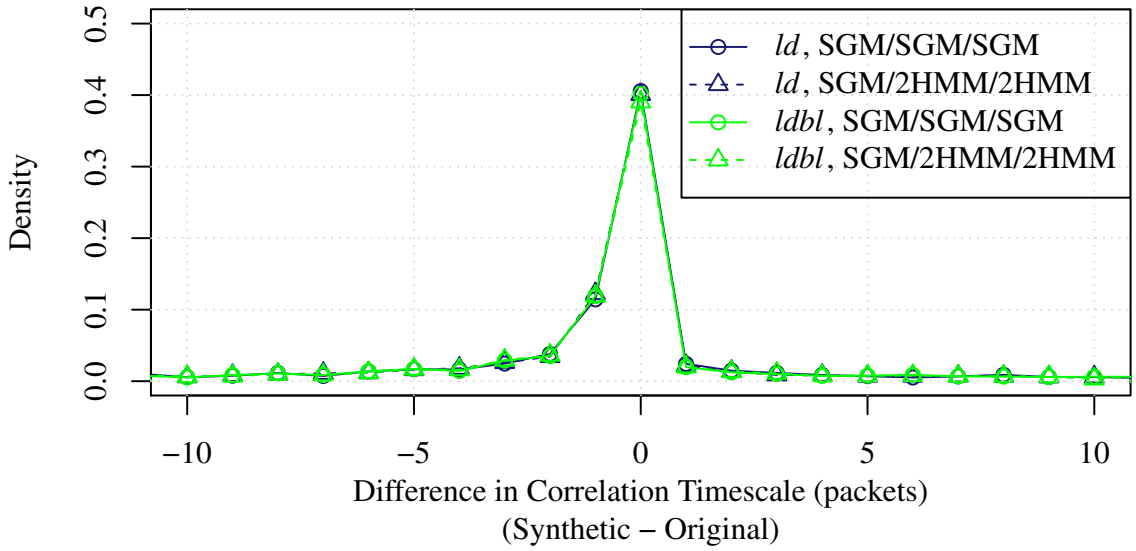


(a) Mean Loss Ratios R_{ML}

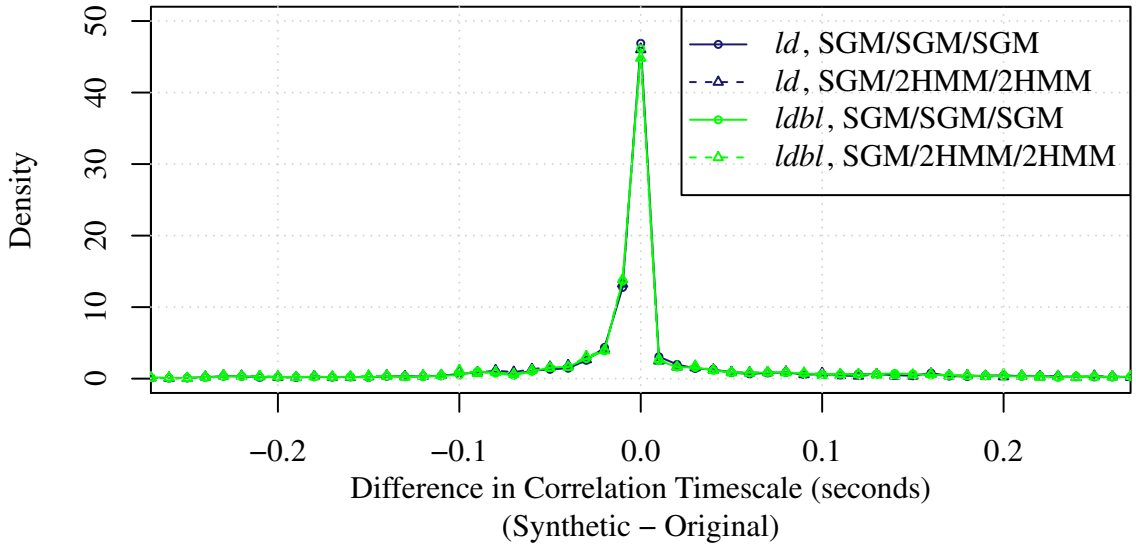


(b) Mean Loss Differences Δ_{ML}

Figure 6.4: Distributions of Mean Loss Ratios and Differences. Synthetic sequences are generated from each trace, and R_{ML} and Δ_{ML} from each of these is calculated.

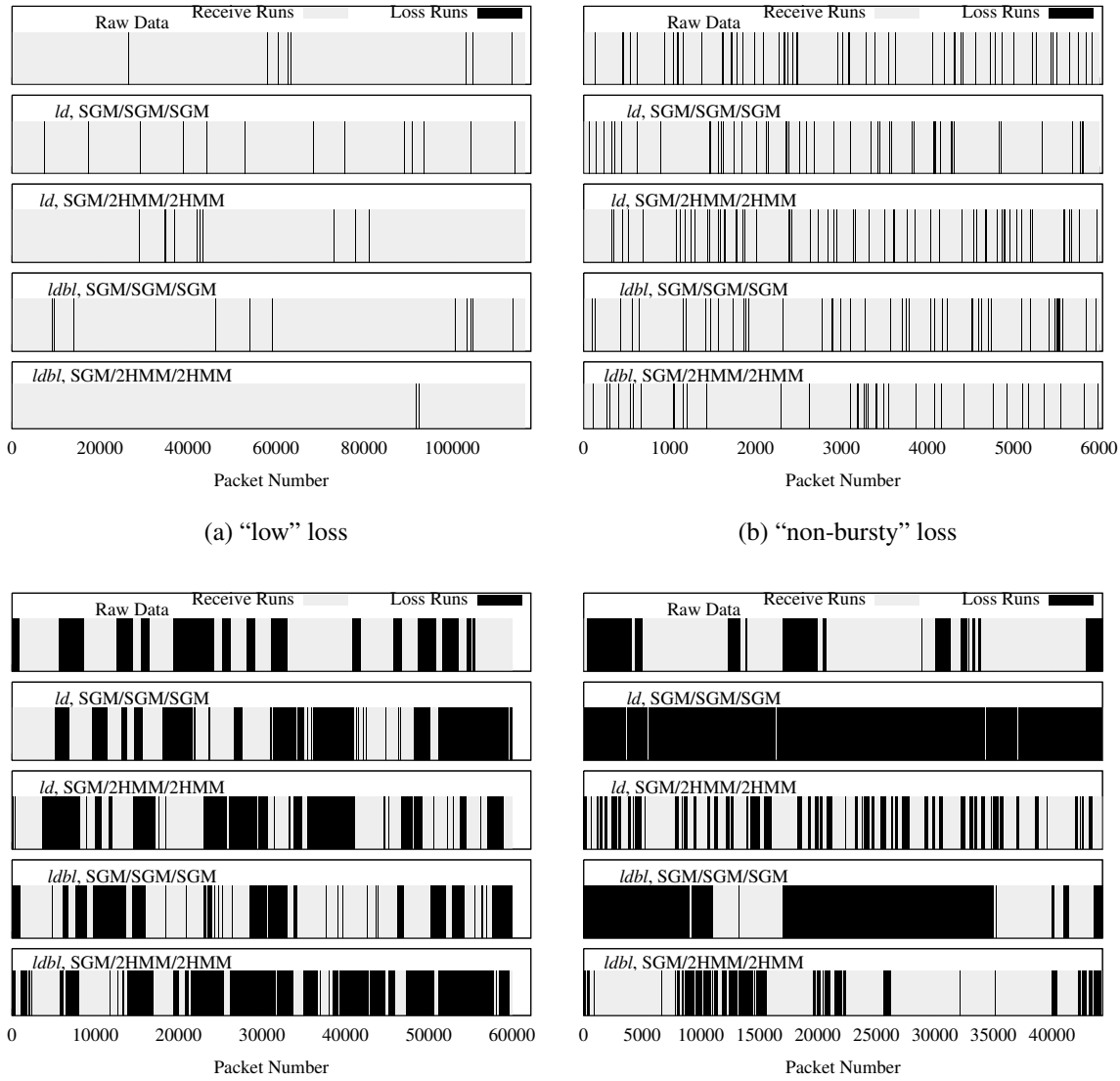


(a) Δ_c in packets



(b) Δ_c in seconds

Figure 6.5: Distributions of Differences in Correlation Timescales (Δ_c). Synthetic sequences are generated from each trace, and Δ_c from each of these is calculated.



(a) "low" loss

(b) "non-bursty" loss

(c) "bursty" loss

(d) "bursty" loss

Figure 6.6: Example Loss Traces and Synthetic Sequences from Two-Level Models (same example traces used with SGM, EGM and HMMs in Figure 5.13, page 110)

reason for this is that the *ld* classifier has identified the whole of this trace are showing "edge congestion", due to high delay, meaning that the two-level model collapses to its "inner" model, the SGM. However, the other models show better performance, with the *ld*, SGM/2HMM/2HMM model showing similar performance to the 2HMM in Chapter 5 (for the same reason), and the models using the *ldbl* classifier identifying different periods of congestion within the traces. These results show that the versatility of the two-level models are more suited to capturing bursty packet loss behaviour.

6.5.3 Parametric Bootstrap Results

This section discusses the results from testing the goodness-of-fit of the two-level models using the parametric bootstrap technique described in Section 5.4. As in Section 5.4, the goodness-of-fit test used is to compare whether the statistic calculated from the raw data falls within the central 95% of the distribution of the synthetic statistics.

To demonstrate this improved performance over all the traces, Figures 6.7 and 6.8 show the results of applying parametric bootstrap to the "non-bursty" and "bursty" traces identified in Chapter 5. 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 showing good fit for these models than the previous models in Figures 6.7 and 6.8 (compared with the SGM, EGM, and HMM results shown in Figures 5.15 and 5.16, on pages 114 and 115).

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 SGM, EGM, and HMMs using only loss data, with the percentiles of the receive run-length distribution still not being well-modelled in the majority of traces. However, the SGM/2HMM/2HMM configuration shows 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 periods, and HMM for congested periods).

These results show that the two-level delay/loss models (using the SGM/2HMM/2HMM configuration) are more accurate than using the previous SGM, EGM, or HMMs. For the

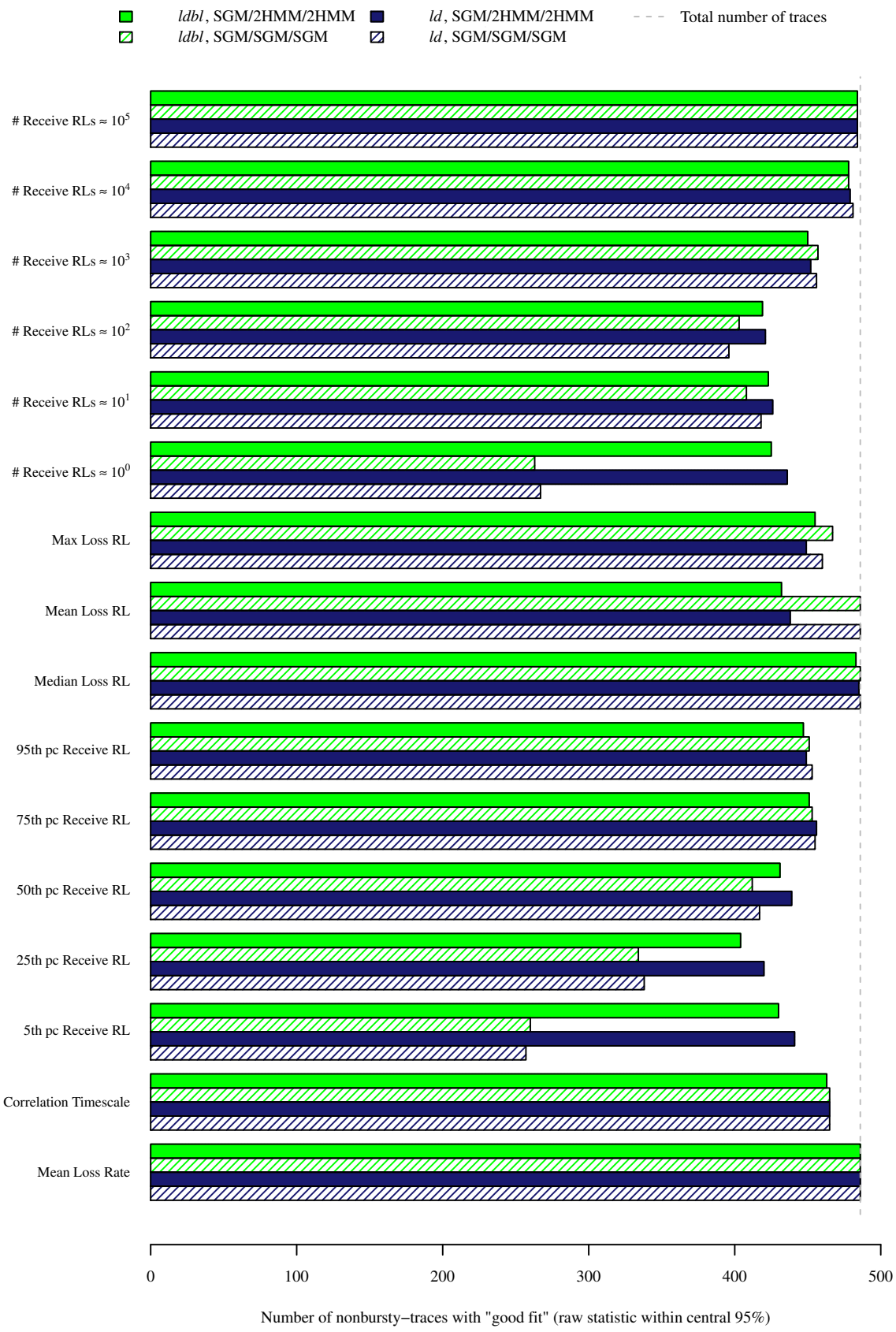


Figure 6.7: Parametric Bootstrap Results (“non-bursty” traces)

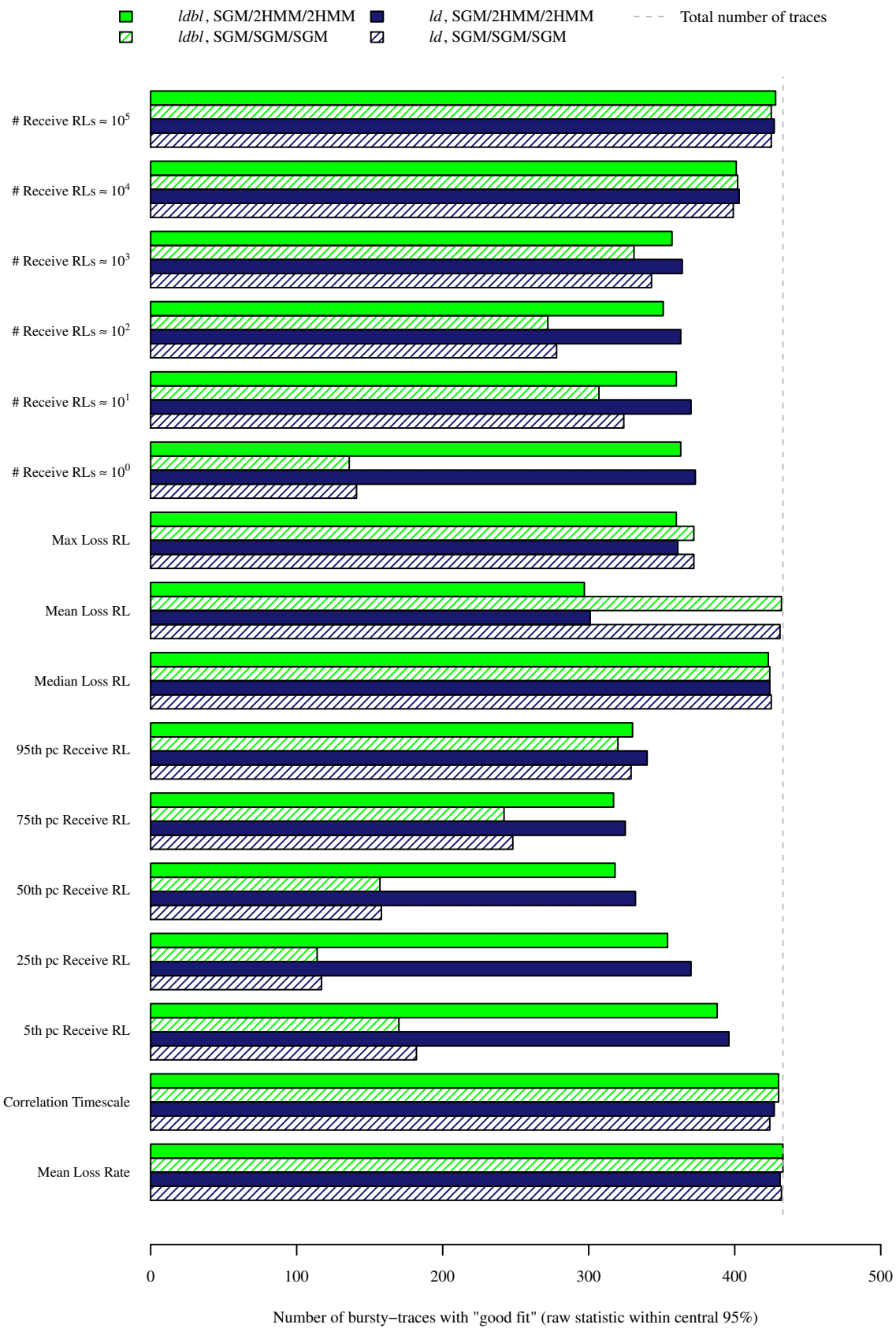


Figure 6.8: Parametric Bootstrap Results (“bursty” traces)

traces previously identified as "non-bursty", the SGM/2HMM/2HMM configuration is 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 examples of Section 6.5.2. Looking at the percentiles of the receive run-length distribution (which the previous models failed to accurately model), the importance of the choice of inner model is made clear. Two-level models with the SGM/SGM/SGM configuration perform better than the SGM and EGM models for all the receive run-length percentiles, but not as well as the HMMs do for the 5^{th} and 25^{th} percentiles. However, by using the SGM/2HMM/2HMM configuration, the performance is better than the SGM, EGM, and HMMs, for all the statistics.

