Congestion Control for Packetised Video in the Internet

Ian John Wakeman

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of the

University of London

Department of Computer Science

University College London

1995

Abstract

There exist a number of algorithms for compressing digital video, such as the ISDN H.261 standard, and MPEG I and II. With all these algorithms, it is possible to vary the degree of compression within the codec and thus the output data rate, by manipulating the parameters of the algorithm. Using intelligent codec implementations, it becomes possible to send compressed video over packet switched networks and adapt the rates at which the compressed video is produced to maximise video quality, whilst protecting the network from congestion. This thesis presents two viable schemes which use results from human factors experiments that prove this is possible with minimal support from the intermediate switches.

First we derive the requirements for transmitting video over packet switched networks, surveying the reported work on human factors, and survey and compare the various algorithms for compressing video and determine their suitability for use over packet switched networks. We produce a taxonomy of congestion control schemes for packet switched networks. We present two novel approaches to congestion control which build on results from human factors experiments: a unicast scheme that we prove is analytically stable under some simple assumptions, and demonstrate working in simulations; and a multicast scheme that is designed to work over heterogeneous hosts and networks, and which has been proven in the wide-spread use of a particular workstation based video conferencing tool, known as IVS.

We conclude with ruminations on the extent to which resource protection schemes need to be fielded in packet switches to support multimedia.

Acknowledgements

Jon Crowcroft has been my supervisor, boss and friend whilst at UCL. His intellectual curiosity and ability to discover new ways of thinking about problems have inspired much of this work, whilst without his patience and good-natured encouragement, the thesis would never have been written.

Nearly everyone in the UCL Computer Science Department seems to have helped me at some point; in particular I'd like to thank Mark Handley, Atanu Ghosh, Dave Lewis, Zheng Wang, Angela Sasse, Simon Arridge, Nermin Ismail, Vicky Hardman, Stuart Clayman and Natalie May.

I'd like to thank Van Jacobson and Sally Floyd at LBL for their guidance and code in building the CBQ router and for inspiring the work in Chapter 5; a footnote from Van in [1] stated that proportional control was impossible in the Internet. I'm very grateful to Thierry Turletti and Jean Bolot for allowing me to hold up releases of IVS whilst I worked on the integration of the congestion control code. I'm deeply indebted to Joel Halpern and Paul Francis for their code and help on Patricia derived algorithms, to the folks at University of London Computer Centre, SuraNet and BBN for their work and help in putting the CBQ code on the FATpipe and to Greg Minshall and other referees for their helpful comments on the parts of this thesis which have made it into papers.

Finally I'd like to dedicate the thesis to my partner Helen. Without her love and support, I would never have finished it.

Contents

List of Figures

List of Tables

Chapter 1

Introduction - A roadmap through the thesis

In this chapter we explain why video will be moved over networks between computers, and why we have chosen to concentrate on the data transport mechanisms from conventional computer networking. We conclude with a description of the work in the subsequent chapters, and explain the original contribution of the thesis.

1.1 Why Video?

The next generation of workstations will be able to manipulate digital video, through the use of built-in digital signal processing capabilities. Video[1] can be used in a number of ways: to enhance computer mediated communication between people such as in a video conference setup; to enhance communication between the application and the user, such as in a travelog educational program; and as a source of raw data to be processed to provide the required information to the user, e.g. for automatic person recognition in surveillance. The source of the video can either be live from a camera, or pre-recorded and stored on some suitable storage medium such as an optical disk.

In video telephony and conferencing, the source of the data is necessarily remote from the user, and so the video must be transmitted over a communication network. In applications using stored video, the storage media for the video may be remote, and the video may have to be retrieved from a *video file server* [2,3], whilst in a sensing application

[1] In this thesis, we will concentrate on video, although similar ideas can be applied to audio data as well.

the video may be captured locally, and then transmitted to a remote machine for the required image processing. In all these cases, a communication network is required. Although it is possible to provide a separate wire for the video, this is not economically viable on a large scale. Instead, it would be preferable to use the existing connections between computers, since these are already in place. This implies a need to support video data over computer networks.

Due to the relatively low bandwidth of today's computer networks compared to raw video - Ethernet typically working at 10 MBit/s whilst uncompressed *common intermediate format* (cif) video uses 36.5 MBit/s - the video must be compressed so as to fit. The compression and coding of the video will take place in what will be termed a *video codec* in this thesis. We will not distinguish between implementation of the codec in specialised hardware, or in software [4]. The demands that the transmission of video places upon the network will depend on the application's requirements of the video.

Compressed video is an inherently variable bit rate (VBR) data stream, since the information content of each frame of video varies. The first frame of a new scene will carry new information, but subsequent frames will enhance the information content of the original frame only slightly. The compression of the subsequent frames can thus use the information carried in the original frame to increase the compression that can be achieved. This results in the compressed data per frame varying with a ratio of up to 15:1 for some compression schemes.

Consider the case of video from a camera being compressed in real time, e.g. for a video conferencing application. The compressed video can be transmitted as it is generated, which will result in a highly bursty stream, or it can be passed into a dual ported buffer, where the sending software extracts the video from the buffer at a rate that suits it, which could be anywhere between the average rate at which the compressed data stream is generated and the peak rate at which the data is generated. The rate chosen depends upon the delay which the application can tolerate in receiving the video, the amount of buffering that can be offered to reconstruct the video stream for decompression and the extent to which the network can absorb bursty data.

Computer networks rely on statistical multiplexing of the bursty traffic generated by computers in order to maximise the efficiency of the network. Since video traffic is also bursty, computer networks would appear to be ideal for the transmission of compressed video traffic. However, a computer network is a shared resource, and if a bursty source transmits without regard to the degree of network utilisation, then not only will the misbehaving connection lose packets due to congestion, but the other users of the network will suffer from the congestion induced by the video source. It has been argued in [5] that the size of network queues in ATM networks will reduce the potential for statistical multiplexing. However, packet networks have packets and queues that are orders of magnitude larger than for cell-based ATM, a packet being typically 512 bytes long and buffer capacity of a megabyte or larger in current multi-protocol routers, and a high statistical multiplexing gain can be achieved, although at the cost of an increased delay.

The network can be protected from these problems in two ways - network resources can be reserved for the video transmission, or the traffic sources can control the rate at which they are generating traffic to protect against congestion. Since current computer networks do not supply the requisite resource allocation schemes for a bandwidth reservation approach and deployment of these mechanisms will be neither cheap nor fast, this thesis will consider the second alternative. Fortunately, the compression schemes that will be used in compressing the video admit to being able to generate variable average data rates, although generally at the cost of reduced quality, and increased processing time. It is likely that these schemes will result in adaptable codecs, where a codec can have its parameters adjusted to suit the characteristics of the output data stream. This thesis will therefore explore the options available for trading the network resource, the output data rate of an adaptive video codec, and the ability of humans to compensate for poor visual quality, so as to increase the total utility of the network and its denizens.

As a certain Jeremy Bentham[2] said, "The Greatest Happiness of the Greatest Number, that is the measure of Right and Wrong" [6]. In packet switched networks, we assume utilitarian measures of what is best, rather than attempting to maximise the ecstasy of a few by dedicating resources to them.

The specific model which we will be considering is shown in Figure 1.1. This shows a conferencing type application, where the video is generated by one of a number of cameras, which is then fed into a codec in a workstation. The workstation then transmits the video through a wide area IP type network to the receiving workstation, which feeds the compressed video stream into its codec and displays the received image on its display device. This model has been chosen since it incorporates characteristics of nearly all other scenarios. The work carried out by Gilge at Berkeley is similar in assumptions,

[2] As your starter for ten, which London college did Jeremy Bentham found?

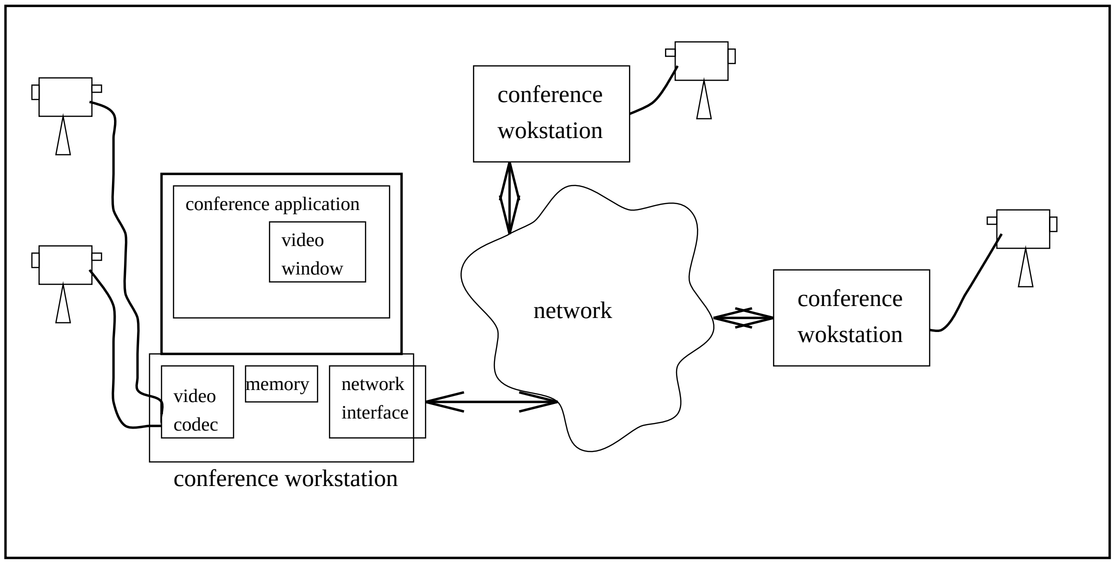


Figure 1.1: Model setup

although different in focus [7].

1.1.1 Why IP?

The Internet Protocol, or IP, is used in this thesis as shorthand to denote a particular cultural approach towards computer networking. A reasonably complete history and description of the protocols and the philosophies behind them can be found in [8].

The fundamental service model of the Internet is embodied in the Best Effort delivery model of the IP datagram. This has allowed the Internet to become a particularly robust technology. The distributed routing protocols have allowed the routing to be dynamic and with no single point of failure, allowing rerouting of flows to adapt to any changes in the network. IP has been structured so as to be independent of any subnetwork technology, by only requiring the most basic of packet delivery services. This has allowed the Internet to grow, since any sub-networking technology can be used to move the packets, and different administrations can choose the most suitable technology for their own goals. The complexity required to meet the service goals of applications, such as a reliable, sequenced stream, has been pushed into the end-systems, through the use of protocols such as TCP. Interoperability problems have been solved by ensuring that working code is publicly available for vendors to build their own code upon.

The robustness and the ability to subsume heterogeneous subnetworks, from modems over POTS running at 300 baud to the Gigabit Networks of today's test-labs and tomorrow's backbones is what makes the IP model so attractive as the basis for future multi-service networks. To make this possible, the delays incurred from packet queues must be controlled. This thesis develops approaches based on end system mechanisms to achieve this control, and develops a framework to discuss the other necessary mechanisms to create a robust multi-service network.

An additional motivation for the use of the IP approach is that it is more than just a philosophy: the IP approach to networking is embodied in the Internet. We therefore have a testbed on which to experiment, and in particular, we have used the global multicast IP network, known as the Mbone[3] to test the scheme developed in Chapter 6.

1.1.2 Why not asynchronous transfer mode?

The broadband network of the future as envisioned by the telecommunications world is based around the services and possibilities offered by the asynchronous transfer mode (ATM). This networking technique is based around the transport of all data in fixed sized cells that are small in size. The standards for ATM are defined by the ITU in the I series of recommendations, with de facto standards defined by the ATM Forum and are best described in [10]. The ITU standards define the cell size to be 48 bytes of data plus 5 bytes of header for switching. The network is based on virtual circuits (VCs), where the cells carry virtual circuit identifiers to enable the switches to look up state for the VC, decide on the output port(s) and rewrite the VC identifier for the next hop.

The fixed sized cell is supposed to make the hardware required for switching easier to build, in that it simplifies buffer management. The VC identifier is selected for each link by the receiving switch, which can choose the identifier to make life easier in looking in up the VC state, either based on a simple table[4] or as hash key. Most importantly, the small cell size is intended to reduce the latency of the path. This can be seen by viewing a switch as an M/D/1 queuing system. It can easily be shown that the mean wait, W, in a queue for a server which takes a fixed period x to service a packet i.e. transmit it,

---

[3]The Mbone is a virtual network overlaid on the Internet to provide multicast routing capabilities. It is intended to be temporary until wide area multicast routing can be integrated in the normal IP routing capabilities. More details can be found in [9].

[4]Although the Telcos' anathema to the use of memory in switches may result in different approaches being used, such as the associative memory lookup used in the Siemens/GPT switch.

where the offered load is ρ is given by

$$W = \frac{\rho x}{2(1-\rho)}$$

Thus it can be seen that if the load remains constant, the number of jobs in the queue remains constant whilst the service time changes. Thus the use of small packets should ensure that latencies through the network due to queuing delays are low, whereas if large packets were used - IP packets have a bi-modal distribution from the size of a TCP acknowledgment in whichever data link encapsulation is required to the maximum transmission unit over the network [11] - the queuing latencies would be large.

However, the value of these so-called advantages is debatable. It is extremely unlikely that the distribution of the packets will scale according to the size of packet. Computers generally manipulate data in large chunks such as a frame of video or a page of memory. When these are sent over the network, they will generally be sent out in large chunks. If they are fragmented into small cells this will introduce a long stream of packets sent out, creating a bursty traffic flow. Measurements of data traffic by Leland et al [12] and preliminary studies of the nature of compressed video by Garrett [13] show that real traffic is likely to be bursty even after aggregation. To protect the cell switches, outgoing traffic will need to be "policed" by the switches in the network, and "shaped" in the end-systems, using parameters such as those from a leaky bucket description. "Policing" is the monitoring of the cell stream and the dropping of cells which fall outside the allowable parameters, whilst "shaping" is the act of buffering cells and metering them out to fit the allowable parameters. The reduced queuing latencies will thus be offset by shaping delays and complication in the end-systems. Even worse is that the use of a hierarchical network structure such as that based on a backbone will require additional shaping to be done within the network, adding complexity in the queue management. Thus the reduction in cost from a fixed size cell is offset by the need to manipulate larger scale structures in shaping and policing. In the end-systems, since the granularity of the structures manipulated are orders of magnitude larger than a cell - typical memory page sizes are 4 KBytes compared to 48 bytes of user data in a cell, the benefits of fixed sized cells are offset by the problems of segmentation and re-assembly in the network device. Therefore, to produce the benefits of a low latency ATM network, computer architectures should be redesigned so that data is manipulated as cells inside the machine, such as in the Desk Area Network [14].

1.2 Thesis Outline and Contribution

In Chapter 2, we examine the service requirements for moving video data across networks, and examine the required characteristics both of the network video compression scheme and of the packet transport system.

In Chapter 3, we develop the requirements in Chapter 2 and examine a selection of video compression schemes. We present an informed decision table of which compression scheme is likely to be most suitable.

In Chapter 4, we present an abstracted model of the service presented to the application, and how this degrades under congestion. We survey other approaches to controlling congestion in packet networks within the framework of this model.

Chapter 5 presents an approach to maximising bandwidth utilisation of the network for continually transmitting sources, such as video sources. We assume that the sources use a hierarchical encoding and can adapt their source coding rates in response to requests from the network congestion control algorithm, and use this adaptability as the controlled input for a proportional control algorithm. We model flows entering or leaving the network as step inputs, and use the algorithm as the basis for admission control, using very little intelligence in the switches. We then present simulation results that demonstrate the algorithm working in a number of scenarios.

Chapter 6 describes a mechanism for scalable control of multicast continuous media streams. The mechanism uses a novel probing mechanism to solicit feedback information in a scalable manner and to estimate the number of receivers. In addition, it separates the congestion signal from the congestion control algorithm, so as to cope with heterogeneous networks. This mechanism has been implemented in the IVS video-conference system using options within RTP to elicit information about the quality of the video delivered to the receivers. The H.261 coder of IVS then uses this information to adjust its output rate, the goal being to maximize the perceptual quality of the image received at the destinations while minimizing the bandwidth used by the video transmission.

In Chapter 7 we conclude with ruminations on how the new multi-service Internet will provide video transmission, drawing on the experience of the work in this thesis. Finally in Appendix A, we describe implementation experience of a scheduler for a router which both protects the robustness and scalability of the Internet, yet allows real-time flows to survive with different reactions to congestion from best effort traffic.

The original contributions of this thesis lie in the development of a novel congestion avoidance and control schemes for the unicast case, shown to be stable in theory and in simulation, and the development of the first scalable approach to multicast congestion control which has been implemented in a widely used tool and shown to reduce congestion.

Chapter 2

Packetised Video - Options for Interaction between the User, the Network and the Codec

2.1 The User - what do they want from video?

Users[1] require modern computing technology to meet three objectives:-

- Applications must be appropriate to the needs of the user.
- The cost of the infra-structure to support the application must not be too high.
- The infra-structure should be sufficiently connected to allow the user (and their application) to communicate with all entities with which they wish to communicate.

In meeting the needs of the user, the application may require the video data it utilises to be transported across a network. Therefore, the demands placed upon the network will depend on the type of video that the application will be using, and how that video is being used.

For example, the variability of the stream will depend on the application: video telephony will generally have only one camera trained on one scene, and so the video statistics will be slowly changing; video conferencing will have several cameras, and so the output video will be selected from a choice of several different sets of evolving

[1] An early version of the work in this and the succeeding chapter was presented in [15]

statistics; a simulation of the view from an aeroplane cockpit may have a set of statistics that change faster; whilst an arbitrary video for editing may have statistics that change rapidly on every scene change[2], or even on every frame depending on the subject matter of the video stream. These statistics will affect the burstiness of the stream. Another implication of changing statistics is that the user and the application will not be able to characterise the statistics of the video accurately, and the demand they will place on the network; the bandwidth demanded must continually be updated to maintain constant quality.

In conferencing and other distributed applications, the video may be distributed to more than one site. This requires the network to support one to many distribution of video, with the possibility of a single site sourcing or sinking many video streams, and multiple sites generating video streams.

It is unlikely that applications will use the video without other sources of related information. In video telephony, there will be an associated audio stream, whilst in a workstation-based video conference, there may be shared applications, such as shared whiteboards, which must be related to the gestures of the users and their speech. All these data streams must be synchronised with each other, so that the user sees related events happening in approximately the correct time ordering. More demanding synchronisation occurs when users at different points in the network must see events simultaneously. The most popular technique for doing this is to buffer the data as it arrives at the end-system and release it at a common playback point [16]. Other possibilities are for the network to adjust the timing relationship of packets as they pass through it, so as to maintain the same temporal relationship within a stream, and thus between streams. This requires the network to provide some form of jitter control, as described in [17]. Another technique [18,19] for maintaining synchronisation between flows is for the data to be multiplexed together at the source, relying on the packet reception being ordered to maintain temporal consistency. This last approach has the disadvantage in that the flows cannot be distinguished, and we cannot take advantage of features unique to the data type to improve the efficiency of the system as a whole.

In addition to separate but related data streams, the video may be augmented with additional side information. As an example, in viewing a video, freezing the picture may result in a low resolution image. However, this image may then be augmented by

[2] Assuming the use of real film, rather than fractal descriptions or other forms of animation.

additional stored information, so that the resolution can be increased. This could also be done to allow the user to compensate for poor image quality interactively, where the user uses a mouse or other device to delimit the area which should be enhanced on screen, and this is used to request additional information to enhance the image from then onwards. Or in an educational video, pointing at an object could result in a database lookup on an associated object key to retrieve relevant information. Although the problem of associated information does not directly impact upon the transmission of video over a network, the sensible use of side information, and the timing of its transmission could both reduce the impact of the video on the network, and increase the utility of the video to the viewer.

If the video stream is being generated in real time, rather than from storage media, then additional constraints must be met. The video streams differ from normal data streams in that their delivery time is more directly dependent upon the requirements of the user, and affect the value of the data to the application and the user. In a conferencing application, the pictures must be produced within a reasonable time, so as to allow the participants to associate the images they see with the actions that caused them, such as a comment producing a grimace in one of the other participants. If the delay is too large then the value of the conference will decrease. Thus there may be a maximum transmission delay associated with the data. There is a conflict of requirements here between the buffering of data for synchronisation and early release for interactive viewing.

2.1.1 Video Quality in the Application Domain

The eventual consumer of the video can be considered as an information processing system [20][3], whether the consumer is a computer based image processing system or a human being. The effectiveness of an information processing system at any task is determined initially by whether there is the necessary information available. Assuming that the information processing system is adequate for the task in hand, then the task can be completed. In the simplest cases, task performance is either success or failure, where failure is a result of insufficient information. In more complex cases, task performance is

[3] David Marr in [21] revolutionised the study of vision by viewing human vision as a processing problem in an AI context. Whilst his theories are not yet universally accepted, they offer a basis for computer scientists to argue about the important parts of an image.

more fuzzily defined, but the difference between low performance and high performance is still determined by a narrow range of available information.

The information within a video signal is used to infer what objects are within the scene, and associated information such as shape, position, colour, size and relative movements. The necessary information depends upon the accuracy with which these objects must be rendered, so that they can be recognised by other parts of the information processing system. When we map the analogue video signal to the digital domain, the accuracy is constrained by the resolution of the picture due to the number of pixels, the representation of the luminance and chrominance, and the frame rate. Each of these four dimensions is quantised in the digital domain, and the necessary accuracy determines boundaries of possible quantisation. If the quantisation is too coarse, then the information processing system cannot determine the objects and their properties with sufficient accuracy. If we define *video quality* to mean the amalgam of the quantisation resolution, chrominance, luminance and frame-rate, then this boundary defines the *minimum quality threshold* for the task.

The video signal carrying the information contains a large amount of redundant information not needed by the information processing system. Compression algorithms work by removing this redundancy. The compression process may also add noise, such as visual artifacts. As long as the resulting compressed image still has sufficient quality for the image processing system, then the task can be performed. Increasing the compression pushes the image closer to the minimum quality threshold, and eventually the resulting video may not carry all the necessary information, and task performance will deteriorate. Ideally, the compression algorithm would be cognizant of how the image is to be used and thus could preserve important aspects of the picture, but this requires further research to achieve. The psycho-visual aspects of human perception are being investigated to improve future compression algorithms [22]. For current widespread compression schemes, such as H.261, the video quality is dependent to the first order on the output data rate of the codec.

Wilson, Wakeman and Smith [23] carried out an experiment to investigate the hypothesis that the necessary video quality for task support exhibits a threshold effect. The experiment attempts to determine comprehension task performance at a cognitive level, where the video is a major component of communication. The subjects were shown video clips of people repeating word lists, stories and so on, with the audio heavily degraded.

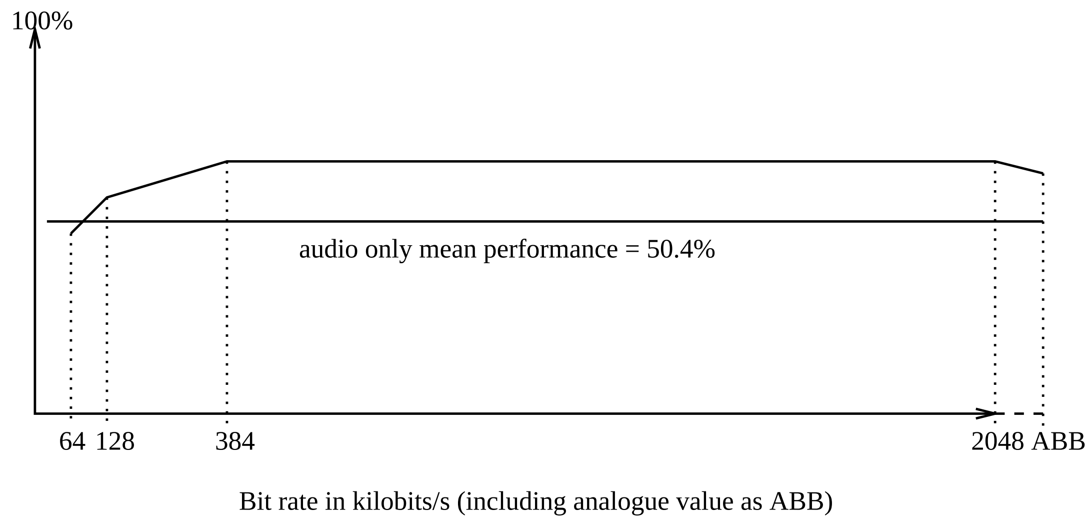


Figure 2.1: Task performance against video quality

The video clips were encoded through a video codec to produce output streams of various output data rates and thus quality. The resulting graph of task performance against video quality is reproduced in Figure 2.1.1. The differences in performance were shown to be significant.

When the subjects were asked about the perceived quality of the video, a similar threshold effect was seen, but shifted to the right, reproduced in Figure 2.1.1. This rightward shift is thought to be the result of two distinct factors:-

- Cognitive dissonance [24] describes the situation where a subject holds two inconsistent thoughts in their head about an action just performed. This state is not to be desired, so they attempt to relieve it through changing their attitudes. In this case the difficulty the subjects encountered in what they assume to be an easy task could result in justificatory attributions of low quality.

- Drawing on the psychology of consumer behaviour, the consumer is not satisfied with the bare minimum necessary to perform adequately - instead the "quality-greedy" consumer wishes to have a product that they perceive to be of a high quality [25]. Since people are very aware of the quality that can be delivered from a broadcast television system, they bring this knowledge to evaluating the quality of a product. In the language of consumer behaviour, not only do they evaluate the product in terms of intrinsic cues, such as how well the product helps in achieving the task in hand, but also using extrinsic cues, such as how does it compare to

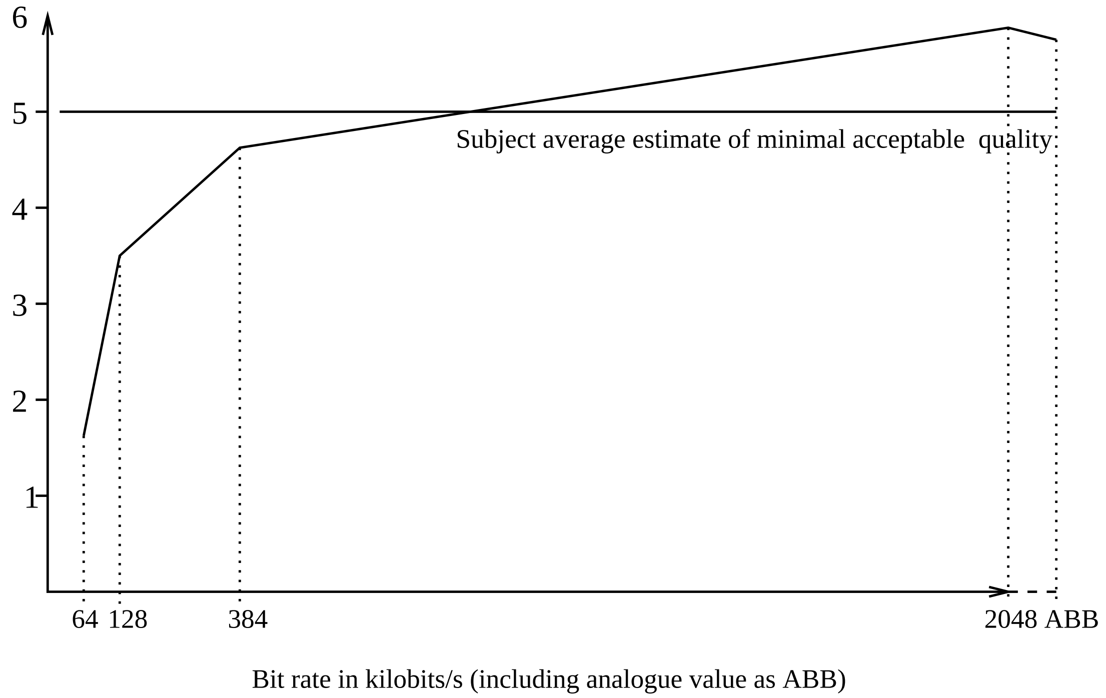


Figure 2.2: Perceived quality against video quality

other related products (such as broadcast TV), and how much does it cost[4]. Thus their evaluation of the video is influenced by comparison to broadcast quality TV, pushing the perceived quality curve to the right.

Frowein et al in [26], report a quality continuum based on the relationship between task performance and frame rate, allowing advice concerning suitable frame rates for selection and design of codecs. This work again showed a threshold effect. This minimum quality threshold is important in the design of congestion control schemes for packetised video since the congestion control system can vary the data rate of the video codec in response to the network, and thus vary the quality of the video as long as the the video contains the sufficient visual information to support the task. However, a work task is composed not only of the physical and cognitive parts, but also of the affective processes [27]. To investigate the totality of the work task, factors affecting the emotional and social state must be investigated. Investigations into the area of computer supported co-operative work (CSCW) have examined the use of video in task performance in a wider social context, but little work has focussed on how quality affects performance. The work within the CSCW community has focussed on the use of video in three different contexts

[4] It is perhaps these extrinsic cues which explain why conferencing systems which do add value to personal communications may not be widely taken up, until picture quality is improved.