6.6 Discussion & Summary

This chapter has presented a new technique for modelling packet loss, which incorporates packet loss and delay data, and classifies the underlying network state to more accurately capture the congestion state and resulting packet loss. Combining classification of the underlying network state with the models evaluated in Chapter 5, I developed a two-level model. The "outer" model captures changes in network conditions (e.g., between different sources of packet loss; access link noise, core congestion, and edge congestion), while the "inner" models capture the packet loss process within each outer state.

The results indicate that both the classification schemes (*ld* and *ldbl*) give similar overall model performance, but that the choice of "inner" packet level models determines the quality of fit. Most importantly, by using the SGM/2HMM/2HMM configuration of inner models, with HMMs for the "congested" states, better performance can be achieved than by just using classification and separate SGM models for each state. This reason for this improved performance is that, even within periods of time classified as "congested", packet loss is still bursty, with losses only occurring when queues overflow. By using the HMM, which incorporates these different behaviours (within the congested state), better performance can be achieved. Moreover, by using the SGM within non-congested periods, the complexity of the modelling process is reduced. By explicitly capturing changes in the network state, the problem of the lack of "constancy" [228] in Internet packet loss can be addressed.

Chapter 7

Evaluating AL-FEC Performance under Residential Packet Loss

As discussed in previous chapters, packet losses lead to degradation in the quality of streaming video applications, unless error recovery mechanisms such as forward error correction (FEC) are used, as described in Section 2.3. In IP-based streaming video and IPTV applications, FEC is generally deployed at the application level (AL-FEC), adding redundant packets to the media stream that can be used to repair loss. To improve transmission efficiency, and to reduce the risk of introducing congestion by increasing the overall data rate, it is desirable to minimise the FEC overhead, while maintaining adequate protection. Finding the correct balance can be difficult, and requires insight into the network conditions. With ongoing deployment of streaming video and IPTV to residential Internet users, it is becoming more 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. Since much of the existing work on FEC performance uses simulation with models like the SGM, I evaluate the accuracy of the SGM and the two-level model presented in Chapter 6 in capturing FEC performance.

This chapter will evaluate the performance of three application-layer FEC schemes applied to RTP-based streaming video. The FEC schemes, which have been standardised by the IETF and implemented in the OpenFEC project (http://openfec.org) are: 2D parity codes (2D) [196]; "Reed-Solomon codes for the erasure channel" (RSE) [119]; and LDPC-Staircase codes [180]. The measurements from Chapter 3 are used to inform trace-driven

simulations of FEC performance, studying residual packet loss rates and fraction of lost packets successfully recovered. The contributions of this chapter are 1) a simulation-based evaluation of application-layer FEC performance on real networks, using traces of unmanaged Internet streaming to residential users (i.e., reflecting the conditions experienced by "over-the-top" streaming video or video conferencing services); 2) an explanation of the differences between these results and those of previous evaluations of these schemes under random packet loss (particularly the effect of bursty packet loss); 3) guidelines for use of FEC, to recommend which FEC schemes and parameter combinations work well under the loss conditions common on residential networks; and 4) a validation of the results of Chapter 6, showing that the two-level model is also more accurate than previous models in terms of FEC performance, a real application where packet loss simulation is used.

This chapter is structured as follows. Section 7.1 outlines related work on evaluating FEC performance. Section 7.2 describes the FEC schemes in OpenFEC. Section 7.3 explains the evaluation methodology, including discussion of FEC parameters and performance metrics. Section 7.4 presents the results of the evaluation. Section 7.5 shows the relationship between trace burstiness and FEC performance, and introduces a new metric to capture this relationship. Section 7.6 shows how the two-level models presented in Chapter 6 allow more accurate simulation of FEC, compared to random loss, and previous models for packet loss. Section 7.7 summarises the chapter.

7.1 Background

The OpenFEC library contains implementations of three FEC schemes; 2D parity codes [196], Reed-Solomon Erasure codes [119], and LDPC-Staircase codes [180]. These schemes were studied under uniform random packet loss from 0–51% by Matsuzono *et al.* [134]. In that work, a lab-scale experiment was set up, with a sender and receiver running a Digital Video (DV) application transmitting data over RTP/UDP/IP [184, 109]. The study evaluated recovery capabilities, latency introduced, and CPU cost incurred by each of the FEC schemes, concluding that LDPC codes with a source block size of ($k = 170$, $r = 85$) give the best trade-off between recovery performance, latency, and CPU load. However, this does not tell us about FEC performance under real-world packet loss conditions, which is what this chapter investigates.

A previous study looking at FEC performance on DSL networks was conducted by Begen [16]; this compares the performance of 1D interleaved parity codes against Raptor codes [186]. Begen uses typical DSL noise models to drive simulations of loss patterns, while this chapter evaluates performance using measured loss traces. Similar models of packet loss were used in a DVB study [58] to evaluate the performance of 1D parity and Raptor codes, describing the relative strengths and weaknesses of both schemes. Luby *et al.* [129] discuss AL-FEC for IPTV, including a discussion of different FEC schemes, and the "layered" approach using 1D parity and Raptor codes, discussed in [58]. Mammi *et al.* [132] investigate SMPTE 1D and 2D parity FEC schemes [196], using a testbed setup and models of random i.i.d. noise and repetitive electrical impulse noise. Kang & Loguinov [103] analytically studied the effect of packet loss on FEC for video streaming, using a number of models for packet loss, but did not consider any Internet packet loss measurements. FEC performance for video on optical backbone networks was studied in [147], finding that a suitable trade-off between repair performance and latency can be hard to achieve. Older work examined performance of FEC for packet audio over the academic backbones [26, 170, 167], using measured traces and performance models.

This chapter focuses on FEC performance using measured data from residential broadband networks, since streaming video systems are often accessed by home users. Previous chapters have shown that the characteristics of the packet loss on these networks has been found to be quite different from uniform random loss, so FEC performance can be expected to be different too.

7.2 Applying OpenFEC to IP-based Streaming Video

In this section, each of the FEC schemes implemented in the OpenFEC framework is discussed. Each of these schemes has been considered by the IETF FECFrame working group for use in protecting media streams, and has received interest from both academia and industry. This section discusses the operation of each of the schemes, describes the overhead and latency they incur.

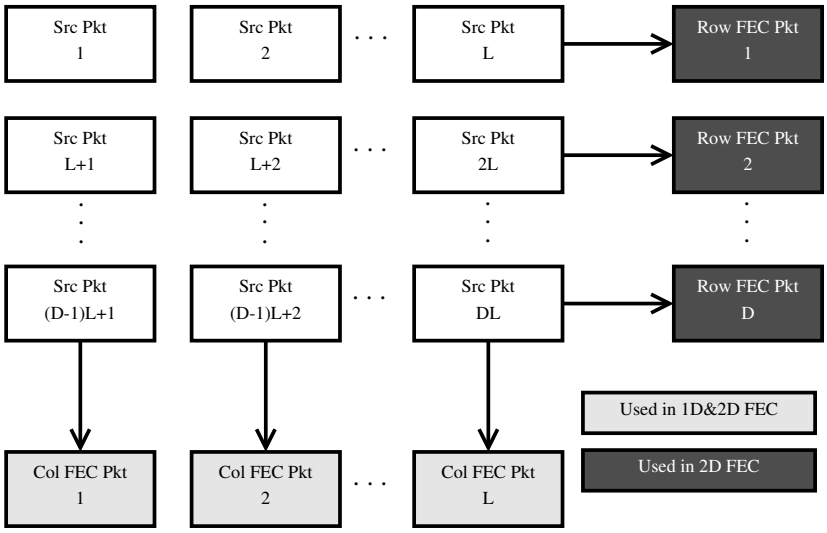


Figure 7.1: Grid Format for Source and Repair Packets in 2D FEC

7.2.1 2D Parity Codes

2D parity codes work by arranging the source packets into a grid of D rows and L columns, and adding repair packets for each row and column, as shown in Figure 7.1. Therefore, a total of $D + L$ repair packets are added to the source packets. If at most one packet is lost in a row or column, then the remaining packets in the row (or column) can be used the recover the loss. SMPTE 2022-1 [196] is a widely deployed example.

This scheme is simple, but has a limited capacity for recovery, since it is only resilient to up to D consecutive losses (i.e., is able to recover from loss bursts up to length D, provided that no other packets were lost within the grid). As there are constraints on the latency that a real-time system can tolerate, the D and L parameters cannot grow too large. In OpenFEC, the 2D codes use square grids (with $D = L$), and limit k to be ≤ 16.

7.2.2 Reed-Solomon Codes

RFC 5510 [119] defines Reed-Solomon FEC schemes for the Erasure channel (RSE codes); the so-called packet-erasure channel implies that packets either arrive successfully or are discarded, which matches the use-case of application-layer FEC running over IP networks. Reed-Solomon codes (originally proposed in [177]) are maximum separable distance codes, meaning that of the n encoding symbols sent (including k source symbols and $n - k$ repair symbols), any k can be used for recovery.

The computational cost of the mathematical operations (using Galois Fields) increases

rapidly with the size of those fields, introducing a practical limitation on the block sizes that can be used. Due to these limitations, this chapter will look only at RSE codes over GF(2^8) (as in [134]).

7.2.3 LDPC-Staircase Codes

A third FEC scheme, LDPC-Staircase codes (defined in RFC 5170 [180]), is suitable for use with large block sizes, and has a relatively low computational complexity. Similar to Raptor codes [186], LDPC-Staircase codes require slightly more than k encoding symbols to be received to allow recovery (i.e., these are not maximum separable distance codes). However, in practice, the fraction of extra symbols required can be quite low when using an appropriate decoding algorithm. Matsuzono *et al.* [134] state that $(k \times 1.05)$ is appropriate, based on experimental evidence [44].

7.3 Evaluation Methodology

The OpenFEC framework provides implementations of the three FEC algorithms under study, as well as a performance evaluation tool, eperftool. This tool allows the evaluation of the FEC schemes on a single machine, simulating the transmission and reception of packets. It is configurable with a number of transmission schemes (which determine the order of transmitted source and repair packets) and loss modes (which determine which packets arrive at the receiver).

The FEC schemes are evaluated using the same binary loss sequences used in Chapters 5 and 6, obtained from the measurements in Chapter 3. To allow evaluation of the FEC schemes with these loss traces, minor modifications were made to eperftool, as follows. A new mode was added, which reads a given loss trace and uses the loss patterns within the trace to decide whether packets will be received or lost. A new transmission mode (which determines the order in which packets are sent) was also added, to support the source and FEC packets for each block being sent together, rather than using the default sending arrangement that sends all source packets (for all blocks) first, then all repair packets. This modification is necessary to prevent large delays when recovering lost packets. Another minor modification, to support the replication of the results of [134], was to add a maximum source block size. This was necessary to support the different k values specified for the LDPC scheme.

Finally, modifications have been made to profile the FEC decoding process, to enable more fine-grained reporting of packet loss and repair than was previously available in eperftool. To this end, the number of source and repair packets received within each block are recorded, to allow calculation of per-block and overall packet loss statistics. When decoding of a block fails, the number of source packets received (and lost) determine the residual loss rate. A further metric of interest in FEC is the delay incurred by waiting for FEC packets to recover lost packets. To capture this, profiling code was added to record the distance between lost packets and the repair packets that recover them. The modifications to eperftool are available at http://martin-ellis.net/research/fec.

7.3.1 FEC Parameters

To begin, I apply the same parameters as [134]; 2D parity with ($k = 16$, $r = 8$), RSE with ($k = 170$, $r = 85$), LDPC with ($k = 170$, $r = 85$), ($k = 500$, $r = 250$), and ($k = 1000$, $r = 500$), and code rate of $\frac{2}{3}$, (i.e., 50% overhead). These parameters are used so that the experiments of [134] can be re-run to validate the experimental setup, and to compare performance of the FEC schemes using real-world loss traces against simulated random loss.

When running a simulation, the total number of source (T_k) and repair (T_r) packets needs to be specified. When re-running the experiments of [134], $T_k = 10000$ and $T_r = 5000$ are chosen, to achieve a code rate of $\frac{2}{3}$, with a reasonable number of blocks for each of the FEC schemes being tested. For the loss traces, the choice of T_k and T_r is determined by the trace length, with $\frac{2}{3}$ of the trace being allocated to the source packets, and the remainder to the repair packets. So, for a trace of length T:

$$T_k = \left\lfloor T \times \frac{2}{3} \right\rfloor, \quad T_r = \left\lfloor T \times \frac{1}{3} \right\rfloor. \tag{7.1}$$

For 2D FEC, there are further limitations, such that T_k must be a multiple of 16 (the largest 2D block size supported in OpenFEC), and T_r is $T_k/2$.

7.3.2 Performance Metrics

In [134], three metrics are used to evaluate performance; residual (post-repair) loss rate, frame delay, and CPU usage. Since I am evaluating the algorithms using eperftool, rather than a separate sender and receiver, I will look at residual loss rate and delay due to FEC.

Measuring CPU usage is not so important, since the goal is not to measure the complexity of the algorithms, but rather to observe how their repair performance is affected by measured loss data from residential networks, and since FEC decoding cost is small relative to the cost of decoding video.

To calculate the delay due to FEC, the number of packets received between the time when the lost packet would have been received, and the receipt of the packet that repairs the loss, is counted. This is more appropriate than the wall-clock time measured for delay in Matsuzono *et al.* [134], since it makes no assumptions about sending rates (as the sending rate increases, the time delay associated with a fixed number of packets decreases), and is not tied to the particular processor or load of the machine doing the calculations. Although delay in terms of packets does not consider the FEC decoding delay, in practice this is expected to be small compared to the packet arrival delay, especially given the low packet sending rates available on residential networks.

7.4 OpenFEC Performance

This section begins by evaluating the FEC schemes under simulated random loss, as in [134], then presents results of applying the same parameters to the loss traces measured in Chapter 3. Then, the effect of applying FEC on delay (i.e., latency incurred by waiting for repair packets to arrive) is considered in terms of time, with a typical packet transmission rate. Finally, another set of FEC parameters, which ensure acceptably low FEC delay, are then considered.

7.4.1 Applying FEC under uniform random loss

Figure 7.2 shows the input packet loss probability against residual (post-repair) packet loss rate (the diagonal represents loss rate with no FEC). This shows that for low loss (up to around 10%), all the schemes perform well, repairing almost all loss. Above 10%, 2D parity FEC starts to show poorer performance, showing steadily higher residual loss rates as the input loss probability increases. Up to around 30% loss, the other FEC schemes continue to perform well. However, above 30%, they show a sharp degradation in performance, with residual loss rates climbing sharply, eventually matching the input loss probability after 35%. Beyond 35% loss, 2D FEC gives slightly lower residual loss rates (since the small block

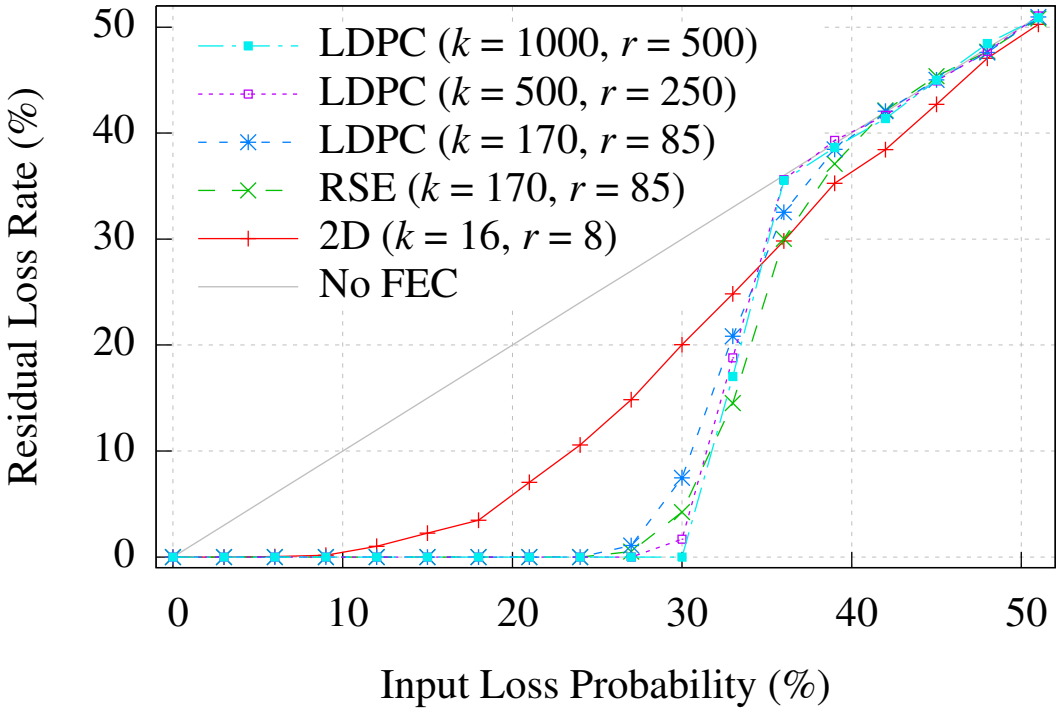
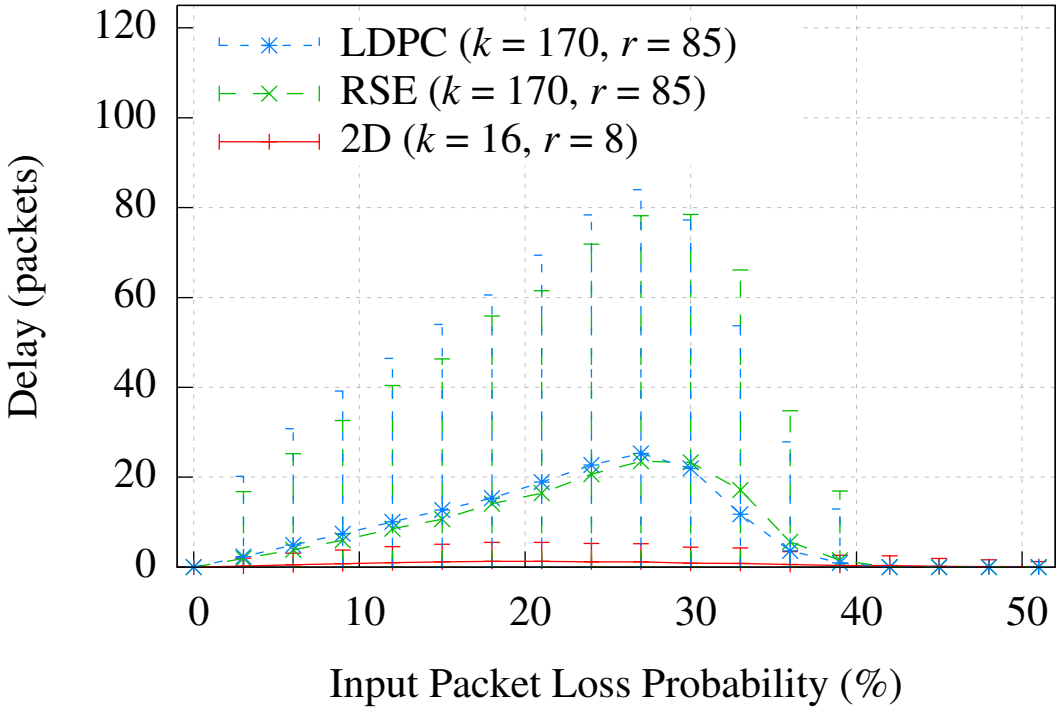


Figure 7.2: Random Loss Probability vs. Residual Packet Loss

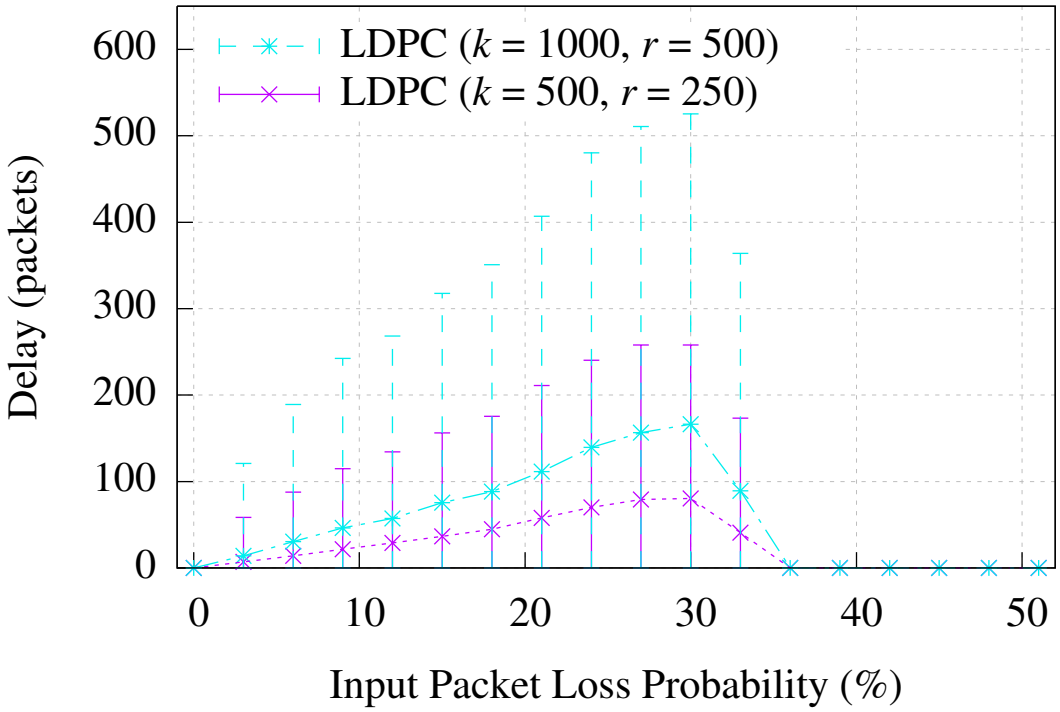
sizes mean that some blocks will be repaired). However, residual loss rates of above 30% will produce unusable video. These results are comparable with those in [134], which also showed the early decline in 2D parity FEC performance, and the "cliff" in performance for the other schemes.

Figures 7.3a and 7.3b show the mean and standard deviation of the FEC delay (in terms of packets). At low loss rates, the delays are generally low (although related to the block size), since most packets are received normally (i.e., with a delay of zero packets). As loss rate increases, the mean (and variance) of the delays increase, with more packets in need of repair, up to the threshold point discussed earlier and seen in Figure 7.2. After this point, since fewer packets are actually recovered, the average delay due to FEC decreases.

The results presented in Figure 7.3 are independent of the packet transmission rate (which will be discussed in more detail in Section 7.4.3). Therefore, they are comparable with, but do not directly replicate those in [134], which assume a particular transmission rate, and include the CPU processing time for the FEC. The impact of transmission rate is discussed in Section 7.4.3.



(a) Mean FEC Delay (small/medium k)



(b) Mean FEC Delay (large k)

Figure 7.3: FEC Delay under Random Packet Loss

FEC Scheme	Num traces	Percentage
2D ($k = 16$, $r = 8$)	3264	85.6%
RSE ($k = 170$, $r = 85$)	3728	97.9%
LDPC ($k = 170$, $r = 85$)	3724	97.8%
LDPC ($k = 500$, $r = 250$)	3780	99.2%
LDPC ($k = 1000$, $r = 500$)	3789	99.5%

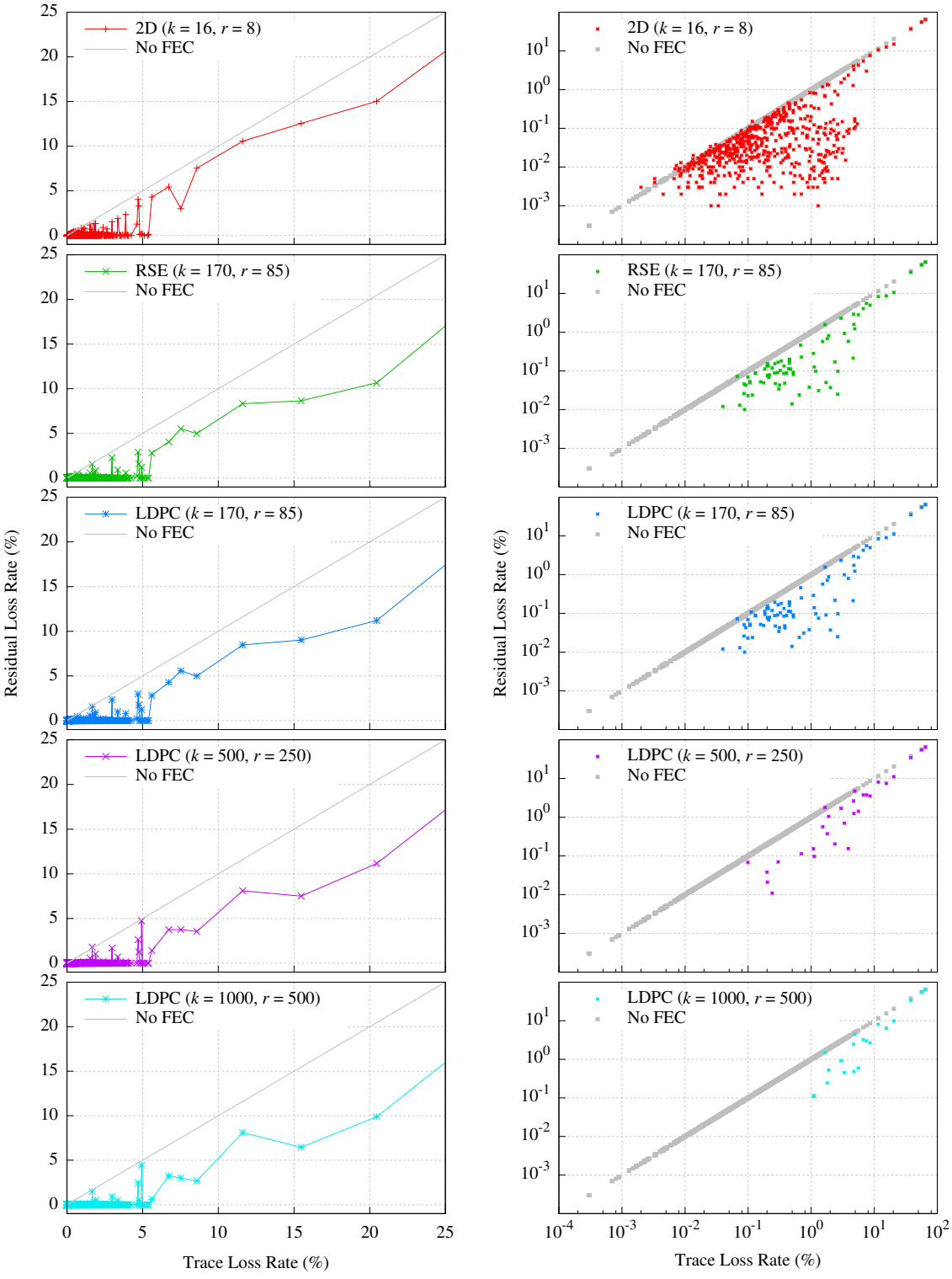
Table 7.1: Number of Traces with 0% Residual Loss Rate

7.4.2 Applying FEC to loss traces

Figure 7.4 shows results of applying the FEC schemes to the loss traces from Chapter 3. Since most of the traces show quite low loss rates, the scale in Figure 7.4a is set to focus on the 0–25% range. Observe that unlike the random loss results (where all the FEC schemes except 2D parity show almost full recovery under 25% loss), there are traces in this range where not all loss is recovered. Table 7.1 gives the percentage of traces, for each FEC scheme, which have 0% residual loss. Note that having such a high percentage of low loss traces is not unusual for a well-engineered network. However, those traces that show significant loss are common enough to have a severe effect on user experience. A common target for IPTV services is to have no more than one visible artefact per two hours (or fewer). This corresponds to packet loss of $\sim 10^{-6}$ [16].

It is clear that the 2D parity code has the poorest performance, as in [134], since it has the lowest fraction of traces being fully repaired (see Table 7.1). This lower fraction produces the higher points on the 2D FEC graph in the top panel of Figure 7.4; some of these are clustered near to the diagonal, showing that the performance is little better than it would be without FEC.

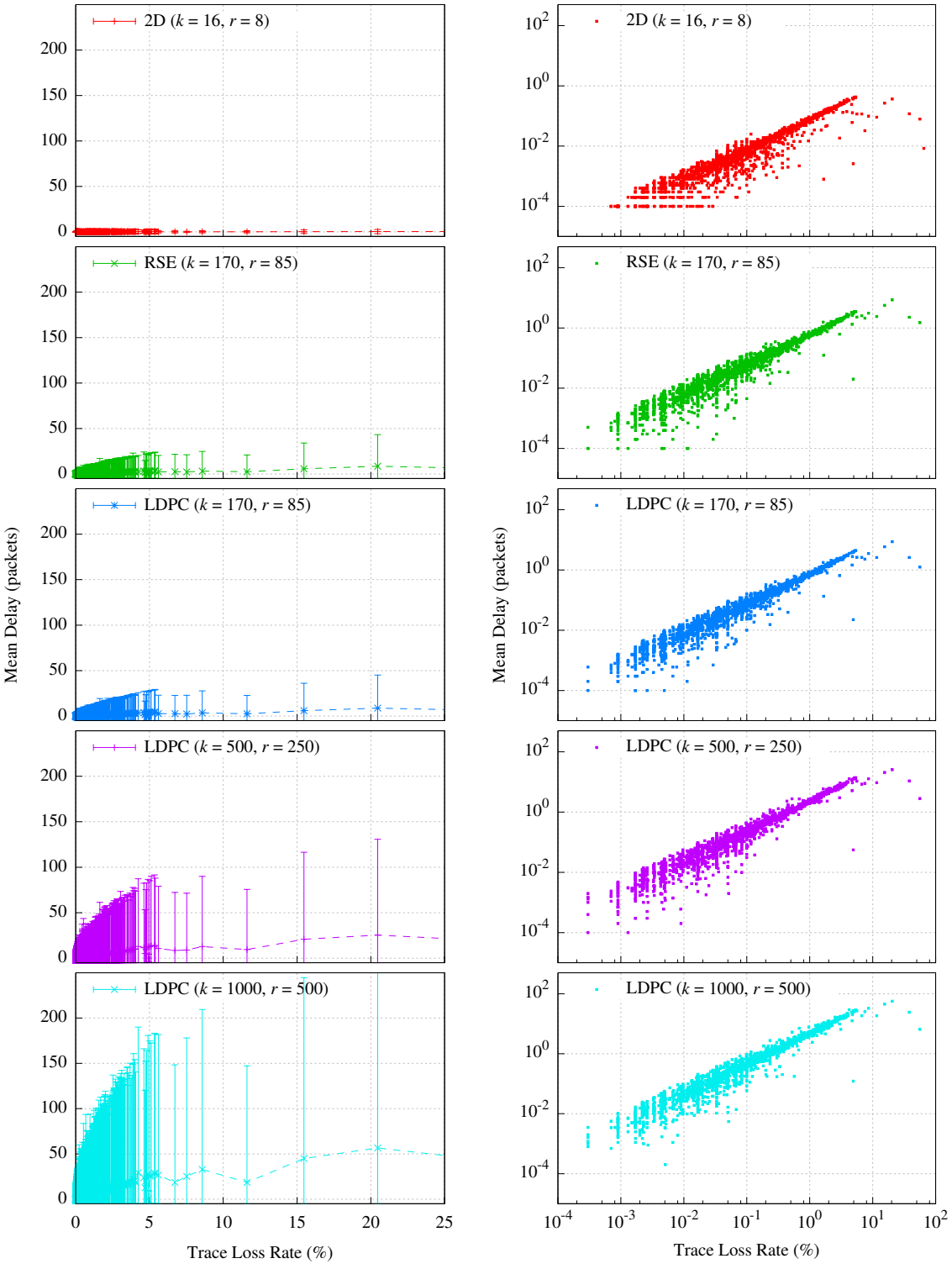
The RSE and LDPC schemes perform equally well for ($k = 170$, $r = 85$), with LDPC performing even better at larger block sizes. However, unlike the random loss results seen in the previous experiments, there are cases of low loss (i.e., less than 10%) where the losses in the trace data cannot be recovered, resulting in residual loss rates above zero. In these cases, the loss traces contain loss patterns that overwhelm the capacity of FEC to recover within certain blocks (e.g., large loss bursts). This result is a reminder that the relationship between packet loss and FEC performance is not straightforward. It is also worth noting that there



(a) Linear scale

(b) Log scale

Figure 7.4: Trace Loss vs. Residual Loss Rate



(a) Linear scale

(b) Log scale

Figure 7.5: Trace Loss vs. Mean FEC Delay

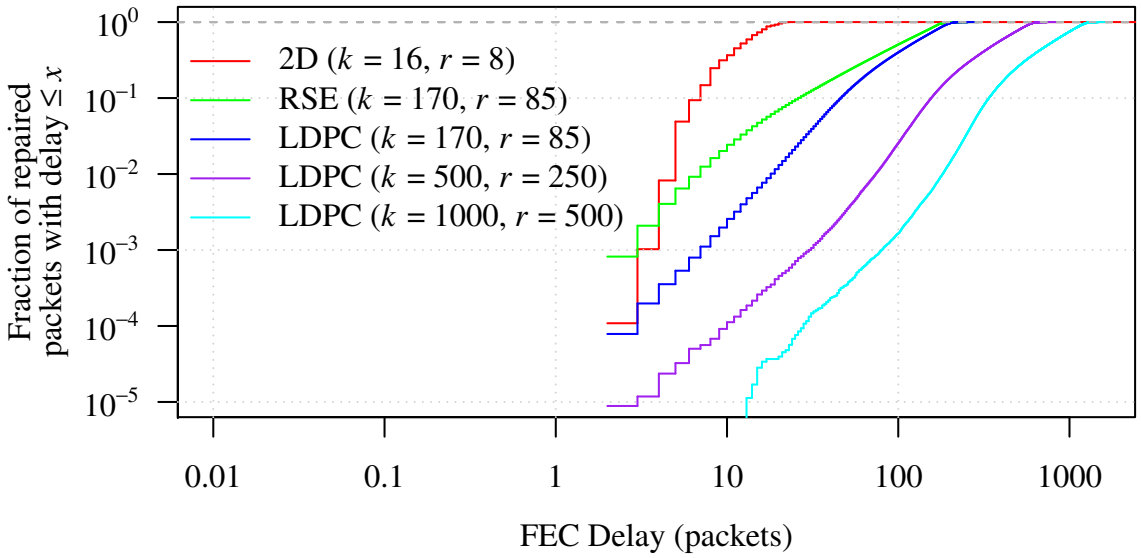


Figure 7.6: Cumulative Distribution of FEC Delays from Loss Traces

are cases where none of the FEC schemes are able to repair the loss (even with a large FEC overhead). In such cases, a retransmission-based recovery mechanism will be necessary.

Figure 7.5 shows trace loss vs. per-trace mean delay (in terms of packets, as discussed in Section 7.3.2). A similar shape can be seen for all the schemes, with the overall trend showing higher mean delays when the trace loss rate is higher (since more packets need to be repaired). Moreover, the mean delay in terms of packets increases with the size of the FEC blocks. The maximum per-trace mean delay is around one packet for 2D FEC ($k = 16$, $r = 8$), around 10 packets for RSE and LDPC with ($k = 170$, $r = 85$), and between 50 and 100 packets for LDPC with ($k = 500$, $r = 250$) and ($k = 1000$, $r = 500$).

Mean delay, while useful, does not tell the whole story, since video quality depends not only on the mean packet delay, but on how often packets exceed their deadline and cause visible distortions to playback. Figure 7.6 shows the cumulative distribution of per-packet delays for each of the FEC schemes. This shows the delays due to FEC repair, for each packet that was repaired. Note that the point where all packets are repaired is close to the FEC block size. This is because the FEC block size determines the worst case for the FEC delay (i.e., in the case where all the repair packets are required for recovery). Given the limitations on time available for FEC recovery, the relatively high worst case delays for the LDPC schemes with ($k = 500$, $r = 250$) and ($k = 1000$, $r = 500$) can limit their applicability.