[28]:-

Non-verbal Communication In a face to face conversation, visual information is used in determining understanding, using cognitive cues such as head nods and visual attention, in determining turn-taking and for communicating emotional state, using affective cues such as facial expression and posture. Whittaker in [28] surveys the existing work where typically, users working on a group task face to face are compared against groups using audio and video channels and solely audio channels. Whittaker concludes that video adds little or no value to task performance, and that audio is the main medium of communication. Recent work by Olson et al [29] has produced evidence that well designed video can enhance distributed working by "a small but significant amount". A theme in most studies, however, is that users perceive the video channel to add value, by offering social presence to the remote workers. Their image on a screen reassures that they are present, and although people can translate cognitive and turn taking cues across to the audio domain, video remains much better at communicating affective cues [28].

Video for Connection Video channels can be used to determine the availability of an individual, such as the "port-holes" system from Dourish [30], the "video-window" and its successor "cruiser" system [31]. A user can look into an office through a remote camera to see if they are present, opening a short duration connection known as a *glance*. Alternatively, an *open link* connection can be maintained, where the video from the remote office is continually available. Although the jury is still out on whether these are useful functions and socially acceptable, it is obvious that the video quality required need not be high. In the port-holes system, the video was a single frame grabbed every ten minutes.

Video-as-data Instead of using video to communicate the image of the person, video can be used to communicate information about the relationships of the work objects themselves, such as in surgery [32]. In these cases, the video shows the object to be manipulated. The necessary quality of video should be sufficient to meet the surgeons' requirements. It is a common fallacy that video quality in medical applications should be as high as possible - in many cases, the video is of a very low quality already, typically in endoscopic surgery and in ultra-sound techniques and the quality loss from compressing the video is very low, especially after signal processing to enhance the image[5]. Alternative uses of video-as-data include the use of images of 3D objects in design tasks [33], where only enough information in the image is required to disambiguate the 3D relationships of the objects to each other.

Although much of the work described above has used standard analogue video, lower quality video could perform equally as well in many cases, such as in the video for connection and video-as-data scenarios. Since cameras and video cards are fast becoming standard items with computer equipment, the uses to which they will be put will be limited by the imagination of the public. Hollan and Stornetta in [34] argue for the use of computer mediated communication not as a replacement for proximate interaction but to provide new modes of communication that will enhance and enrich human interaction. Similarly, computer mediated video communication is not a replacement for face to face interaction, since obviously the field of vision is restricted to the field of the camera, and image quality is necessarily degraded, but a separate mode of communication, whose usage and utility will depend on context, and thus where the necessary quality will vary from application to application.

2.1.2 Delivery Requirements for Video

The fact that video quality can be translated almost directly into bandwidth prevents us from supplying the highest possible quality to the user. Since video codecs are likely to be adaptable in the quality of video that they produce, we are able to trade network bandwidth against quality, using the adaptability of the user to adjust the stream injected into the network so as to produce the best of all possible worlds.

The scalability of the video stream that the flexible user allows can be used in a number of ways:-

- Transmission of video data over a packet network is subject to the possibility of error, and of loss of information. The options for protecting the video against loss will be more fully explored in the next section, but the major feature which distinguishes video data from other data is that much of the information in the data stream is redundant, and often irrelevant. Loss of data need therefore only result in a decreased quality of video, rather than loss of video.

[5] Legal considerations may well preclude the proper engineering of the video delivery system, since medical practitioners may be reluctant to stand up in court and admit they intentionally threw information away from their images.

- Since the network is shared by a number of users, the quality for each user can be set so as to allow the maximum number of users on the network, subject to the constraint of meeting their minimum quality standard.

- If the network is driven into congestion[6], then the video source can reduce its input data rate to allow the network to recover, since the user can tolerate the reduced quality, as long as it does not fall below their minimum quality threshold.

Whilst maximising the video quality is the primary aim of this thesis, the video is only one part of the application. As has already been pointed out, in much human communication, it is the audio component that contains the most important information. Thus, when an application is using a finite communication resource to send multiple data streams, such as audio and video, the system must enable the application to make policy decisions about how to prioritise the data streams and allocate bandwidth accordingly. We shall return to this aspect of policy in Chapter 6.

Cost is an important factor. The tariffing of packet switched networks will ensure that the user does not want to pay for a service that they do not find usable. The fixing of a minimum quality threshold, below which they are unwilling to stray, should help in setting the Quality of Service parameters with which the network must be provided. Conversely the user will not want to pay for quality that they do not find necessary, so there may be a requirement for a maximum quality threshold.

Users want connectivity to other users. This affects the choice of networking technology, since basing the transmission upon a networking technology to which a remote user cannot easily connect, removes one of the major uses of packet video for human communication. This probably removes the possibility of directly using ATM, since this is unlikely to be widely available for some years yet. The best candidate for networking technology is the well-known IP (Internet Protocol), since this can be carried over any number of media (including ATM), and is robust and adaptable to the needs of the video. This will be further explored in the next section.

Connectivity with other users also requires the video to be compressed and coded in a manner that can be decoded by the other end. This suggests that the coding scheme used should be a *standard*. Unfortunately, video coding standards do not currently

[6] *Congestion* occurs when the offered load to the network exceeds the ability of the network to process it and queues grow large, increasing the transit time through the network.

support the features necessary for efficient transmission over packet networks, since they are aimed at fixed bit rate transmission from CD-ROMs or over telephone networks. This will change in the future, as networked video becomes more pervasive.

2.2 Transmitting Compressed Video over the Network

In order to understand the impact that the use of video has on packet networks, let us examine the sequence of actions that need to be undertaken to send video data from one host to another. This is a one-to-one relationship; one-to-many connections will be discussed later.

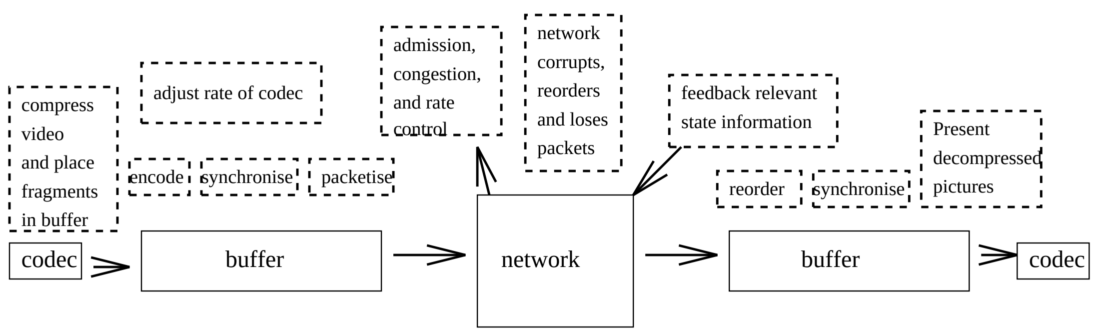


Figure 2.3: Actions required to transmit data

First the connection must be established, ensuring that the routing function in the network can transfer packets between the two ends points. Then the end-points must ensure that they can understand and use the encodings in the protocol they are using to talk each other - this can either be implicit from accepting the connection, or explicitly negotiated where the two end points exchange information about their capabilities.

The network must then be checked to ensure that there is sufficient available bandwidth in the network for the video to be sent. This is termed *admission control* and will be discussed later, as it is related to the problems of congestion control and avoidance.

Once both ends of the connection are established, the video data can be sent from the transmitting codec. The codec produces fragments of compressed pictures[7] at a time, signalling in some way to the transmission software that the fragment is available. The transmission software encapsulates these fragments as necessary, adding any timing

[7] We will be following the MPEG convention, and calling each frame of video a picture, to avoid confusion with the use of frame to indicate a time slot.

information that might be necessary to coordinate with other data streams (such as audio), and any hooks that refer to other information available. The fragment is then packetised in a manner understood by the destination eg in RTP format [35]. Narrow-band ISDN has also defined a framing format as H.221 [36] but because it is designed for low delay coding and framing of a bit stream, it is difficult to implement efficiently on a word oriented general purpose computer.

Note that it is possible that the codec may provide whole pictures as output, but this is unlikely to be as useful. Most compression algorithms produce parts of the picture at a time, which can be transmitted as they appear. Transmission of the fragments as they appear reduces delay, and increases the potential for statistical multiplexing.

Packetisation for video is not as problematic as packetisation for audio, because the data rate is much higher (MPEG B-pictures coming in at around 2800 to 3000 bytes). At 512 bytes per packet, six packets would be required to send the smallest MPEG picture. The delay in collecting a full packet of data would be small compared to the delay in encoding and compressing the video - at most a sixth of a picture time, whereas for the low data rate of audio, the packetisation delay becomes significant.

Since we are currently considering a datagram network in which there is no resource reservation, when data is sent into the network, the rate at which the data is transmitted into the network must be controlled in some way to prevent the network becoming congested.

Whilst traversing the network, data packets may be corrupted, lost or re-ordered. Corruption of the bits in the packet is usually detected by checking that a calculated checksum of some sort over the packet corresponds to a checksum sent with the packet. A separate checksum is computed over the network header to ensure that the data is delivered to the correct place. If the packet is corrupted, the data has traditionally been tossed away, since incorrect data was of no value. However, since video data is naturally very redundant, it may be still possible to use the data depending on the degree of corruption and the extent to which the data has been compressed (the higher the compression, the lower the redundancy in the data). Determining the extent of the corruption is possible, but it implies sophisticated checksumming algorithms such as Bose-Chauadhuri-Hoquenghem (BCH) or Reed-Solomon encodings [37]. It is unlikely that the effort in determining how corrupted data could be used will yield good returns, since modern networks have very low bit error rates, due to advanced wiring technologies such as optical fibre.

Re-ordering in the network is a minor problem, since the packets may be numbered with a sequence number as they are sent (either labeling bytes as in TCP, or packets as in TP4). The only problem that may occur is that the size of the sequencing field is insufficiently large so that there is danger of the sequence number wrapping around whilst packets from a previous epoch are still in existence.

A more challenging problem is the need to cope with loss in the network, given the packets may be re-ordered and delayed arbitrarily. The detection of lost packets has usually been achieved by adopting a timeout. If the packet does not arrive within the timeout period then it is very likely that the packet has been lost. Note that this is a probabilistic approach, and the timeout period may be tuned to trade-off probability of misreporting a packet as lost against the reduction in waiting time for learning about packet loss. How large the timeout should be will then depend on the loss recovery technique and the delay requirements of the application.

The loss recovery strategy can utilise a number of techniques, some of which have been used in traditional protocols, others of which are possible only because the required information in video data is smeared across all of the bytes in a picture.

Forward Error Correction: The packets can be buffered and have a Forward Error Checksum applied across all the packets, aimed at recreating burst errors such as occur in a lost packet. The complexity and delay that is required in implementing this may be incompatible with the delay constraints of the application [38,39], especially since the packets will tend to be around 512 bytes in size.

Retransmission: This approach has generally been disregarded in the context of real time data, but whether it is useful depends on the round trip time in the network. If after detection of the lost packet, it is possible to wait another round trip time for retransmission of the original data, then this becomes a viable technique. Assuming that there is a total target delay budget of 500 ms (an estimated maximum beyond which conferencing deteriorates in quality), then it would appear unlikely that the timeout for detecting a lost packet would allow this to be viable in the wide area, where round trip times are in the order of tens to hundreds of milliseconds, although it can be done in local area networks.

Update: Once data has been lost, which will generally correspond to a spatial area in the picture, a request can be sent to the sender to update the area that has been corrupted (assuming that the pictures are encoded in the temporal dimension and therefore cannot be recovered by later packets). This would result in the sender sending up to date information about the area of the picture corresponding to the lost data[8]. It is hoped that the user would only see a temporary blemish that would last on the order of a round trip time. The effectiveness of this technique will depend on the degree of interdependence of the data in the picture and subsequent pictures [40]. This will be explored in the next section.

Image Processing: After a picture has been decoded, it is possible that the image data that has been lost can be approximated using image processing techniques, varying from averaging the surrounding pixels and filling in the blemish with the average through to interpolation of the picture in the frequency domain. Again the efficacy of this will depend on the degree of compression that the data has undergone. The cost in complexity and delay may also be high.

Both of the last techniques require that the information remaining within the picture after the packet has been lost still be sufficient that the picture can still be interpreted meaningfully by a person. Therefore it behoves us to ensure that when data in the picture is lost, it is lost in order of decreasing importance to decoding the picture. This leads us to the concept of hierarchical decoding. It is envisioned that the coding will be in the form of a quality hierarchy, where the lowest level of the hierarchy contains the minimum information for intelligibility, and succeeding layers of the coding hierarchy add increasing quality to the scheme. A trivial example of this would be temporal subband encoding of a 44 kHz audio stream, where the stream is split into subbands where the lowest contains an 8 kHz pcm encoding of the stream, whilst the other bands have information that will augment bands to higher rate (and thus better quality) audio streams[9].

When the data is packetised, only data from within a particular level of the hierarchy would be placed in a single packet (or all the data from combined levels). The low levels of encoding are most important for intelligibility to the end-user, so these packets can be marked as more important than higher level packets, and this information can be used

[8]Thus it can be seen that we are taking advantage of the *idempotent* nature of the video, in that a repeated operation is at least as effective as receiving the original lost data.

[9]Note that low frequencies contain the most important information for human perception of images and sound (frequency being in the spatial domain for images, and in the temporal domain for audio).

by routers in deciding which packets to junk. IP already has precedence bits available in its ToS field, although these are not widely used [41].

A hierarchical coding scheme is also of importance in enabling stored compressed video to be sent at variable rates. Since stored video is not generated by a codec, the codec cannot be told to reduce its rate when required. Instead, depending on the bandwidth available, the appropriate levels of quality could be filtered from the stored compressed video and sent to their destination.

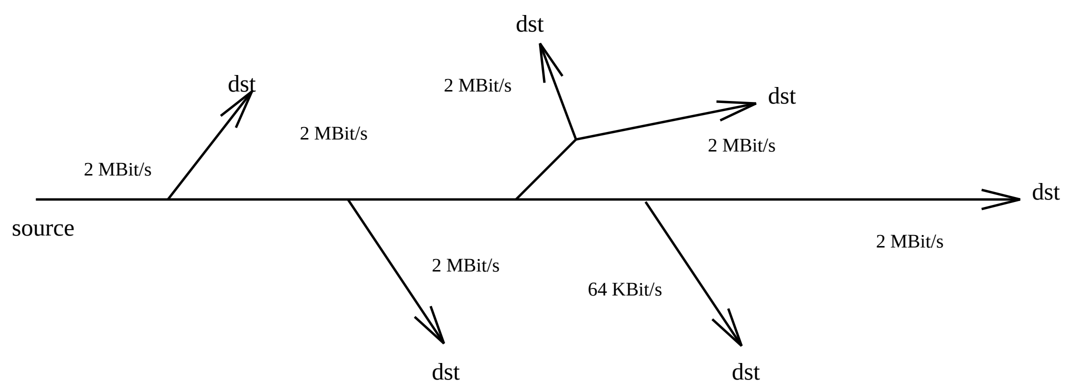


Figure 2.4: Example multicast tree

Multicasting in the wide area network will require the video to pass over a variety of links of possibly varying bandwidth. In Figure 2.4, we observe that one part of the constructed multicast tree has a significantly lower bandwidth than the other branches. This leaves us with the dilemma of how to transmit so that each user receives the greatest possible bandwidth without forcing any link into congestion. Once again hierarchical coding provides us with an elegant solution. If we packetise each level in separate packets, when the multicast stream encounters a low bandwidth path to some of the end-users, the low level packets can be sent down the low bandwidth path, whilst the high level packets are not.

In the current Internet environment this requires that information be gained about the current bandwidth available on branches of the multicast tree, so that decisions can be made about which levels of the hierarchy should be sent (or received).

How the routers discover which packets to filter is an interesting problem in determining the semantics of multicast addresses. It is feasible to send each level in a different multicast group, where the address is used as an indication of which level of the video hierarchy is in the packet. The addressing information then becomes the name of a service that the receiver registers to receive. Alternatively, the routers could be made more intelligent and be able to determine which packets to filter using level colouring of the packets. The solution to the problem must take into account the size of the address space (assigning addresses to levels of a hierarchy will use a globally used finite resource quickly) and the additional complexity that must be fielded in routers and end systems. An approach to this problem based on a priori knowledge of the network is taken in [42].

Now we come to discussing admission control, rate control and congestion control. A common theme running through all of these areas, as well as some of the other topics discussed above, is the requirement to estimate the currently available bandwidth in the network, and to adapt so that the network resource is shared amongst all available users. In admission control, we want to determine if there is sufficient bandwidth in the network so that every user's minimum bandwidth requirement is met. In congestion control we want to ensure that we are not over-using the bandwidth available in the network and forcing queues to grow, whilst rate control is controlling the load that the data is introducing into the network and is the tool used to optimise bandwidth allocation.

If we are using a FIFO packet network, where the only information about the state of the network comes implicitly from what is known about packets traversing the network, estimation of the available bandwidths must be done using patterns of packet loss and the time taken to traverse the network. Note that each end system can only know about what happens to packets it receives, so any algorithm that is used must be conservative in how it reacts to good news about available bandwidth and pessimistic when it hears about bad news.

When there are multiple receivers, the sender must choose from which receiver or group of receivers to determine network state. If the number of receivers is small, then the sender can use all the receivers. However, as the number of receivers grows, the processing load on the sender increases, and the bandwidth used by the return paths to the sender increases, and if the control scheme is to scale, the sender must select a receiver group that is representative. But this implies a loss of information, and so precision in determining the required rate. Therefore, for multiple receivers, we must balance processing and bandwidth requirements against accuracy. We discuss this further in Chapter 6.

The long lived nature of video streams, typically in the order of minutes, if not hours, will make the problem of admission control more akin to human scheduling rather than the usual problems of computer scheduling. This will allow the setup time of the video flow to be on the order of a few round trip times, rather than the minimum possible. This will allow the system to signal its intention to start transmitting

One proposed scenario is to use information about the round trip time for packets in the network, the currently offered local bandwidth and the pattern of loss. The proposal is to use a variation on the Slow Start scheme from Van Jacobson [1] and the Tri-Start scheme from Zheng Wang and Jon Crowcroft [43]. In the proposed scheme, a new admission to the network is aggressive about sending at its minimum required rate for a few round trip times, which acts as signals to the other users of the network by way of increasing their loss rate and their round trip time that someone else requires entry into the network. The other users backoff their transmitting rate to their minimum required rate, whereupon the new user determines whether modeling the loss it sees is small enough to allow the video to be sent. If not, then it desists, otherwise it continues. The bandwidth of the network is used by a slow incremental increase in the rate at which the users are sending, backing off when it is detected that the bandwidth is over utilised. A full description of this scheme and results of simulation can be found in Chapter 5.

When to change the rate at which data is generated can be implicit in a control loop working on the buffer occupancy between the codec and rate control mechanism, or it can be explicit in messages between the rate control and the codec control.

Side information with less stringent delay requirements or higher reliability requirements, such as pro-active codebook updates for pre-recorded Vector Quantised video, may be transmitted out of band in a more conventional reliable transport protocol, such as TCP. Whilst it is conceivable that this could be done in band with the video data, modularity of the system and the performance constraints on the transfer of video data would make it preferable to do this over a separate logical data channel, as long as the information was sent sufficiently far in advance to allow the information to be ready when required.

Secondary characteristics of the network that must be examined are what might happen due to the periodic and continuous nature of the video stream. Will the streams become synchronised, so that the entry of packet bursts into the network occur at the same time? Correlation of entry points of the video into the stream might result in higher loss of packets than was expected and result in poor performance. We touch upon this question briefly in Chapter 6. How will transient data traffic affect the video streams? We study the effects in simulation in Chapter 5, but the modeling of data traffic in a network is still in its infancy, and much work remains before convincing results can be found. Will the continuous changes in quality be tolerated by the user? Since the current ISDN codecs exhibit a higher variability of quality in order to maintain a Constant Bit Rate stream than we assume for our policing mechanisms, the answer is yes for now, but eventually, consumer led demand may require a constant quality for their video.

2.2.1 Video in Resource Reservation Networks

In resource reservation networks, it is assumed that the characteristics of the data flow will be known before starting to transmit, and that the network knows enough about how its resources are committed to say whether the flow should be admitted or not. The problem with this approach is that it is a static allocation of resources. If all users only request their minimum requirement, and stick to their request, the users are not getting the best quality that is available, when the network is under-utilised. If users continually try to increase their quality by requesting new allocations, then when a new user attempts to use the network they may be refused access when the minimum consumption would allow them in, but the attempt to maximise utilisation of the network temporarily fully books the network.

It is possible to ameliorate this problem by splitting requests for resources into hard requests for minimum quality and soft requests for enhanced quality, but this complicates the network resource controller. Alternatively a two tier system is possible, where the minimum quality is resource allocated, and excess quality is shared using an adaptive scheme similar to that outlined above[10]. Which is the most viable approach depends on the expense necessary to field resource allocation networks. If ATM is fielded in a form that allows the user to negotiate bandwidth at a sensible cost and in a reasonably timely manner, then resource allocation will eventually become dominant. But since ATM is unlikely to be widespread in an appropriate form within the next five years, there will

[10]We have ignored the problem of accounting and how the cost of the call is charged, as the extent to which this influences the engineering of the network depends on whether the carrier wishes to make a profit or simply serve the customer. However, the guarantee of a minimum quality may provide the basis of a charging mechanism, both for resource reservation networks, and reservation-less schemes, such as described in [44].

still be a place for adaptive bandwidth allocation.

Chapter 3

Choice of a Video Compression Scheme

From the arguments in Chapter 2, an important facet of any compression algorithm is how well it will fit a hierarchical coding scheme. In particular it is necessary to answer the questions: what are the size of rate changes we can use; how do these affect quality; and how fast can they be achieved. These are all important factors in determining the suitability of a particular compression scheme to use as a network video codec.

Some codecs can utilise side channels to reduce the amount of data that is sent in the primary bit rate, such as code book exchange for vector quantisation. This is another feature that may help in determining suitability.

In this section we will describe the pertinent features of a number of video compression schemes which use both intra-frame (compresses using information only within the current picture) and inter-frame techniques (compresses using information from previous or future pictures), and will attempt to evaluate them for their suitability for adaption to a hierarchical coding scheme and implementation as a network codec.

3.1 Preprocessing of the video

Even before the video is passed to the compression engine, there exist possibilities for adjusting the bit-rate of the video that is finally produced. Obviously, since the bit rate is directly proportional to the frame-rate of the input video, reducing the frame-rate (sub-sampling the frames) will produce a proportionate reduction in the bit rate. Since for many conferencing applications, it is unnecessary to run at the standard 25 frames/second of television, this option is immediately available. The frame-rate which the user finds acceptable for conferencing can be as low as 2-3 frames/second. The required quality for a number of different types of tasks is explored in [23].

Another aspect of the preprocessing that can be used for reducing the bandwidth of the final image is to reduce the resolution of the image. This can either result in a smaller displayed image, or a reduced resolution image, in which the values of the pixels that are sent are interpolated from the surrounding pixels. This method of bandwidth reduction may be quite promising, since the bandwidth is reduced as the square of the linear reduction in size, and the bandwidth of the full image is proportional to the area, rather than the length. How the users react to this is not known as yet.

The quantisation levels that are applied to the input video for analogue to digital conversion of the chrominance and luminance components are additional candidates for adjustment. Since the chrominance is of lesser psycho-visual importance, less information needs to be sent for the chrominance information than for luminance, as is already in the common intermediate format defined by CCIR Recommendation 601. Further reduction in bandwidth can be obtained by more coarsely quantifying the chrominance, since the luminance image is suitable for video conferencing.

Of course, preprocessing can be extended still further, filtering noise out from the image, and so eliminating camera shake. However, this is unlikely to be of great importance, since the processing of the compression engine is designed to eliminate noise from the image as soon as possible.

3.1.1 Interframe Coding

Video is composed of a number of pictures of a scene taken at short intervals which, when shown one after another, appear to run seamlessly. Since the intervals between pictures are short (40 milliseconds for 25 frames/second video), the composition of the pictures will not change very much. The obvious technique to take advantage of this is to subtract one frame from another and send the difference, or *delta*, of the frames. This technique forms the basis of the majority of inter-frame or temporal encodings.

However, this technique is not very efficient since small changes in lighting and camera shake combine to add random noise and small movements to the two pictures, so that the corresponding pixels in the two pictures are different. To rid the deltas of the effects of random noise, the most common technique is to use a threshold above which the differences become significant. Also, if we are differencing, we will probably be working on larger units than single pixels, such as a 8x8 block of pixels, so if the differences of the blocks are compared (using a mean squared error approach), then differences due to random motion will be reduced as well.

Differences between frames that are not due to noise, are probably due to objects in the scene moving or changing. To cope with these, inter-frame coding techniques use some form of motion estimation. The algorithms used for motion estimation are perhaps the most compute-intensive part of any compression algorithm, since they require continual matching of the current frame to previous frames. Current compression standards such as MPEG and H.261 use block matching techniques, where a block in the current picture is moved around in the previous (or future, in the case of MPEG) picture and the position with the best correlation is taken as the old position of the block. The difference between the current block and the old block is taken, and if this is below a threshold value, then the current block is assumed to be the old block under some form of translation. The motion vector of this translation is then sent, possibly with some information about the error between the two blocks. Another technique for motion compensation is the Recursive Displacement Algorithm from Netravali [45].

To adapt the bit rate of the compressed video, it is possible to change the threshold below which blocks are accepted as being translations of former blocks. Increasing the threshold will increase the compression, at the expense of reduced detail and more visible block artifacts in the picture. Another possibility is to increase the search area of the the block matching technique so that fast moving objects are more accurately represented. This may be expensive in processing power and delay, and thus may require a reduced picture rate, which in turn would increase the visibility of the block artifacts.

Using temporal encoding techniques introduces problems for transmission across a network, since they introduce interdependence between different pictures. If a packet from a particular level of the hierarchy is lost, this should not affect any levels below it, since these packets have more important information. Conversely, the loss may have a knock on effect to the levels in the hierarchy above it, since these carry less important information, and if a level is losing packets, then the levels above should already be losing packets. At the same level, if there is dependence between pictures, then a loss of a packet will influence all pictures after that until a fully intra-coded picture is sent. However, this cannot be avoided since temporal encoding works on intra-compression achieved on a picture. From this we deduce that temporal compression at a particular level should not rely on information transmitted previously at a higher level.

3.1.2 Transform Coding - H.261 and MPEG

In transform coding, the image is transformed to a different domain space, in which image co-efficients are not as correlated with each other. There exist a number of transforms which attempt to approximate the ideal transform which would reduce all correlation amongst the coefficients, such as the Discrete Fourier Transform, the Hadamard Transform, as used in the video conferencing tool NV [46,47], or the optimum transform, the Karhunen Loeve Transform [48]. However, the last of these, although optimum, requires knowledge of the statistics of the image to create, and so is unreasonably slow for real-time coding purposes. It is convenient that, like the Fast Fourier Transform, the Discrete Cosine Transform is suitably close to the ideal transform, and is relatively fast to perform. For this reason it has been adopted for use in the H.261 and MPEG video coding standards.

The compression in transform coding comes from the quantisation or discard of the transformed coefficients. The quantisation table that is used is obviously an opportunity for adjusting the bit rate of the final generated scheme - changing to a coarser quantisation table is already possible in some of the emerging H.261 chipsets eg in the GEC-Plessey Semiconductor H.261 chipset [49], where two quantisation tables are available, as well as the facility to select the weighting of the quantisation values on each separate macro-block.

Transform coding typically works on small blocks of pixels, such as 8x8 blocks, and is combined with some form of motion compensated differencing. Both the H.261 video compression scheme proposed by the CCITT for video telephony over ISDN [50] and the MPEG compression scheme for consumer products [51] use a block matching motion compensation algorithm. H.261 uses backwards reference only and suggests that a reference picture that is intra-coded only be transmitted every 132 pictures. MPEG is designed for higher quality and has applications which do not require real-time encoding, so it is able to define use of both a backward reference and a forward reference. Pictures which are intra-coded only are termed I pictures, pictures which are encoded using only backward references are termed P pictures (for Predictive), whilst pictures which are encoded from interpolation of both a backward reference and a forward reference are termed B pictures (for Bidirectional). A typical sequence of MPEG pictures is shown in figure 5.

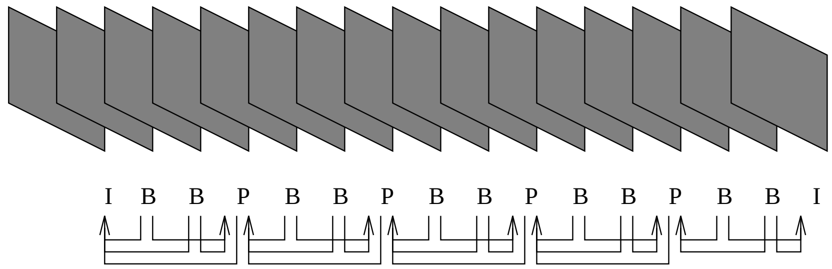


Figure 3.1: A typical MPEG group of pictures (The direction of arrows indicate which picture depends on which)

Mapping of the H.261 encoding onto a hierarchical scheme has already been suggested in a number of papers dealing with error control of VBR video on ATM [52–54]. In these schemes, conventional H.261 encoding is used as the base layer which is said to be given a guaranteed delivery service. Then an enhancement layer is created on top of the base level that increases the fidelity of the coding. In the proposed schemes of Ghanbari and Morrison, the enhancement layer consists of a DPCM encoding of the error between the compressed picture as represented by the H.261 encoding and the original. Results in [53] suggest that this can accept loss in the enhancement layer of up to 10% before the quality is visibly reduced.