7.4.3 Calculating FEC Latency (in seconds)

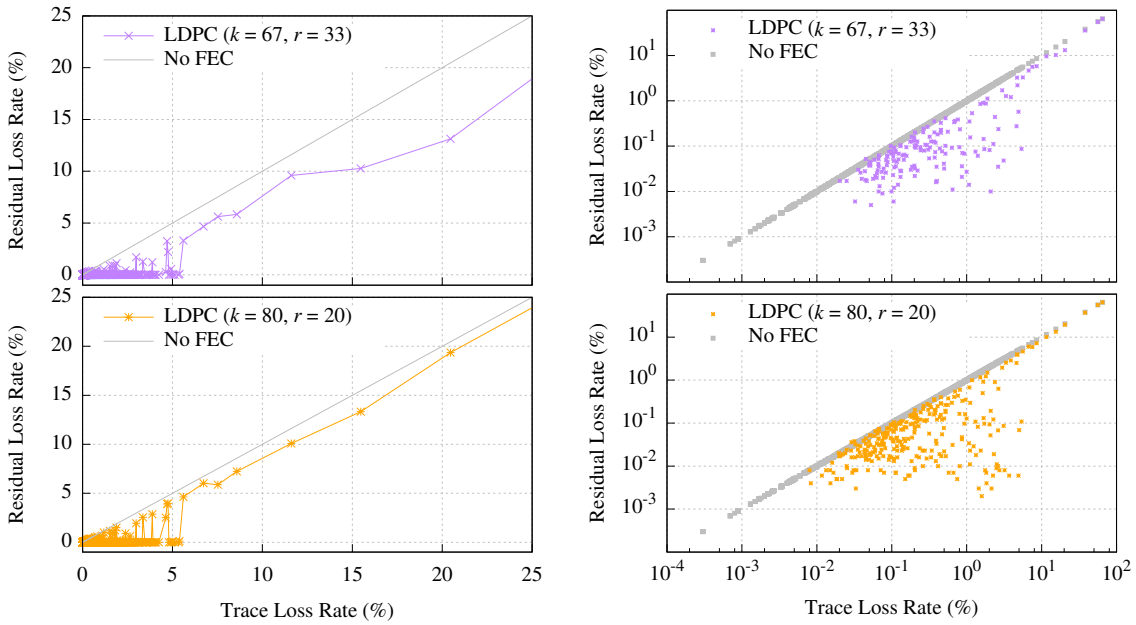
To recommend which of these schemes is most appropriate for use on residential networks, this section looks at their ability to recover from losses, and the delay they introduce. So far, FEC delay has been considered in terms of the number of packets sent in the stream between packets being lost and being recovered. However, by considering the sending rate of these packets, FEC delays in terms of time can be calculated. Considering sending rates of 5Mb/s, which can be received by the typical residential broadband user ([115] found that average home downstream bandwidth was 6.2Mb/s), and 1316-byte RTP packets carrying MPEG video, leads to a packet rate of 500 packets per second (pps). Using this rate, the latency of the FEC can be estimated. To achieve the overall 430ms latency bound suggested in [114], a constraint of 200ms for the FEC latency seems appropriate, allowing time for other components of channel-change latency. Assuming this sending rate of 500pps, this means that the upper bound on FEC delay is 100 packets.

Figure 7.6 shows that the fraction of *repaired* packets exceeding the 200ms limit for FEC delay (100 packets at 500pps) is 50% for RSE ($k = 170$, $r = 85$), and 60% for LDPC ($k = 170$, $r = 85$), although only 0.075% (RSE) and 0.09% (LDPC) of *all* packets exceed the threshold. Recall that the worst case FEC delay is around the FEC block size k. Therefore, to apply these schemes in a streaming application with real-time latency constraints, FEC block sizes should be closer to the latency bound for the application (i.e., 100 packets).

7.4.4 Applying FEC to loss traces (using smaller FEC blocks)

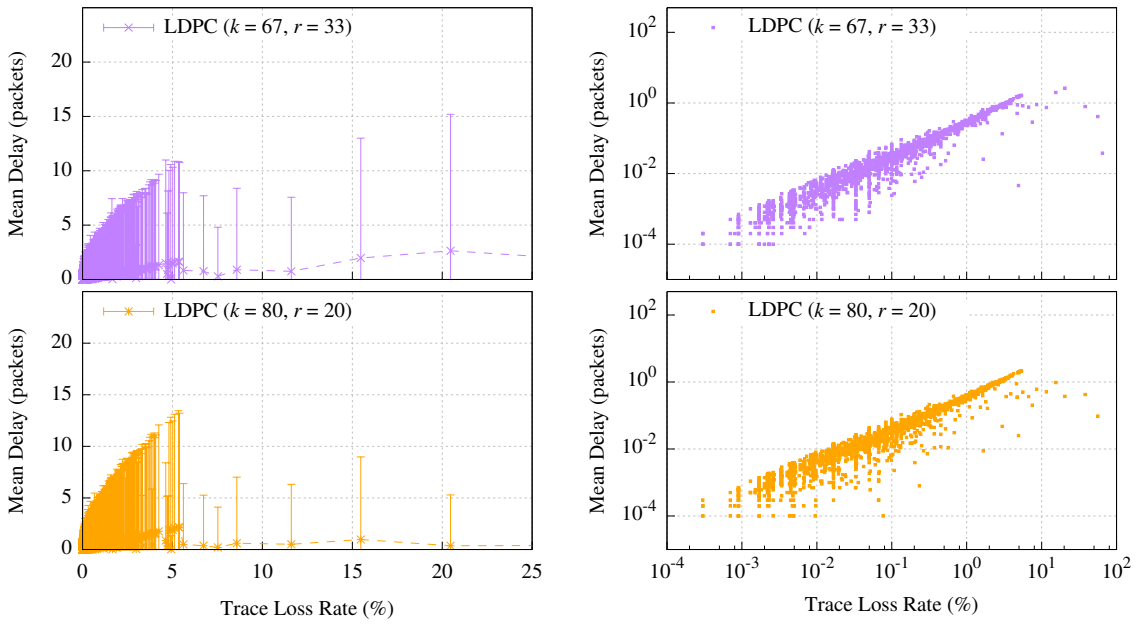
This section presents results using parameters with smaller block sizes, to reduce the worst-case delay and measure the effect on recovery performance. Although the RSE codes had similar recovery and delay performance to LDPC, since [134] recorded much higher CPU overhead for the RSE scheme, (due to the higher computational complexity of the Reed-Solomon algorithm), I focus on the LDPC codes. The parameters used were ($k = 67$, $r = 33$) and ($k = 80$, $r = 20$), with 50% and 25% overhead, respectively (since many traces have low loss rates, I also look at the effect of reducing FEC overhead).

Figure 7.7 shows the loss rate of each trace, plotted against the residual loss rate obtained after FEC recovery of that trace, as in Figure 7.4. This figure suggests that loss recovery of LDPC with smaller FEC block sizes is not too much worse than with the larger LDPC



(a) Linear scale (b) Log scale

Figure 7.7: Trace Loss vs. Residual Loss Rate (smaller FEC blocks)



(a) Linear scale (b) Log scale

Figure 7.8: Trace Loss vs. Mean FEC Delay (smaller FEC blocks)

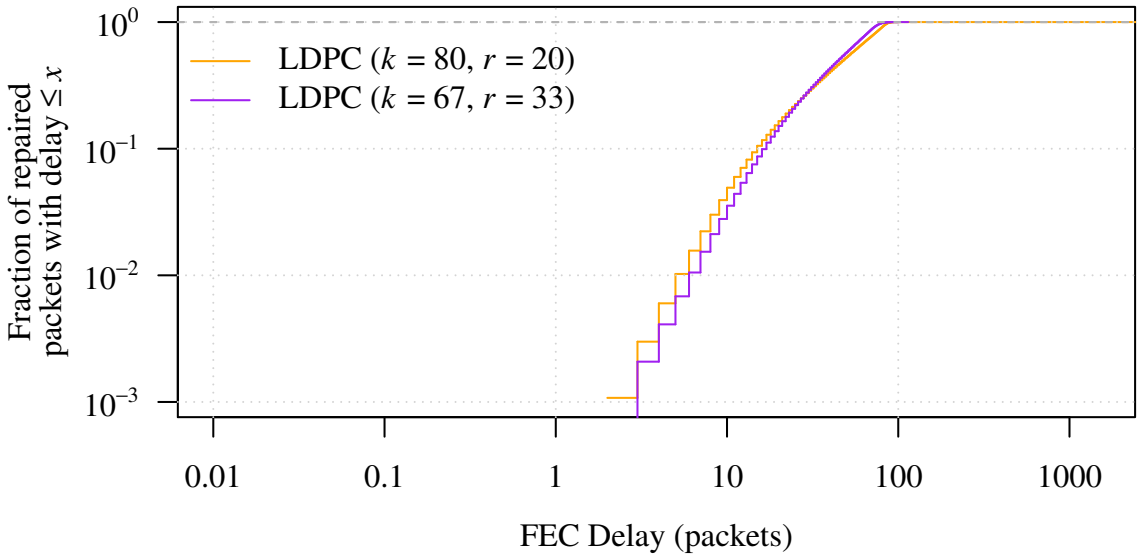


Figure 7.9: Cumulative Distribution of FEC Delays from Loss Traces (smaller FEC blocks)

FEC Scheme	Num traces	Percentage
LDPC ($k = 67$, $r = 33$)	3641	95.6%
LDPC ($k = 80$, $r = 20$)	3508	92.1%

Table 7.2: Number of Traces with 0% Residual Loss Rate (smaller FEC blocks)

($k = 170$, $r = 85$) scheme. Table 7.2 shows the fraction of fully repaired traces using these parameters with smaller FEC blocks; 96% for LDPC ($k = 67$, $r = 33$), and 92% for ($k = 80$, $r = 20$). The results show that reducing the worst-case delay to acceptable levels with smaller FEC blocks does not impact recovery too much; recall Table 7.1, which showed the LDPC ($k = 170$, $r = 85$) scheme fully repaired 98% of traces.

Figures 7.8 and 7.9 show that, as expected, delays are lower on average, and that the worst case FEC delay is never larger than 100 packets.

7.4.5 Summary

Considering the higher CPU overhead for the RSE scheme recorded in [134], the results of this section agree with the conclusions of [134] that LDPC codes are the most suitable choice among the schemes compared. However, the ($k = 170$, $r = 85$) parameters recommended in [134] result in latency that is too high for practical use (a delay bound of 200ms at 5Mb/s).

By applying parameters with a smaller FEC block size ($k = 67$, $r = 33$) and ($k = 80$, $r = 20$), the results show that the worst-case FEC delay is reduced, while not harming recovery performance too much.

It should also be noted that none of the FEC schemes recover *all* lost packets when the loss is too high. In such cases, it is necessary to use retransmission alongside FEC to recover loss, although doing so will add one round-trip time of latency, to account for the retransmission request. Note that infrastructure to support retransmission is likely already present in the system, to support rapid channel-change [212].

7.5 The Effect of Burstiness on FEC Performance

Comparing the results between random loss (Section 7.4.1) and the measured loss traces (Section 7.4.2), it is clear that performance is different. While most of the loss traces show loss rates less than 10%, FEC performance on the traces does not match that of random loss below 10%. The difference is due to the loss patterns in the traces, which exhibit bursty rather than uniform random loss, as described in Chapters 4 and 5. This section looks at metrics to define the burstiness of a trace, and investigates how well these metrics correlate with FEC performance.

7.5.1 Metrics for Burstiness

An intuitive choice for a burstiness metric is the *mean loss burst length*, the average length of a packet loss burst. This metric has been used in previous work on evaluating FEC performance [70, 148]. However, as discussed in Chapter 5, average loss burst lengths can be misleading, since some of the traces with the poorest performance have short mean burst lengths. The reason for this is that there are periods where many packets are lost in short bursts next to short bursts of received packets; while the loss bursts are short, the overall loss patterns are highly bursty.

To measure this effect, and better capture the loss burstiness in the traces, a different metric, β_{win}, is calculated. To calculate β_{win} for a trace, it is split into windows, and the number of lost packets and distinct loss bursts in each window are counted. In a particular window, if the number of lost packets exceeds a threshold N, or the number of loss bursts

exceeds M, that window is classified as bursty. β_{win} is defined as the fraction of windows classified as bursty, using this technique:

$$\beta_{win} = \frac{\textit{\#windows where (\#losses } > \textit{ N) or (\#bursts } > \textit{ M)}}{\textit{total \#windows}} \tag{7.2}$$

A window size of 16 packets is chosen, since this corresponds to the smallest of the block sizes of the FEC schemes being compared. Also, if the packets are being transmitted at 500 packets per second (as discussed in Section 7.4.3), the 16 packet window equates to 32ms, which is roughly equal to one frame of video playback at 29.97fps ($1/29.97 \approx 33.33$ms). $N = 4$ packets was chosen since the 99^{th} percentile of all loss burst lengths in the loss traces is 4, suggesting that loss bursts longer than this are unusual and should be recorded. The M parameter acts as a threshold for how bursty the windows should be; $M = 4$ packets was chosen since 4 loss bursts in a window of 16 suggests loss is reaching the point where repair is ineffective (as seen in Figure 7.2).

A similar metric for burstiness in voice-over-IP calls was proposed in Section 4.7.2 of RFC 3611 [68]. This metric uses a threshold G_{min} to divide the trace into periods of bursts and gaps, where a burst is defined as: "the longest sequence that (a) starts with a lost or discarded packet, (b) does not contain any occurrences of G_{min} or more consecutively received (and not discarded) packets, and (c) ends with a lost or discarded packet" [68].

In the following section, these three metrics are calculated for each of the loss traces, and correlated with FEC performance. The accuracy of the metrics of burstiness in predicting FEC performance is then compared.

7.5.2 Comparing Burstiness Metrics

Figure 7.10 shows the results of plotting the three burstiness metrics (mean burst length, RFC 3611 burstiness metric, and β_{win}), against the residual loss rate after FEC recovery. To assess the correlation between the burstiness metrics and the residual loss rate, Pearson's product-moment correlation coefficient was computed.

The results shown in the left panels demonstrate the poor performance of mean burst length as a predictor of FEC performance, with a wide range of residual loss rates resulting from traces with loss mean burst lengths, and a very low correlation value. The reason for this is as discussed before; most individual loss bursts are short, but when they are placed close together, they have a severe impact on FEC performance.

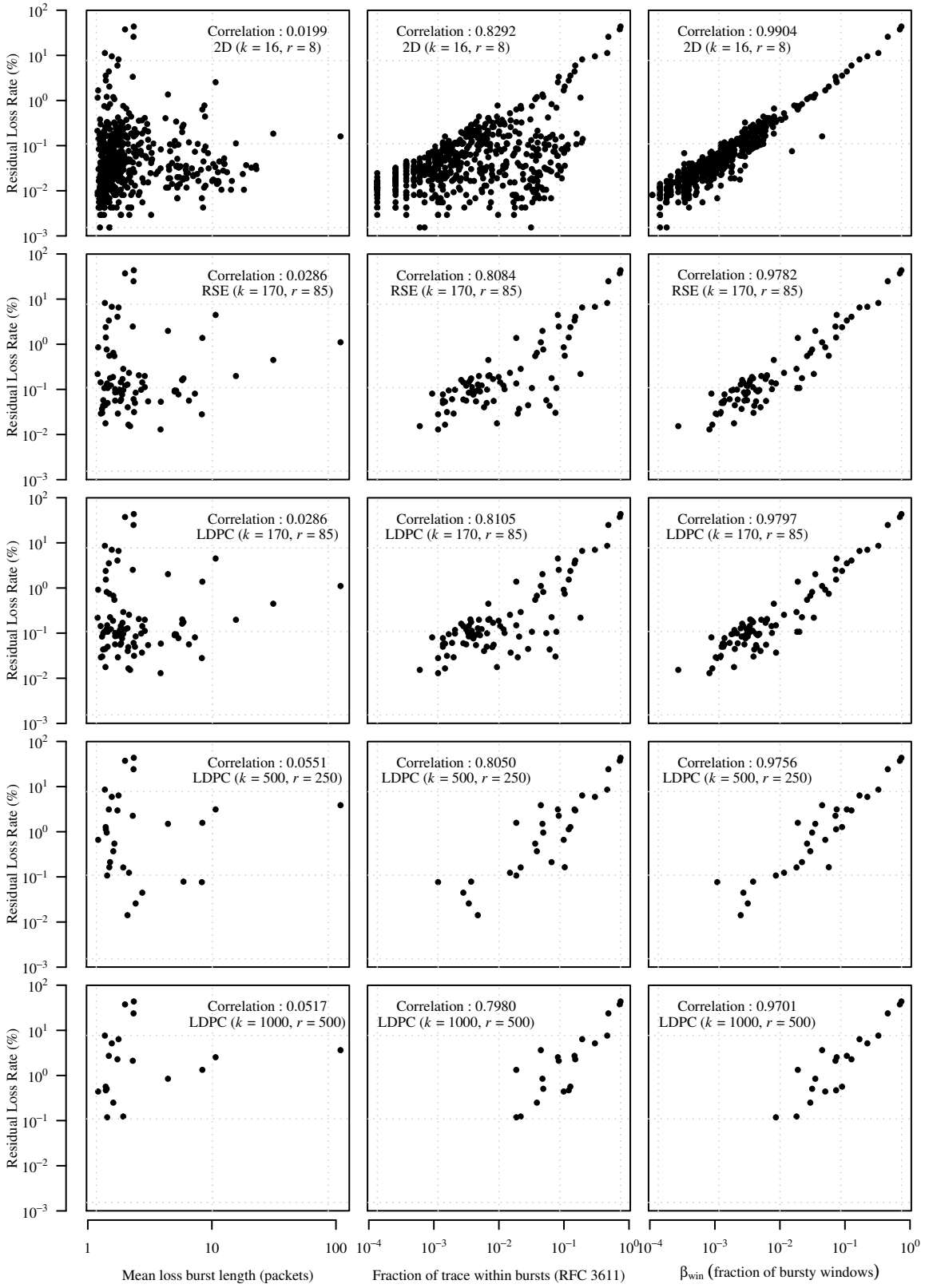


Figure 7.10: Correlating FEC Performance with Loss Burstiness Metrics. Left panels: mean loss burst length; Middle panels: RFC 3611 burstiness metric; Right panels: window-based burstiness metric β_{win}.

The middle panels of Figure 7.10 show better performance from the metric defined in RFC 3611, but these are not as good as the performance of β_{win}, shown in the right panels. The poorer performance is probably due to the RFC 3611 metric being too sensitive, and describing a whole period as bursty, even if FEC recovery might be possible. β_{win} is more accurate since it considers how much loss is tolerable. Adjusting the threshold length, G_{min} used in the RFC 3611 metric can achieve better performance, but the correlation with FEC performance is only stronger than β_{win} when G_{min} is ≤ 3, at which point the RFC 3611 metric does not distinguish between "good and poor quality periods", as was the aim in [68].

Figure 7.10 shows that β_{win} is a better metric for FEC performance than both mean loss burst length and the burstiness metric presented in RFC 3611, correlating strongly with the residual loss rate. This suggests that β_{win} might be used as a predictor of FEC performance (and hence as a crude metric for video quality), allowing the application to adapt to network conditions. An interesting topic for future work would be to adapt the code rate of FEC (i.e., the ratio of video packets to repair packets) using β_{win}, since this might allow adaptation to happen faster than waiting for the FEC recovery process.

7.6 Simulating FEC Performance with Packet Loss Models

Chapter 6 presented a two-level model, which was shown to be more accurate in modelling packet loss statistics than previous models. In this section, the accuracy of the two-level model is compared to the SGM and uniform random loss, in terms of the FEC performance 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., [70, 71, 149, 103, 208]), this section demonstrates that the two-level model is more suitable when simulating the packet loss patterns of residential broadband networks.

7.6.1 Comparing Models to Raw Data

To compare performance of the models, FEC simulations using eperftool were conducted, using the uniform random loss mode (as in Section 7.4.1), and traces generated by the two models, SGM and *ld*, SGM/2HMM/2HMM. To demonstrate the improved accuracy of the

two-level models of Chapter 6 using FEC as an example application, the FEC performance results obtained from these synthetic sequences is compared to the results obtained from the original raw data.

So, for each trace, we have the FEC performance measured from:

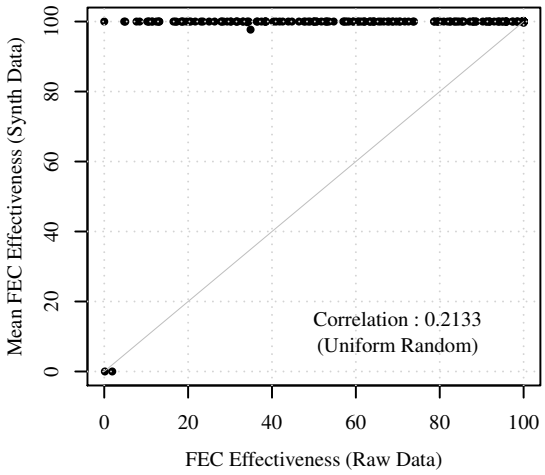
- the original raw trace (discussed in Section 7.4)
- uniform random data (with a loss probability derived from the raw trace)
- the SGM synthetic sequence (with model parameters derived from the raw trace)
- the *ld*, SGM/2HMM/2HMM synthetic sequence (with model parameters derived from the raw trace)

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

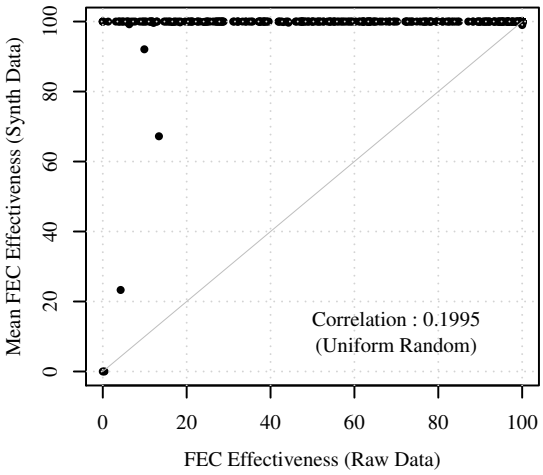
7.6.2 Improved Accuracy of Two-Level Models

Figure 7.11 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, since it more clearly demonstrates the differences between performance of the models.

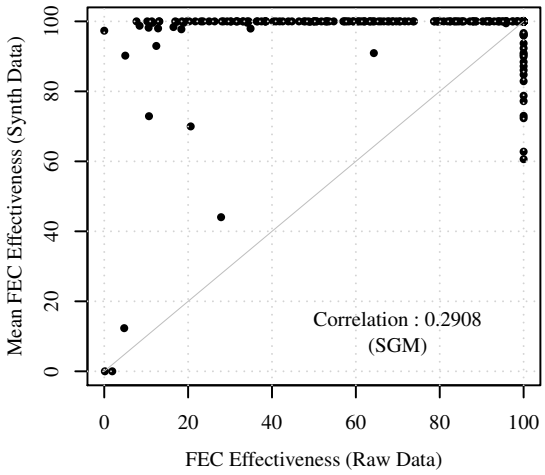
From each raw data trace, ten synthetic sequences were generated (within eperftool for uniform random loss model, and using the approach used in Chapters 5 and 6 for the SGM and *ld*, SGM/2HMM/2HMM models). Each point on the scatter plots represents a measurement trace, and shows the FEC effectiveness obtained from the raw data (x-axis), and the mean of the FEC effectiveness obtained from the ten 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.



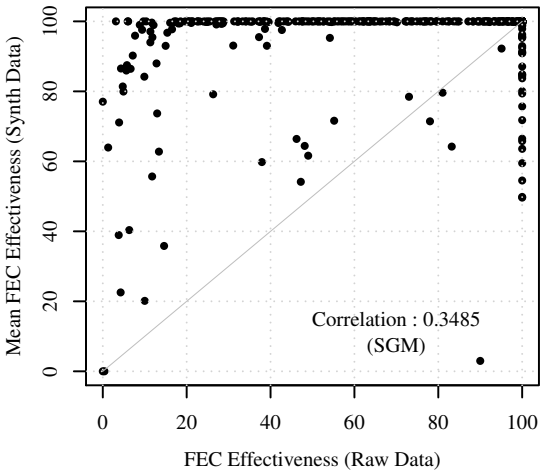
(a) Uniform Random; LDPC ($k = 67$, $r = 33$)



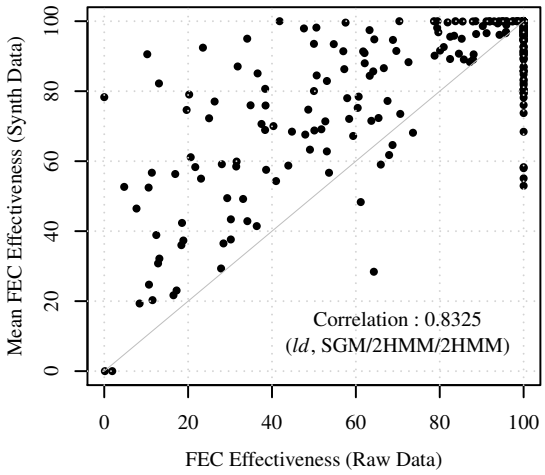
(b) Uniform Random; LDPC ($k = 80$, $r = 20$)



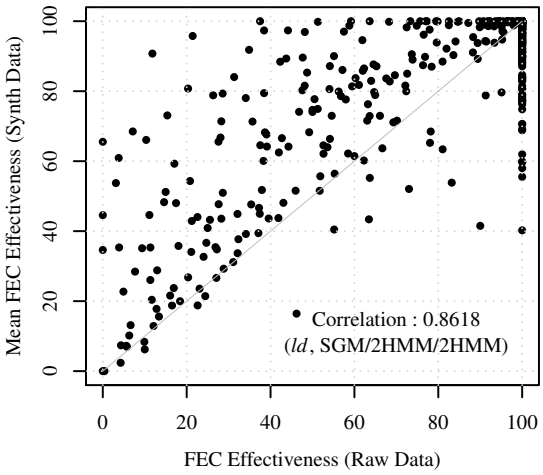
(c) SGM; LDPC ($k = 67$, $r = 33$)



(d) SGM; LDPC ($k = 80$, $r = 20$)



(e) *ld*, SGM/2HMM/2HMM; LDPC ($k = 67$, $r = 33$)



(f) *ld*, SGM/2HMM/2HMM; LDPC ($k = 80$, $r = 20$)

Figure 7.11: FEC Performance on Raw and Synthetic Data

The top row of Figure 7.11 shows the results from uniform random loss, the middle row shows results from the SGM model, and the bottom row shows results from the *ld*, SGM/2HMM/2HMM model. The left column shows results from the LDPC ($k = 67$, $r = 33$) scheme, and the right column shows results from LDPC ($k = 80$, $r = 20$).

From Figures 7.11a and 7.11b, 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 discussed in Section 7.5, but is visible in the line present across the top of Figures 7.11a and 7.11b, corresponding to the case where the synthetic sequence resulted in 100% FEC recovery, while the original raw data sequence was not fully recovered. Figures 7.11c and 7.11d 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 Figures 7.11e and 7.11f, tends to generate synthetic sequences that show FEC performance more similar to the original, with the points clustering closer 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 and 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 $\sim$0.2 (for uniform random loss) and $\sim$0.3 (for the SGM).

7.6.3 Summary

This section has applied the two-level model presented in Chapter 6 to simulation of FEC recovery, comparing its accuracy in terms of FEC performance using the real data. Compared to the previous models (both uniform random loss, and the widely used SGM model), the two-level model gives more accurate FEC performance results, suggesting it is more suitable for simulation of FEC performance on video streaming over residential broadband networks.

7.7 Discussion & Summary

In this chapter, I have evaluated the performance of three AL-FEC schemes; 2D, RSE, and LDPC-Staircase codes, under loss conditions measured for IPTV-like traffic on residential broadband networks. I have shown that since the measured loss is bursty, the FEC performance does not simply follow the input loss rate as suggested in [134], but is also affected by the loss burstiness. Looking at burstiness, I have also demonstrated that the mean burst length is not sufficient to predict FEC performance, since it does not consider the periods of short, clustered loss bursts. I have presented a window-based loss burstiness metric, β_{win}, which better correlates with residual loss rate than the burst loss metric presented in [68].

I have found that LDPC-Staircase codes give the best trade-off between recovery, latency and computational cost (a finding consistent with that of [134]), although the worst-case delays incurred when ($k = 170$, $r = 85$) are too high to be used for Internet streaming. However, the results show that smaller block sizes can be used without harming recovery performance too much. Since there are periods of loss that are unrecoverable using FEC (even with high levels of overhead), it is also clear that retransmission-based recovery should also be used to recover from these periods of loss.

Finally, I have used the application of FEC performance evaluation as a case study to validate the improved accuracy of the two-level model presented in Chapter 6. The results show that future work on simulating FEC performance on networks with bursty packet loss behaviour (such as the residential broadband networks I have measured) would benefit from using the two-level model, rather than existing approaches such as the SGM. Another potential use for the new models is in adaptive FEC, where the receiver uses the classification and modelling mechanisms presented in Chapter 6 as input to a control loop that determines which FEC parameters should be used. 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.

Chapter 8

Conclusions and Future Work

Video streaming has become a ubiquitous application in today's Internet, and transmitting high quality video to home users is increasingly important. However, the performance characteristics of high quality video streaming over these residential users' access links (typically DSL or Cable) are not well understood; therefore, accurately simulating network performance or evaluating new video applications can be difficult.

In this dissertation, I have presented my contributions to solving this problem. I have collected and published packet level measurements of streaming high bit-rate IPTV-like traffic across the Internet to residential users, giving new insight into the packet loss and delay performance of video streaming applications. I have used these measurements to show that existing models for packet loss simulation can be inaccurate, and introduced a new model that allows more accurate simulation of packet loss. I have also shown the differences in forward error correction performance between previous studies (which made assumptions about network conditions) using real measurement data, and explained these differences.

This chapter is structured as follows. Section 8.1 recaps the thesis statement presented in Chapter 1, and describes how it has been addressed. Section 8.2 lists the contributions of this dissertation. Section 8.3 outlines directions for future work, describing improvements that can be made to the work presented here, and exploring how it fits into other areas of research. Section 8.4 gives a summary of this chapter, and concludes the dissertation.

8.1 Thesis Statement

In this section, I repeat my thesis statement from Section 1.1, and indicate how it has been addressed, with reference to the preceding chapters. The thesis statement is restated as follows:

> I assert that packet loss simulation for streaming over residential networks is inaccurate because existing models do not capture the bursty nature of packet loss on these networks. To demonstrate this assertion, I will show the inaccuracy of existing Markov chain models under bursty packet loss by testing their goodness-of-fit against real packet loss data. Then, to overcome the limitations of existing models, I will develop a new model that more accurately models bursty packet loss.
>
> As a first step towards demonstrating this assertion, I will capture the packet loss and delay characteristics of residential networks by performing new measurements. Then, using these measurements I will:
>
> - Evaluate existing models for simulating packet loss, and show that these models can be inaccurate since they generate synthetic sequences that have different properties to the real data. This is important since when the models are used for simulation, the results obtained will not reflect reality. Moreover, I will explain what types of network performance cannot be accurately represented by existing models.
> - Develop a new, more accurate model for packet loss simulation that explicitly models changes in packet loss and delay behaviour. This new model will demonstrate that by better understanding the network performance, simulation accuracy can be improved.
> - Evaluate the performance of forward error correction (FEC) schemes for Internet video under real packet loss conditions, and show that more realistic simulation of FEC performance is possible by using the new packet loss model. This will demonstrate that the new model can be applied to a realistic application, and show the benefits of doing so.

To improve our understanding of the performance of video streaming over residential networks, and improve the accuracy of packet loss simulation in this context, I did the following.

In Chapter 3, I collected a dataset of streaming high bit-rate IPTV-like RTP traffic towards receivers connected to the Internet using residential ADSL and Cable links. The chapter describes the process involved in choosing experimental parameters, as well as practical considerations for conducting active measurements to residential users.

Chapter 4 presented a high-level analysis of these measurements, showing that the packet loss and delay characteristics vary significantly between different links, and can fluctuate over time. Indeed, characteristics can vary over long time scales between the two years of the measurements, over medium time scales between different times of day, and even over short time scales within individual traces. Another finding of Chapter 4 was the clear difference between the performance of capacity estimation techniques on ADSL and Cable, suggesting that although there is not a clear distinction for packet loss and delay characteristics, the links are quite different, and might be distinguished using the packet-pair technique.

In Chapter 5, I used the packet level loss measurements to evaluate the accuracy of existing, widely used models for packet loss. Using a simulation-based approach, I demonstrated that the existing models do not capture the wide range of loss conditions present in the measurement data. In particular, the switches between periods of relatively infrequent packet loss and periods of bursty, correlated loss are not well-captured by existing models. I showed that in order to accurately model these loss behaviours, a new model needs to understand the underlying state of the network (i.e., whether it is currently congested or not), and the transitions between these.

Chapter 6 introduced such a model, which I designed to explicitly take into account the transitions between network states observed in the loss data. Using the same evaluation technique as in Chapter 5, I showed that the new model accurately models a larger number of traces than the previous models, producing loss patterns that more closely match the switches between periods of infrequent packet loss and bursty, correlated loss. Moreover, in terms of a range of summary statistics, I have shown that the new models have better "goodness-of-fit" than the previous models, demonstrating that they are more accurate.

Finally, in Chapter 7, I showed that by understanding the network better, more realistic performance evaluation of network applications can be conducted, presenting a case study

where the performance of forward error correction schemes was tested. I showed that the schemes perform quite differently under real packet loss conditions than previous studies have suggested, and demonstrated that the new packet loss model presented in Chapter 6 enables more realistic simulation.

8.2 Contributions

The contributions of this dissertation are as follows:

A new dataset of IPTV-like RTP streaming to residential Internet users.

I have conducted and published new packet level performance measurements of streaming high bit-rate RTP traffic towards residential receivers. High-level analysis of this dataset to compare access link types (i.e., ADSL and Cable) shows that the links can be quite different in terms of packet loss and delay behaviour, with some showing very predictable performance and others being quite variable (e.g., showing time-of-day variation). Moreover, the analysis shows no clear difference in loss and delay behaviour between ADSL and Cable, with individual links behaving differently from each other (e.g., some links show lots of congestive loss, or spikes in queueing delay during the evening, while other links of the same type do not). There is, however, a difference between the performance of packet pair capacity estimation, which seems quite accurate for ADSL (but not for Cable).

An evaluation study of existing Markov chain packet loss models.

This work used the measurements of packet loss to evaluate the accuracy of existing packet loss models (Simple Gilbert Model and Extended Gilbert Model, and Hidden Markov Models) on residential broadband links. I found that in some traces, the packet loss conditions were able to be modelled by these existing models, but that in a significant number of traces, the existing models performed poorly. A key feature of the traces where existing models are inaccurate is that there appear to be changes in state, where the loss behaviour switches into a mode of bursty, correlated loss. The problem with existing models is that they do not capture these state changes.

Introduction of a new two-level model for packet loss, using loss and delay.

This work developed a new two-level model to explicitly capture the state changes

that caused problems for the SGM, EGM and HMMs. The two-level model classifies periods of each trace into states ("uncongested", "core congestion", "edge congestion") according to the packet loss and delay behaviour, and estimates separately the model parameters for the periods of different states. The evaluation results of this new two-level model show that it improves on the accuracy for those traces that were not well-modelled before.

An evaluation study of FEC algorithms under packet loss on residential links.

This work demonstrated how a number of forward error correction schemes proposed for use in video streaming perform under real loss conditions measured from residential links. I found that the FEC schemes perform differently than previous studies have suggested due to burstiness in packet loss, and presented a new metric for packet loss burstiness, β_{win}, which is a good indicator of FEC performance, showing higher correlation with FEC recovery than existing metrics. I have also demonstrated the benefits of the two-level model introduced in Chapter 6, by showing that simulating FEC performance using this model gives a more accurate insight into FEC effectiveness than using uniform random loss, or the SGM model evaluated in Chapter 5.

8.3 Future Directions

The work in this dissertation has demonstrated how the evaluation of Internet video systems can be improved by better understanding the performance of residential networks, in terms of more accurate simulation (for developing new video techniques) and more realistic performance evaluation (for trying out the performance of existing techniques). However, the work presented in the preceding chapters provides a number of opportunities for future research, which will be described in this section.

8.3.1 Measurement

One obvious direction for future work is to expand the range of measurement data, to study the performance of different ISPs in different geographic locations. Due to the challenges in widespread deployments, integrating the study of video streaming performance into existing

large-scale Internet measurement projects such as [115, 204] may be the best opportunity to understand the larger picture of video streaming performance on the Internet as a whole.