MPEG can obviously utilise a two layer encoding scheme similar to that described for H.261, but the division of pictures into I, P and B pictures also allows for the implementation of a coarser layering onto the pictures themselves, where the I pictures are at layer 1, P pictures at layer 2 and B pictures at layer 3, since this is the ordering suggested by their decoding dependencies.

3.1.3 Subband Coding

Subband encoding is based around dividing the picture into its component spatial frequencies, then quantising the coefficients describing the band images according to their importance, lower frequencies being more important than high frequencies. There are a number of variants of this technique, like the pyramid encoder [55], in which the image is successively subsampled and then subtracted from the result to get effective bandpass filtering, or purer subband encodings such as [56] and [57], in which quadrature mirror filters are used to filter the images. These filters allow perfect reconstruction of the filtered image.

Subband encoding is a naturally scalable compression scheme, in that the rate can be adjusted by changing the quantisation tables for each frequency, or by dropping less important frequencies altogether.

The obvious and probably the best mapping of a subband coding to a hierarchical encoding is for each frequency of the subband to be mapped to a separate level. If inter-frame encoding is used, then it must be designed so as not to introduce any upwards dependencies.

3.1.4 Vector Quantisation

If we divide a picture into blocks and then use the coefficients which describe the block as a vector, the vector space in which the block vectors exist would not be evenly populated by the blocks. We can then divide the vector space into subspaces selected to provide equal probability of a random vector being in any of the subspaces. We can then use a prototype vector (which is usually the "centre of gravity" of the subspace) to represent all blocks whose vectors fall into the subspace. By this technique, we can both decorrelate the video picture and quantise in one step.

The Vector Quantisation process consists of:-

1. Codebook generation. The division of the vector space into subspaces is the most processor intensive and important part of the Vector Quantisation technique. It is generally done using a training picture (or set of training pictures) whose statistics are similar to those of the target pictures for compression. A common procedure is the Linde-Buzo-Gray (LBG) algorithm [58] in which an iterative search is carried out to locate the optimum division of the vector space, starting from an initial partition. Other codebook generation techniques use neural networks and simulated annealing [59].

2. A copy of the codebook is then passed to the other end, where a label is used to represent the vector.

3. Matching. The image is divided into blocks, and then is mapped to the vector in the codebook that is closest to it, where the representation is by the label in the codebook.

4. Decoding is then by looking up the label in the code book and using the given vector to represent the picture on the screen.

Enhancements to this technique include using a different codebook for different types of vectors, such as for edges and others, or by updating the codebook as the statistics of the image change [60].

Vector quantisation is used in the Picturetel proprietary SG3 algorithm, which uses a form of motion compensated hierarchical vector quantisation, implemented in software on custom programmable digital signal processors [61]. The "Cell" encoding scheme from Sun Microsystems uses vector quantisation for the colour maps - cells of 4 by 4 pixels are mapped using a block truncation code [62].

The rate of vector quantisation can be changed by changing the codebook (a larger codebook requiring more bits to be sent), or by adjusting the size of the blocks that are encoded (which requires different codebooks). However, replenishment or updating of the codebook at the decoding end can be done out-of-band from the data transmission. All that is required is that the new codebook be received by the far end before it is used, which can be achieved by any reliable message passing system. Labeling of the vectors will then require additional information about which codebook they come from.

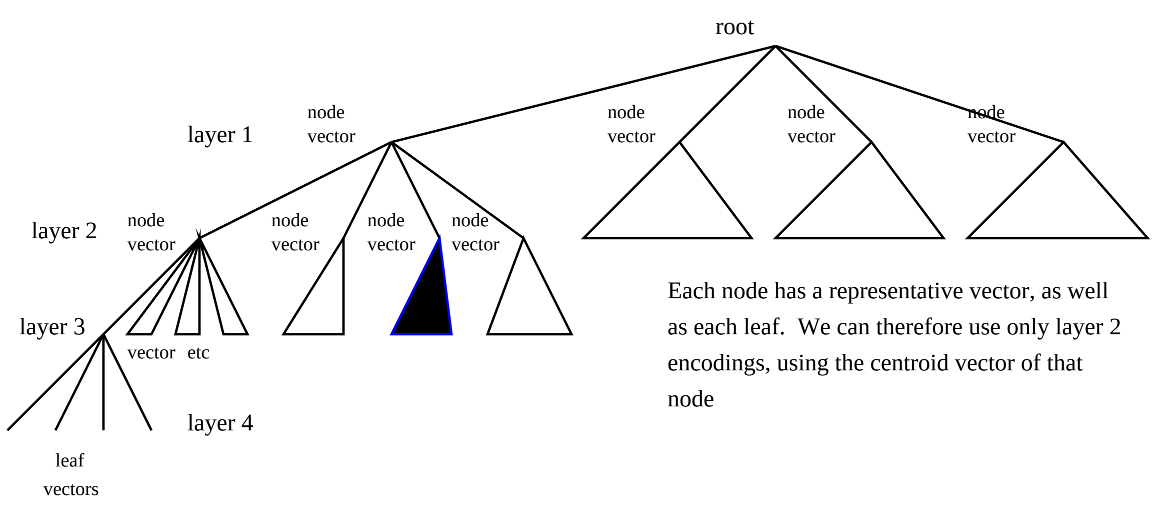


Figure 3.2: Example of a hierarchical vector quantisation tree

One technique for a hierarchical mapping of vector quantisation is to compare the quantised picture against the original picture and then to send the difference as an additional level. Another possibility is to split the vector space into subspaces at several different layers, so giving a hierarchy of codebooks and lookups that must be performed. Each block would then have a multi-part label, which could be truncated as required. As an example, see Figure 3.2. Imagine that we wished eventually to split our vector space into 256 subspaces. If at each level we split the subspace into four subspaces according to codebook generation technique, we would then have four levels of hierarchy, with a centroid vector representing each subspace at each layer. This would give a codebook of size 340 vectors, using 2 bits to encode each block for each layer.

3.1.5 Region Encoding

Region encoding is a technology that can either be applied on its own, or in combination with other compression schemes. Region encoding divides a picture into regions and then either sends the mean grey-scale (or chrominance and luminance for colour) and a description of the region [63]. Alternatively, the picture can be split into foreground and background scenes, where more detail is compressed in the foreground scenes than in the background scenes, using conventional techniques like motion compensated quantised DCT or vector quantisation [64]. This technique may become important in human communication in that it can use *a priori* information about the important features of the image - the face - to enhance the image. Alternatively, the user might be able to delimit regions that they want to enhance, and region encoding might select the objects within the image that should be tracked.

Intel's Digital Video Interactive (DVI) is based upon this concept, [65,66]. The image is analysed into regions, which are then also split into regions, and so on, until the regions can be mapped onto certain basic shapes to fit the required quality or bandwidth. The chosen shapes can be reproduced well at the decoder. The data sent is a description of the region tree and of the shapes at the leaves. This is an assymetric coding, which requires large amounts of processing time to encode. Full quality DVI is described by Production Level Video (PLV), which is produced at dedicated centres, whilst compression in real time is achieved at limited amounts of effectiveness by the Real Time Video (RTV) standard.

Selection of bit rate can be achieved by adjusting the thresholds of what constitutes a boundary and what does not for the pure region encoding scheme, whilst for a shape fitting scheme like DVI the exactness of the shape matching can be adjusted to allow regions to be approximated to a greater or lesser extent. The exact nature of DVI is, however, proprietary, so speculation on techniques to make DVI scalable is necessarily constrained.

If we are using a tree to construct our coding scheme, then obviously each layer of branching of the tree can be fitted to a level in a hierarchical encoding scheme. Alternatively, thresholds can be set on the sharpness of contours when detecting the edges of each region, so that we switch to having regions which are at various layers depending on the degree to which they are identifiable.

3.1.6 Lossless Compression

After the picture has been compressed using one of the above compression schemes, it is represented by a set of co-efficients. However, since the compression scheme for practical purposes must decompose the picture and treat it as separate units, such as the blocks in H.261, there will still be redundancy in the co-efficients. These co-efficients can therefore be further compressed. Typically, this will be through the use of a Huffman encoder, where the statistics of a number of frames are used in creating a set of codes for a number of different co-efficient and run-length pairs [37].

3.2 Comparison of the Video Compression Schemes

In selecting a compression scheme for use over a network, we have a number of factors that must be taken considered:-

1. Efficiency: The compression ratio for a given picture quality must produce a bit rate that will fit within typical current digital network bandwidths - from 64 kBit/s upwards.

2. Cost of the required processing power: Since we are aiming to place the codec inside of each workstation, this requires that the codec algorithm be cheap in monetary terms. The cheapest possibility would be to have an algorithm that could be executed on a general purpose processor, and allow for the increase in power of processors to help us, but this may not be possible. Also, the image processing required for compression may be useful in other applications, and so the cost of a special piece of hardware may be amortised across the workstation.

3. Suitability for Hierarchical coding: Hierarchical encoding is required for protection against errors and for use in bandwidth limited networks. Some of the compression schemes mentioned above are naturally scalable (subband encoding) whilst others can be made scalable by hierarchical encoding of error signals left after coarser encodings.

4. Symmetry: Our model application is for workstation-based video conferencing. This requires a workstation to decode remote signals and to encode the output of local cameras - typically for an n-way conference, this will amount to n decodes and 1 encode. Whilst a degree of assymetry is permissible in the compression scheme, since we are likely to have more remote signals than local signals, the encode operation must still be possible with the resources available on the workstation.

5. Granularity of scalability: In the ideal case, the compression scheme would be tunable to any given bandwidth, but it is likely that this cannot be achieved without compromising the efficiency of the codec.

6. Reduction in rate due to side information: We might be able to reduce the data rate of the stream by judicious use of out of band transfer of side information, such as for slowly changing background, or by the transfer of codebooks.

The comparison we have adopted is a tabular attempt using a ranking of the four intra-frame compression schemes discussed for the 6 factors above, weighted as indicated, along with a short discussion of the reasoning behind our ranking below. The use of the decision table is an attempt to focus attention on the issues which have been debated above.

The relative efficiency of the compression of the techniques are similar, apart from the real time version of DVI, which is currently inferior to the others (although this might change). For cost, both software and hardware are currently available in the market place for the DCT algorithms and the region encoding, and the amount of processing power required is helped in the case of region encoding by having a processor specifically designed for it [67]. Both subband and vector quantisation require specialist signal processing capability at the moment to perform adequately.

The hierarchical efficiency of the four schemes is obvious from the descriptions above - subband being naturally hierarchical, with VQ being hierarchical with little effort, whilst

Factor	Transform (DCT)	Subband	Vector Quantisation (VQ)	Region (DVI)
Efficiency (3)	2	2	2	1
Cost (2)	3	2	2	3
Hierarchy (2)	1	3	2	1
Symmetry (1)	3	2	1	1
Granularity (1)	1	2	2	2
Side Information (1)	1	1	3	2
Weighted Totals	19	21	20	16

Table 3.1: Decision table for the compression techniques (larger ranking is better)

the other two codings must be forced into a hierarchy, based essentially on differencing errors (although this may not be the case for DVI). VQ is the most obvious assymetric encoding, along with region encoding. Subband and transform encodings are both reasonably symmetric. Granularity depends very much on the implementation, but it appears that the parameters of all but transform encoding will be easy to adjust. Finally, only VQ and region encoding have side information, which accounts for their ratings.

Attempting to distinguish the best of the coding algorithms from a network perspective, it is unsurprising that subband encoding comes out winner, with VQ just behind. These are the most naturally hierarchical of the encodings, and it is this feature that best suits a compression scheme for transmission over a lossy network. The work of Karlsson and Vetterli reported in [68] reaches the same conclusion about the efficacy of subband coding for networks.

3.3 Conclusion

In the previous and this chapter we have examined the requirements of users of compressed video, and examined how these can be used to increase the flexibility for transmission of video across ordinary packet networks. Several requirements of the compression schemes have been brought out in the synergy of the user and the network, foremost of which is the necessity of a compression scheme that can be hierarchically encoded for multicast transmission and for error resilience. Several schemes have been examined in the light of these requirements, and it was judged that subband compression and vector quantisation are likely to be the most effective as a packet network video compression scheme.

Chapter 4

Congestion Management in Packet Switched Networks - a survey

Before we introduce our approaches to solving the problems of congestion for variable bit rate video in packet switched networks, we shall first introduce the conceptual framework in which these schemes have been designed. We first describe our model of the network, its services and the processes that must be managed within the network to provide these services. We shall then describe an abstraction useful in thinking about the techniques used to manage these processes and in determining when they are necessary. We shall describe other work in managing these processes within the context of this abstraction.

4.1 A network service model

A network can be viewed as a network of interlinked routers where packets enter the router, the next hop of the route is determined from information in the packet and in the router's tables and the packet is placed on an output queue to be transmitted on the link to the next hop router. The link may be anything from a multiple access Local Area Network, such as Ethernet or FDDI, through Non-Broadcast Multiple Access (NBMA) networks, based on technologies such as ATM, through to a point to point link such as a leased line. Applications run as processes on computers which are connected to the network. These application processes need to communicate information to other processes connected to the network.

The problem of network design is to offer the required services to the applications at the minimum overall cost, where cost is not only the capital costs of buying and installing the bandwidth, but also of maintenance and upgrading and introducing new technology and so on.

The services to be provided by the network to the applications can be described as follows:-

Move packets from one application process to another This is the basic service, and requires routing functionality within the network. In a multiply connected network in which there is more than path from one host to another, the selection of the route can be intimately connected with the provision of Quality of Service to the application.

Meet the deadlines of the packets Applications can have deadlines imposed on the delivery of their packets. These deadlines can be hard in which the packets must be delivered before a given time, such as in real time process control [69], or conferencing applications with no adaptive buffering, such as codecs designed to be attached to the telephone network e.g. telephones or ISDN video codecs. Another common term for this is *guaranteed* delay. Alternatively, the deadlines may be soft, in which the packets should be delivered as quickly as possible to allow the application to work efficiently, such as *adaptive* buffering conference applications such as the Visual Audio Tool from LBL [70] or IVS from Inria [4]. Finally there may be no deadlines for packets from traditional data applications such as FTP, termed *elastic* applications in the glossary of the integrated services working group of the IETF [71].

The delay experienced by the packet in the network consists of the transmission and propagation delay of the packet on the output link, the processing delay from the router, and the queuing delay waiting for the output link. The fixed delays from the router and the output link can be controlled by the routing process, using the known bandwidth and delay parameters as part of the routing function, whilst controlling the queue lengths is the major concern of this thesis, and the major problem that must be addressed by what we shall term Congestion Management. The various facets of Congestion Management will be described below.

Meet the throughput requirements of the flows Some applications may have fixed throughput requirements for the data stream, such as fixed bit rate video codecs [4], or telemetry streams from monitoring devices, whilst others may have a minimum rate that must be satisfied such as bandwidth adaptive codecs. Finally, conventional data applications have "just give me as much throughput as possible" requirements. The extent to which these requirements can be satisfied by the network are determined by the bandwidth of the network and by the data rates offered by the other applications. The application and the network must have some mechanism for exchanging details of the rate the network can accept and the rate that the application wants, either explicitly or implicitly. Ensuring that the traffic actually offered by the applications allows the network to meet the requirements of the applications using the network is the provenance once again of congestion management[1].

High reliability The packets given to the network must be delivered to the destination with high probability. The current state of transmission technology produces an error rate from physical causes that is very low, making the loss of packets from buffer exhaustion in the routers the primary cause of packet loss. Thus once again, QoS management is necessary to control congestion and to meet this requirement. Note that the routing mechanism can cause transitory packet loss if it changes routes often.

Low cost As is the case with all engineering, the choice of design must factor in the cost. All congestion problems can disappear with the provision of very high bandwidth links anywhere congestion might appear, but this can become prohibitively expensive very quickly. The basis of all engineering is selecting from the solution space a solution which meets requirements in a cost effective manner.

Fairness The definition of fairness is problematic. When the demand exceeds the finite amount of resources, as happens in the case of congestion, then some consumer(s) will not get their demands satisfied. For a system to be fair in this case, then either

[1]We have carefully avoided using the term admission control here. Whilst a formal admission control scheme can serve these requirements, the goal of this thesis has been to determine the minimum engineering necessary to evolve existing computer networks to multi-service networks, and we wish to avoid biasing the mind of the reader.

there must be no systematic refusal to the user, such as might occur if they occupy a disadvantaged position in the network topology, or there needs to be some scheme to allocate priority, based either on a common goal or using some monetary measure to unify importance attached to a particular demand. Since it is unlikely that in the heterogeneous network there will be very much commonality of goals across organisations, it appears likely that money will be used to justify decisions, and the network will have a pricing scheme designed to control congestion. An obvious policy is to charge users more in periods of congestion, such as the smart auction suggested by Mackie-Mason and Varian [72,73], where each user bids a price for their traffic to get through, the bids and their associated bandwidth requirements are ordered and the price chosen is the bid in the list where the bandwidth demands of all higher prices sum to the available bandwidth. All users who are prepared to pay a higher price can use the bandwidth and pay the chosen price.

Shenker in [74] defines a general utility function for each user and then shows using game theory that at a single switch, when we can't assume co-operative behaviour, it is only a fair queuing scheduling or one of its derivatives that can guarantee that the resultant Nash equilibrium is an approach to a Pareto optimum[2]. This follows from the degree of isolation that results from fair queuing. Fair queuing works by reflecting back the congestion that the user creates by reducing the available resources solely for the congesting user. In economic terms, this provides incentive compatibility for the users to adjust their resource demands to match their requirements.

In this thesis, when there is too much demand for the resources, resolution is punted to being a policy issue for the application and for the administration of the network, and we only attempt to alleviate the situations in which the schemes systematically disadvantage a particular user.

Finally, any technology changes in the network must co-exist with the existing technology. Since the network is not owned by any single entity, it is difficult to co-ordinate a single transition so technology must not require all network components to support a feature. Another goal for any technology is to be scalable and robust. In particular, transient failure of components should not cause catastrophic failure, leading to the use of

[2] A Pareto optimum is a distribution such that there is no other allocation which any user would prefer.

soft state in components which can be reconstructed easily after rebooting a component.

4.2 Routing for Quality of Service

The use of the expected Quality of Service (QoS) in selecting a route is not a new idea - however, the problems of maintaining and selecting routes within the Internet have been sufficiently difficult to date that QoS has not been widely used as a factor in selecting a route[3]. But with the emergence of technologies that can support policy routing such as the Border Gateway Protocol (BGP) [75], the Inter-Domain Policy Routing [76] and their successors and derivatives such as IDRP and the scalable Inter-Domain routing architecture from Estrin and Rekhter, the use of QoS in selecting a route now becomes possible. However, the amount of processing required to compute multi-metric routes still remains very large, since it can be shown to be NP-complete [77]. A worse problem is that the routing is designed to be dynamic, and adapt to problems within the path. If a route is selected that can support a specified QoS, and the route then fails, we are left with the problem of routing along paths which may not support the QoS whilst we recompute a path which can support the specified QoS.

When we combine dynamic re-routing with resource reservation, then the problem becomes more difficult again - loss of a link on a path means that the resources should be reserved on the new path (if possible) and the resources freed on the segment of the old path that we are no longer using. This problem vanishes if one can assume that re-routing is very rare, as the ATM world appears to, but in the Internet, re-routing takes place rather often; anecdotal experience with the Mbone suggests that a route changes every few minutes. This will obviously present problems for placing resource reservation technologies within the current Internet.

4.3 The regimes of Congestion Management in the network

In considering the mechanisms that we can use in providing congestion management, we borrow abstractions from control engineering. We regard the network as a black

[3]OSPF can use static information to construct routing tables optimising delay or bandwidth, or any available static metric within a domain

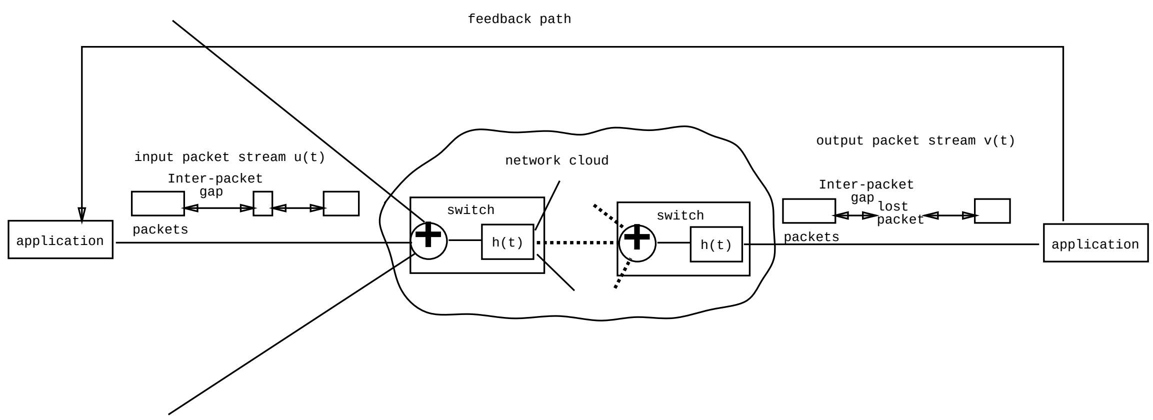


Figure 4.1: Network Control Model

box into which we place an input stream, characterised by the size of packets and the inter-packet gap (a sequence of vectors). The output stream must match the delay and throughput goals of the application, i.e. the input stream must not be over-distorted beyond the capacity of the rectifiers at the application. The control diagram of the network as perceived by the application can be seen by regarding the network cloud as the black box in Figure 4.1. Note that we are assuming a feedback path. Typically this feedback path would be provided by control messages from routers in the network or by messages from the destination and would be used to derive information about the behaviour of the network. The transfer function of the network is a function both of static parameters such as the bandwidth of the links, and dynamic parameters such as the routes and data rates of the flows currently using the network. This implies that the transfer function is time varying. An alternative formulation of the control problem from the view of a router can be seen inside the network cloud in Figure 4.1. The incoming streams that are routed through the router are mixed together by the queuing mechanisms and pass through the black box of the router. In this case the transfer function of the router is known[4], and we know the set of input streams that are permissible before distortion sets in. Finally, the most sophisticated viewpoint allows us to adjust the transfer function of the router by changing the parameters used in manipulating the packet stream. Note that although we sum the incoming stream, this summation does not imply that the

[4] In the case of FIFO queuing, the function is not mathematically tractable, but its behaviour can be modeled; for Fair Queuing and other scheduling mechanisms based on Generalised Processor Sharing, the transfer functions are mathematically tractable

individual packet flows cannot be treated separately.

Using this control model, we have three approaches. We can apply open loop control to the network, by using our knowledge of the transfer functions of the component routers and the individual flows that are already passing through the router to restrict the set of flows to those whose QoS requirements we can meet. This abstraction matches the flow admission control schemes. When possible, open loop control is used in conjunction with manipulation of the transfer functions of the routers by adjusting the algorithms used to schedule packets on the output queue. This allows us to downgrade flows with flexible QoS requirements to allocate bandwidth and processing power to flows with inflexible QoS requirements and thus allow more flows in. Finally, if the QoS requirements of the applications are flexible, we can apply closed loop control around the network, where each application adjusts its input packet stream to react to the changing transfer function of the network. These are the three modes of congestion management we will examine.

The dynamic nature of the transfer function of the network comes from the combination of all the queuing processes on the constituent links. The queuing process is a non-stationary process whose statistics change due to three factors:-

- The non-stationary nature of the packet streams. This has been observed by a number of studies such as Leland et al [12]. Note that if the observation of the self-similarity of the streams is correct, then the queues will change over a wide range of time frames.
- The entrance and termination of packet flows. The startup and completion of the packet streams generated by the applications will change the statistics of the queuing process at a coarse time frame.
- Changes in traffic rates due to any control algorithms implemented either by the network or by the application.

The queuing processes are fundamental to the provision of QoS. If no queues exist in the network, then the packets pass through the network with constant delay and no loss. If queues exist, then the delay experienced by a packet will be variable - to offer delay bounds, then the queue sizes must be controlled to ensure that the total queuing delay does not exceed the allowed bound. Finally, if the queues are allowed to grow unbounded, then eventually the buffers of the routers will be exhausted and packets will be lost. Thus all congestion management schemes attempt to control the queuing processes. This provides us with another way to view the congestion management schemes; categorised by the techniques and the time frame upon which they hope to influence the queues.

4.3.1 Admission control

We view admission control as a process where the source offers a description of the flow it will be introducing to some "network entity" and the network entity then assesses whether the flow can be admitted into the network and returns an answer to the source, possibly requiring the source to modify the proposed flow. The details of this view can be implemented in a number of ways. The description of the flow can be of varying complexity. At the minimum it must contain the destination address, so that the network entity can search for possible routes. With a set of routes, the network entity must then apply its decision making algorithms using the information supplied with the flow description and the information known about the routes. Thus, the necessary information in the flow description depends upon the decision algorithms. Craig Partridge attempts to second guess which algorithms will be used in [78], requiring parameters such as the mean rate, the peak rate and the burst rate for leaky bucket specifications, and delay requirements. If a route can be found that supports the flow specification, the route, either as a source route or as a route identifier such as a Virtual Circuit Identifier is supplied to the source. Congestion is avoided by not allowing a flow into the network that would overload the links.

The admission control process can be a negotiation process: when the network entity cannot meet the requirements, it provides a description of what it can achieve, and the application decides whether to take that.

The mathematical basis upon which admission control decisions are made has produced two equivalent definitions. "Equivalent bandwidth" comes from the work of Kelly [79] and Guerin et al [80]. Sources are modeled as either on and off at their peak rate, and are characterised by the peak rate, the mean rate and the duration of an on period. Under certain assumptions (exponential distribution of burst periods), an equivalent capacity can be calculated for the flow according to the required probability of buffer overflow. Ensuring that the allocated equivalent capacities to flows doesn't exceed the available bandwidth will then prevent queue overflow. The equivalent bandwidth calculation and similar calculations have been used as the basis for a number of real-time admission schemes, such as [81,82].

Alternatively, one can work using the delay bounds obtained for a particular form of fair queuing using the work of Parekh on Generalized Processor Sharing [83]. Generalised Processor Sharing is a scheduling strategy in which the packets are served according to the share they have been allocated. Since packets are indivisible, the quantum of sharing used is the packet, and GPS is approximated by Packet-by-Packet Generalised Processor Sharing (PGPS). Parekh shows that PGPS leads to guaranteed delay bounds across arbitrary topology networks when flows are constrained by leaky bucket characterisations, and that the delay bounds can be calculated solely in terms of a leaky bucket characterization of the flow. Thus given the leaky bucket, one need only calculate the delay bound through each switch to generate delay bounds, and therefore calculate whether one can support the flow. However, Parekh does require that each switch support PGPS or some variant scheduler which supplies the requisite sharing and quarantining of flows.

The characterization of the network used to calculate whether the flow should be admitted can be based either on the history of flow admission requests made, or on measurements of the current state of the network. Whilst the first has the advantage that the sums can be done off-line from the network, the actual state of the network may not be close to the model of the flow requests, so most work suggests using the measured state. However, taking measurements complicates the engineering of the network. Another question is what granularity of measurements should be made, and how should the measurements be abstracted? Lazar et al in [84] suggest that there should be an abstraction mechanism for taking measurements at the level of the packet scheduling and presenting data to the admission control process.

The admission control scheme requires an interaction with the routing process and a (possibly highly distributed) network entity to calculate whether the flow should be admitted. If the duration of the flow is not much longer than the delay in setting up the flow, then network users may be dissatisfied with the overhead in using the network.

Admission control also requires the flow to be characterised in some way. Whilst that may be possible for pre-recorded data, or devices with known output characteristics, there exist a number of applications, most notably real time compressed video, for which it is difficult to characterise the output data rate. In these cases, the tail may end up wagging the dog, as the application varies quality so as to keep within the contract made with the network.

Finally and most importantly, the admission process is intimately connected with the route taken. If the route changes due to component failure, then new decisions must be made as to whether the flow can still be supported. If the flow admission process is explicit, then the route change must be recognized and a new request must be made.

4.3.2 Flow and Congestion Control

When applying closed loop control to influence the input rate of sources according to the conditions in the network, there are a number of assumptions in any congestion control scheme. It is necessary for:-

- the sources to be well-behaved and comply with the scheme, so that the utility of the whole is improved;
- the scheme to maximise utilisation by taking up unused bandwidth;
- the scheme to reduce and control long term queue lengths;
- the applications to be adaptive in the required data rates;
- the scheme to be stable across the multitudinous sources and switches which make up the network.

Whilst the transport protocols used in networks are under some form of administrative control, such as the TCP implementations in various forms of Unix and for other boxes[5], compliance with the protocol for the common good can be assumed. However, with nearly all the schemes described below, a single source can achieve a greater proportion of the bandwidth by disregarding the appropriate behaviour and forcing other sources to reduce their share of the bandwidth. In future networks, the users cannot be presumed to be altruistic, and policing must ensure their compliance with the scheme, either at the boundary to the network, or by isolating the data streams within the switches. In all the work described below, the above points are generally covered.