Another approach is to use passive measurement of streaming performance in homes (e.g., using instrumented home gateways such as [205, 144]). This could be used with both real streaming traffic from actual Internet video applications, as well as with test traffic designed to measure the capabilities of the network. The use of intelligent end-points (either home gateways or set-top boxes) present an opportunity for "always-on" monitoring of network performance, and might be also augmented to monitor video quality.

8.3.2 Modelling

The two-level model presented in Chapter 6 demonstrated that by taking the network states into account, more accurate packet loss simulation is possible. However, further work into the model could focus on improved classification algorithms, inner packet loss models, or models for transitions between outer states, since each of these can be replaced separately. A related idea is that this approach for modelling different packet loss states (e.g., due to noise or congestion) could be extended to include other states. For example, a nice extension would be to incorporate the effect of wireless networks (e.g., home 802.11 or mobile wireless networks), which are likely to have different loss characteristics. Developing improved classifiers and inner packet loss models within the general two-level model framework presented in Chapter 6 would be a useful step in this direction.

Finally, another direction for modelling is to extend the packet loss models discussed in this work, to also provide a model for delay (e.g., a similar approach to [182]), to give useful input for evaluation and simulation of delay-sensitive mechanisms like algorithms for improving channel change, or optimising the size of de-jitter buffers. To do this, a more detailed understanding of the time-series behaviour of delay, and the relationship between loss would be required. For example, given the close relationship between loss and delay seen throughout this dissertation, the approach taken in [182] (where the packet delays are drawn from a Gamma distribution) is unlikely to be suitable, since the trends and correlations between the delay values, and between delays and losses, would be hard to capture. Therefore, any work in this direction needs to make a careful study of the delay behaviour; not just in isolation, but in terms of the rest of the system.

8.3.3 Applications

Other future work can include extending the work in Chapter 7 on FEC performance by developing adaptive FEC mechanisms that react to changes in network performance (according to the network classification made by the algorithms presented in Chapter 6). Such an approach could allow the ratio of video data to FEC to be adjusted, to use more FEC when conditions degrade (i.e., sacrificing video bit-rate for error resilience).

Another useful application for the models is for anomaly detection. Together with the "always-on" monitoring discussed in Section 8.3.1, this would allow receivers to continually monitor the network conditions, building a profile of the "typical" behaviour of the network (taking into account long-term effects and trends). Then, anomalous behaviour that does not conform to the expectations of the model might be reported, allowing the operators to more easily identify faults.

Since this dissertation has focused on the effect of network performance on RTP-based video streaming, another useful direction for future work would be to similarly study the effect of the network on HTTP-based streaming (e.g., DASH), since this is a particularly active area of research at present. Topics in this area might include understanding the effect of the network on throughput, retransmissions, and buffer occupancy at receivers, and distinguishing congestive and non-congestive losses. The packet loss model described in Chapter 6, combined with the capacity estimates presented in Chapter 4 might be used as a starting point to simulate TCP retransmissions and throughput under realistic conditions, although the accuracy of these models for this scenario would need to be verified. Since the mechanisms used in HTTP streaming are somewhat more complicated (including TCP retransmissions and congestion control, and application level rate adaptation and buffer management), understanding the effect of the network on these mechanisms will be beneficial. The techniques presented in this dissertation provide the foundations to do this.

8.4 Summary & Conclusions

In this dissertation, I have presented new insight into the network performance of video streaming applications as perceived by residential Internet users. Using new measurements of network performance, I have shown that existing models for packet loss are insufficient to express the full range of packet loss behaviours seen on real networks. By understanding the

network behaviour in-depth (i.e., the congestion state in different parts of the network) using the end-to-end packet loss and delay observations, I developed a new model that explicitly considers these network states. Recognising the relationship between packet level observations and trends (i.e., individual loss and delay values) and the overall state of the network is key to the new model. Finally, by understanding the packet loss behaviour of the network, I have also made a realistic evaluation of FEC performance, showing an example of how application behaviour is linked to that of the network.

The findings presented in this dissertation demonstrate the importance of understanding network performance, both for providing realistic simulation and evaluating the performance of existing applications. In doing so, I have improved packet loss simulation for video streaming to residential users, and provided a means for future work to realistically evaluate the performance of these applications.

Bibliography

[1] V. Adhikari, S. Jain, Y. Chen, and Z.-L. Zhang. How Do You "Tube"? In *SIGMETRICS '11: Proceedings of the 2011 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 137–138, San Jose, CA, USA, June 2011.

[2] R. J. Adler, R. E. Feldman, and M. S. Taqqu, editors. *A Practical Guide to Heavy Tails: Statistical Techniques and Applications*. Birkhaüser, 1998.

[3] V. Aggarwal, A. Feldmann, and C. Scheideler. Can ISPs and P2P Users Cooperate for Improved Performance? *ACM SIGCOMM Computer Communication Review*, 37(3):29–40, July 2007.

[4] S. Agrawal, C. N. Kanthi, K. V. M. Naidu, J. Ramamirthram, R. Rastogi, S. Satkin, and A. Srinivasan. Monitoring Infrastructure for Converged Networks and Services. *Bell Labs Technical Journal*, 12(2):63–78, August 2007.

[5] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. C. Begen. What Happens When HTTP Adaptive Streaming Players Compete for Bandwidth? In *NOSSDAV '12: Proceedings of the 22nd ACM SIGMM Workshop on Network and Operating Systems Support for Digital Audio & Video*, Toronto, Canada, June 2012.

[6] S. Akhshabi, A. C. Begen, and C. Dovrolis. An Experimental Evaluation of Rate Adaptation Algorithms in Adaptive Streaming over HTTP. In *MMSys '11: Proceedings of the 2nd Annual ACM SIGMM Conference on Multimedia Systems*, pages 157–168, San Jose, CA, USA, February 2010.

[7] S. Ali, A. Mathur, and H. Zhang. Measurement of Commercial Peer-To-Peer Live Video Streaming. In *Proceedings of Workshop in Recent Advances in Peer-to-Peer Streaming*, August 2006.

[8] G. Almes, S. Kalidindi, and M. Zekauskas. A One-way Delay Metric for IPPM. *Internet RFCs*, RFC 2679, September 1999.

[9] G. Almes, S. Kalidindi, and M. Zekauskas. A One-way Packet Loss Metric for IPPM. *Internet RFCs*, RFC 2680, September 1999.

[10] A. Aurelius, C. Lagerstedt, and M. Kihl. Streaming media over the Internet: Flow based analysis in live access networks. In *BMSB 2011: Proceedings of the IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, Nuremberg, Germany, June 2011.

[11] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz. A Comparison of Mechanisms for Improving TCP Performance over Wireless Links. *IEEE/ACM Transactions on Networking*, 5(6):756–769, December 1997.

[12] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, M. Stemm, and R. H. Katz. TCP Behavior of a Busy Internet Server: Analysis and Improvements. In *INFOCOM 1998: Proceedings of the 17th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 252–262, San Francisco, CA, USA, April 1998.

[13] G. Baltoglou, E. Karapistoli, and P. Periklis Chatzimisios. Real-World IPTV Network Measurements. In *ISCC 2011: Proceedings of the IEEE Symposium on Computers and Communications*, pages 830–835, Kerkyra, Greece, June 2011.

[14] N. Becerra Yoma, C. Busso, and I. Soto. Packet-loss modelling in IP networks with state-duration constraints. *IEE Proceedings-Communications*, 152(1):1–5, February 2005.

[15] A. Begen. RTP Payload Format for 1-D Interleaved Parity FEC. *Internet RFCs*, RFC 6015, October 2010.

[16] A. C. Begen. Error Control for IPTV over xDSL Networks. In *CCNC 2008: Proceedings of the 5th Annual IEEE Consumer Communications & Networking Conference*, Las Vegas, NV, USA, January 2008.

[17] A. C. Begen, N. Glazebrook, and W. Ver Steeg. A Unified Approach for Repairing Packet Loss and Accelerating Channel Changes in Multicast IPTV. In *CCNC 2009:*

Proceedings of the 6th Annual IEEE Consumer Communications & Networking Conference, pages 1–6, Las Vegas, NV, USA, January 2009.

[18] A. C. Begen, N. Glazebrook, and W. Ver Steeg. Reducing Channel Change Times in IPTV with Real-Time Transport Protocol. *IEEE Internet Computing*, 13(3):40–47, May/June 2009.

[19] A. C. Begen, C. Perkins, and J. Ott. On the Scalability of RTCP-Based Network Tomography for IPTV Services. In *CCNC 2010: Proceedings of the 7th Annual IEEE Consumer Communications & Networking Conference, Special Session on IPTV Toward Seamless Infotainment*, Las Vegas, NV, USA, January 2010.

[20] A. C. Begen, C. Perkins, and J. Ott. On the Use of RTP for Monitoring and Fault Isolation in IPTV. *IEEE Network*, 24(2):14–19, March/April 2010.

[21] J. S. Bendat and A. G. Piersol. *Random Data: Analysis and Measurement Procedures*. John Wiley & Sons, Third edition, 2000.

[22] S. Bhattacharyya. An Overview of Source-Specific Multicast (SSM). *Internet RFCs*, RFC 3569, July 2003.

[23] J. A. Bilmes. A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. Technical Report TR-97-021, International Computer Science Institute, April 1998.

[24] J. Bolot, S. Fosse-Parisis, and D. Towsley. Adaptive FEC-Based Error Control for Interactive Audio in the Internet. Submission to *IEEE INFOCOM*, March 1998.

[25] J.-C. Bolot. End-to-End Packet Delay and Loss Behavior in the Internet. In *SIGCOMM '93: Conference proceedings on Communications architectures, protocols and applications*, pages 289–298, San Francisco, CA, USA, September 1993.

[26] J.-C. Bolot, H. Crépin, and A. V. Garcia. Analysis of Audio Packet Loss in the Internet. In *NOSSDAV '95: Proceedings of the 5th International workshop on Network and Operating Systems Support for Digital Audio & Video*, pages 163–174, Durham, NH, USA, April 1995.

[27] J.-C. Bolot, T. Turletti, and I. Wakeman. Scalable Feedback Control for Multicast Video Distribution in the Internet. In *SIGCOMM '94: Proceedings of the conference on Communications architectures, protocols and applications*, pages 58–67, London, UK, August 1994.

[28] M. S. Borella. Measurements and Interpretation of Internet Packet Loss. *Journal of Communications and Networks*, 2(2):93–102, June 2000.

[29] C. Bovy, H. Mertodimedjo, G. Hooghiemstra, H. Uijterwaal, and P. Van Mieghem. Analysis of End-to-end Delay Measurements in Internet. In *PAM 2002: Proceedings of the Passive & Active Measurement Workshop*, Fort Collins, CO, USA, March 2002.

[30] J. M. Boyce and R. D. Gaglianello. Packet Loss Effects on MPEG Video Sent Over the Public Internet. In *MULTIMEDIA '98: Proceedings of the 6th ACM International Conference on Multimedia*, pages 181–190, Bristol, UK, September 1998.

[31] E. Brosh, S. A. Baset, D. Rubenstein, and H. Schulzrinne. The Delay-Friendliness of TCP. In *SIGMETRICS '08: Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 49–60, Annapolis, MD, USA, June 2008.

[32] M. Cha, G. Choudhury, J. Yates, A. Shaikh, and S. Moon. Case Study: Resilient Backbone Design for IPTV Services. In *Proceedings of the IPTV Workshop, International World Wide Web Conference*, Edinburgh, UK, May 2006.

[33] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon. I Tube, You Tube, Everybody Tubes: Analyzing the World's Largest User Generated Content Video System. In *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 1–14, San Diego, CA, USA, October 2007.

[34] M. Cha, P. Rodriguez, S. Moon, and J. Crowcroft. On Next-Generation Telco-Managed P2P TV Architectures. In *Proceedings of the 7th International Workshop on Peer-to-Peer Systems*, Tampa Bay, FL, USA, February 2008.

[35] Y.-F. Chen, Y. Huang, R. Jana, H. Jiang, M. Rabinovich, B. Wei, and Z. Xiao. When is P2P Technology Beneficial for IPTV Services? In *NOSSDAV '07: Proceedings of the*

17th International workshop on Network and Operating Systems Support for Digital Audio & Video, Urbana, IL, USA, June 2007.

[36] M. R. Chernick. *Bootstrap Methods: A Practitioner's Guide*. John Wiley & Sons, 1999.

[37] B.-Y. Choi, S. Moon, Z.-L. Zhang, K. Papagiannaki, and C. Diot. Analysis of point-to-point packet delay in an operational network. *Computer Networks*, 51(13):3812–3827, September 2007.

[38] Cisco. Delivering Video Quality in Your IPTV Deployment. White Paper, 2008.

[39] P. Coverdale, S. Möller, A. Raake, and A. Takahashi. Multimedia Quality Assessment Standards in ITU-T SG12. *IEEE Signal Processing Magazine*, 28(6):91–97, November 2011.

[40] D. Croce, T. En-Najjary, G. Urvoy-Keller, and E. W. Biersack. Fast Available Bandwidth sampling for ADSL links: rethinking the estimation for larger-scale measurements. In *PAM 2009: Proceedings of the 10th Passive and Active Measurement Conference*, pages 67–76, Seoul, South Korea, April 2009.

[41] D. Croce, M. Mellia, and E. Leonardi. The Quest for Bandwidth Estimation Techniques for Large-Scale Distributed Systems. *ACM SIGMETRICS Performance Evaluation Review*, 37(3):20–25, December 2009.

[42] M. Crovella and B. Krishnamurthy. *Internet Measurement: Infrastructure, Traffic, and Applications*. John Wiley & Sons, 2006.

[43] A. Cuadra-Sanchez. End-to-End Quality of Service Monitoring in Convergent IPTV Platforms. In *NGMAST '09: Proceedings of the 3rd International Conference on Next Generation Mobile Applications, Services and Technologies*, pages 303–308, Cardiff, UK, September 2009.

[44] M. Cunche and V. Roca. Optimizing the Error Recovery Capabilities of LDPC-staircase Codes Featuring a Gaussian Elimination Decoding Scheme. In *SPSC 2008: Proceedings of the 10th International Workshop on Signal Processing for Space Communications*, pages 1–7, Rhodes, Greece, October 2008.

[45] E. de Souza e Silva, R. M. M. Leão, and R. R. Muntz. Performance Evaluation with Hidden Markov Models. In *Performance Evaluation of Computer and Communication Systems. Milestones and Future Challenges*, pages 112–128. Springer, 2011.

[46] L. De Vito, S. Rapuano, and L. Tomaciello. One-Way Delay Measurement: State of the Art. *IEEE Transactions on Instrumentation and Measurement*, 57(12):2742–2750, December 2008.

[47] N. Degrande, D. De Vleeschauwer, and K. Laevens. Protecting IPTV Against Packet Loss: Techniques and Trade-Offs. *Bell Labs Technical Journal*, 13(1):35–51, May 2008.

[48] C. Demichelis and P. Chimento. IP Packet Delay Variation Metric for IP Performance Metrics (IPPM). *Internet RFCs*, RFC 3393, November 2002.

[49] M. Dischinger, A. Haeberlen, K. P. Gummadi, and S. Saroiu. Characterizing Residential Broadband Networks. In *IMC '07: Proceedings of the 7th ACM SIGCOMM Internet Measurement Conference*, pages 43–56, San Diego, CA, USA, October 2007.

[50] F. Dobrian, A. Awan, D. Joseph, A. Ganjam, J. Zhan, V. Sekar, I. Stoica, and H. Zhang. Understanding the Impact of Video Quality on User Engagement. In *SIGCOMM '11: Proceedings of the ACM SIGCOMM 2011 Conference on Data Communication*, pages 362–373, Toronto, Canada, August 2011.

[51] R. Doverspike, G. Li, K. Oikonomou, K. K. Ramakrishnan, and D. Wang. IP Backbone Design for Multimedia Distribution: Architecture and Performance. In *INFOCOM 2007: Proceedings of the 26th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1523–1531, Anchorage, AK, USA, May 2007.

[52] R. Doverspike, G. Li, K. N. Oikonomou, K. K. Ramakrishnan, R. K. Sinha, D. Wang, and C. Chase. Designing a Reliable IPTV Network. *IEEE Internet Computing*, 13(3):15–22, May/June 2009.

[53] C. Dovrolis, P. Ramanathan, and D. Moore. What do packet dispersion techniques measure? In *INFOCOM 2001: Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 905–914, Anchorage, AK, USA, April 2001.

[54] C. Dovrolis, P. Ramanathan, and D. Moore. Packet-Dispersion Techniques and a Capacity-Estimation Methodology. *IEEE/ACM Transactions on Networking*, 12(6):963–977, December 2004.

[55] A. B. Downey. Evidence for long-tailed distributions in the Internet. In *IMW '01: Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 229–240, San Francisco, CA, USA, November 2001.

[56] F. P. Duarte, E. de Souza e Silva, and D. Towsley. An adaptive FEC algorithm using Hidden Markov Chains. *ACM SIGMETRICS Performance Evaluation Review*, 31(2):11–13, September 2003.

[57] N. Duffield, A. Morton, and J. Sommers. Loss Episode Metrics for IP Performance Metrics (IPPM). *Internet RFCs*, RFC 6534, May 2012.

[58] DVB. DVB Application Layer FEC Evaluations. DVB BlueBook A115 - TM 3783, May 2007.

[59] E. O. Elliott. Estimates of Error Rates for Codes on Burst-Noise Channels. *Bell System Technical Journal*, 42(5):1977–1997, September 1963.

[60] M. Ellis and C. Perkins. Packet Loss Characteristics of IPTV-like Traffic on Residential Links. In *CCNC 2010: Proceedings of the 7th Annual IEEE Consumer Communications & Networking Conference, Workshop on Emerging Internet Video Technologies*, Las Vegas, NV, USA, January 2010.

[61] M. Ellis, C. Perkins, and D. P. Pezaros. End-to-End and Network-Internal Measurements of Real-Time Traffic to Residential Users. In *MMSys '11: Proceedings of the 2nd Annual ACM SIGMM Conference on Multimedia Systems*, San Jose, CA, USA, February 2011.

[62] M. Ellis, D. P. Pezaros, T. Kypraios, and C. Perkins. Modelling Packet Loss in RTP-based Streaming Video for Residential Users. In *LCN 2012: Proceedings of the 37th Annual IEEE Conference on Local Computer Networks*, Clearwater, FL, USA, October 2012.

[63] M. Ellis, D. P. Pezaros, and C. Perkins. Performance Analysis of AL-FEC for RTP-based Streaming Video Traffic to Residential Users. In *PV 2012: Proceedings of the 19th International Packet Video Workshop*, Munich, Germany, May 2012.

[64] ETSI. Digital Video Broadcasting (DVB); Transport of MPEG-2 TS Based DVB Services over IP Based Networks. European Telecommunications Standards Institute, August 2009.

[65] J. Evans, A. C. Begen, J. Greengrass, and C. Filsfils. Toward Lossless Video Transport. *IEEE Internet Computing*, 15(6):48–57, November/December 2011.

[66] M. Filippone and G. Sanguinetti. Information Theoretic Novelty Detection. *Pattern Recognition*, 43(3):805–814, March 2010.

[67] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot. Packet-Level Traffic Measurements from the Sprint IP Backbone. *IEEE Network*, 17(6):6–16, November/December 2003.

[68] T. Friedman, R. Caceres, and A. Clark. RTP Control Protocol Extended Reports (RTCP XR). *Internet RFCs*, RFC 3611, November 2003.

[69] B. D. Fritchman. A Binary Channel Characterization Using Partitioned Markov Chains. *IEEE Transactions on Information Theory*, 13(2):221–227, April 1967.

[70] P. Frossard. FEC Performance in Multimedia Streaming. *IEEE Communications Letters*, 5(3):122–124, March 2001.

[71] P. Frossard and O. Verscheure. Joint Source/FEC Rate Selection for Quality-Optimal MPEG-2 Video Delivery. *IEEE Transactions on Image Processing*, 10(12):1815–1825, December 2001.

[72] J. Gettys and K. Nichols. Bufferbloat: Dark Buffers in the Internet. *Communications of the ACM*, 55(1):57–65, January 2012.

[73] E. Gilbert. Capacity of a Burst-Noise Channel. *Bell System Technical Journal*, 39(5):1253–1265, September 1960.

[74] J. Greengrass, J. Evans, and A. C. Begen. Not All Packets Are Equal, Part 1: Streaming Video Coding and SLA Requirements. *IEEE Internet Computing*, 13(1):70–75, January/February 2009.

[75] C. D. Guerrero and M. A. Labrador. On the applicability of available bandwidth estimation techniques and tools. *Computer Communications*, 33(1):11–22, January 2010.

[76] C. D. Guerrero and M. A. Labrador. Traceband: A fast, low overhead and accurate tool for available bandwidth estimation and monitoring. *Computer Networks*, 54(6):977–990, April 2010.

[77] H. Haddadi. *Topological Characteristics of IP Networks*. PhD thesis, University College London, November 2008.

[78] G. Haßlinger and O. Hohlfeld. The Gilbert-Elliott Model for Packet Loss in Real Time Services on the Internet. In *Proceedings of the 14th GI/ITG Conference on Measurement, Modeling and Evaluation of Computer and Communication Systems*, pages 269–283, Dortmund, Germany, April 2008.

[79] F. V. Hecht, T. Bocek, R. G. Clegg, R. Landa, D. Hausheer, and B. Stiller. LiveShift: Mesh-Pull P2P Live and Time-Shifted Video Streaming. In *LCN 2011: Proceedings of the 36th Annual IEEE Conference on Local Computer Networks*, Bonn, Germany, October 2011.

[80] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross. A Measurement Study of a Large-Scale P2P IPTV System. *IEEE Transactions on Multimedia*, 9(8):1672–1687, December 2007.

[81] D. Hoffman, G. Fernando, V. Goyal, and M. Civanlar. RTP Payload Format for MPEG1/MPEG2 Video. *Internet RFCs*, RFC 2250, January 1998.

[82] O. Hohlfeld. Statistical Error Model to Impair an H.264 Decoder. Master's thesis, Technische Universität Darmstadt, March 2008.

[83] G. Hooghiemstra and P. Van Mieghem. Delay Distributions on Fixed Internet Paths. Technical Report 20011031, Delft University of Technology, Delft, The Netherlands, October 2001.

[84] K. Imran, M. Mellia, and M. Meo. Measurements of Multicast Television over IP. In *LANMAN 2007: Proceedings of the 15th IEEE Workshop on Local & Metropolitan Area Networks*, pages 170–175, Princeton, NJ, USA, June 2007.

[85] ISO. Generic Coding of Moving Pictures and Associated Audio Information: Video. ISO/IEC 13818-2:2000, December 2000.

[86] ISO. Generic Coding of Moving Pictures and Associated Audio Information: Systems. ISO/IEC 13818-1:2007, October 2007.

[87] ISO. Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats. ISO/IEC 23009-1:2012, 2012.

[88] ITU-T. Methods for subjective determination of transmission quality. International Telecommunication Union, Telecommunication Standardization Sector, August 1996. Rec. ITU-T P.800.

[89] ITU-T. Advanced video coding for generic audiovisual services. International Telecommunication Union, Telecommunication Standardization Sector, November 2007. Rec. ITU-T H.264.

[90] ITU-T. Third-generation transmission systems for interactive cable television services – IP cable modems: MAC and Upper Layer protocols. International Telecommunication Union, Telecommunication Standardization Sector, July 2007. Rec. ITU-T J.222.2.

[91] ITU-T. Third-generation transmission systems for interactive cable television services – IP cable modems: Physical layer specification. International Telecommunication Union, Telecommunication Standardization Sector, July 2007. Rec. ITU-T J.222.1.

[92] ITU-T. Requirements for the support of IPTV services. International Telecommunication Union, Telecommunication Standardization Sector, January 2009. Rec. ITU-T Y.1901.

[93] M. Jain and C. Dovrolis. Ten Fallacies and Pitfalls on End-to-End Available Bandwidth Estimation. In *IMC '04: Proceedings of the 4th ACM SIGCOMM Internet Measurement Conference*, pages 272–277, Taormina, Sicily, Italy, October 2004.

[94] M. Jain and C. Dovrolis. Path selection using available bandwidth estimation in overlay-based video streaming. *Computer Networks*, 52(12):2411–2418, 2008.

[95] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley. Measurement and Classification of Out-of-Sequence Packets in a Tier-1 IP Backbone. *IEEE/ACM Transactions on Networking*, 15(1):54–66, February 2007.

[96] V. Janardhan and H. Schulzrinne. Peer assisted VoD for set-top box based IP network. In *P2P-TV '07: Proceedings of the 2007 workshop on Peer-to-peer streaming and IP-TV*, pages 335–339, Kyoto, Japan, August 2007.

[97] A. Jean-Marie, Y. Calas, and T. Alemu. On the compromise between burstiness and frequency of events. *Performance Evaluation*, 62(1-4):382–399, October 2005.

[98] P. Ji, B. Liu, D. Towsley, Z. Ge, and J. Kurose. Modeling frame-level errors in GSM wireless channels. *Performance Evaluation*, 55(1-2):165–181, January 2004.

[99] P. Ji, B. Liu, D. Towsley, and J. Kurose. Modeling Frame-level Errors in GSM Wireless Channels. In *GLOBECOM 2002: Proceedings of the IEEE Global Telecommunications Conference*, pages 2483–2487, Taipei, Taiwan, November 2002.

[100] W. Jiang and H. Schulzrinne. Modeling of Packet Loss and Delay and Their Effect on Real-Time Multimedia Service Quality. In *NOSSDAV '00: Proceedings of the 10th International workshop on Network and Operating Systems Support for Digital Audio & Video*, Chapel Hill, NC, USA, June 2000.

[101] D. Joumblatt, R. Teixeira, J. Chandrashekar, and N. Taft. HostView: Annotating end-host performance measurements with user feedback. *ACM SIGMETRICS Performance Evaluation Review*, 38(3):43–48, December 2010.

[102] D. Joumblatt, R. Teixeira, J. Chandrashekar, and N. Taft. Performance of Networked Applications: The Challenges in Capturing the User's Perception. In *W-MUST '11: Proceedings of the 2011 ACM SIGCOMM Workshop on Measurements Up the Stack*, pages 37–42, Toronto, Canada, August 2011.

[103] S.-R. Kang and D. Loguinov. Modeling Best-Effort and FEC Streaming of Scalable Video in Lossy Network Channels. *IEEE/ACM Transactions on Networking*, 15(1):187–200, February 2007.

[104] R. Kapoor, L.-J. Chen, L. Lao, M. Gerla, and M. Y. Sanadidi. CapProbe: A Simple and Accurate Capacity Estimation Technique. In *SIGCOMM '04: Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 67–78, Portland, OR, USA, August 2004.

[105] K. Kerpez, D. Waring, G. Lapiotis, J. B. Lyles, and R. Vaidyanathan. IPTV Service Assurance. *IEEE Communications Magazine*, 44(9):166–172, September 2006.

[106] H. Khlifi and J.-C. Grégoire. Low-complexity offline and online clock skew estimation and removal. *Computer Networks*, 50(11):1872–1884, August 2006.

[107] L. Kleinrock and W. E. Naylor. On Measured Behavior of the ARPA Network. In *Proceedings of the National Computer Conference*, pages 767–780, Chicago, IL, USA, May 1974.

[108] J. Klotz. Statistical inference in Bernoulli trials with dependence. *The Annals of Statistics*, 1(2):373–379, March 1973.

[109] K. Kobayashi, A. Ogawa, S. Casner, and C. Bormann. RTP Payload Format for DV (IEC 61834) Video. *Internet RFCs*, RFC 3189, January 2002.

[110] T. Kohno, A. Broido, and K. Claffy. Remote Physical Device Fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2(2):93–108, April-June 2005.

[111] A. Konrad, B. Y. Zhao, and A. D. Joseph. A Markov-Based Channel Model Algorithm for Wireless Networks. *Wireless Networks*, 9(3):189–199, May 2003.

[112] A. Konrad, B. Y. Zhao, and A. D. Joseph. Determining model accuracy of network traces. *Journal of Computer and System Sciences*, 72(7):1156–1171, November 2006.

[113] R. Koodli and R. Ravikanth. One-way Loss Pattern Sample Metrics. *Internet RFCs*, RFC 3357, August 2002.

[114] R. Kooij, K. Ahmed, and K. Brunnström. Perceived Quality of Channel Zapping. In *Proceedings of the 5th IASTED International Conference on Communication Systems and Networks*, pages 155–158, Palma de Mallorca, Spain, August 2006.

[115] C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson. Netalyzr: Illuminating The Edge Network. In *IMC '10: Proceedings of the 10th Usenix/ACM SIGCOMM Internet Measurement Conference*, pages 246–259, Melbourne, Australia, November 2010.

[116] T. Kupka, P. Halvorsen, and C. Griwodz. An Evaluation of Live Adaptive HTTP Segment Streaming Request Strategies. In *LCN 2011: Proceedings of the 36th Annual IEEE Conference on Local Computer Networks*, pages 608–616, Bonn, Germany, October 2011.

[117] R. Kuschnig, I. Kofler, and H. Hellwagner. An Evaluation of TCP-based Rate-Control Algorithms for Adaptive Internet Streaming of H.264/SVC. In *MMSys '10: Proceedings of the 1st Annual ACM SIGMM Conference on Multimedia Systems*, pages 157–168, Phoenix, AZ, USA, February 2010.

[118] R. Kuschnig, I. Kofler, and H. Hellwagner. Improving Internet Video Streaming Performance by Parallel TCP-based Request-Response Streams. In *CCNC 2010: Proceedings of the 7th Annual IEEE Consumer Communications & Networking Conference*, Las Vegas, NV, USA, January 2010.

[119] J. Lacan, V. Roca, J. Peltotalo, and S. Peltotalo. Reed-Solomon Forward Error Correction (FEC) Schemes. *Internet RFCs*, RFC 5510, April 2009.

[120] K. Lakshminarayanan and V. N. Padmanabhan. Some Findings on the Network Performance of Broadband Hosts. In *IMC '03: Proceedings of the 3rd ACM SIGCOMM Internet Measurement Conference*, pages 45–50, Miami Beach, FL, USA, October 2003.

[121] S. Lederer, C. Müller, and C. Timmerer. Dynamic Adaptive Streaming over HTTP Dataset. In *MMSys '12: Proceedings of the 3rd Annual ACM SIGMM Conference on Multimedia Systems*, pages 89–94, Chapel Hill, NC, USA, February 2012.

[122] S. Lederer, C. Müller, and C. Timmerer. Towards Peer-Assisted Dynamic Adaptive Streaming over HTTP. In *PV 2012: Proceedings of the 19th International Packet Video Workshop*, pages 161–166, Munich, Germany, May 2012.

[123] Y. Li, Y. Zhang, and R. Yuan. Measurement and Analysis of a Large Scale Commercial Mobile Internet TV System. In *IMC '11: Proceedings of the 11th Usenix/ACM*

SIGCOMM Internet Measurement Conference, pages 209–224, Berlin, Germany, November 2011.

[124] Z. Li, X. Zhu, A. C. Begen, and B. Girod. Peer-Assisted Packet Loss Repair for IPTV Video Multicast. In *MM '09: Proceedings of the 17th ACM international conference on Multimedia*, pages 401–410, Beijing, China, October 2009.

[125] J. Liu and M. Crovella. Using Loss Pairs to Discover Network Properties. In *IMW '01: Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 127–138, San Francisco, CA, USA, November 2001.

[126] J. Liu, I. Matta, and M. E. Crovella. End-to-End Inference of Loss Nature in a Hybrid Wired/Wireless Environment. In *WiOpt'03: Proceedings of Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, Sophia-Antipolis, France, March 2003.

[127] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang. A Case for a Coordinated Internet Video Control Plane. In *SIGCOMM '12: Proceedings of the ACM SIGCOMM 2012 Conference on Data Communication*, Helsinki, Finland, August 2012.

[128] D. Loguinov and H. Radha. End-to-End Internet Video Traffic Dynamics: Statistical Study and Analysis. In *INFOCOM 2002: Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 723–732, New York, NY, USA, June 2002.

[129] M. Luby, T. Stockhammer, and M. Watson. Application Layer FEC in IPTV Services. *IEEE Communications Magazine*, 46(5):94–101, May 2008.

[130] A. Mahimkar, Z. Ge, A. Shaikh, J. Wang, J. Yates, Y. Zhang, and Q. Zhao. Towards Automated Performance Diagnosis in a Large IPTV Network. In *SIGCOMM '09: Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication*, pages 231–242, Barcelona, Spain, August 2009.

[131] G. Maier, A. Feldmann, V. Paxson, and M. Allman. On Dominant Characteristics of Residential Broadband Internet Traffic. In *IMC '09: Proceedings of the 9th*

Usenix/ACM SIGCOMM Internet Measurement Conference, pages 90–102, Chicago, IL, USA, November 2009.

[132] E. Mammi, G. Russo, and A. Neri. Evaluation of AL-FEC performance for IP television services QoS. In *Proceedings of SPIE*, volume 7529, 75290X, San Jose, CA, USA, January 2010.

[133] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, Y. Ganjali, and C. Diot. Characterization of Failures in an Operational IP Backbone Network. *IEEE/ACM Transactions on Networking*, 16(4):749–762, August 2008.

[134] K. Matsuzono, J. Detchart, M. Cunche, V. Roca, and H. Asaeda. Performance Analysis of a High-Performance Real-Time Application with Several AL-FEC Schemes. In *LCN 2010: Proceedings of the 35th Annual IEEE Conference on Local Computer Networks*, pages 1–7, Denver, CO, USA, October 2010.

[135] J. McDougall, J. Joseph, Y. Yi, and S. Miller. An Improved Channel Model for Mobile and Ad-Hoc Network Simulations. In *CIIT 2004: Proceedings of the 3rd IASTED International Conference on Communications, Internet, and Information Technology*, pages 352–357, St. Thomas, VI, USA, November 2004.

[136] B. Melander, M. Björkman, and P. Gunningberg. A New End-to-End Probing and Analysis Method for Estimating Bandwidth Bottlenecks. In *GLOBECOM 2000: Proceedings of the IEEE Global Telecommunications Conference*, pages 415–420, San Francisco, CA, USA, November 2000.

[137] M. Mellia and M. Meo. Measurement of IPTV traffic from an operative network. *European Transactions on Telecommunications*, 21(4):324–336, June 2010.

[138] M. Mignon, K. Bouckhout, J. Gahm, and A. C. Begen. Scaling Server-Based Channel-Change Acceleration to Millions of IPTV Subscribers. In *PV 2012: Proceedings of the 19th International Packet Video Workshop*, Munich, Germany, May 2012.

[139] K. Miller, E. Quacchio, G. Gennari, and A. Wolisz. Adaptation Algorithm for Adaptive Streaming over HTTP. In *PV 2012: Proceedings of the 19th International Packet Video Workshop*, Munich, Germany, May 2012.

[140] D. L. Mills. Network Time Protocol (Version 3) Specification, Implementation and Analysis. *Internet RFCs*, RFC 1305, March 1992.

[141] S. B. Moon, J. Kurose, P. Skelly, and D. Towsley. Correlation of Packet Delay and Loss in the Internet. Technical Report 98-11, Department of Computer Science, University of Massachusetts Amherst, Amherst, MA, USA, January 1998.

[142] S. B. Moon, P. Skelly, and D. Towsley. Estimation and Removal of Clock Skew from Network Delay Measurements. Technical Report 98-43, Department of Computer Science, University of Massachusetts Amherst, Amherst, MA, USA, October 1998.

[143] S. B. Moon, P. Skelly, and D. Towsley. Estimation and Removal of Clock Skew from Network Delay Measurements. In *INFOCOM 1999: Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 227–234, New York, NY, USA, March 1999.

[144] R. Mortier, B. Bedwell, K. Glover, T. Lodge, T. Rodden, C. Rotsos, A. W. Moore, A. Koliousis, and J. Sventek. Supporting Novel Home Network Management Interfaces with Openflow and NOX. In *SIGCOMM '11: Proceedings of the ACM SIGCOMM 2011 Conference on Data Communication*, pages 464–465, Toronto, Canada, August 2011.

[145] A. Morton, L. Ciavattone, G. Ramachandran, S. Shalunov, and J. Perser. Packet Reordering Metrics. *Internet RFCs*, RFC 4737, November 2006.

[146] A. Mukherjee. On the Dynamics and Significance of Low Frequency Components of Internet Load. Technical Report MS-CIS-92-83, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA, December 1992.

[147] S. Naegele-Jackson, P. Holleczek, and J. Reinwand. Multi-layer Performance Measurements over Optical Testbeds and QoS Provisioning for High-Bandwidth Video Applications. In *BROADNETS 2006: Proceedings of the 3rd International Conference on Broadband Communications, Networks and Systems*, pages 1–10, San Jose, CA, USA, October 2006.