[5]TCP implementations are an interesting case to consider. The slow start algorithm in TCP is not necessary to produce inter-operability, but the Host Requirements RFC [85] makes implementation mandatory. A carrot to implementors is that a complete implementation is freely available and can be used. One can also appeal to the meta-game where if an implementor sells a TCP without congestion control, then in the presence of congestion due to a large number of these implementations with congestion control, they all lose. The only explicit stick is general deprecation from the community.

The original primary use of flow control loops was to prevent buffer overrun at the receiver, such as in X.25 and TCP, and was based upon credit based flow control schemes as used in nearly all ARQ protocols. The problem of congestion was recognized, and the flow control loop was subverted to use in protecting the network against the sources. An excellent survey by Gerla and Kleinrock of this early work can be found in [86].

The increased performance and wider applications of computer systems motivated researchers to generate new transport protocols in the eighties, such as NETBLT [87], VMTP [88] and XTP [89]. These protocols were designed to match the rates of sources and sinks, and so used rate control schemes. But both the proliferation of networked computers and the increased performance led to congestion within networks, and the pioneering work of Ramakrishnan and Jain [90] and Van Jacobson [1] used credit based flow control schemes to adapt sources to the available bandwidth in the network. These schemes reduced queues in switches within the network by using signals indicating queues were building in the switches. The signal was an explicit bit in the header fed back by the receiver in the DECbit scheme, or lost packets for TCP. The control action taken by the sources was an additive increase when no indication of congestion was received and a multiplicative decrease when a congestion indication was received. A number of later theoretical studies have proven that this is stable under various conditions [91–95].

The work of Jacobson and Jain inspired a new wave of congestion control schemes, where the basic techniques for data traffic congestion control were improved and honed. Jain used delay as the feedback signal for an enhancement on the DECbit scheme [96]. Wang and Crowcroft supplement the basic Van Jacobson Slow Start with information about what the optimal data rate is [43]. Rose modified the DECbit scheme in [97]. In the Vegas TCP, techniques from Tri-S and other implementation tricks have been used to produce a "better" congestion avoidance scheme for TCP [98].

Other work has factored in the expected large variation in the bandwidth of links on the network and the new "real-time" media streams that would use the high bandwidth networks, such as Kanakia adapting a video codec in [99]. There have been a number of approaches to the problem using control loop theory. Keshav in [100] demonstrates that proportional feedback can be used to maintain stability when fair queuing is used in the switches to isolate the flows. Benmohamed in [101] demonstrates the theory behind the design of a controller for a single queue, but using information that it is infeasible to derive in a real network. Park in [102] independently derived a similar model to that described in Chapter 5, but uses a different congestion signal.

Other approaches taken include the Loss-Load curves of Williamson for rate based schemes [103,104], and the mechanisms proposed for congestion control in the Available Bit Rate service proposed in the ATM Forum, excellently summarized in [105], where many of the techniques used in the above approaches have been transferred to the ATM world.

We have discussed congestion control as an end-to-end phenomenon. However, it is possible to implement a hop-by-hop scheme. In this scheme, information about the queue lengths are fed back hop by hop, and each router takes action to reduce its output rates. This results in faster reduction of queues at the bottleneck link, since the preceding nodes reduce their output rates faster, but at the expense of spreading the queues back along the chain of switches to the source as in [106–108]. Whilst in the local area this may be acceptable, since utility is a shared enterprise function, delaying all cross traffic preceding the bottleneck in the wide area may be unacceptable to users, since they may have to reduce their traffic rates unnecessarily. Further, consideration of the inter-linked control loops which have the output function modified by the succeeding router produces a combined transfer function that has a higher order characteristic equation then a single end to end loop, and in general reduced stability - although still stable. Along with the additional complexity that has to be added to each switch to implement a hop by hop approach, these reasons have motivated the end to end approach taken in the work in this thesis.

When the source is sending to a group of receivers, then the problem becomes more complicated since the sender has a choice of control loops in controlling its rate, one for each receiver. In order to scale with the number of receivers, both the processing load per receiver, and the bandwidth used to return information should remain approximately invariant. There has been no published work to date on how the correct control loop should be selected. We describe one approach in Chapter 6.

4.3.3 Scheduling

The ordering of the packets in the queues within the switches is the fundamental mechanism to achieve control over the offered QoS. A switch receives a packet, looks up information within the packet to decide which queue to place the packet on and adjust the queue management state accordingly. Then a scheduler decides from which queue to send a packet when the link becomes free. At the minimum a switch must use information to decide on how to route the packet, either using an address in the case of datagram protocols such as IP, or by a key to looking up state within the switch in the case of the VCI in ATM. A distinction between a key and an address is that the address is unchanged throughout the passage of the packet from source to destination, whilst the key is changed at each switch to suit the lookup engines of the next hop switch[6]. However, since the switch must read information from the packet, it could also use this information in managing the queues.

The scheduling of packets within a switch adjusts the timing relationship between packets both within the same flow and across flows. Using the analogy from the control model, the function determining the adjustments is the transfer function of the network. Individual flows with real time requirements can be protected from congestion by using separate queues and thus isolating them from the congesting traffic, whilst other traffic may suffer increased delay or even loss. With the use of an appropriate admission control policy and scheduling techniques, congestion problems can be avoided.

Since the scheduler must be invoked for every packet, the performance of the switch may suffer if the scheduling code is too complex. Historically, the IP world has made the engineering decision to optimise the switching path and use very simple scheduling mechanisms, since no guarantees were required about loss or re-ordering of packets. But the increased speed of processors and memory has increased the processing possible per packet and more complicated scheduling mechanisms are now possible without affecting the switching speed. As an engineering problem, the choice of scheduling algorithm is to find one which protects flows, doesn't impact switching speeds, and can work with other primitive FIFO switches.

A number of scheduling mechanisms have been described in the literature that can be used to protect traffic with delay requirements from best effort data traffic, described well by Zhang in [17]. Golestani's "Stop and Go" mechanism gives guaranteed delay by the use of slots and a non-work conserving scheduler [109], whilst the equivalent mechanisms of Virtual Clock [110] and Fair Queuing [111] give protection by isolating the flows from sharing the jitter from each other. Other variations include fifo+ [112],

[6] This distinction is a little arbitrary, since many addressing schemes use variations on source routing where a pointer is advanced in the packet to the next address to be used in switching the packet. In this case, all necessary keys are put into place by the source.

in which packets are re-ordered according to how far they have drifted from their target transit time, Earliest-Due-Date used in the Tenet work [113], where the packet with the earliest deadline is always served first and Hierarchical Round Robin [114] in which priority is used as the basis for the hierarchies in a round robin server.

Besides ordering the packets so as to spread the delay appropriately, another option open to the switch is to drop packets. Indeed, if the switch runs out of buffers because of congestion, then this is the only option. The naive approach when buffers are exhausted is to drop the last packet. However, since dropped packets are used as a signal of congestion, the switch can do better by dropping packets before buffers are exhausted, and can select the flow to drop the packet from so as to cause them to reduce their sending rates. Floyd and Jacobson explore the potential performance improvements that come from implementation of the Random Early Drop policy in [115]. In the ATM world, the imposition of higher level structure on the cell stream means that when one cell is dropped from the structure, all other cells can be dropped from the structure. This idea came from the Fairisle project [116,117] and the practical effects have been explored by Romanow and Floyd in [118].

Common to nearly all the above approaches is the classification of the service according to the individual flow. This is a major weakness, since this does not scale well on the backbone links. In the ATM world, this problem is tackled through the use of Virtual Paths, in which Virtual Circuits with similar Quality of Service requirements are identified by a single Virtual Path Identifier and are treated the same.

In the world of datagrams, the only equivalent approach yet identified is Class Based Queuing from Van Jacobson and Sally Floyd. This is discussed more fully in Appendix A, where we describe implementation experience of a CBQ router.

4.4 Conclusion

We have given the basic model and abstractions which have been used for the work in this thesis. We have described the service requirements of the network, and then described three areas in which control can be asserted to meet these service goals in the presence of contention for resources, working at different time-scales. The first area working at the longest time scale are open loop control techniques based on admission control. Operating in the regions of round trip times are closed loop control techniques where sources adapt their rate according to signals of how congested the network is. Finally at the smallest granularity of time is the adjustment of the transfer function of the network by adjusting the parameters of the schedulers within the switches. We use this model for discussing related work.

In Chapters 5 and 6, we build on datagram technology and describe closed loop technologies for controlling the effects of congestion. One of the constraints in our problem space is that the network is administratively diverse, and thus we cannot assume the existence of a commonly agreed upon admission control scheme for all switches. However, we do believe that sufficient numbers of end-systems can implement algorithms to protect the network, based on the experience of Slow-Start in TCP. We expect schedulers to be implemented in a piecemeal approach over the network, and whilst they will help, assuming that they can correctly identify flows with real-time requirements, sophisticated schedulers are not assumed in this work. We describe the packet classifying and scheduling techniques of a CBQ router in Appendix A.

Chapter 5

A Combined Admission and Congestion Control Scheme for Variable Bit Rate Video

5.1 Introduction

The[1] advent and rise of multimedia technology will increasingly require the transmission of continuous media data over computer networks. The unfettered introduction of these flows into the network, and subsequent unconstrained transmission would lead to congestion and unacceptable service for all users of the network. Many researchers have suggested that the network should move towards a resource reservation model, where sources request reservation of resources from the network, characterising the load that they will offer in some way. However this requires intelligence and sophistication in the network, a departure from the prevalent model of a computer network which will take a significant period of time to come about.

In this chapter we attempt to reduce the required intelligence of the network (and thus the cost) by devolving responsibility for protecting network resources onto hosts - the host determines the state of the network and decides whether it should introduce an additional flow into the network. The host is then responsible for adjusting the source coding rate to share the available bandwidth in a fair manner. Whilst this may not be the most efficient way of protecting network resources, it is easily implemented in host

[1] An earlier version of the work in this chapter appeared in [44].

software without changing the philosophy or infra-structure of the data network and so economically attractive.

To accomplish this we have developed a more sophisticated control theoretical model of the network than the Slow Start algorithm [1] and the DECBit scheme [96]. We posit a second order proportional control scheme, where we adjust the global proportional controller constant according to the number of users in the network. We also constrain the variances of the incoming data flows so that the variance of the data stream as a whole is bounded (assuming that the data flows are independent). We show that when the network becomes congested this scheme remains stable. The only modifications that are required to the routers are that they recognise designated bits in the network level header indicating low delay packets and low importance packets which can be preferentially dropped, such as the ToS bits in IP [41].

We assume that the continuous media stream is variable bit rate, constant quality video, subject to the source shaping mechanisms described below. It should be noted that the current commercially available codecs produce video with considerably greater quality variation than would be expected with the scheme described below. In addition, we claim that users require a minimum quality from their video, and gain increased satisfaction from any increase in quality over the minimum quality threshold. This claim is supported by user studies in [23]. Since bandwidth is (to a first approximation) proportional to the quality of a variable bit rate video stream, we have a flow that has a minimum bandwidth requirement but can vary its bandwidth above that threshold to maximise quality for the user, within the bounds of fairness.

In the next section we discuss the models of congestion that we have adopted and discuss approaches to controlling them. In Section 5.3 we present the algorithms used for congestion and admission control algorithm, and present analysis that shows the limits of stability, and indicates some of the dynamic performance of the algorithms. In Section 5.4 we discuss the required instrumentation for the control, and the effects of non-linearity and inaccuracy. In Section 5.5, we present the results of simulations of the algorithm.

5.2 Source and Network Models

A connectionless datagram network can be viewed as a network of interlinked routers, where the performance of any route through the network is dominated by a single bottleneck link. In this chapter we will be focusing on how to control the queues at this bottleneck link. As described in Chapter 4, in the absence of any controls, the queue size is a non-stationary random process where the non-stationarity comes from three different sources:

non-stationary applications Sources change the statistics of their output data flow arbitrarily according to the characteristics of the application.

non-stationary sources Sources change their data flow according to congestion control and avoidance schemes.

source admission Sources enter and leave the network.

We attempt to design a system which will make the queue length at the bottleneck a stationary process around a known fixed point.

We assume that the flows generated by the sources are the output of a codec, where the codec can adjust the source coding it applies to the raw data to set the output to any desired average rate. In addition, it is assumed that the coding scheme used has at least two hierarchical levels, where the base level contains all necessary information to recreate the picture, and higher levels add information to improve picture quality. The host can thus adjust the source coding rate of the input flow in the various levels to trade-off bandwidth in the network against quality of the stream. The variance of the codec output stream is unknown. We also assume that we know the output rate of the codec corresponding to the minimum quality required by the user.

Since the sources generate at a given average rate, it then becomes natural to rate-control the data entering into the network. It is assumed that the codec and the network interface are coupled by some form of elastic buffer, where the occupation of the buffer is fed back to control the rate of the codec.

If the data was fed out at the average rate and no more, then the codec would be run as a constant bit rate video stream. This is not desirable, since this increases the delay from the video source to the destination and forces the codec to change its rate and thus the quality of the video stream more than is necessary. To alleviate this variability in quality, we allow the bit rate of the data stream to vary within defined limits.

A flow entering the network is typically constrained by a "leaky bucket" mechanism, where each packet that is transmitted requires a coin to be thrown into a metaphorical bucket. The bucket has hole in the bottom out of which coins drain at some specified rate. The user is allowed to transmit a packet as long as the bucket is not full. In this way, the user is allowed to burst a bucket load of packets into the network, but then must refrain from transmitting at anything higher than the coin drainage rate (the long term average data rate), or less if the user wants to burst more data into the network. The variance of the data flow is constrained in this way to a worst case of an impulse stream where the impulse is the size of the bucket, and the inter-impulse gap is the bucket size divided by the average drainage rate.

A single level of leaky bucket is likely to have a higher burstiness than is really required for large packets. To reduce the short term burstiness of the stream, a nozzle of a width greater than the average data rate may be placed on the output packet flow so that large bursts of packets are not fed directly into the network to overflow the switch buffers. This is implemented by defining an interval where one packet per interval would equal the required average rate, and then allowing a maximum number of packets to be sent per interval. In combination with a leaky bucket scheme, both the short and long term burstiness are controlled at a reasonable cost[2].

We have already discussed traffic shaping techniques, where we constrain the variance of the flows entering the network to prevent a single burst hitting the bottleneck and overflowing the queue. This operates at a granularity that is finer than one round trip time (rtt), where the queue can increase to the point where it overflows and drops packets and then recover before any of the sources can be notified to reduce their data rates.

The next level of control of the size of the queue operates in the time granularity where feedback of the queue sizes is possible, and the sources can adjust their data rates to prevent the queues growing too large. This is the time domain which is traditionally called congestion control and avoidance[3]. A fuller discussion can be found in Chapter 4. We describe the congestion control scheme as a second order multi-variable control scheme that attempts to set the bottleneck queue size to a reference size. Our scheme is based upon the estimation of the queue lengths at the bottleneck server, and can be viewed as an extension to the approach Keshav adopted in controlling the queue length

[2] It is possible to design the implementation of the protocol so that the scheme becomes self clocking by the arrival of incoming acknowledgment packets (subject to the constraints of "ack clumping" [119]), and a timer interrupt arrives rarely.

[3] Later in this chapter we will be deliberately creating a queue at the network, so we are not avoiding congestion, just controlling it.

in a fair queuing network [100] and [120]. Our approach extends the ad hoc attempts to govern the operating point in [121] and [43] by using a proportional control scheme, and is similar to that of Bolot and Shankar in [122], except that we consider normal FIFO queuing as opposed to fair queuing, and attempt to control the queue length explicitly.

The last level of queue control is at the level of actually admitting flows into the network. In resource reservation networks, this is done by the intelligence of the network based on the current state of the network and the parameters of the new flow. In our scheme we use the congestion control mechanism and the minimum rate threshold to fix the minimum bandwidth that is available in the network. If a new flow starts at its required minimum bandwidth, this can be viewed as a step input to the control system, which then adjusts the source rates to maintain the reference queue size. If there is sufficient excess bandwidth for the new source's minimum data rate, then the new flow is accepted. If there is not, then because the new flow's packets are tagged as being of lesser importance then older flows, the new flow's packets are preferentially dropped and the source discovers that it cannot obtain its required quality of service.

An important aspect of any congestion control scheme is the fairness exhibited in allocating the bandwidth of the network. In this chapter we will define fairness as saying that any source is allowed into the network as long as its minimum bandwidth threshold can be satisfied. The division of the bandwidth above the minimum will then depend on a number of factors, which we discuss below.

An aspect of rate controlled flows that has not received much attention as yet is the problem of correlated flows. Whilst the sum of the rates offered by all sources may be less than the bottleneck bandwidth, if all the sources release a packet burst into the network so that they arrive at the bottleneck at the same time, the bottleneck queue may yet overflow (or some of the sources will receive disproportionately high delays). To prevent this, we introduce random phase changes to the sources, where the inter-packet gap is a random variable drawn from a Normal distribution with mean of the desired gap, and a small variance. This also reduces the risk of the simulations below exhibiting the traffic phase effects described in [123].

5.3 The Control Model

5.3.1 Controlling the Queue Lengths

The aim of our control is to maintain the unfinished work in the system at a constant point (and thus also the queue size). This corresponds to attempting to fix the delays in a system at a set-point, and so tune the queuing delays to the desired levels, although they must be the same for all sources sharing the bottleneck.

We approximate the computer network as a first order linear system. The queuing network is modeled as a single server queue, with a service rate equal to the bottleneck switch in the network[4]. The input to the server is then the sum of the load offered by each of the individual flows. The number of flows is an unknown number, N.

We are making the following assumptions in this model, some of which are needed to make the math tractable - in practice, many of these assumptions can be violated:

1. We are assuming the system to be heavily loaded and can thus use the fluid flow approximation. This assumes that the unfinished work in the system from one interval to another is equal to the work that the system does subtracted from the unfinished work that is introduced. It also assumes that the system never empties of work, and that the work entering the system can be approximated by the mean arrival rate. Note that since we intentionally maintain queues at the bottleneck, so that the server never goes idle, this is a valid assumption.

2. We choose the sampling intervals to be the round trip time, and that the rates are per sampling interval.

3. The round trip time remains approximately constant i.e. that the propagation delay dominates the combined transmission delay. With reference to item 1, although we maintain persistent queues, the length of the queues is kept low so that this can be true.

4. We are disregarding any observation noise in the system[5].

[4]Note that we assume that the lowest bandwidth link will be the bottleneck, thus modeling the networks of yesteryear, where high speed LANs fed low speed WANs, rather than the fat pipe high speed WANs of the bright new tomorrow.

[5]If the eigenvalues of the system and the observation models are distinct, we can apply the Separation theorem and ignore this. In the observer model described later this is obviously true.

symbol	explanation	units
$u(k)$	unfinished work at bottleneck at time k	packets
$X_i(k)$	packets presented by source i from control interval $k-1$ to k	packets
μ	work done in interval $k-1$ to k (constant over time)	packets
$\omega(k)$	variation in unfinished work due to noise	packets
$rtt(k)$	round trip time at time k	seconds
$\lambda_i(k)$	Operating point: the approximate rate at which a source transmits	packets/second
α	weight in tracking the operating point	dimensionless
K_i	proportional constant used at source i	dimensionless
N	The unknown number of flows	dimensionless
u_T	Target unfinished work at bottleneck	packets

Table 5.1: Symbol Table

5. We are assuming constant packet size, and thus a constant round trip time if there are no queuing delays.

6. We can discover the service rate of the bottleneck, and the propagation delay of the return path to the source.

We shall discuss these assumptions and their validity in this and succeeding sections. The basic equation that we use in our analysis is

$$u(k+1) = u(k) + \sum_{i=1}^{N} X_i(k) - \mu + \omega(k) \tag{5.1}$$

where $u(k)$ is the unfinished work at the bottleneck measured in packets queued at time k, $X_i(k)$ is the work offered in the interval from $k-1$ to k by source i, μ is the work removed from the system in the interval (the bottleneck rate in packets per second divided by the sampling interval), and $\omega(k)$ is the effects of noise that we are attempting to counteract. This is the noise that arises from the variability of the sources and from the effects of cross-traffic before the bottleneck.

We observe the unfinished work in the system by observing the round trip times and subtracting the minimum round trip discovered to show the queuing delays in the bottleneck. However, the round trip times record the unfinished work that was present in the previous sampling interval, so

$$rtt(k) = u(k-1)/\mu + rtt_{MIN}$$

The control we implement is a second order control scheme, so that the $\sum_{i=1}^{N} X_i()$ converges to the bandwidth of the bottleneck. We use a coarse adjustment to the rate every control period, and track changes to the operating point using an exponential estimator of the actual rate transmitted. The sum over all the sources of their operating point variables in normal operation will approximately equal the bottleneck bandwidth. The operating point is here measured in packets/control interval.

$$\begin{aligned} X_i(k) &= \lambda_i(k-1)rtt(k) + K_i(u_T - u(k-1)) \\ \lambda_i(k) &= \lambda_i(k-1) + \alpha K_i(u_T - u(k-1))/rtt(k) \end{aligned} \tag{5.2}$$

We can use a multi-variable model to determine the dynamics of the system, describing it in a state space format as:

$$\boldsymbol{x'}(k+1) = \boldsymbol{A'x'}(k) + \boldsymbol{c'} \tag{5.3}$$

by selecting the following as our state vector:

$$\boldsymbol{x'}(k) = \begin{pmatrix} u(k) \\ u(k-1) \\ \lambda_1(k) \\ \vdots \\ \lambda_N(k) \end{pmatrix}$$

with the following as our matrix:

$$\boldsymbol{A'} = \begin{pmatrix} 1 & -\sum_{i=1}^{N} K_i & 1 & 1 & 1 & \cdots & 1 \\ 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & -\alpha K_1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -\alpha K_2 & 0 & 1 & 0 & \cdots & 0 \\ & & & \vdots & & & \\ 0 & -\alpha K_N & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

$$
\boldsymbol{c}' = \begin{pmatrix} \sum_{i=1}^{N} K_i u_T - \mu \\ 0 \\ \alpha K_1 u_T \\ \vdots \\ \alpha K_N u_T \end{pmatrix}
$$

$\boldsymbol{A}'$ degenerates into the following form, since the behaviour of the individual rates tracks the sum of the rates, where $K = \sum_{i=1}^{N} K_i$ and $\lambda(k) = \sum_{i=1}^{N} \lambda_i(k)$.

$$
\boldsymbol{x}(k) = \begin{pmatrix} u(k) \\ u(k-1) \\ \lambda(k) \end{pmatrix}
$$

$$
\boldsymbol{A} = \begin{pmatrix} 1 & -K & 1 \\ 1 & 0 & 0 \\ 0 & -\alpha K & 1 \end{pmatrix}
$$

$$
\boldsymbol{c} = \begin{pmatrix} K u_T - \mu \\ 0 \\ \alpha K u_T \end{pmatrix}
$$

If the K is fixed, then for the system to be stable around the fixed point $\boldsymbol{x}_{eq} = (u_T, u_T, \mu)^T$, the eigenvalues of the matrix $\boldsymbol{A}$ must have magnitude strictly less than one. Thus from algebra, we get

$$
\sum_{i=1}^{N} K_i < 1 \tag{5.4}
$$

and

$$
\frac{1 - \sqrt{1 - \sum_{i=1}^{N} K_i}}{\sum_{i=1}^{N} K_i} < 1 - \alpha \tag{5.5}
$$

However, we do not implement a strictly linear scheme for the reasons discussed below. Instead, we choose K to be fixed when $u(k-1) < u_T$, and let $K = \lambda(k)/\mu$ when $u(k-1) \geq u_T$. The stability of the total system can be analysed, by linearising the nonlinear part at the equilibrium point, to produce the following linear model:

$$
\boldsymbol{B} = \begin{pmatrix} 1 & -1 & 1 \\ 1 & 0 & 0 \\ 0 & -\alpha & 1 \end{pmatrix}
$$

The magnitude of the eigenvalues of $\boldsymbol{B}$ are not all strictly less than one, so the equilibrium point itself is unstable. However, if we consider small orbits around the equilibrium point, where the system oscillates above and below the equilibrium point using the various control laws, such that

$$x(k+2) = ABx(k)$$

the system will be asymptotically stable if the eigenvalues of the combined system AB are strictly less than one. The characteristic equation of the joint system is

$$\sum_{i=1}^{N} K_i - 2\alpha \sum_{i=1}^{N} K_i + \alpha \sum_{i=1}^{N} K_i - 2\alpha(1 + \sum_{i=1}^{N} K_i)\lambda + (1 - \sum_{i=1}^{N} K_i)\lambda^2 - \lambda^3 = 0 \tag{5.6}$$

The limits of α and $\sum_{i=1}^{N} K_i$ for this to be true are weaker than for the pure linear system, as can be seen in Figure 5.1. However, even in an unstable system, for "small" deviations above the equilibrium point the nonlinearity will ensure that the system goes into a stable orbit about the equilibrium point, rather than shooting off to infinity as an unstable linear system would. The dynamics of this are too complex to be easily captured by our model, although they have been verified in simple simulations. In addition the real world restrictions on the size of the unfinished work - $0 \leq u(k) \leq$ queue-size - ensures that the queue sizes remain bounded.

5.3.2 Proportional Control Constant

In the model above we showed the range that $\sum_{i=1}^{N} K_i$ must have for asymptotic stability. How can we set the distributed values so as to achieve this? To do this we observe that we have an estimate of the bottleneck bandwidth, and that if we have successfully entered the network that the sum of all the minimum bandwidth requirements of the sources sharing the bottleneck must be less than the bandwidth of the bottleneck. This allows us to set the proportional control constant according to the ratio of the minimum required bandwidth (λ_{MIN}) and the bottleneck rate when increasing the data-rate. To ensure that the sum is strictly in the region of stability (and to ensure that large oscillations do not occur), we multiply the estimated K_i by a target K_T selected from the region of guaranteed stability.

When we are decreasing the rate, we want to decrease the bandwidth required to the minimum as quickly as possible, both to ensure that new sources can enter the network, and that the delay is reduced as quickly as possible. For this reason we set K_i to be equal to the ratio of the current transmission rate (λ_i), and the bottleneck bandwidth. The sum over all the sources will be approximately one, and so we will probably over-compensate for any excess noise - but this is not a bad thing - it means we empty queues quicker, and reduce "rush hour" effects.

We now show (without very much mathematical rigour) that the use of a downwards K_i that is proportional to the bandwidth leads to a single stable equilibrium distribution of the bottleneck bandwidth over all the sources.

Assume that the system is currently at the equilibrium point defined above. Let there be a perturbation in u, of size δ. We shall now trace through the trajectory of u and a λ_i. Note that we can represent the value of the summed rates by a single rate. Assume at k = 0, that we are in equilibrium with $u(0) = u_T$ and a distribution of $\lambda_{i,eq}$ such that $\sum_{i=1}^{N} \lambda_{i,eq}(0) = \mu$.

k	$u(k)$	$u(k-1)$	λ_i
0	u_T	u_T	$\lambda_{i,eq}$
1	$u_T + \delta$	u_T	$\lambda_{i,eq}$
2	u_T	$u_T + \delta$	$\lambda_{i,eq}(1 - \frac{\alpha\delta}{\mu})$
3	$u_T - (1+\alpha)\delta + \frac{\alpha\delta^2}{\mu}$	u_T	$\lambda_{i,eq}(1 - (\frac{\alpha\delta}{\mu})^2)$
4	$u_T - (1 + \alpha - \frac{\alpha\delta}{\mu}(1-\alpha))\delta$	$u_T - (1+\alpha)\delta + \frac{\alpha\delta^2}{\mu}$	$\lambda_{i,eq}(1 - (\frac{\alpha\delta}{\mu})^2)$

At this time, the value of $u(k-1)$ will be less than u_T, since we can choose δ arbitrarily small. From this time the linear system kicks in, and we can analyse this best using the Z-transform. We can represent the process at the bottleneck using the following difference equation -

$$u(k+1) = 2u(k) - (1 + \sum_{i=1}^{N} K_i)u(k-1) + (1-\alpha)\sum_{i=1}^{N} K_i u(k-2) + \alpha \sum_{i=1}^{N} K_i u_T$$

where

$$u(0) = u_T - (1 + \alpha - \frac{\alpha\delta}{\mu}(1-\alpha))\delta$$

$$u(-1) = u_T - (1+\alpha)\delta + \frac{\alpha\delta^2}{\mu}$$

$$u(-2) = u_T$$

$$\lambda_i(0) = \lambda_{i,eq}(1 - (\frac{\alpha\delta}{\mu})^2)$$

Taking the Z-transform, we obtain:-

$$U(z) = \frac{-(u(0) + zu(-1) + z^2 u(-2))}{1 - 2z^{-1} + (1 + \sum_{i=1}^{N} K_i)z^{-2} + (1-\alpha)\sum_{i=1}^{N} K_i z^{-3}}$$

This has a solution of the form:

$$u(k) = A\gamma_1^k + B\gamma_2^k + C\gamma_3^k$$

where the γ_i are the roots of the characteristic equation in the denominator of the Z-transform, and the constants ABC represent the initial conditions. Since we have chosen the values of $\sum_{i=1}^{N} K_i$ and α so that the roots have a magnitude strictly less than one, the overshoot is guaranteed to be less than $A+B+C$. Substituting this in to the transformed equation for the time averaged operating point, we obtain:-

$$\Lambda(z) = \frac{\alpha K_i z^2 U(z)}{1-z} + \frac{\lambda(0)}{1-z}$$

By the final value theorem,

$$\lambda(\infty) = \lim_{z\to 1}(1-z)\Lambda(z) = \frac{\alpha K_i(u(0)+u(-1)+u(-2))}{2\sum_{i=1}^{N} K_i - \alpha\sum_{i=1}^{N} K_i} + \lambda(0) \tag{5.7}$$

From this it can be seen that the increase in operating point across all sources is a common constant multiplied by the individual proportional constant. Similarly, the decrease is a common constant multiplied by the previous bandwidths. If a source has less than its proportionate share, then a perturbation to the equilibrium position will force a decrease proportional to its current share, and it will then accrue new bandwidth according to its proportional constant, leading to an increased share. If a source has a larger share, then a perturbation will lead to a source getting a larger decrease, whilst only increasing according to its proportional constant. This demonstrates when the system is stochastic, that of the solutions to the equations, the system will tend to a unique distribution of the bottleneck bandwidth[6].

Further, the use of this scheme in the stochastic system described means that the effects of upstream or downstream cross traffic can be alleviated. This is described further below when we describe the simulations[7].

5.3.3 Admission Control

We model our video stream as consisting of two different levels of coding. The base level consists of the necessary information to give the minimum level of quality required by

[6] if anyone can think of a decent Lyapunov function that produces manageable results, I'd be very glad to hear of it.

[7] since we can't find an adequate mathematical model.

the user. This is expected to have a roughly constant time average bandwidth requirement. The second level consists of coded video that enhances the base quality level, and whilst not actually necessary for the user to perform their "task", it does improve their satisfaction.

Once a stream of video has discovered that there is sufficient bandwidth to transmit, then it transmits the two levels of video quality at different level of "importance", using the Type of Service flags discussed previously, the minimum base level being classified as more "important". This is used by the switches to decide which packets to trash when the switch buffers overflow because of increasing queue sizes. This method of error control for video has been discussed before in [54].

When a stream first starts transmitting, it transmits only the base level of the coded video, and transmits this at the lower level of "importance" for a given period. It is also transmitted with the "low delay" option of the ToS field set. We can thus model the start of the new flow as a step reduction in bandwidth of the bottleneck bandwidth, which is detected by the sources as increasing queue sizes. The sources will then react using the congestion control scheme described above, reducing their level of usage of the bandwidth at the bottleneck. If there is sufficient excess bandwidth at the network to allow the new flow in then no packets will be lost by the new flow. If there is insufficient bandwidth then the new flow will lose packets and should desist from transmitting. Setting the "low delay" option increases the perceived unfinished work that the sources see, as in the Tri-S scheme of Wang [124], and makes the congestion control algorithm operate faster, as well as allowing the new source to get a good estimate of the minimum round trip time. The period for which the packets should be sent at "low importance" is calculated as follows:

$$\text{start up time} = \frac{2\lambda_{MIN} rtt(k)}{\lambda_{bottleneck}}$$

This follows from assuming that the buffering at a switch will be of the order of the bandwidth delay product for that link, and thus transmitting for this period of time will ensure that new source injects sufficient packets into the network so that the bottleneck queue is guaranteed to overflow if there is insufficient bandwidth for the new flow[8].

Note that since the base level of established streams are transmitted at "high importance", we can guarantee the minimum level of video to the users. Since the flows

[8] This calculation of the admission period is conservative, and consistent with data buffering requirements. Depending on the rtts of the sources sharing the bottleneck, this period could be reduced

are stochastic in nature, it is possible that the entrance of a new flow might coincide with a quiet period for the current flows, and thus erroneously enter the network causing the sum of the minimum bandwidths to be greater than the bottleneck rate. This can be prevented if we can discriminate between flows either by colouring according to their claim on the resources, or by timestamping packets with the start time of the connection, so that if a choice of dropping packets of equal importance is to be made, then packets from the newest connection could be made preferentially, in combination with Random Early Drop Gateways [115].

An alternative approach to admission control could use the minimum required bandwidth as the single parameter in a simple resource reservation scheme, in which each switch kept a sum of the minimum bandwidths of the flows using it, and compared new requests against this and the known bandwidth of the switch. The congestion control scheme described above would then share out any remaining bandwidth.

5.4 Implementation

5.4.1 Instrumentation

The sources require three measurements from the network:

propagation delay This is obtained by periodically sending packets with a low delay option set[9]. These packets avoid queuing delays and the minimum rtt is obtained as the minimum rtt seen. This value should gradually be aged out.

bottleneck rate Assuming that the bottleneck is the server with the lowest rate, this can be obtained from sending out two back-to-back packets as described in [100]. The rate is given by dividing the packet size by the difference in round trip times of the two packets. Some proposed routing schemes allow for line bandwidth and delay discovery by "snmping"[10] to each router on a path and requesting the interface speed. If this is available, this may provide a more reliable scheme for bottleneck discovery.

[9] This could also be measured by analysing the differences in delay suffered by packets of different size, although this requires considerable statistical analysis.

[10] by using the Simple Network Management Protocol or some other out of band signalling technique.

unfinished work Given the minimum round trip time calculated above, the unfinished work is the difference between an estimate of the round trip time and the minimum round trip time. What estimate of the round trip time to use is discussed below. We assume that every packet is acknowledged so that we get a rtt estimate per packet sent.

The round trip time is assumed to come from the following equation

$$rtt(k) = u(k-1)/\mu + rtt_{MIN} + \epsilon(k) \tag{5.8}$$

where $\epsilon(k)$ is observation noise. The distribution of the observation noise is asymmetric, since it comes from queuing delays from other nodes along the route. This means that a Kalman filter cannot be used to obtain a good estimate of the round trip time. Alternative techniques are based around robust filtering techniques such as median filtering, where the value is obtained by taking the median value over a fixed number of values in the series, or by taking an exponentially weighted moving average from the values. The simulations presented below use a simple average over the rtts recorded in the last control period.

The rtt measurements are robust, in that they are averaged over a number of measurements, and if they are inaccurate, then their effect will be cancelled quickly. However, the bottleneck bandwidth measurement is more fragile. If the measured value is too low from the gap between the back to back packets being lengthened somehow, then the source will attempt to set the queue at the bottleneck to a higher value than the other sources, and will underestimate the work at the queue. It can be seen that the effective value of K_i will thus be higher, which may push the system into oscillation, since the stability of the system is guaranteed by the downwards proportional constant proportional to the bandwidth. Conversely, if the gap between the packets was shortened, by the front packet being forced to wait in a queue an abnormally long time, then the work at the queue will be perceived as large and the source will obtain a reduced share of the bandwidth. Inaccuracies in the minimum round trip measurement have similar effects.

5.4.2 The Search for the Operating Point

To actually reach the operating point we examine the format of the unfinished work versus load graph in Figure 5.3. Because we have constrained input variances, the graph will have very low queue sizes when the server is being run well below the bottleneck data rate. Therefore, whenever the measured queue sizes are below some threshold, we increase the operating point by the amount that we increase the data rate to compensate for the errors. Because our measurements of the queue lengths lag by as much as two round trip times, we use the rates of two control periods behind to ensure that we do not set the operating point too optimistically and create a large overshoot on the queue. When the queue sizes pass the threshold, we alter the operating point as an exponentially moving average of the adjusted rates. In this way the operating point will track long term deviations of the system. Analysis and simulation shows that a value of the weighting of 1/8 is a reasonable value. The performance of the system is very similar to slow start, in that we increase the data rate linearly when the system is uncongested, adding a constant to the data rate each round trip time.

if $((u_T - u(k)) > \frac{4}{5}u_T)$

  $\lambda_i(k+1) = X_i(k-2)/rtt(k)$

else

  $\lambda_i(k+1) = \frac{7}{8}\lambda(k) + \frac{1}{8}X_i(k)/rtt(k)$

5.4.3 Practical Effects of Nonlinearities

If we examine the equation governing the behaviour of the queue lengths including the use of parameters in setting the new rates,

$$u(k+1) = u(k) + \frac{(rtt_{MIN} + u(k)) \sum_{i=1}^{N} K_i(u_T - u(k-1))}{(rtt_{MIN} + u(k-1))} + \omega(k)$$

we see that the unfinished works of the previous and current epochs are used as multipliers for the new work. If the transmission delay is a significant portion of the rtt, then the subsequent rtts cannot cancel and the system becomes non-linear. This will tend to increase the rate at which rtts change. We can bound overshoots due to this change in rtt by combining a window flow control scheme with the rate based congestion control scheme and setting the window to constrain the packets in the system to $\lceil rtt(k)X(k)\rceil$, or by using the target work plus propagation delay as a minimum time period over which to introduce the new work.

The use of different proportional constants ensures that the system remains bounded, if not strictly stable, with a large degree of measurement inaccuracy in the system. If the measurements of the source err so as to make the linear system unstable, then the use of the downwards proportional controller will ensure that the system develops a limit cycle, rather than unbounded growth. Whilst analytically very difficult to prove, it can be seen in simulation of the equations.

The final source of nonlinearity is that the intervals over which the sources take action vary, and do not have a constant relationship. It is felt that this can be swept into the noise term in the analysis, and merely increase the gain margin that must be achieved.

5.4.4 Extending the approach to small groups

In Chapter 4 we pointed out that extending congestion control mechanisms to large groups of receivers is a difficult problem. The mechanism described can be used for small groups since a new state machine can be instantiated for each receiver, and the source can send at the minimum bandwidth from the receivers' state machines. However, it cannot scale to large groups for two reasons:-

- Since a new state machine is required for every receiver, the processing load on the sender increases at least linearly with the number of receivers.
- Each receiver must send regular acknowledgments back to the sender to allow the sender to estimate the rtt and the bottleneck bandwidth. Whilst each individual acknowledgment stream is a small fraction of the sender stream, the sum from all the receivers would soon exceed the transmitted information signal, and would itself be a cause of congestion.

For larger groups, the techniques used to return the congestion signal to the sender must scale with the number of receivers. We describe one such mechanism in Chapter 6.

5.5 Simulation

The simulation experiments reported are based around the topology shown in Figure 5.4. There is a single bottleneck link at which queues form. The effect of a bottleneck forming at a link which is not the lowest bandwidth link is discussed at the end of this section. The other parameters used in the simulations are shown in Table 5.2. The sources of data used in the simulations generate packets of a fixed size, with a gap between packets drawn from a normal distribution with a mean determined by the algorithm and a coefficient

Target K (K_T)	0.75
Target packets	5
in bottleneck (u_T)	
Minimum rate	5 packets/second
Packet size	512 bytes
bottleneck rate	62 packets/second
Minimum rtt	40.21ms
No bit errors	

Table 5.2: Parameters used in all simulations

of variation of 0.1. Since a real traffic source would be decoupled from the network interface via a queue of sorts anyway, a normal traffic source suffices to demonstrate that the control algorithms work.

In each of the simulations we focus upon the dynamics of the rate changes in the hosts and the queue lengths in the bottleneck switch for the outgoing link.

Scenario 1: Three Sources In this scenario, shown in Figures 5.5 and 5.6, all three sources have the same minimum bandwidth requirement. All sources start at the same time and consume the bottleneck bandwidth within 4 seconds. The peak queue length is 11 packets. As can be seen, the initial conditions for each of the sources vary, and so the path to the equilibrium points are not equal. In addition, since we have noise components, the equilibrium is constantly being disturbed, and the result is the observed stochastic process around the equilibrium points.

Examining the round trip times, we can see that the initial delay grows high, and then settles down to within the predicted limits. The rtts experienced by each of the sources can be seen to be very similar.

Scenario 2: 3 sources with unequal bandwidth requirements In this simulation we show that the base rates tend to an equilibrium point. Source 1 has a minimum bandwidth of 15 packets/second, whilst the other two sources have minimum bandwidths of 5 packets/second. As the analysis above predicts, the equilibrium distribution is approximately in proportion to the relative values of the minimum bandwidths, as can be seen from the graph in Figure 5.7 (3:1:1).

Scenario 3: Effect of varying rtts In this simulation we exhibit the independence of the equilibrium points from the rtt, and show that the base rates will tend towards the equilibrium distribution. Source 2 has a minimum rtt of 89.6 ms whilst the other two sources have minimum rtts of 40.21 ms. It can be seen in Figure 5.8 that although because of phase effects one of the sources gains an initial advantage over the other sources, the distribution of the bandwidth does eventually become fair. Simulations not shown here over a wide range of rtt differences corroborate this.

Scenario 4: Effects of cross-traffic Here we introduce a fourth source that regards a switch on the path for source 1 as a bottleneck, as shown in Figure 5.12. As can be seen from the graph in Figure 5.9, the sources once again appear to discover an equilibrium. Note that the log scale tends to exaggerate the noise in source 1.

That the sources tend to an equilibrium point where the source suffering from cross-traffic is not at its minimum bandwidth, is a surprising result. If we model the cross-traffic as a constant reduction in the target bandwidth in a deterministic mathematical model, the suffering source will tend eventually to its minimum bandwidth allocation. However, if we introduce noise into the system, then mathematical modeling shows us the reason for this phenomenon.

It can be shown that when the queue length is above the target length, then since the control constant is proportional to the current rate of the source, sources will reduce their rate so that the queue length drops below the target point in a fixed period independent of the number of sources, and that the amount their rates are reduced is proportional to their previous rate. However, the rise time to the target queue length is dependent on the amount below the target queue length and on the fixed constants proportional to the minimum bandwidths. Thus when we view the queue length as a random process, the time average of the process will drop below the target length as a source gains a disproportionate share of the bandwidth. This will allow the source suffering from cross-traffic to compensate for the effects of the cross-traffic, and so get a share of the bottleneck above its minimum rate.

Scenario 5: Dynamics of start up In this simulation we focus upon the dynamics of the start up, shown in Figures 5.10 and 5.11. In this scenario, two sources share the bottleneck until a third source starts, whose minimum bandwidth requirement is

15 packets/s. The start of transmission of the delayed source can be regarded as a step function reduction in the bandwidth of the bottleneck from the viewpoint of the other sources, since the initial packets are transmitted with the low delay indication set. After two intervals, these are no longer transmitted, and the delayed source will see the real work at the bottleneck. The queues at the bottleneck climb to 16 packets, before returning back down. The rates of the three sources then move gradually to their equilibrium points after this.

The round trip times seen by the sources, apart from the initial low estimates that the delayed source receives, show that the sources all perceive approximately equal rtts. Note that this is not necessarily the case for every instance, since each realisation of the processes may have different initial phase relationships.

Scenario 6: Displaced Bottleneck In this scenario, we use the network shown in Figure 5.13. Here, the bottleneck in the network is not the link with the lowest rate. The measurement techniques we are using assume that the bottleneck link is the lowest bandwidth link, and so our control is out by the ratio of the bottleneck link to the lowest bandwidth link. This means that the queue sizes are increase proportionately, and the source with the lowest bandwidth link in the path dominates the behaviour of the queue as shown in Figure 5.14.

Whilst this behaviour is not ideal, the use of the minimum bandwidth entry requirement prevents starvation of any source, and whilst the queues in terms of packets may rise, the total unfinished work in the system is still bounded by the work that would be contained in a target queue at the lowest bandwidth link, since we prevent any source from transmitting faster than its estimated bottleneck size. This behaviour qualifies as "fair" under our previous definition. Note that this behaviour goes away if sources are isolated from each other, using a queuing discipline such as "Fair Queuing", in which case our scheme degenerates into a variation of Keshav's control mechanism [120].

Scenario 7: Poisson Data Traffic In this scenario, we show how the congestion control scheme interacts with data traffic, represented here by a Poisson source (without traffic shaping) shown in Figure 5.15. The Poisson source takes the same route as source three. Whilst the Poisson traffic increases the variance of the queue length process, it can be seen that the queue process remains stable and has a known variance.

5.5.1 SPSS Modeling of the round trip times

We took a sample series of the measured round trip times from scenario 1, with an increased coefficient of variation (0.5) in the source and used the SPSS-X statistical analysis package to derive a model for the data. This is directly related to the queue size, and has the advantage that it is calculated at the sampling instance of the queue size. However, it is a lagging indicator, since it is constructed from measurements in the sampling interval $k-2$ to $k-1$. It has so far proved difficult to capture the exact measurements of the queue size.

The autocorrelation and partial autocorrelation plots are shown in Figures 5.16 and 5.17. The lack of any significant partial autocorrelations after lag 3 suggests that the series would be best modeled by a third order autoregressive process, such as:

$$\begin{aligned} x(k+1) &= CONSTANT + \\ & AR1\, x(k) + AR2\, x(k-1) + \\ & AR3\, x(k-2) + \omega(k) \end{aligned}$$

as predicted by the multivariate model of the sources. The constants $AR1$,$AR2$ and $AR3$ are the empirically measured values from the modeling process, whilst $x(k)$ is the rtt time series. The results of the modeling are shown in Table 5.3.

model	parameter	B	SEB	T-RATIO
	AR1	1.11356	.01168	95.355612
ar(3)	AR2	-.17879	.01748	-10.228626
	AR3	-.16073	.01168	-13.761878
	CONSTANT	114828.02490	1212.95357	94.668113

Table 5.3: Parameters from modeling series as an ar(3) process

Surprisingly, the $AR1$ constant is shown to be larger than one. This is due to the non-linearity of the round trip times and so the sampling period. Since when the queue length is increasing in size, the sampling period gets longer, the queues will be larger than expected. This will be reflected in increased values for the $AR1$ constant and increased instability. However, this is counteracted by the increased damping that results from using the current rate as the basis for the downward proportional constant, making that larger than would be the case for just the minimum bandwidth basis.

The residual sample autocorrelations and partial autocorrelations are plotted in Figures 5.18 to 5.19 for the third order model. From inspection, it is gratifying to discover that the third order model fits the data well.

5.6 Conclusions

In this chapter, we have described a new congestion control scheme for continuous sources using hierarchical encodings. We have shown that it leads to a stable stochastic process for the queue length at the bottleneck, leading to a low jitter arrival pattern at the destinations, and have demonstrated in simulation that the scheme will tend to an equilibrium distribution of bandwidths to all the sources sharing the bottleneck. We have also shown that the stability of the system is independent of the number of sources in the system (assuming that the entrance requirements are satisfied) and independent of the path lengths of the sources sharing the bottleneck.

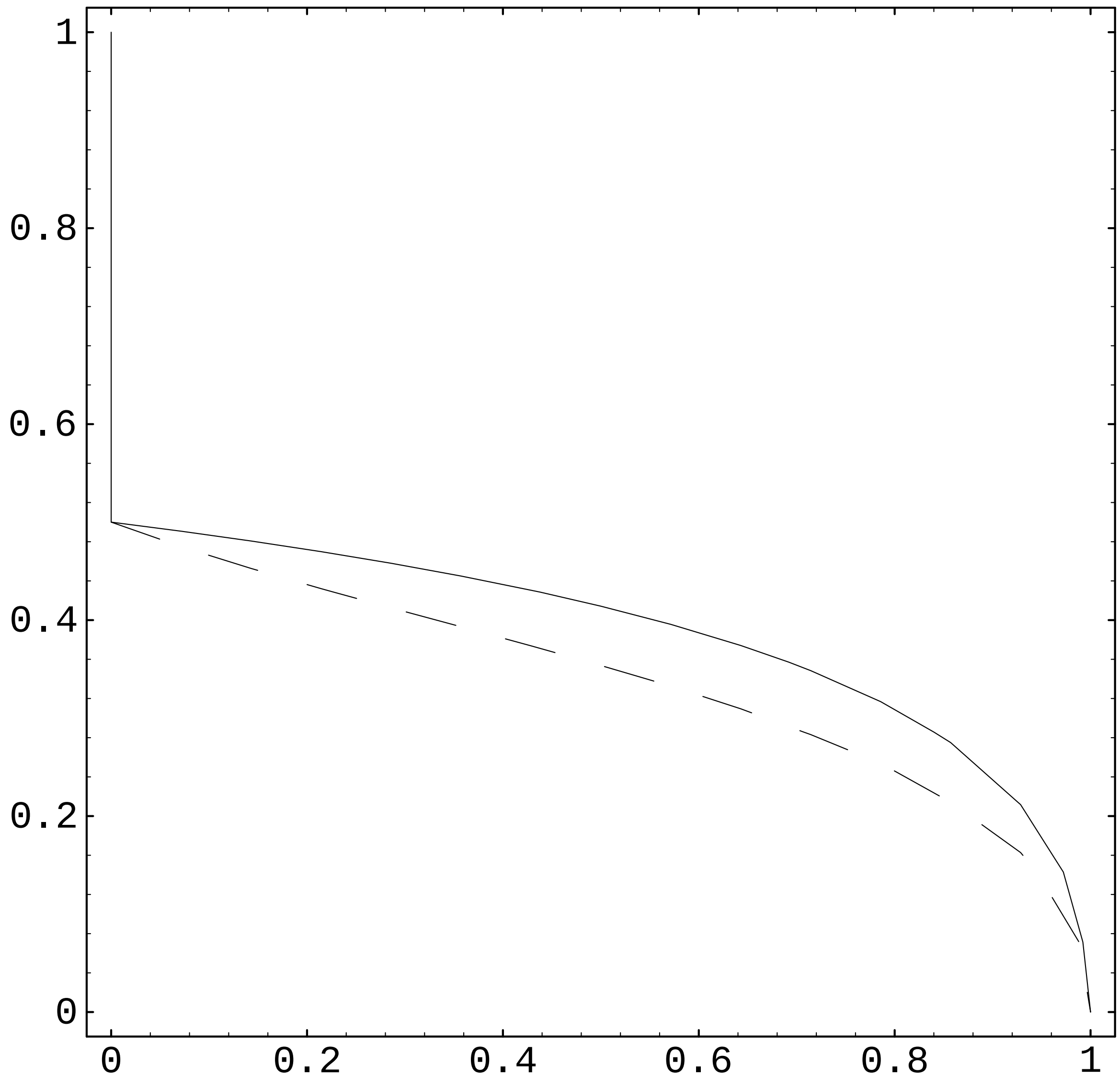


Figure 5.1: Plot of K on the x axis and α on the y axis for stability - the area beneath the solid line is for the linear system, whilst the dashed line circumscribes the nonlinear asymptotically stable area.

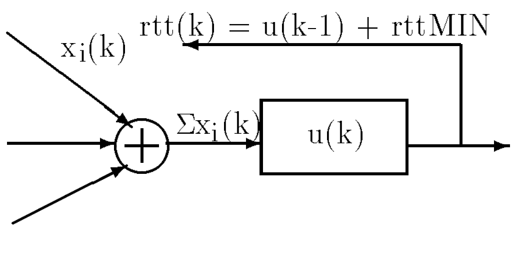


Figure 5.2: Reference model

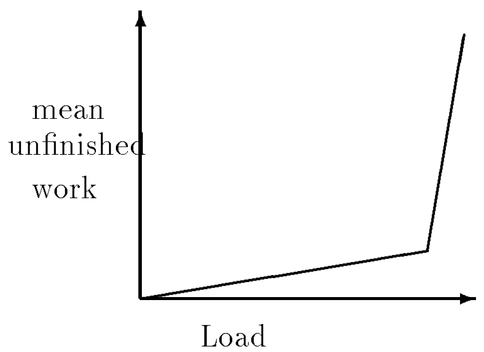


Figure 5.3: Reference delay graph

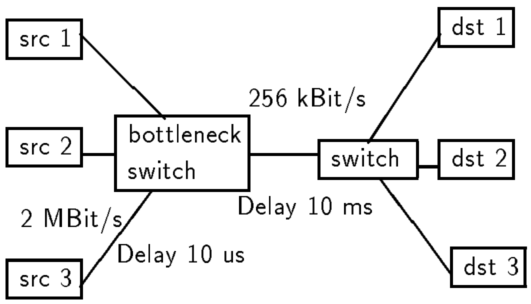


Figure 5.4: Basic scenario

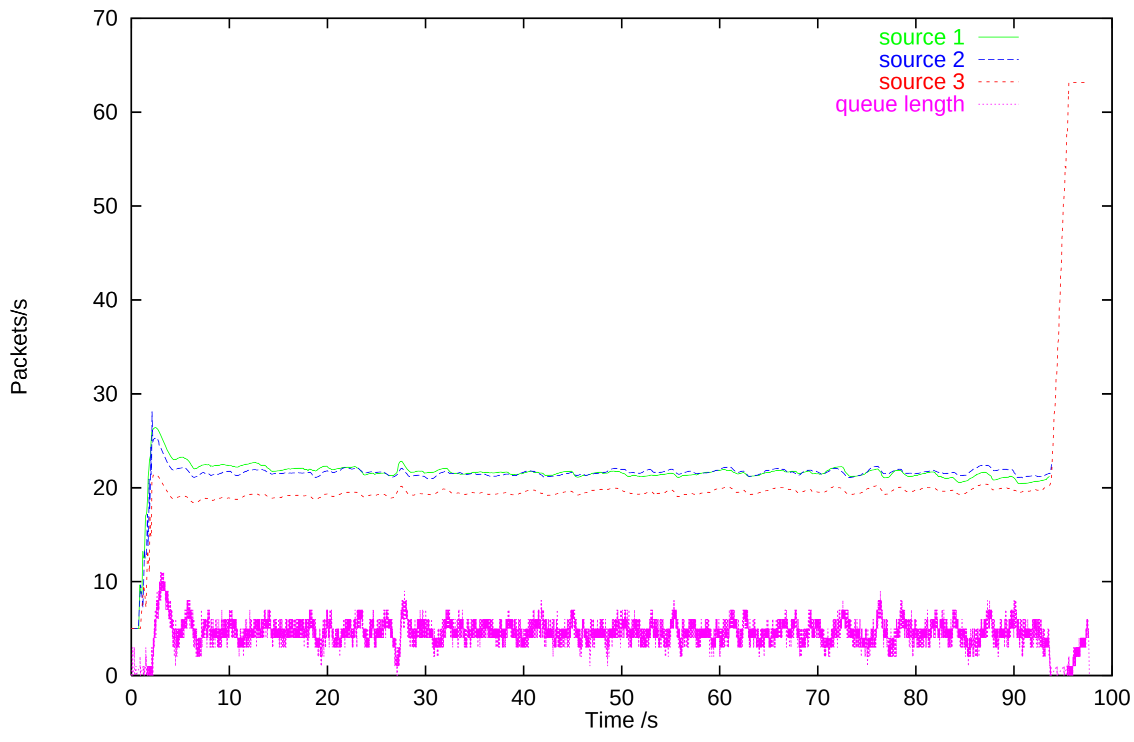


Figure 5.5: Scenario 1: 3 Sources with equal minimum bandwidth requirements - plot of base rates and queue size

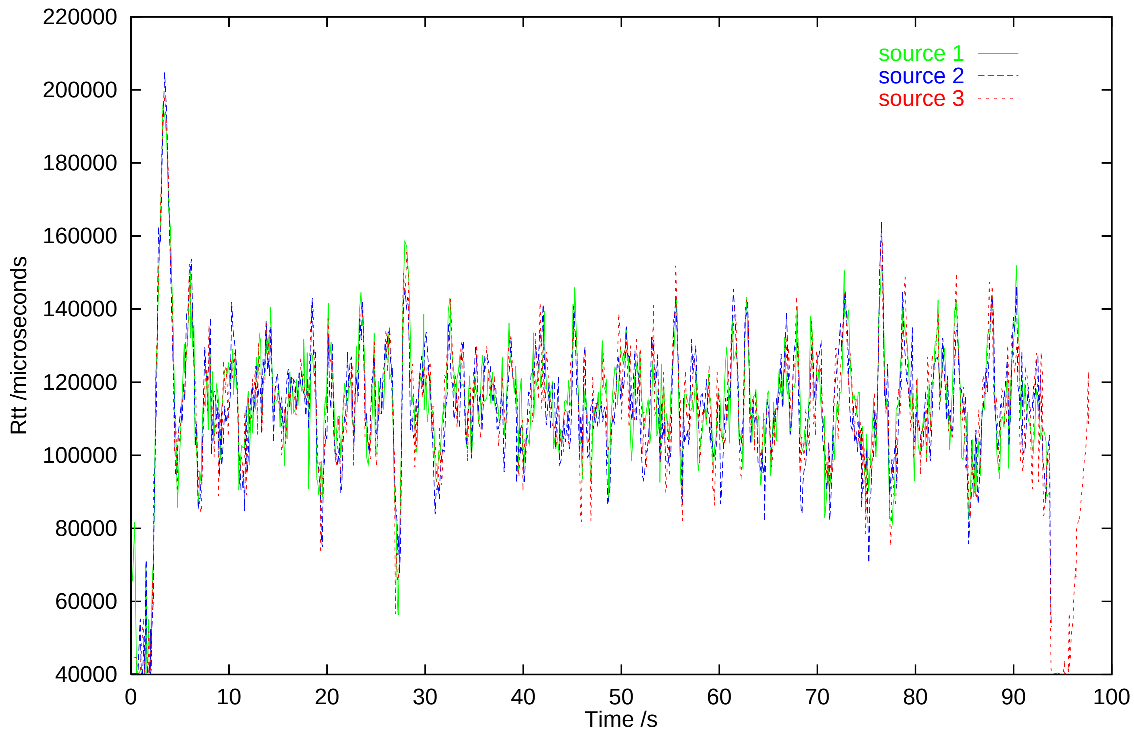


Figure 5.6: Scenario 1: 3 Sources with equal minimum bandwidth requirements - plot of measured round trip times