[148] A. Nafaa, Y. Hadjadj-Aoul, and A. Mehaoua. On Interaction between Loss Characterization and Forward Error Correction in Wireless Multimedia Communication. In *ICC 2005: Proceedings of the IEEE International Conference on Communications*, pages 1390–1394, Seoul, South Korea, May 2005.

[149] C. Neumann, V. Roca, A. Francillon, and D. Furodet. Impacts of Packet Scheduling and Packet Loss Distribution on FEC Performances: Observations and Recommendations. In *CoNEXT '05: Proceedings of the ACM conference on Emerging network experiments and technology*, pages 166–176, Toulouse, France, October 2005.

[150] H. X. Nguyen and M. Roughan. Rigorous Statistical Analysis of Internet Loss Measurements. In *SIGMETRICS '10: Proceedings of the 2010 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 361–362, New York, NY, USA, June 2010.

[151] H. X. Nguyen and M. Roughan. Rigorous Statistical Analysis of Internet Loss Measurements. Technical report, University of Adelaide, 2010. http://adelaide.edu.au/directory/hung.nguyen.

[152] H. X. Nguyen and P. Thiran. Network Loss Inference with Second Order Statistics of End-to-End Flows. In *IMC '07: Proceedings of the 7th ACM SIGCOMM Internet Measurement Conference*, pages 227–239, San Diego, CA, USA, October 2007.

[153] K. Nichols, S. Blake, F. Baker, and D. Black. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. *Internet RFCs*, RFC 2474, December 1998.

[154] OECD. OECD Broadband Statistics, June 2011. http://www.oecd.org/sti/ict/broadband. Last Accessed: 22/05/2012.

[155] J. Ott, J. Chesterfield, and E. Schooler. RTCP Extensions for Single-Source Multicast Sessions with Unicast Feedback. *Internet RFCs*, RFC 5760, February 2010.

[156] R. Pantos and W. May. HTTP Live Streaming. IETF, September 2012. Work in Progress.

[157] K. Papagiannaki, S. Moon, C. Fraleigh, P. Thiran, and C. Diot. Measurement and Analysis of Single-Hop Delay on an IP Backbone Network. *IEEE Journal on Selected Areas in Communications*, 21(6), August 2003.

[158] V. Paxson. End-to-End Routing Behavior in the Internet. In *SIGCOMM '96: Conference proceedings on Applications, technologies, architectures, and protocols for computer communications*, pages 25–38, Palo Alto, CA, USA, August 1996.

[159] V. Paxson. End-to-End Internet Packet Dynamics. In *SIGCOMM '97: Proceedings of the ACM SIGCOMM Conference on Applications, technologies, architectures, and protocols for computer communication*, pages 139–152, Cannes, France, September 1997.

[160] V. Paxson. *Measurements and Analysis of End-to-End Internet Dynamics*. PhD thesis, University of California, Berkeley, April 1997.

[161] V. Paxson. On Calibrating Measurements of Packet Transit Times. Technical Report LBNL-41535, Lawrence Berkeley National Laboratory, Berkeley, CA, USA, March 1998.

[162] V. Paxson. Strategies for Sounds Internet Measurement. In *IMC '04: Proceedings of the 4th ACM SIGCOMM Internet Measurement Conference*, pages 263–271, Taormina, Sicily, Italy, October 2004.

[163] V. Paxson, A. Adams, and M. Mathis. Experiences with NIMI. In *PAM 2000: Proceedings of the Passive & Active Measurement Workshop*, Hamilton, New Zealand, April 2000.

[164] V. Paxson, G. Almes, J. Mahdavi, and M. Mathis. Framework for IP Performance Metrics. *Internet RFCs*, RFC 2330, May 1998.

[165] V. Paxson, J. Mahdavi, A. Adams, and M. Mathis. An Architecture for Large-Scale Internet Measurement. *IEEE Communications Magazine*, 36(8):48–54, August 1998.

[166] C. Perkins. *RTP: Audio and Video for the Internet*. Addison-Wesley Professional, 2003.

[167] C. Perkins, O. Hodson, and V. Hardman. A Survey of Packet Loss Recovery Techniques for Streaming Audio. *IEEE Network*, 12(5):40–48, September/October 1998.

[168] M. Pietrzyk, L. Plissonneau, G. Urvoy-Keller, and T. En-Najjary. On Profiling Residential Customers. In *TMA 2011: Proceedings of the Traffic Monitoring and Analysis Workshop*, pages 1–14, Vienna, Austria, April 2011.

[169] L. Plissonneau and E. Biersack. A Longitudinal View of HTTP Video Streaming Performance. In *MMSys '12: Proceedings of the 3rd Annual ACM SIGMM Conference on Multimedia Systems*, pages 203–214, Chapel Hill, NC, USA, February 2012.

[170] M. Podolsky, C. Romer, and S. McCanne. Simulation of FEC-Based Error Control for Packet Audio on the Internet. In *INFOCOM 1998: Proceedings of the 17th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 505–515, San Francisco, CA, USA, April 1998.

[171] J. Poikonen and J. Paavola. Comparison of Finite-State Models for Simulating the DVB-H Link Layer Performance. In *ISWCS 2005: Proceedings of the 2nd International Symposium on Wireless Communication Systems*, pages 153–157, Siena, Italy, September 2005.

[172] J. Poikonen and J. Paavola. Error Models for the Transport Stream Packet Channel in the DVB-H Link Layer. In *ICC 2006: Proceedings of the IEEE International Conference on Communications*, pages 1861–1866, Istanbul, Turkey, June 2006.

[173] R. Prasad, M. Murray, C. Dovrolis, and K. Claffy. Bandwidth estimation: metrics, measurement techniques, and tools. *IEEE Network*, 17(6):27–35, November/December 2003.

[174] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. ISBN 3-900051-07-0.

[175] L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[176] R. Raghavendra and E. M. Belding. Characterizing high-bandwidth real-time video traffic in residential broadband networks. In *WiOpt '10: Proceedings of the 8th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, pages 597–602, Avignon, France, June 2010.

[177] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, June 1960.

[178] J. Rey, D. Leon, A. Miyazaki, V. Varsa, and R. Hakenberg. RTP Retransmission Payload Format. *Internet RFCs*, RFC 4588, July 2006.

[179] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell. pathChirp: Efficient Available Bandwidth Estimation for Network Paths. In *PAM 2003: Proceedings of the Passive & Active Measurement Workshop*, San Diego, CA, USA, 2003.

[180] V. Roca, C. Neumann, and D. Furodet. Low Density Parity Check (LDPC) Staircase and Triangle Forward Error Correction (FEC) Schemes. *Internet RFCs*, RFC 5170, June 2008.

[181] K. Salamatian and S. Vaton. Hidden Markov Modeling for network communication channels. In *SIGMETRICS '01: Proceedings of the 2001 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 92–101, Cambridge, MA, USA, June 2001.

[182] P. Salvo Rossi, G. Romano, F. Palmieri, and G. Iannello. Joint End-to-End Loss-Delay Hidden Markov Model for Periodic UDP Traffic Over the Internet. *IEEE Transactions on Signal Processing*, 54(2):530–541, February 2006.

[183] H. Sanneck and G. Carle. A Framework Model for Packet Loss Metrics Based on Loss Runlengths. In *MMCN 2000: Proceedings of the SPIE/ACM Multimedia Computing and Networking Conference*, pages 177–187, San Jose, CA, USA, January 2000.

[184] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. *Internet RFCs*, RFC 3550, July 2003.

[185] J. Seedorf and E. Burger. Application-Layer Traffic Optimization (ALTO) Problem Statement. *Internet RFCs*, RFC 5693, October 2009.

[186] A. Shokrollahi. Raptor Codes. *IEEE Transactions on Information Theory*, 52(6):2551–2567, June 2006.

[187] P. Siebert, T. N. M. Van Caenegem, and M. Wagner. Analysis and Improvements of Zapping Times in IPTV Systems. *IEEE Transactions on Broadcasting*, 55(2):407–418, June 2009.

[188] M. Siekkinen, D. Collange, G. Urvoy-Keller, and E. W. Biersack. Performance Limitations of ADSL Users: A Case Study. In *PAM 2007: Proceedings of the 8th Passive and Active Measurement Conference*, Louvain-la-neuve, Belgium, April 2007.

[189] F. Silveira and E. de Souza e Silva. Modeling the short-term dynamics of packet losses. *ACM SIGMETRICS Performance Evaluation Review*, 34(3):27–29, December 2006.

[190] F. Silveira and E. de Souza e Silva. Predicting packet loss statistics with hidden Markov models. *ACM SIGMETRICS Performance Evaluation Review*, 35(3):19–21, December 2007.

[191] F. J. Silveira Filho and E. de Souza e Silva. A method for predicting packet losses with applications to continuous media streaming. In *Proceedings of the Brazilian Symposium on Computer Networks*, 2006.

[192] T. Silverston and O. Fourmaux. Measuring P2P IPTV Systems. In *NOSSDAV '07: Proceedings of the 17th International workshop on Network and Operating Systems Support for Digital Audio & Video*, Urbana, IL, USA, June 2007.

[193] T. Silverston, O. Fourmaux, K. Salamatian, and K. Cho. Measuring P2P-TV systems on both sides of the world. In *ICME 2010: Proceedings of the IEEE International Conference on Multimedia & Expo*, 2010.

[194] J. E. Simsarian and M. Duelk. IPTV Bandwidth Demands in Metropolitan Area Networks. In *LANMAN 2007: Proceedings of the 15th IEEE Workshop on Local & Metropolitan Area Networks*, pages 31–36, Princeton, NJ, USA, June 2007.

[195] D. E. Smith. IP TV Bandwidth Demand: Multicast and Channel Surfing. In *INFOCOM 2007: Proceedings of the 26th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 2546–2550, Anchorage, AK, USA, May 2007.

[196] Society of Motion Picture and Television Engineers. Forward Error Correction for Real-Time Video/Audio Transport Over IP Networks. SMPTE 2022-1-2007, 2007.

[197] J. Sommers, P. Barford, N. Duffield, and A. Ron. Improving Accuracy in End-to-end Packet Loss Measurement. In *SIGCOMM '05: Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 157–168, Philadelphia, PA, USA, August 2005.

[198] J. Sommers, P. Barford, N. Duffield, and A. Ron. A Geometric Approach to Improving Active Packet Loss Measurement. *IEEE/ACM Transactions on Networking*, 16(2):307–320, April 2008.

[199] H. H. Song, L. Qiu, and Y. Zhang. NetQuest: A Flexible Framework for Large-Scale Network Measurement. *IEEE/ACM Transactions on Networking*, 17(1):106–119, February 2009.

[200] A. Soni García, J. Ott, M. Ellis, and C. Perkins. Virtual RTCP: A Case Study of Monitoring and Repair for UDP-based IPTV Systems. In *PV 2012: Proceedings of the 19th International Packet Video Workshop*, Munich, Germany, May 2012.

[201] M. A. Stephens. Tests Based on EDF Statistics. In R. B. D'Agostino and M. A. Stephens, editors, *Goodness-of-Fit Techniques*, pages 97–193. Marcel Dekker, Inc., 1986.

[202] T. Stockhammer. Dynamic Adaptive Streaming over HTTP – Standards and Design Principles. In *MMSys '11: Proceedings of the 2nd Annual ACM SIGMM Conference on Multimedia Systems*, pages 133–144, San Jose, CA, USA, February 2011.

[203] J. Strauss, D. Katabi, and F. Kaashoek. A Measurement Study of Available Bandwidth Estimation Tools. In *IMC '03: Proceedings of the 3rd ACM SIGCOMM Internet Measurement Conference*, pages 39–44, Miami Beach, FL, USA, October 2003.

[204] S. Sundaresan, W. de Donato, N. Feamster, R. Teixeira, S. Crawford, and A. Pescapè. Broadband Internet Performance: A View From the Gateway. In *SIGCOMM '11: Proceedings of the ACM SIGCOMM 2011 Conference on Data Communication*, pages 134–145, Toronto, Canada, August 2011.

[205] J. Sventek, A. Koliousis, O. Sharma, N. Dulay, D. Pediaditakis, M. Sloman, T. Rodden, T. Lodge, B. Bedwell, K. Glover, and R. Mortier. An Information Plane Architecture Supporting Home Network Management. In *IM 2011: Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management*, pages 1–8, Dublin, Ireland, May 2011.

[206] S. Tao, J. Apostolopoulos, and R. Guérin. Real-Time Monitoring of Video Quality in IP Networks. *IEEE/ACM Transactions on Networking*, 16(5):1052–1065, October 2008.

[207] S. Tao and R. Guérin. On-Line Estimation of Internet Path Performance: An Application Perspective. In *INFOCOM 2004: Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1774–1785, Hong Kong, China, March 2004.

[208] P. U. Tournoux, E. Lochin, J. Lacan, A. Bouabdallah, and V. Roca. On-the-Fly Erasure Coding for Real-Time Video Applications. *IEEE Transactions on Multimedia*, 13(4):797–812, August 2011.

[209] R. Turner and L. Liu. *hmm.discnp: Hidden Markov models with discrete non-parametric observation distributions*, 2012. R package version 0.1-8.

[210] T. N. M. Van Caenegem, K. O. Struyve, K. Laevens, D. De Vleeschauwer, and R. Sharpe. Maintaining Video Quality and Optimizing Video Delivery Over the Bandwidth Constrained DSL Last Mile Through Intelligent Packet Drop. *Bell Labs Technical Journal*, 13(1):53–68, May 2008.

[211] S. V. Vasudevan, X. Liu, and K. Kollmansberger. IPTV Architectures for Cable Systems: An Evolutionary Approach. *IEEE Communications Magazine*, 46(5):102–109, May 2008.

[212] B. Ver Steeg, A. Begen, T. Van Caenegem, and Z. Vax. Unicast-Based Rapid Acquisition of Multicast RTP Sessions. *Internet RFCs*, RFC 6285, June 2010.

[213] B. Wang, J. Kurose, P. Shenoy, and D. Towsley. Multimedia Streaming via TCP: An Analytic Performance Study. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 4(2), May 2008.

[214] M. Watson, T. Stockhammer, and M. Luby. Raptor Forward Error Correction (FEC) Schemes for FECFRAME. *Internet RFCs*, RFC 6681, August 2012.

[215] W. Wei, B. Wang, and D. Towsley. Continuous-time hidden Markov models for network performance evaluation. *Performance Evaluation*, 49(1-4):129–146, September 2002.

[216] W. Wei, B. Wang, D. Towsley, and J. Kurose. Model-based Identification of Dominant Congested Links. In *IMC '03: Proceedings of the 3rd ACM SIGCOMM Internet Measurement Conference*, pages 115–128, Miami Beach, FL, USA, October 2003.

[217] W. Wei, B. Wang, C. Zhang, J. Kurose, and D. Towsley. Clarification of Access Network Types: Ethernet, Wireless LAN, ADSL, Cable Modem or Dialup? *Computer Networks*, 52(17):3205–3217, September 2008.

[218] S. Winkler. Video Quality Measurement Standards - Current Status and Trends. In *ICICS 2009: Proceedings of the 7th International Conference on Information, Communications and Signal Processing*, Macau, China, December 2009.

[219] Y. J. Won, M.-J. Choi, B.-C. Park, J. W. Hong, H.-W. Lee, C.-K. Hwang, and J.-H. Yoo. End-User IPTV Traffic Measurement of Residential Broadband Access Networks. In *E2EMON 2008: Proceedings of the 6th IEEE Workshop on End-to-End Monitoring Techniques and Services*, Salvador, Bahia, Brazil, April 2008.

[220] Y. J. Won, J. W.-K. Hong, M.-J. Choi, C.-K. Hwang, and J.-H. Yoo. Measurement of Download and Play and Streaming IPTV Traffic. *IEEE Communications Magazine*, 46(10):154–161, October 2008.

[221] Y. Xiao, X. Du, J. Zhang, F. Hu, and S. Guizani. Internet Protocol Television (IPTV): The Killer Application for the Next-Generation Internet. *IEEE Communications Magazine*, 45(11):126–134, November 2007.

[222] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz. P4P: Provider Portal for Applications. In *SIGCOMM '08: Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*, pages 351–362, Seattle, WA, USA, August 2008.

[223] M. Yajnik, J. Kurose, and D. Towsley. Packet Loss Correlation in the MBone Multicast Network. In *GLOBECOM 1996: Proceedings of the IEEE Global Telecommunications Conference*, pages 94–99, November 1996.

[224] M. Yajnik, S. Moon, J. Kurose, and D. Towsley. Measurement and Modelling of the Temporal Dependence in Packet Loss. In *INFOCOM 1999: Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 345–352, New York, NY, USA, March 1999.

[225] X. Yang, C. Zhu, Z. G. Li, X. Lin, and N. Ling. An Unequal Packet Loss Resilience Scheme for Video Over the Internet. *IEEE Transactions on Multimedia*, 7(4):753–765, August 2005.

[226] X. Yu, J. W. Modestino, and X. Tian. The Accuracy of Gilbert Models in Predicting Packet-Loss Statistics for a Single-Multiplexer Network Model. In *INFOCOM 2005: Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 2602–2612, Miami, FL, USA, March 2005.

[227] Y. Yu and S. L. Miller. A Four-State Markov Frame Error Model for the Wireless Physical Layer. In *WCNC 2007: Proceedings of the IEEE Wireless Communications & Networking Conference*, pages 2055–2059, Hong Kong, China, March 2007.

[228] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker. On the Constancy of Internet Path Properties. In *IMW '01: Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, pages 197–211, San Francisco, CA, USA, November 2001.

[229] Y. Zhao, Y. Chen, and D. Bindel. Towards Unbiased End-to-End Network Diagnosis. *IEEE/ACM Transactions on Networking*, 17(6):1724–1737, December 2009.

[230] M. Zink, K. Suh, Y. Gu, and J. Kurose. Watch Global, Cache Local: YouTube Network Traffic at a Campus Network – Measurements and Implications. In *MMCN 2008: Proceedings of the SPIE/ACM Multimedia Computing and Networking Conference*, San Jose, CA, USA, January 2008.

[231] W. Zucchini and I. L. MacDonald. *Hidden Markov Models for Time Series: An Introduction Using R*. Chapman & Hall/CRC, 2009.