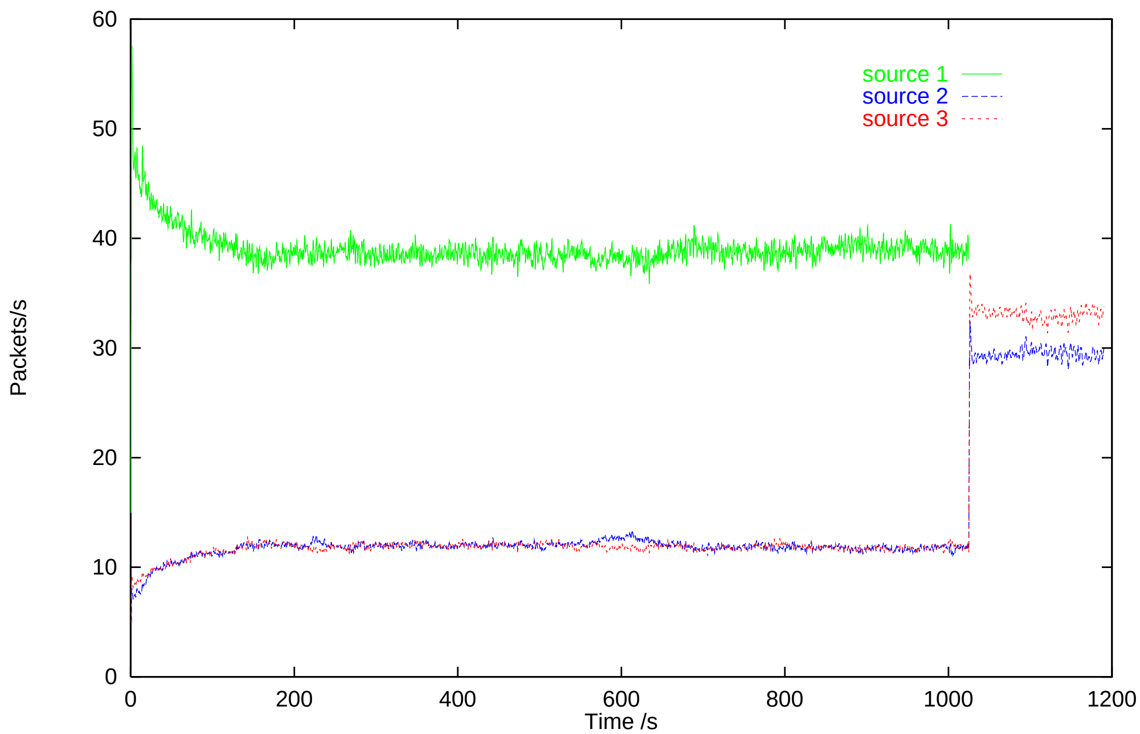


Figure 5.7: Scenario 2: 2 Sources with equal minimum bandwidth requirements and one with increased bandwidth requirements - plot of base rates

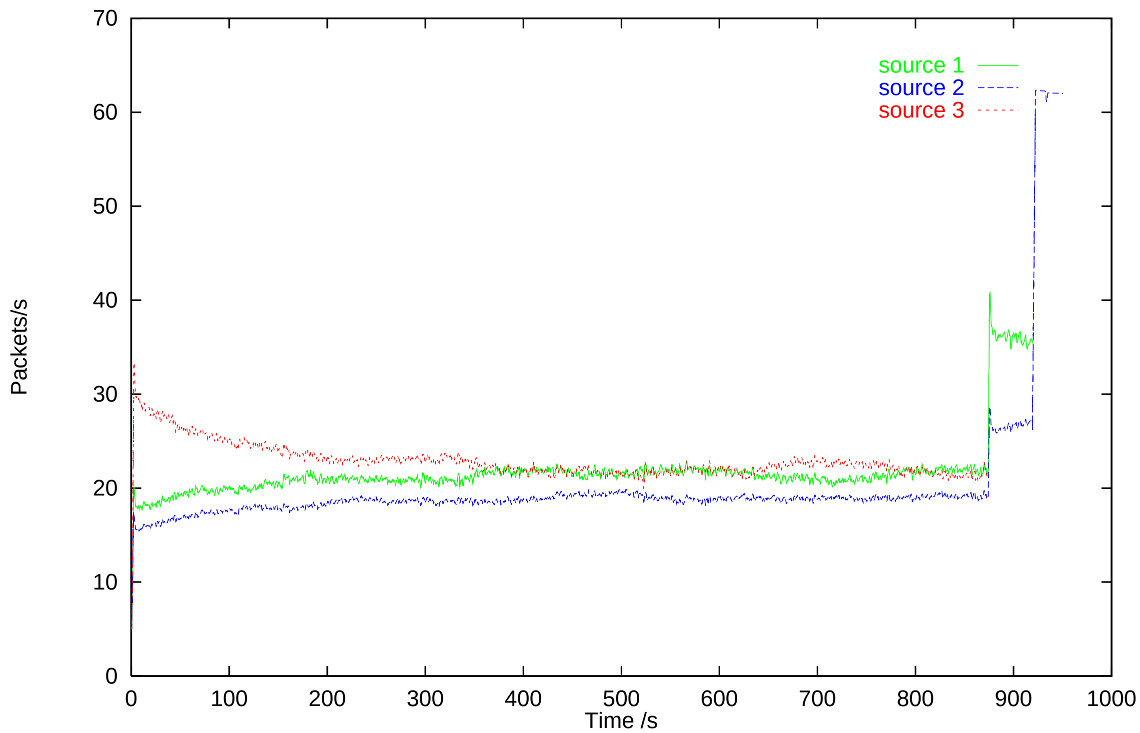


Figure 5.8: Scenario 3: 2 Sources with equal rtts and 1 source with a larger rtt - plot of base rates

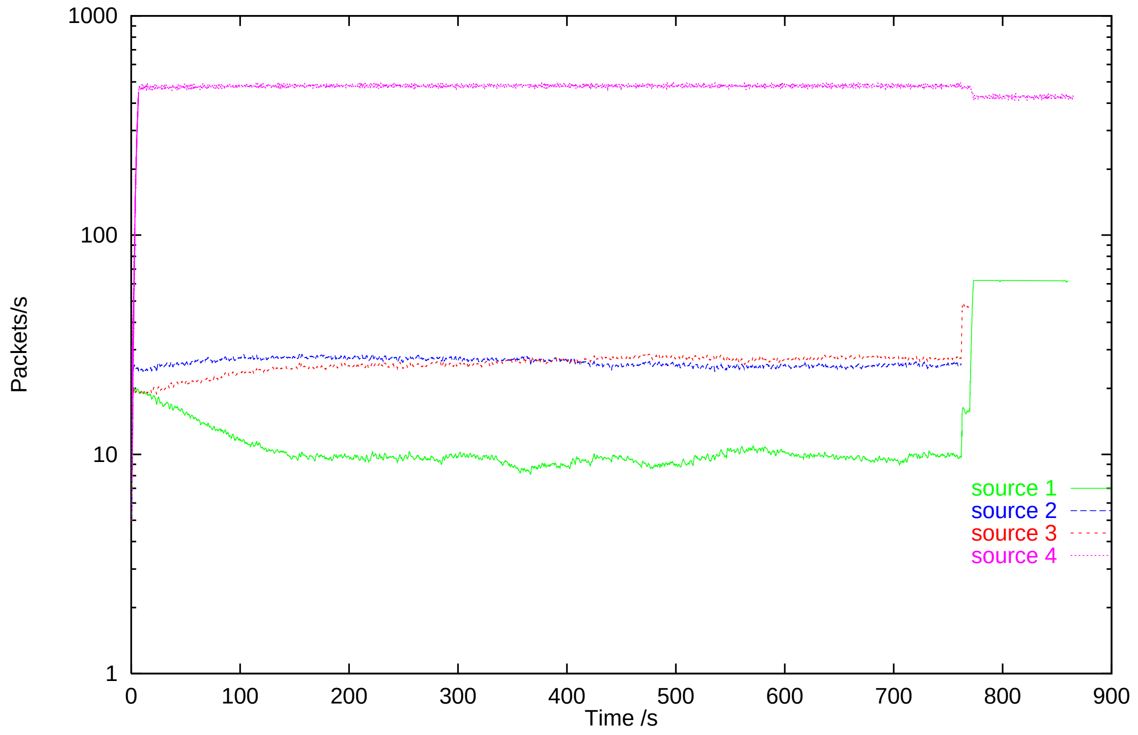


Figure 5.9: Scenario 4: 3 Sources with equal minimum bandwidth requirements but one suffering cross-talk from a fourth source - plot of base rates

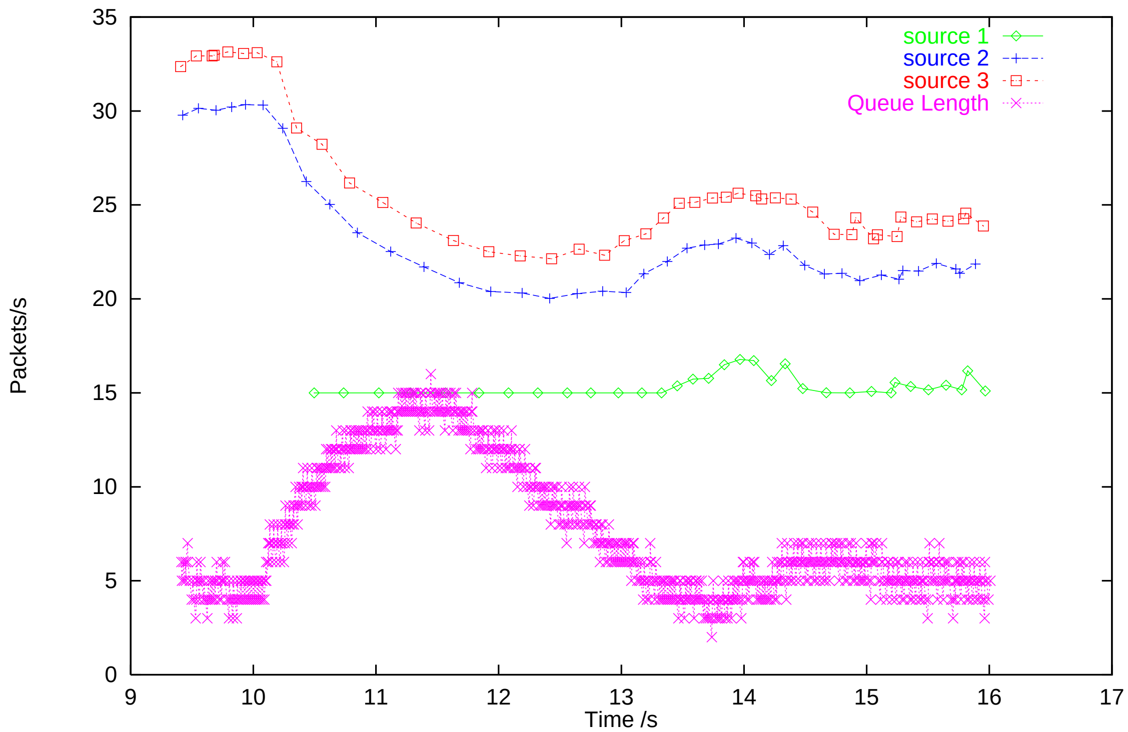


Figure 5.10: Scenario 5: Closeup of dynamics of 1 source starting when 2 sources already share the bottleneck - plot of queue length and packet rates

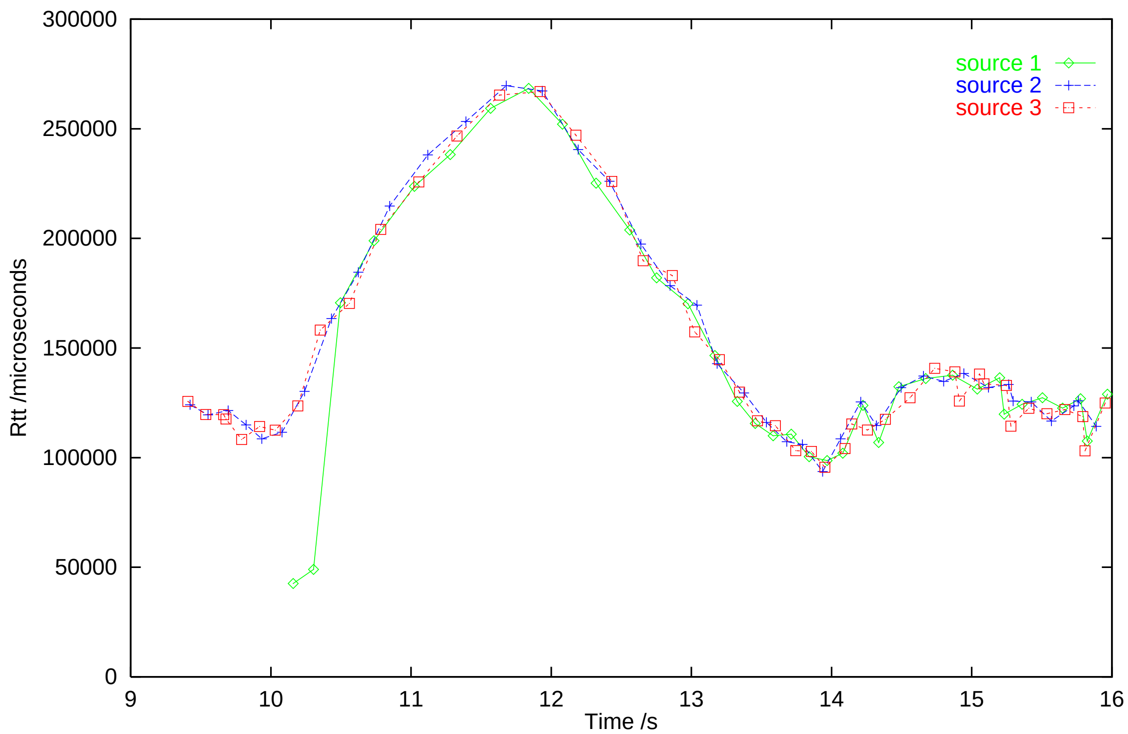


Figure 5.11: Scenario 5: Closeup of dynamics of 1 source starting when 2 sources already share the bottleneck - plot of rtts

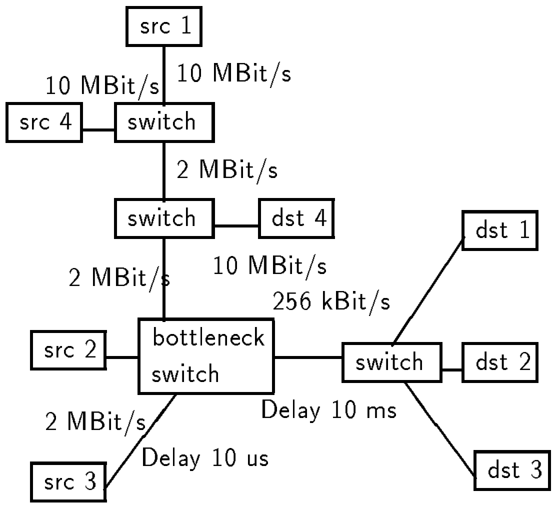


Figure 5.12: Cross Traffic scenario

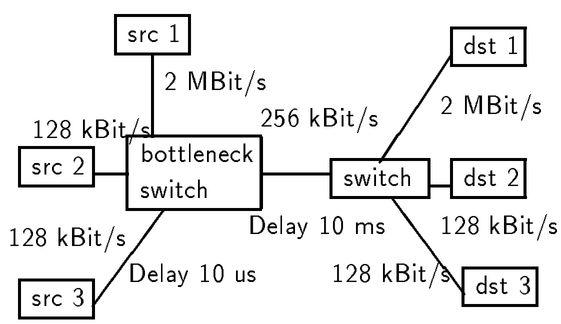


Figure 5.13: Scenario for displaced bottleneck

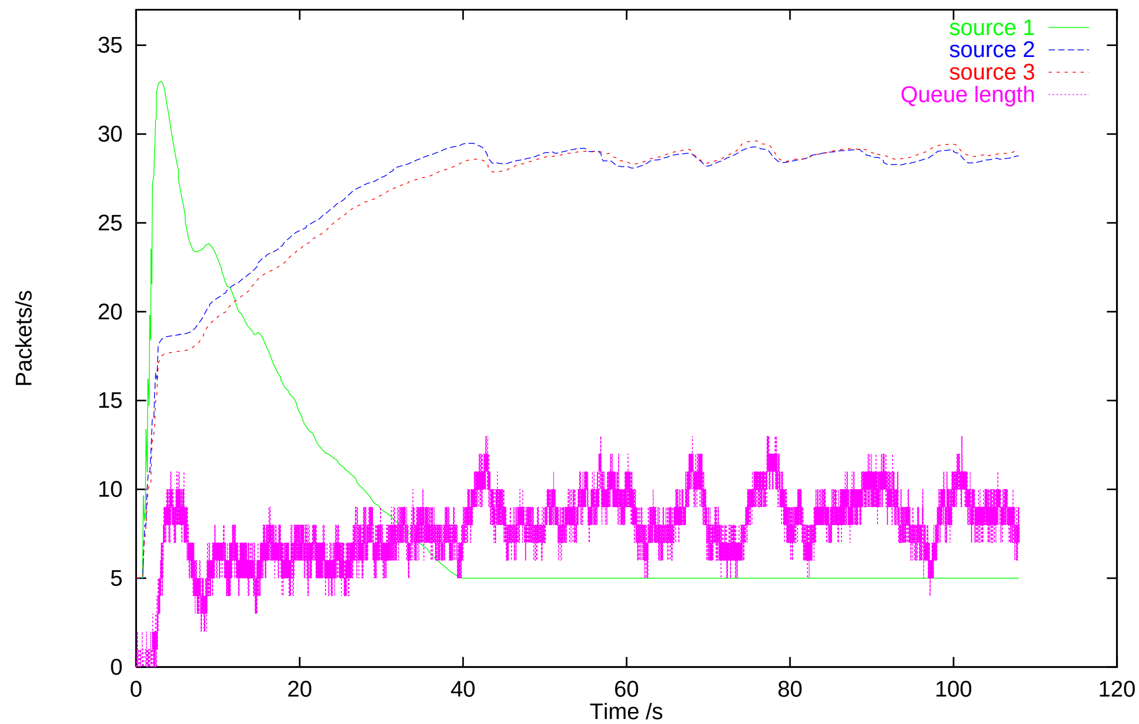


Figure 5.14: Scenario 6: Displaced bottleneck plots of base rate and queue length

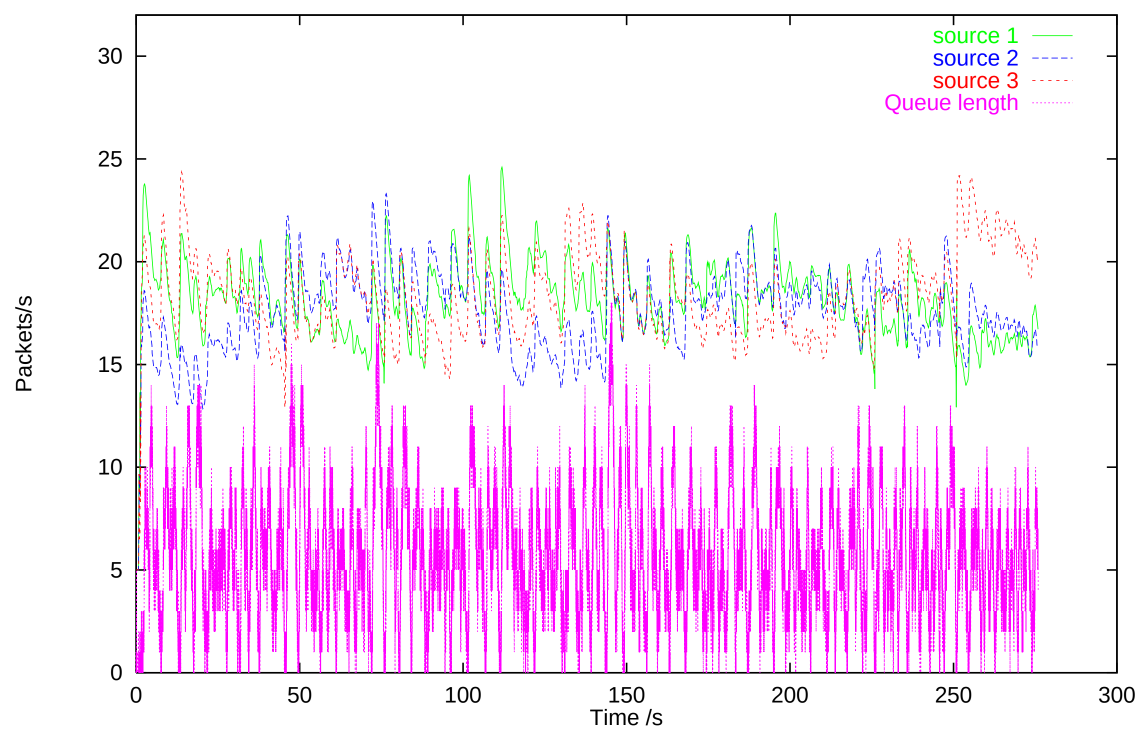


Figure 5.15: Scenario 7: Plots of base rate and queue length when sharing with Poisson traffic

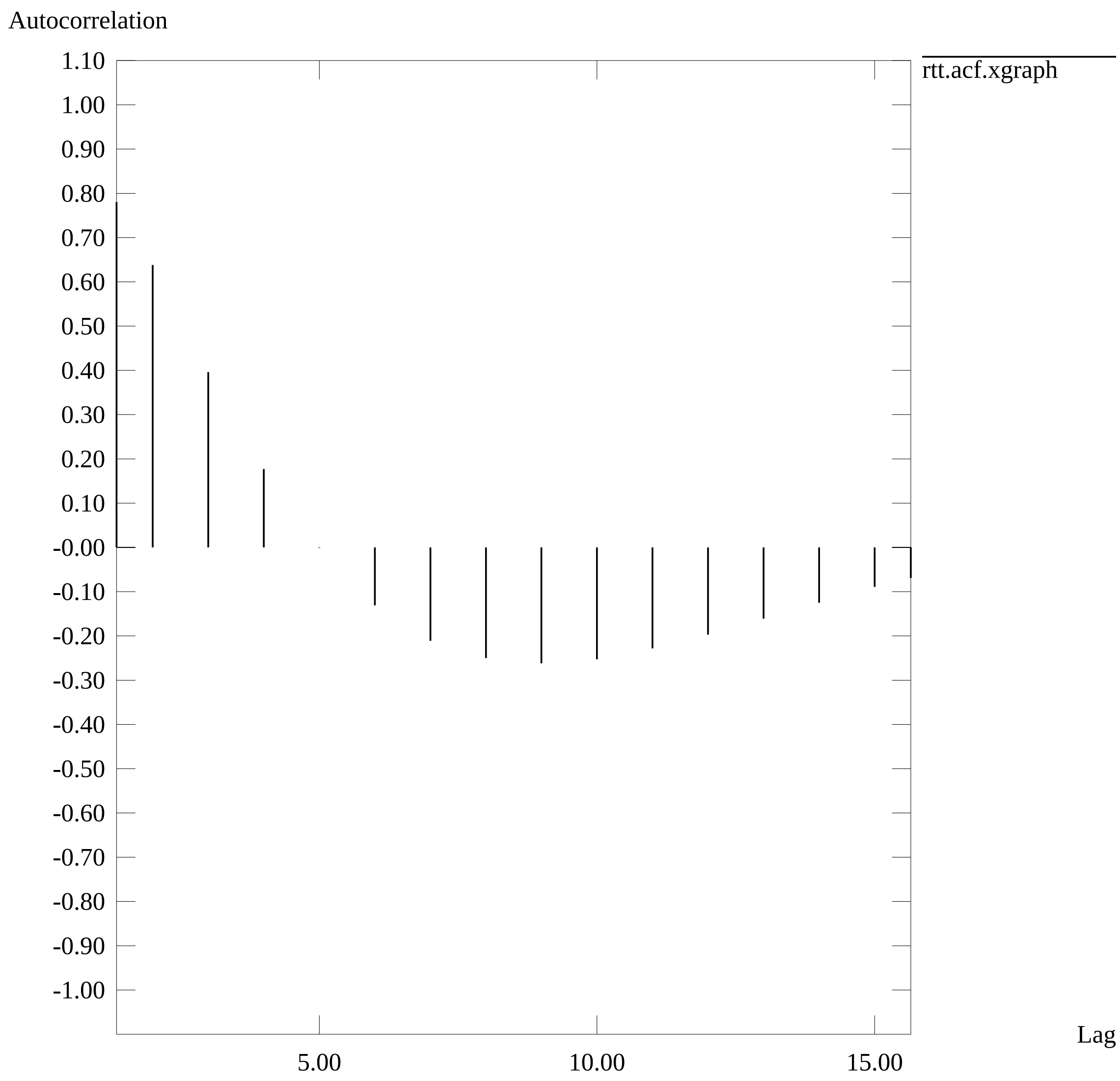


Figure 5.16: Autocorrelation plot for the sample rtt process. Standard Error = 0.012

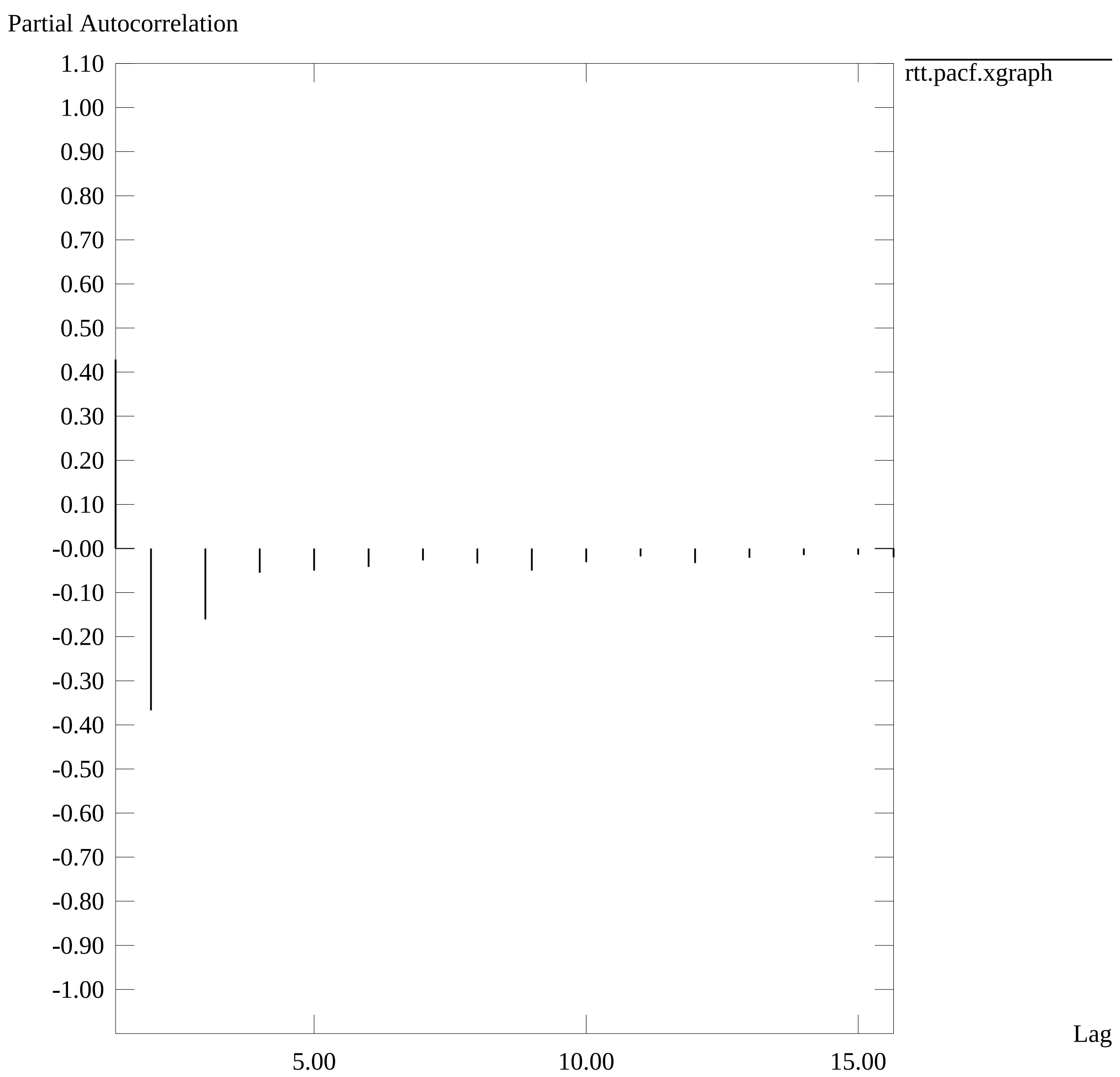


Figure 5.17: Partial Autocorrelation plot for the sample rtt process. Standard Error = 0.012

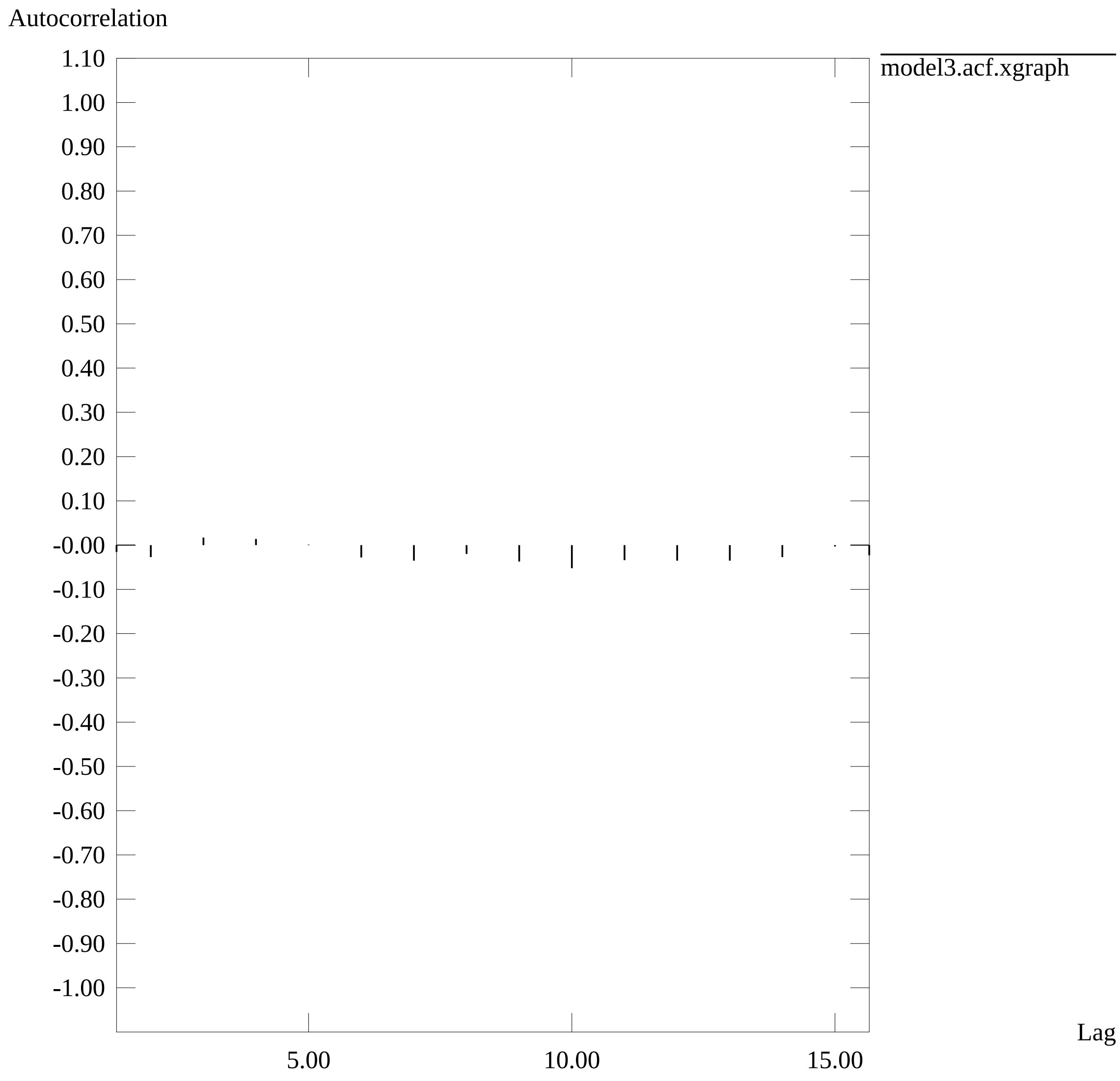


Figure 5.18: Autocorrelation plot for the residuals after modeling as arma(3,0,0). Standard Error = 0.012

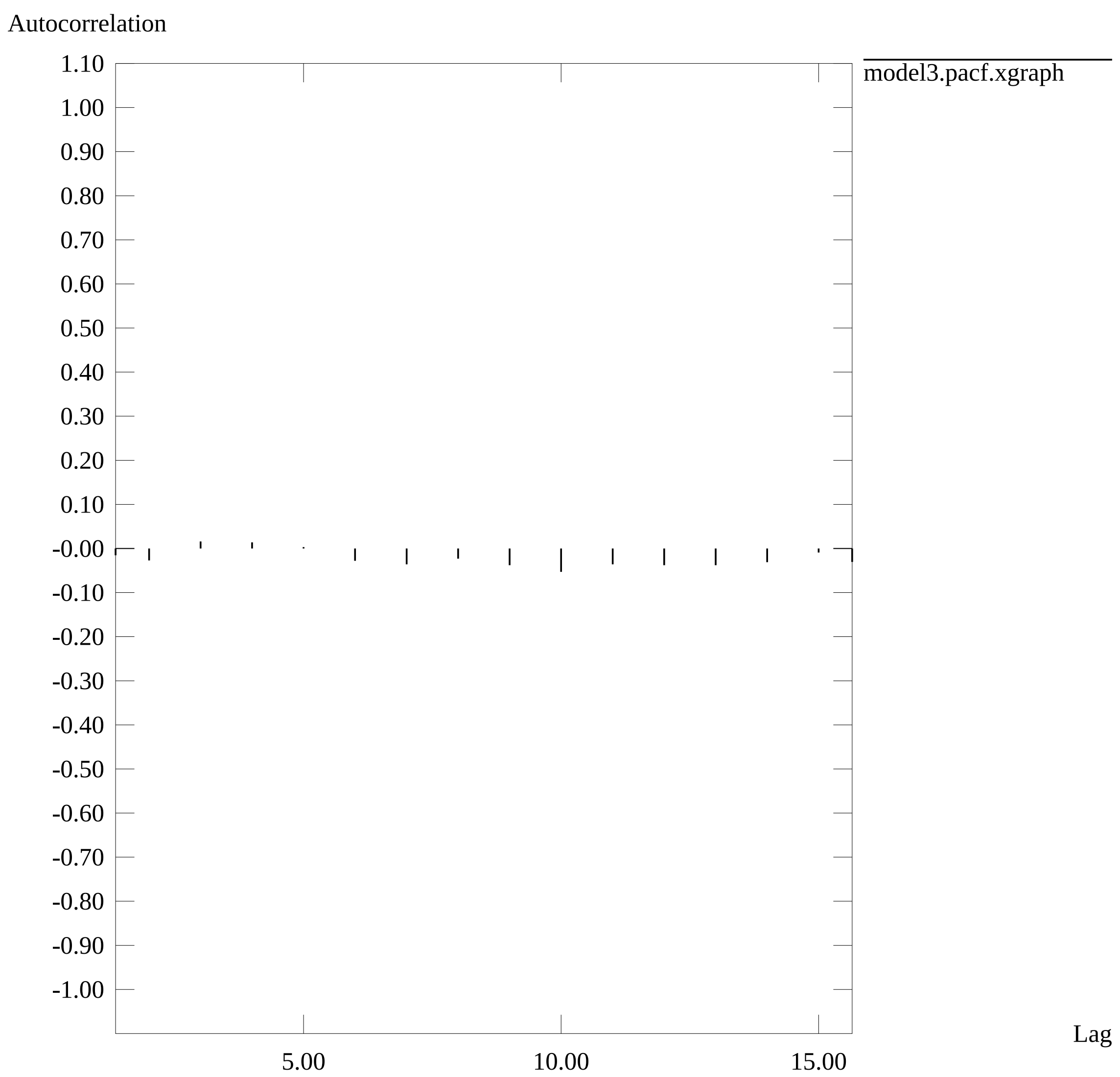


Figure 5.19: Partial Autocorrelation plot for the residuals after modeling as arma(3,0,0). Standard Error = 0.012

Chapter 6

Scalable Feedback Control for Multicast Video Distribution in the Internet

6.1 Introduction

In[1] Chapter 5, we described a unicast congestion control scheme. We described briefly how the scheme could be used for small groups. However, since each receiver must send acknowledgments back to the source, it cannot scale to large groups. In this chapter we describe a mechanism for end to end congestion control that scales to large groups of receivers.

Multicast packet transmission has now been available in the Internet for some time [127]. The examination of how it is being used brings one to the conclusion that multicast distribution is not being widely used for the traditional bulk transfer applications of computer communications, but rather for the distribution of so-called real-time traffic, i.e. packet video and voice, along with resource location querying. Multicast groups for voice and video distribution currently have up to hundreds of recipients attached to them (although this number is expected to increase, for such applications as TV distribution), for which the end consumers of the data are people.

Ergonomic studies and anecdotal evidence from the Internet demonstrate that people can use audio or video signals as long as the information content is above some minimum

[1] An early version of the work in this chapter appeared in [126].

level which depends on the task in hand (e.g. as shown in [23,128]). Thus, it is possible to transmit audio and video signals with lower bandwidth requirements at the expense of a slight degradation in user satisfaction, although user task performance will not be significantly degraded until the information content in the signal slips below the minimum level mentioned above. One approach to distributing real-time streams is then to adjust the bandwidth, or rate, of a source based on the prevailing conditions in the network. These conditions change with time because connections are set up and terminated, and because sources do not send at a constant rate. The rate adjustment mechanism must be of course informed of such changes. One way is for the source to receive feedback about the state of the network and to control the rate at which packets are sent into the network accordingly.

We propose to use this approach to control sources of real-time traffic. In this chapter, we consider specifically sources of video traffic, i.e. video coders. Through the use of the mechanisms described here, we can prevent congestion of the Internet. This is important because the increased computing power of workstations and the availability of audio and video applications such as VAT [70], NV [47], NEVOT [129] and IVS [130] has led to a huge increase in the real-time traffic in the Internet. IETF meetings [131], seminars (e.g. the MICE [128] and Xerox seminars), shuttle launches, etc., are now regularly audio- and video-cast. The uncontrolled transmission of audio and video streams would easily (it already has done so on a few occasions) swamp the resources of the Internet, cause congestion, and lead to unacceptable service for all users of the network.

We note that our approach does not require special support from the network such as resource allocation. This is in contrast to another approach to delivering real-time streams, which has focussed on changing the network architecture to meet the expected bandwidth and delay requirements of audio/video applications by introducing new admission control and switch scheduling mechanisms (e.g. [112] [132]). These mechanisms may eventually be implemented if the Internet moves to a resource reservation model, but they are not expected to be available in the very near future.

Our proposed feedback-based approach is an end to end control loop, using the taxonomy of Chapter 4. However, the use of wide area multicast for the delivery of the real time streams creates additional problems in getting feedback from the receivers. It is important to get timely notification of congestion, but if the congestion is close to the source then all receivers will detect the congestion and will send a notification to the source generating an implosion of messages at the source [133,134]. For this reason, a control mechanism such as that described in Chapter 5 is not possible. Instead, we require a mechanism for soliciting feedback information in a scalable way from the receivers.

Given this feedback information, it is important to relate the state to the entire group of receivers within the context of the application. If only a single receiver is suffering congestion on its last hop in the delivery tree, should the transmitter degrade the video for the entire group? Or should it request that the troubled receiver leave the video group, and rely on some other information stream? Thus mechanisms are needed to estimate the number of receivers suffering, and for the application to use this information in deciding how to adjust its output rate.

The rest of the chapter is organized as follows. In Section 6.2, we discuss issues related to congestion control for real-time traffic in a multicast environment.

In Section 6.3, we describe the scalable feedback mechanism. We demonstrate a prototype working in a real network in Section 6.3.4. In Section 6.4, we describe the video control mechanism used in IVS. In Section 6.5, we evaluate the performance of our feedback control mechanism and discuss its limitations. We delay the discussion of related work until Section 6.6 in order to build up sufficient context to compare our results to others published in the literature. Section 6.7 concludes the chapter.

6.2 Congestion control for real-time applications in a multicast environment

Feedback control mechanisms are used in the Internet to control the unicast distribution of non real-time traffic, specifically in TCP. There, the feedback information is packet losses detected by timeouts or multiple acknowledgments at the source, and the control scheme is Van Jacobson's dynamic window scheme [1]. The control of multicast real-time data presents a new set of problems.

The goal of TCP and other mechanisms for data traffic is to maximize throughput and minimize packet delay or loss, or equivalently to optimize some function of throughput and delay such as power. The goal for real-time traffic is to optimize the utility of the information delivered to the receivers. These goals are not necessarily equivalent. Therefore, the mechanisms proposed for the control of data traffic cannot be expected to be suitable for real-time traffic.

We observed earlier that real-time applications differ from network applications in that they require a certain minimum level of service to offer utility to the end user, whereas traditional applications will take whatever service they can obtain. Thus, there is a floor to the rate at which a real-time source can transmit and still send a useful stream. This presents the problem of who to satisfy when two applications compete for the same bandwidth, and whose combined minimum bandwidth requirements, i.e. combined floor rates, exceed the available bandwidth. The decision is a policy issue, and it is up to external forces to resolve the problem, either by turning off an application, or by negotiating with the network provider to get more bandwidth.

One solution is to say that who pays the piper calls the tune, i.e. whoever has administrative control over the network should be able to decide who shall get priority. However, in the absence of a visible reservation scheme - one in which a manager can place "policy" constraints as to who can make reservations when, and who can pre-empt who - alternative mechanisms must be used, as well as a default decision making process. The mechanism we describe in this chapter cannot distinguish between flows, but it does make the problem of congestion visible to the application and to the user. It is then up to the application or user to sort out the policy issues as to who can use the available bandwidth[2].

We also make congestion visible to the user in order to cope with the problems of using heterogeneous applications, e.g. an audio and a video application. This is intended to meet the needs of video conferences, in which the audio is generated by some other application, or where the video streams are prioritized according to some conference management protocol[3]. By allowing the user to discover when congestion is experienced they can institute the conference policy on the priority of the particular video stream. For instance, they may choose to suggest to the congested users that they leave the video stream, and use only audio.

Control mechanisms for real-time traffic must be able to work in a multicast environment, and specifically to scale up with the number of receivers in a multicast group. Clearly, it is only if we push the responsibility for detection of congestion in part of the multicast tree onto the receivers that we can expect any scalability in a scheme.

[2] The class based queuing mechanism described in Appendix A is one possible solution to this problem.

[3] For example in MICE conferences, the audio is handled by VAT, and the video is handled by IVS.

Although receiver-based detection of congestion is possible, it is rarely seen since unicast communication generally involves sending information about data received back to the sender anyway, which can be processed by the sender and used to initiate control actions. Another advantage for the receiver-based congestion detection is that the receiver can use metrics suitable for the local network technologies, creating the possibility of heterogeneous congestion signals in use on a single multicast distribution tree. To aid in the design of a generic algorithm, we use generic network state variables, namely UNLOADED, LOADED and CONGESTED. We use such variables because the heterogeneity of today's Internet gainsays any attempt to use a single metric to determine state. Instead we allow each host to make a local decision as to how it shall measure the state of the network, using an appropriate local measure.

The congestion control algorithm then becomes a gradual increase in bandwidth from the source to either the maximum useful rate or to the rate that drives one of the receivers to decide that the network is LOADED. The source then transmits at this rate, continually polling its users in a scalable manner to ensure that the network doesn't become congested and can take advantage of when the network is no longer LOADED. When a receiver detects that the network is CONGESTED, it tells the receiver, and the sender takes the appropriate action.

6.3 A scalable feedback mechanism

In this section, we describe a mechanism for eliciting feedback information from the receivers in a multicast group. We consider the general case when the size of the group is not known by the source or by any of the receivers. In some cases, additional information is available about the group. For example, many conferencing applications use a tight session control in managing the conference, in which case the identities of all the members are known. This information can be used to tune the feedback mechanism, and in particular to select good initial values. However, this information is not required since, as will be shown, the algorithm can be used to estimate both the size of the group and the number of receivers experiencing a particular network state.

6.3.1 Avoiding implosion

Soliciting information from receivers in a multicast group of indeterminate size might create a so-called implosion problem, in which a potentially large amount of feedback information is sent almost synchronously from the receivers back to the source [133]. Solutions to this problem have included probabilistic querying, randomly delayed responses, and an expanding scoped search [134].

In a probabilistic scheme, a receiver responds to the request from a source with a given probability. Typically, the request is sent again by the source after some timeout interval if no reply has been received. The probabilistic scheme is easy to implement, but it has limitations. For example, the source is not guaranteed to receive the worst news from the group within a certain time period. Furthermore, it is not clear how to set the probability of replying when the size of the group is not known. However, one advantage is that it is possible to bias the probability associated with a receiver according to the importance of the receiver.

In the random delay response scheme, each receiver delays the time at which it sends its response back to the source by some random amount of time. Clearly, this scheme is not sufficient to prevent the implosion problem, especially if the random delay is chosen too small. However, the scheme is very appealing, in the sense that it allows to receive the responses from all the receivers in the multicast group, if the delay can be adapted using some knowledge of the size of the group. This is done in VAT to time the state multicasts from multiple instantiations. There, the delay is set so that the bandwidth used by the state announcements is 1% of that by the audio stream [70].

In the increasing scope search scheme, the time-to-live (ttl) of the packets sent by the source is gradually increased. Therefore, packets travel further and further along the branches of the multicast tree. This scheme is clearly efficient when all that is required is to find the receiver closest to the source. However, we typically need to locate the receiver with the worst view of the state of the network. This receiver is unlikely to be located close to the source given the typical topology of a wide area network. Furthermore, the scheme is not expected to scale up well since the distribution of receivers in the ttl bands is not uniform, as most receivers are likely to be close to the source if only because of time-zone differences.

Our mechanism attempts to combine the best part of the above schemes. We describe it next.

6.3.2 Description of the mechanism

The mechanism consists of a series of rounds of probing of the multicast group, in which we find the state of the network corresponding to the worst positioned receiver, estimate the number of receivers and elicit the worst round trip time (rtt) to any receiver in the group so that we know how to time the series of rounds. We term a series of rounds an epoch. Once the state of the network is discovered, we signal upwards to the application for the adjustment of the rate of the output stream if necessary, and we terminate the current epoch. How the rate is adjusted is left unspecified, since this is a matter of policy for the application. A specific example is described in Section 6.4.

The algorithm we describe here relies on probabilistic arguments for scalability. Both the source and all the receivers generate random[4] keys of length 16 bits at the start of an epoch (this size is justified in Section 6.5.1). When the source wishes to solicit responses from its receivers, it sends out its key and a number indicating how many of the digits of the key are significant. Initially, all digits are significant. If the the source key equals the receiver key, using the declared number of significant digits, then the receiver can send a response, according to the rules described below. If there is no response within a timeout set at twice the largest round trip time in the receiving group, then the number of significant digits is reduced by one, and the source issues the same key for another time period, and so on, until either the required number of responses indicating congestion has been received[5], or we have passed a round in which no digits were declared significant, in which case any receiver can send a response. This ends the epoch.

Figure 6.1 shows the header of the packets sent by the source. Figure 6.2 shows the header of the packets sent by the receivers. Our algorithm requires that packets sent by the source include a timestamp and a sequence number, both of which are available in RTP[6].

The receiver responds on matching keys in two cases to allow the sender to estimate the size of the group and to learn the worst case state of the network.

[4] Regenerating keys at the beginning of every epoch eliminates the possibility of bias.

[5] This is currently set at one, although a more sophisticated analysis may use more responses

[6] We are currently carrying these headers as application-defined options in RTP.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|S|R| u |Sta| u |rttshft|keyshft|         key                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     maximum rtt in milliseconds                               |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

S = SIZESOLICITED; R = RTTSOLICITED; Sta = STATE from {UNLOADED, LOADED, CONGESTED}; u = unused bits; rttshft = the number of seconds to pick a delay from $2^{rttshft}$; keyshft = number of bits to use in matching the keys; maximum rtt = the maximum round trip time yet found in milliseconds.

Figure 6.1: Format of packet sent by the source

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|S|R| u |Sta| u |  delay in milliseconds                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     timestamp (seconds)       |     timestamp (fraction)      |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

S = response to SIZESOLICITED; R = RTTSOLICITED; Sta = STATE from {UNLOADED, LOADED, CONGESTED}; u = unused bits; delay = delay in a RTTSOLICITED response, 0 otherwise; timestamp = timestamp of the source packet that forced the response.

Figure 6.2: Format of packet sent by receivers

- If the SIZESOLICITED bit is set in the header sent out, then a matching receiver will send a response back to the sender. This allows the sender to estimate the receiver group size, since there is a simple relationship between the average round upon which a receiver first matches the key, and the size of the group. The SIZESOLICITED bit is unset in all packets sent out in subsequent rounds to receiving a response, to prevent unwanted packets coming back.

- The current worst state of the network seen by the source is sent out in the STATE field. If the receiver matches the key and the state that they perceive in the network is worse than the current advertised state from the sender, then the receiver will send back a response, reporting the state that they currently see. The sender will then set the state sent out to the worst case reported state.

Once a receiver has responded in a given epoch, it will not allow responses to the same match for a period equal to the advertised maximum rtt. If the same match is made after this period, then it responds again. This protects the scheme against problems of lost responses.

When the sender receives a response containing a LOADED state, it sets the new state in all packets that it sends out, and continues to solicit responses. The application is presumed to be continuing to send at the optimum rate for the health of the network and the application. If a response indicates CONGESTION, then the sender signals upwards to the application (in the hope that the application will do something sensible such as reduce its bandwidth), and terminates the current series of solicitations. It then commences a new epoch of solicitations, with the state reset to UNLOADED and the SIZESOLICITED bit set. If the sender manages to get all the way through the series of rounds, including the no matching required round without hearing of any part of the tree in a LOADED or CONGESTED state, then it signals upwards to the application that the network is UNLOADED, and that the application can increase its rate if it wishes. Once an epoch is completed, the sender initiates a new epoch. By continually probing the receivers with every packet, the sender will receive timely notification of network state changes.

Because the sender has an estimate of when the first match is likely to occur, the sender can adjust the number of significant digits sent out with the key so as to sensibly set the number of rounds taken to determine the state of the network in the first of a series of rounds. This significantly reduces the time taken to discover and react when things are going wrong. In addition, the round at which a response with a given state is received can be used as a hint to build an estimate of the number of receivers in a particular state. These estimates can be tracked using time averaging techniques.

6.3.3 Maximum round trip time discovery

Round trip time discovery is implemented by the receiver saving the timestamp of a packet, drawing a random delay from a uniform distribution from a time period set by the sender, waiting for this delay, then returning a packet containing the original timestamp and the actual delay which the receiver waited. Since the sender knows the approximate number of receivers it is sending to, it can select a time period so as to get a reasonable rate of responses back. In terms of the packet headers, if the sender sets the RTTSOLICITED flag in the header, then if the receiver is not currently waiting to send a response, it selects a delay using the size of the shift that the sender has set in the RTTSHIFT field to set the time period as $2^{RTTSHIFT}$ seconds from which it draws the random waiting period.

The sender solicits rtt responses over periods defined as $2^{RTTSHIFT}$ seconds. It sets the maximum rtt as the worst received. To ensure the maximum rtt reflects the current state of the network and the receiver group and to eliminate the effects of outliers, the maximum rtt is gradually aged out every rtt solicitation period. The maximum rtt is used to determine the timeouts and duration of everything in the algorithm, to ensure that every receiver will respond.

6.3.4 Prototyping in a real network

In this section we present results from monitoring the scheme working within a real network. Whilst this does not "prove" the scheme in any deep way, it is evidence of how the scheme would work for larger groups over real networks. The scenario is intended to show the effects of adding a number of receivers behind a low bandwidth link to a stable group on a high bandwidth link. We show a simulated source transmitting to a group of receivers, and we present a time series of how the bandwidth of the data traffic changes in response to control traffic from the receivers, and how the estimation of the group size changes. We used a packet generator, generating packets with an inter-packet gap drawn from a normal distribution with a mean set at the nominal rate set by the congestion control code. This generator ran over code implementing the transmit mechanism. The receivers implemented the receive mechanism and simply threw the packets away as they were received. The congestion meter used in the receivers was a simple loss based mechanism. The source would increase its rate by 1 packet/s after an unloaded epoch, and would reduce its rate by a half upon a congested epoch. Since an epoch is directly proportional to the measured rtt, and the rtt would become larger as the network became more congested, then the increase rate is less than linear.

The experimental scenario can be seen in Figure 6.3. Two separate Ethernets are connected by a multicast router and by a unicast router, through which their respective

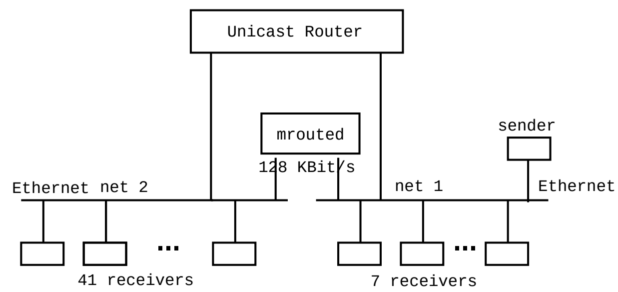


Figure 6.3: Demonstration scenario for the prototype

traffic flows. The interfaces for the multicast traffic are rate-limited to 128 KBit/s (the multicast router thus emulates a slow link), using the facilities of the 3.6 release of the mrouted code [135][7]. The unicast traffic is limited by the 10 MBit/s bandwidth of the Ethernet. The demonstration was run when the network was very lightly loaded. A single source is located on Net 1, along with 7 different receivers. The source and the receivers on Net 1 start at time=0. After 297 seconds, receivers are started on Net 2. These start up sequentially, being started via the *rsh* facility under unix. A total of 41 receivers are on Network 2. The sender sends UDP packets with 400 bytes of data plus 12 bytes of header, whilst the receiver generates a UDP packet with 12 bytes of header. In Figure 6.4, the data sent by the source can be seen to reach the maximum rate set and stay there. The slight non-linearities come from the difference in the measured maximum round trip time. After the additional receivers on Net 2 start, the source quickly adapts and the sending rate varies in the usual saw tooth associated with Slow Start congestion control. It should be noted that variations of sending rate between 64 and 128 KBit/s over 50 seconds would be noticed by the user as a periodic blip in quality. More sensitive measures of congestion could keep the queues within the LOADED region, and thus maintain a constant rate.

The control traffic in Figure 6.4 consists of solicited responses to estimate the size of the group, responses due to congestion and responses solicited to estimate the size of the round trip time. The sender attempts to set the period from which the receivers draw their random waits such that 10 responses are received per second. The estimator of the

[7] At the time of the experiment, the mrouted code had a bug which exhausted the buffers from the interface after a "while", where "while" depends on the prevailing traffic. For this reason the maximum rate used by the sender is fixed.

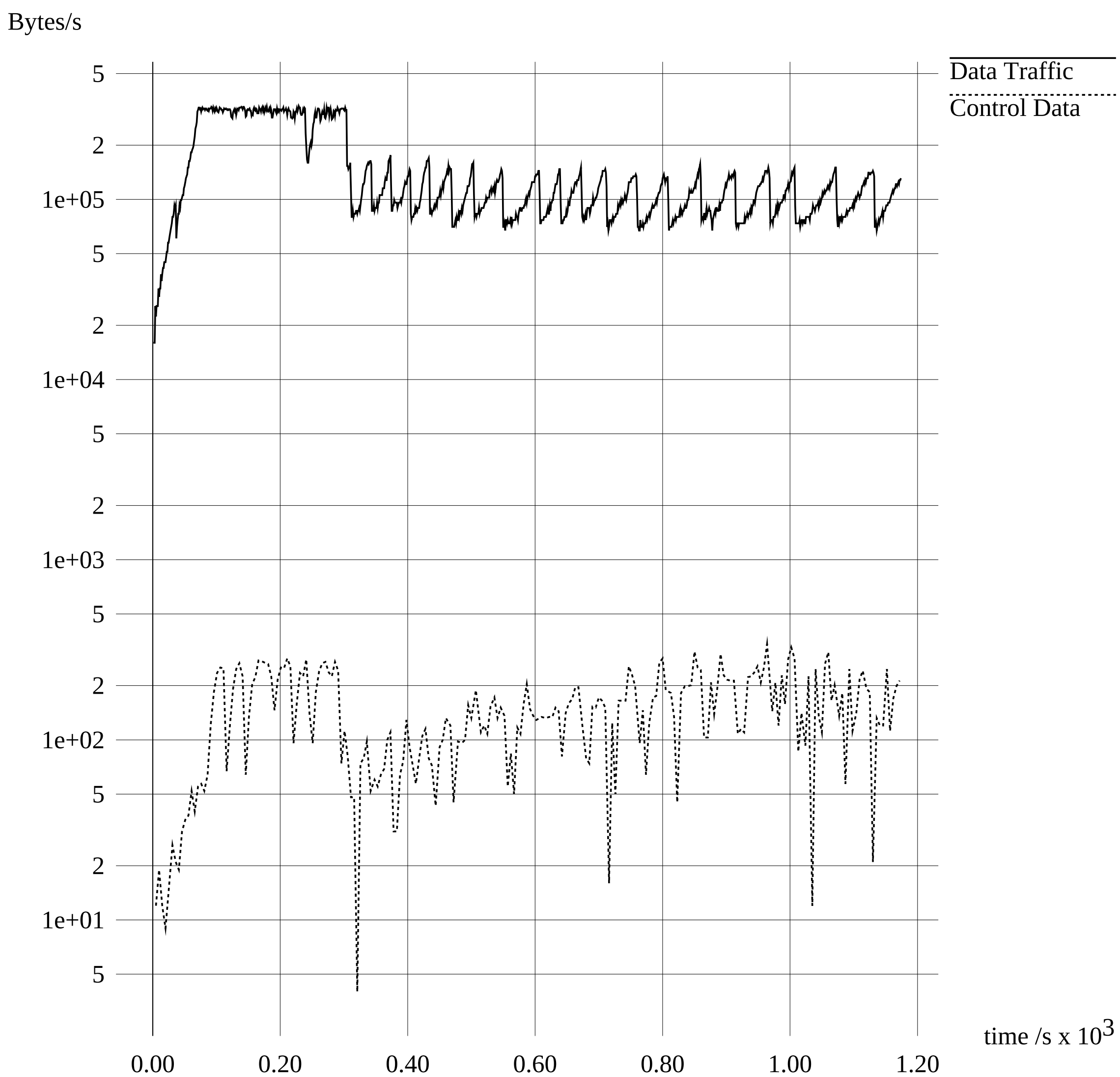


Figure 6.4: Bandwidth used for data and control traffic

receivers is a lagging estimator, and subject to fluctuations, so the traffic for rtt responses is random and tends to be higher than the expected 10 responses per second. Only one solicited size response is seen per epoch. In Figure 6.5, we can see the estimation of the

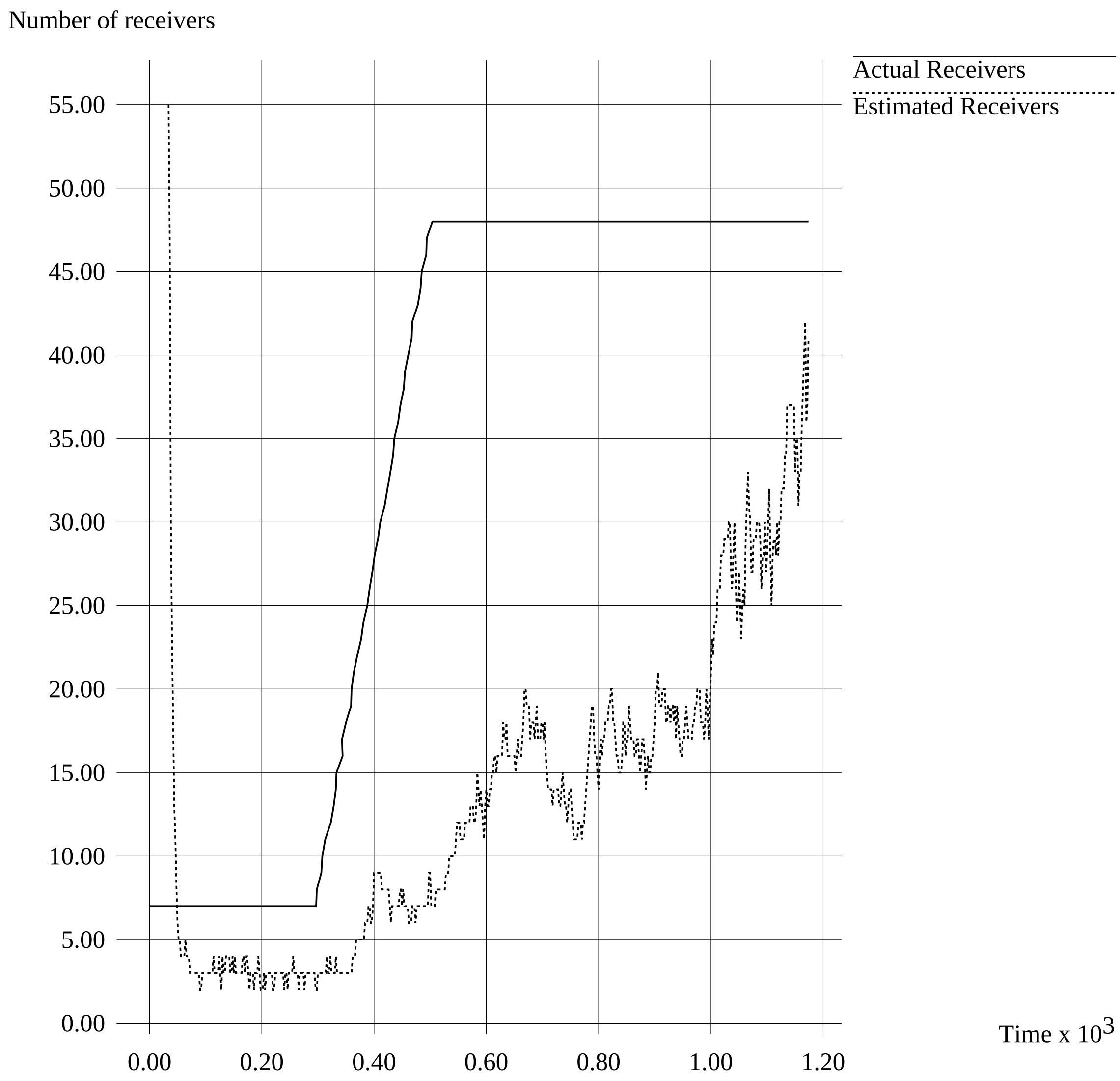


Figure 6.5: Estimated number of receivers

number of receivers. It should be noted that the actual number of receivers - 48 - still places us in the realm where there is inaccuracy in the approximation detailed in Section 6.5.1. If we have 48 receivers, then the expected first round for a response is round 11.06. If we use the approximation formula for this value, then we calculate that there should be 40 receivers. Thus it should be expected that the value should be at least 8 less. It can be seen in the graph that the estimate was starting to approximate this value.

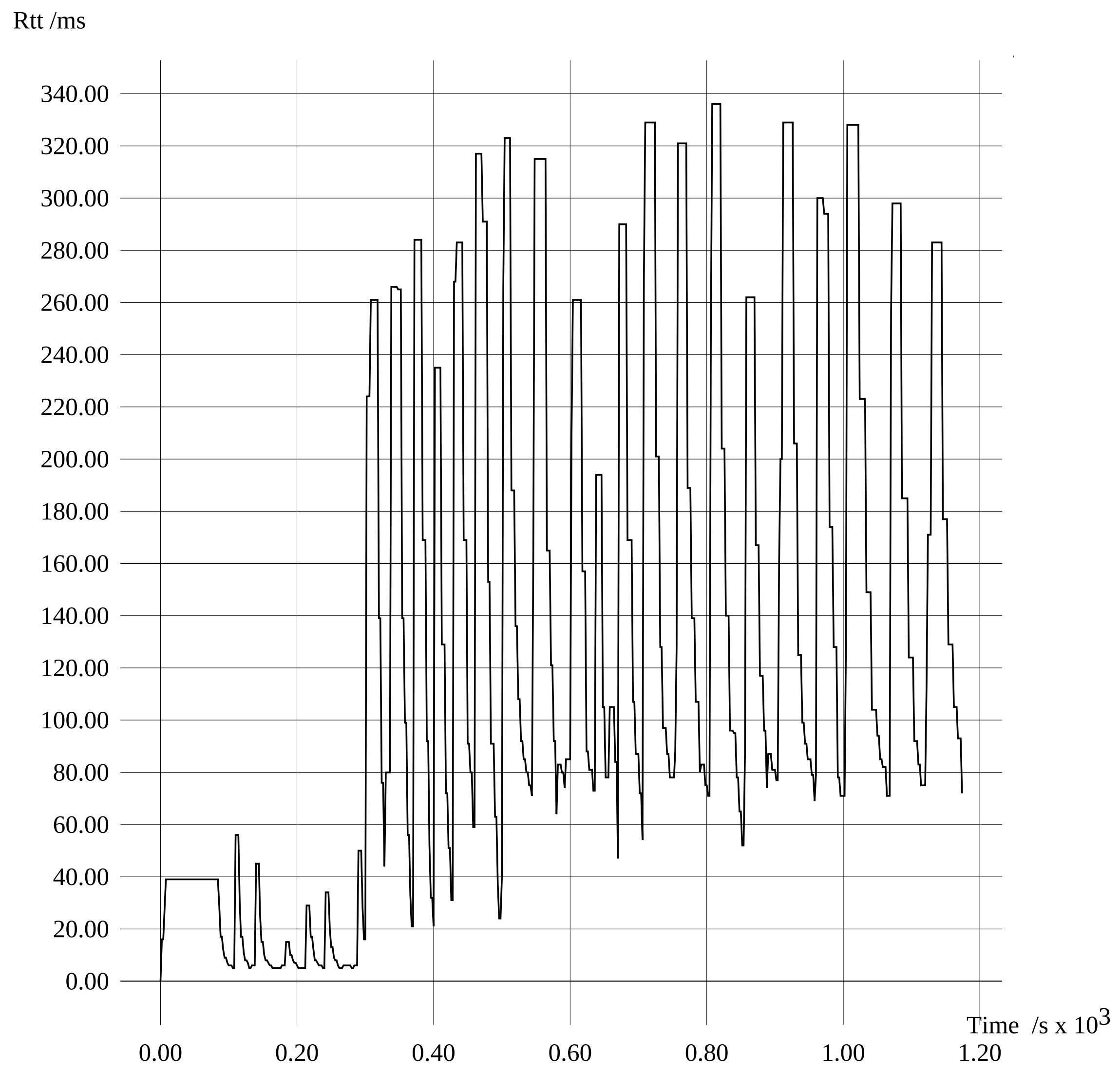


Figure 6.6: Measured worst case round trip times

In Figure 6.6, we can see how the measured worst case rtt varies at the transmitter. The rtt varies according to how congested the link is, slowing the rate of increase as the link becomes more loaded. Note that the sender decreases its rate as soon as the receiver signals pass the congestion threshold policy (in this case a single receiver congested), so there is not unnecessary delay.

6.4 An implementation of feedback congestion control for multicast video

In Section 6.3, we described a mechanism to provide scalable feedback in large multicast environments. In this section, we describe how this mechanism is used to control the video coder in IVS. In Section 6.4.1, we describe how to control the output rate of a coder by adjusting parameters in the coding process. In Section 6.4.2, we describe how the output rate of the IVS coder are controlled using the feedback information.

The mechanism described in Section 6.3 will provide the signals for congestion control. However, testing the mechanism was problematic. Since the mechanism is designed to scale to large groups in the order of tens of thousands, any useful simulation experiments would require tens of thousands of simulated hosts and a network complicated enough to support large groups. Current network simulation tools do not provide such support. It was also difficult to see how the source adaptation policies could be examined in simulation, since they need to satisfy user demands. Instead, the mechanism was tested in field trials, since the current Mbone tools didn't use any congestion avoidance mechanisms, and political decisions within the MICE project [136] decided that any control was better than none. A prototype test tool was built to validate the actual algorithm over small groups, and then the mechanism was integrated into IVS. IVS is a video conferencing tool for workstations from Thierry Turletti at Inria. It is designed around a software video codec for the H.261 standard.

6.4.1 Output rate control in video coders

Many algorithms have been proposed and standardized for the coding and transmission of video traffic. Example standards include ISO JPEG for single frame images, and ISO MPEG and CCITT H.261 for moving images [137]. A fuller survey of the choices for a network codec are described in Chapter 3. The use of a standard coding algorithm allows for inter-operation with hardware codecs and other software implementations, and it is for this reason that IVS is designed around the H.261 algorithm. We recognise that it is not the most suitable video coding algorithm for network video tools.

A central part of a codec is the compression/coding algorithm. In H.261, a picture is divided into blocks of 8×8 pixels. A discrete cosine transform (DCT) converts the blocks of pixels into blocks of frequency coefficients. These coefficients are quantized and then encoded using a Huffman encoding technique. In addition, images can be coded using intra-frame or inter-frame coding. The former encodes each picture in isolation. The later encodes the difference between successive pictures [50].

The H.261 standard documents describe how parameters of a coder can be adjusted to change the output rate of the coder. In IVS, we adjust three such parameters: the refresh rate; the quantizer; and the movement detection threshold.

The refresh rate characterizes the speed at which frames are grabbed from the camera. Decreasing the refresh rate decreases the average output rate of the coder. However, it also decreases the frame rate and hence the quality of the video delivered at the receivers.

The quantizer characterizes the granularity used to encode the coefficients from the discrete cosine transformation. Increasing the quantizer is equivalent to encoding the frequency coefficients more coarsely, and thus reducing the quality of the transmitted image. However, it is also equivalent to reducing the number of bits used to encode pixels, and thus reducing the output rate of the coder.

The movement detection threshold characterizes the blocks in a frame which are "sufficiently different" from those in the previous frame. If the threshold value increases, then the number of blocks which are considered to have changed since the last frame decreases. Therefore, the number of bytes required to encode each image decreases. However, increasing the threshold decreases the sensitivity of the coder to movement and yields lower image quality.

Thus adjusting the parameters of the video coder is an easy way, particularly in a software coder such as IVS, to trade off a lower output rate (i.e. lower bandwidth requirements) for a lower image quality. The specific requirements of a video application will dictate which of the three parameters described above should be adjusted. The refresh rate is adjusted if the precision of the rendition of individual images is important. The quantizer and the movement detection threshold are adjusted if the frame rate or the perception of movement is important. These requirements are taken into account in IVS as follows. The source specifies the maximum rate at which the video stream can leave the coder, which we denote by max_rate, and a mode. The mode characterizes which parameters are adjusted in the coder. In the *Privilege Quality* mode (PQ mode), only the refresh rate is adjusted. The coder then waits for a sufficient amount of time before encoding the next image in a sequence so that the output rate stays below max_rate. In the *Privilege Rate* mode (PR mode), only the quantizer and the movement detection threshold are adjusted.

6.4.2 Feedback control of the H.261 coder in IVS

We next describe the mechanism which controls the output rate of the H.261 coder in IVS using the feedback information. We consider first the feedback information and then the control algorithm used by the coder.

In Section 6.3, we described the mechanism used by the receivers to send information about the state of the network. However, we did not specify the assignment of the generic variables LOADED, UNLOADED, and CONGESTED. Recall that our goal here is to maximize the perceptual quality of the image received at the destinations while minimizing the bandwidth used by the video transmission. Therefore, we need to map the state variables to the perceived quality of the received image.

A subjective measure of the perceptual quality, referred to as the Mean Opinion Score (MOS), has been defined and used extensively to design and compare video coding algorithms [22]. However, a MOS-based feedback mechanism would be impractical, since it would have to include the user in some kind of continual rating procedure. We thus have to rely on objective measures. Unfortunately, objective measures typically do not reflect the user's perception of an image [22]. The signal to noise ratio (SNR) is an objective measure of the spatial quality of the image. However, numerous experiments have shown that it is an imperfect measure because the perceptual quality in a sequence of frames depends on the quality of each frame in the sequence. Another objective measure of perceptual quality is the loss rate of blocks encoded by the H.261 coder. Yet another objective measure is the frame rate, i.e. the rate at which video frames arrive at the destinations.

In the absence of a reliable objective measure of perceptual quality, we characterize the quality of the video delivered to the receivers by a function of all the measures mentioned above, namely the SNR, the block loss rate, and the frame rate. However, we note that the SNR cannot be computed by the receivers since it requires that the original image be available. The loss rate of the encoded blocks cannot be computed by the receivers either since the coder only knows which blocks in a frame were actually encoded. However, the loss rate for the blocks can be crudely approximated by the packet loss rate[8]. Finally, we note that the rate at which frames are sent by the source

[8] It would be possible to measure the loss rate for the groups of blocks (GOBs) at the receivers.

is known by the coder. Therefore, it seems reasonable to choose a feedback information based on measured packet losses at the receivers[9]. Specifically, each receiver measures an average packet loss rate observed during a time interval equal to an epoch. If the receiver can respond according to the rules in Section 6.3 (i.e. if its key matches the source key over the appropriate number of significant digits), it sends the state (i.e. UNLOADED, LOADED, or CONGESTED) corresponding to the measured loss rate.

An optimization of the information feedback mechanism when there is little danger of implosion has also been implemented. If the number of receivers is below a threshold, receivers send a negative acknowledgment (NACK) whenever they detect a packet loss. Upon receipt of a NACK, the source encodes the blocks which were in the lost packet using intramode coding. Observe that this will result in a refresh of the lost blocks (as opposed to a retransmission of these blocks, since the encoding is done on a new frame), and hence faster and better error recovery. The threshold is currently set at 10 receivers. An analysis of the feasibility of this sort of scheme can be found in [133].

Once the source has received feedback information from receivers, it must convert the values into a global measure that it can use to adjust its output rate. The approach used within IVS is to reduce the output rate only if greater than a threshold percentage of the receivers are suffering excessive packet loss. We use the probabilistic properties of the probing algorithm to detect this, using the difference between the round in which a response to the SIZESOLICITED request was sent and the round in which a CONGESTED match was found. The logarithmic relationship obtained in Section 6.5.1 can be used to show that the above difference is related to the relative proportion of congested receivers. The current incarnation of IVS sets this difference to 6 rounds in a "large multicast" environment (i.e. when the number of receivers is large), or equivalently the control reacts when more than 1.4% of the receivers experience congestion (refer to Section 6.5.1).

In an earlier version of IVS, the source reacted when up to than 50% of the receivers experienced congestion. Clearly, this could lead to situations where up to half the destinations receive low quality video. Both choices of the 50 or 98.6 percentile are *ad hoc*, and

---

However, there is no direct relationship between the number of GOBs in a packet and the number of blocks actually encoded in the packet. This is because a GOB header must be sent by the source even if no block in the GOB was actually encoded. Therefore, the GOB loss rate is not a good approximation for the loss rate of blocks. Refer to [50] for details.

[9] Note that this criterion does not take into account possible random losses [138] which would be mistakenly considered as indicative of network congestion.

none seems adequate in a network where links can have widely different bandwidths, and where loss rates can be widely different on different branches of the multicast tree. This suggests that more elaborate criteria be used by the source. Ideally, the source should be able to single out the parts (i.e. branches) of the multicast tree that experience high loss rates, and to treat these branches separately. One way to do this is to use sub-band coding [22]. However, this would require some cooperation from the routers. Furthermore, the H.261 standard is ill-suited to support sub-band coding[10]. Nevertheless, this approach is promising, and we are currently investigating it.

We now describe the control algorithm used by the video coder. Control actions are taken by the coder at the end of an epoch. The control algorithm strives to keep the average "quality" across the receivers above some value, or equivalently it strives to keep the loss rate below some tolerable value for the majority of receivers so as to keep the network in the LOADED region. This is done by adjusting the maximum output rate of the coder max_rate.

In TCP, the window size, and to a first approximation the rate at which packets are sent into the network, is adjusted using a linear increase/multiplicative decrease algorithm. This is done so as to ensure stability and to provide both efficiency and fairness in the sense that n sources sharing the same link would be allocated on average a fraction $1/n$ of the capacity of the link [139]. While this might be non-optimal for video streams (refer to our discussion in Section 6.2), we choose to be conservative and we use a similar linear increase/multiplicative decrease algorithm. In this algorithm, max_rate is decreased by half if the fraction of congested receivers $frac_{CONGESTED}$ is greater than some threshold value, which we denote by $thresh_{CONGESTION}$. max_rate is increased by a fixed value $rateIncrement$ if all the receivers detect the network as UNLOADED. We also make sure that the output rate is always larger than some minimum rate to guarantee a minimum quality of the video-conference at the receivers. Thus, the control algorithm is as follows:

If $frac_{CONGESTED} > thresh_{CONGESTION}$

$max\_rate = \max(max\_rate/2, min\_rate)$

else if $(frac_{UNDERLOADED} == 100\%)$

$max\_rate = rateIncrement + max\_rate$

[10] However see [54]. Other hierarchical schemes can be developed.

In IVS, the *rateIncrement* is set to 10 kb/s. We set default values for min_rate, $thresh_{CONGESTION}$, and for the maximum value of max_rate to 15 kb/s, 1.4% and 150 kb/s, respectively. Note that these figures are heavily dependent on the policy decisions used in the network. For example, it is reasonable to set the value of $thresh_{CONGESTION}$ to much higher than 1.4% when the number of receivers is small. Different policies may result in different figures or even different adjustment algorithms.

6.5 Evaluating our feedback control mechanism

In this section, we evaluate the performance of our feedback control mechanism from Section 6.5.1.

6.5.1 Evaluating the feedback mechanism

Let n denote the number of receivers in the tree, and i denote length of the key, expressed in bits, sent by the source. We now derive the average round in which we get a first hit, i.e. the round in which the key of a receiver matches the key of the source over the appropriate number of significant digits. We number rounds in an epoch starting from 0. Let r_j denote the number of replies sent back to the source during round j and p_j denote the probability a receiver replies during round j. We assume that no message is lost in the network. The probability that we get at least one response in a round, given that there were no responses in the previous round is

$$\Pr(r_j > 0 | r_{j-1} = 0) = 1 - (1 - p_j)^n$$

where

$$\begin{aligned} p_j &= 2^{-i}; j = 0 \\ &= 2^{j-1}/(2^i - 2^{j-1}); j > 0 \end{aligned}$$

This is because on the jth attempt, we have already tried 2^{j-1} numbers in the search space[11]. The average round $E(\text{First})$ when we get a first hit is then

$$\begin{aligned} E(\text{First}) &= \sum_{j=1}^{i} j(1 - (1 - p_j)^n)(1 - 2^{j-1-i})^n \\ &\quad +1 - (1 - 2^{-i})^n \end{aligned}$$

[11] Note that we would have $p_j = 2^{j-i}$ for $j \geq 0$ if a new key were generated at the source every round, as opposed to every epoch.

Figure 6.7 shows the graph of $\log E(\text{First})$ as a function of n for $0 \le n \le 10000$. The

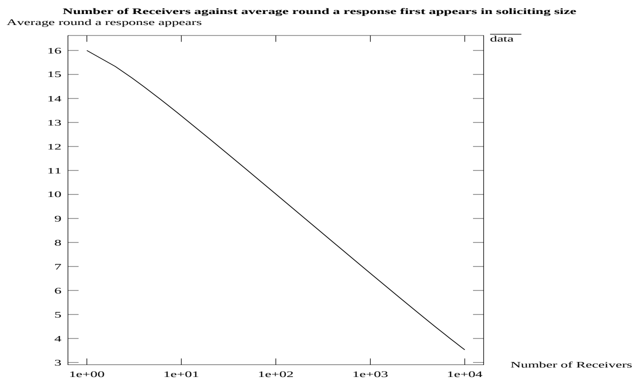


Figure 6.7: Expected round in which we get a hit vs number of receivers

graph can be fitted to a straight line, and we obtain

$$n \sim e^{16.25 - E(\text{First})/1.4} \tag{6.1}$$

A simple manipulation of this equation shows that a difference (in means) of 6 rounds corresponds to a ratio of sample sizes of around 1 in 72, or approximately 1.4%. In IVS, we assume that the network is congested if the difference between the round in which a response to the SIZESOLICITED request was sent and the round in which a CONGESTED match was received is equal to 6. The result above shows that this occurs when approximately 1.4% of the receivers are in the CONGESTED state (refer to Section 6.4.2).

The model above also shows that our algorithm essentially eliminates the implosion problem. Consider a simplified algorithm in which an epoch ends as soon as a reply is received at the source[12]. In this case, we have

$$\Pr(r_j = m) = \Pr(r_j = m | r_{j-1} \neq 0) \Pr(r_{j-1} = 0)$$

The conditional probability $\Pr(r_j = m | r_{j-1} \neq 0)$ is distributed according to binomial distribution with parameter $p = 2^{j-1}/(2^i - 2^{j-1})$ and number n. The table below shows the probability of receiving more than 10 replies during the different rounds in an epoch, when the total number of receivers is 10 000. The rounds not shown have probabilities lower than 10^{-10}. The numerical values clearly indicate that the probability of receiving a large number of replies during the first round in which a hit occurs is very low.

[12] In the real algorithm, a reply only changes the state advertised by the sender, which in turn reduces the potential population that can reply to subsequent source messages.

Round	4	5	6	7	8
Prob.	$2\ 10^{-8}$	$4\ 10^{-6}$	$9\ 10^{-5}$	$2\ 10^{-5}$	$3\ 10^{-9}$

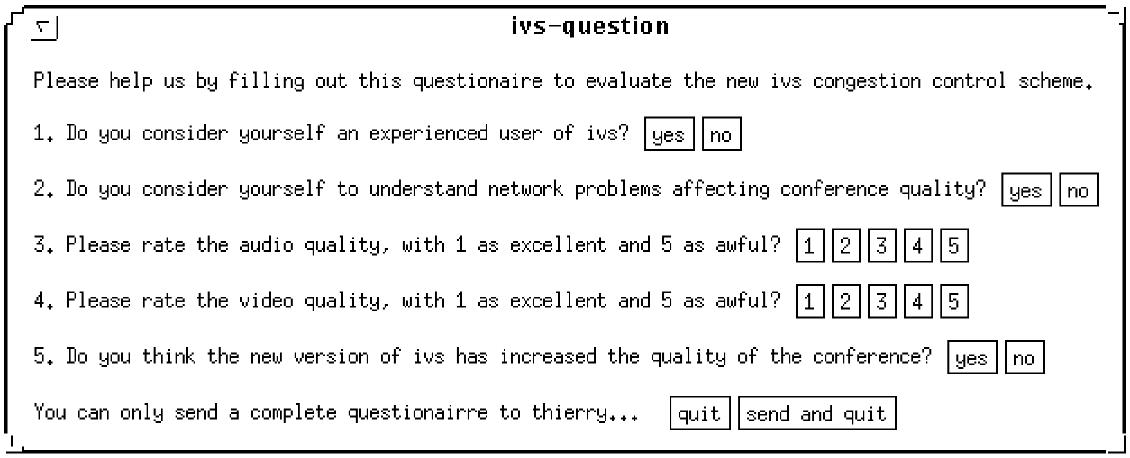
The scalability of the scheme is illustrated by Equation (7.1), i.e. by the logarithmic relation between the probability of receiving a reply and the number of receivers. For comparison purposes, consider the random delay scheme with 10 000 receivers. If the source is to receive no more 10 replies per second in response to one of its requests, then the random delay has to be drawn from the range 0-1000 seconds . The upper bound of one thousand seconds on receiving feedback information effectively rules out adapting the source data rate to network conditions. However, in our scheme, the largest maximum response time is equal to 32 times the worst case round trip time. For a typical worst case rtt of 500 milliseconds, the worst case state of the receivers can be found within 16 seconds. Since the probability of a response is dependent on the population size and the rules of the algorithm will always allow the first congested receiver to respond, the more receivers who are congested, the faster a response is returned.

As we mentioned earlier, a low bandwidth at the source translates into a lower image quality. There remains to quantify this bandwidth-gained/quality-lost tradeoff. The main problem is to find a way to estimate the quality of the video data delivered to the receiver. We argued in Section 6.4.2 that the loss rate is a good indication of this quality. Experiments show that the control mechanism decreases the bandwidth requirements as well as the loss rate at the receiver, as long as the video traffic makes up a significant part of the total traffic on the path from INRIA to UCL.

However, the loss rate is not enough to evaluate the quality of the video at the receivers. Therefore, we have carried out an informal evaluation of the effectiveness of our control scheme within the MICE project using questionnaires to discover user satisfaction with the quality of the images delivered to the receivers during the seminars and weekly meetings held between MICE partners. The questions asked can be seen in Figure 6.8.

Our results indicate that 90% of the time, our pool of 13 users believe that the conference quality has been improved by the new scheme. In the wider pool of 25 users with a number using the tool in the US, 34 from 47 responses believes the scheme to have made a difference in the quality of the conference.

Since our primary goal in implementing the scheme has been to protect the network rather than to increase quality for individual users, the only noticeable effect of the


ivs-question

Please help us by filling out this questionaire to evaluate the new ivs congestion control scheme.

1. Do you consider yourself an experienced user of ivs? yes no

2. Do you consider yourself to understand network problems affecting conference quality? yes no

3. Please rate the audio quality, with 1 as excellent and 5 as awful? 1 2 3 4 5

4. Please rate the video quality, with 1 as excellent and 5 as awful? 1 2 3 4 5

5. Do you think the new version of ivs has increased the quality of the conference? yes no

You can only send a complete questionairre to thierry... quit send and quit

Figure 6.8: The IVS evaluation questionaire - an X-windows dump of the form

scheme for the users is to reduce degradation of audio quality in periods of congestion. The results are encouraging in this respect. However, they are at most indicators that the scheme is improving network congestion and that users will accept variable quality as a result of congestion; as an experiment, there are too many uncontrolled factors to draw any hard conclusions. The video quality from IVS is dependent upon the performance of the video frame-grabber and the loading on the workstation, whilst the Mbone and the Internet are still variable in their connectivity and load. Further work building on the experiments in [23] is required to investigate the acceptability of varying the video quality to control congestion.

A limitation of our scheme is that the congestion signal may become a global synchronisation point. It is argued in [140] that synchronisation of events in a large network may be an emergent behaviour, and it is unlikely to be beneficial. Since congestion on a link is reported quickly in our scheme, and the behaviour of our scheme is synchronised by this event, wide-spread use may result in cycles of bandwidth increase and decrease by video sources. Engineering this cyclic behaviour out requires randomness to be introduced into the source response. One possibility is for the source to change its bandwidth with a probability weighted by the expected receivers undergoing congestion.

6.6 Related work

Until recently, video was typically transmitted over CBR networks such as the narrowband ISDN. Since the rate of a video sequence can vary rapidly with time, the problem was to obtain from a variable rate sequence a constant rate stream of data that could be sent into the network. This is typically done by sending the video stream into a buffer which is drained at a constant rate. The amount of data in the buffer is used as a feedback information by a controller which adapts the output rate of the coder to prevent buffer overflow or underflow (e.g. [141]). Recently, packet-switched networks have been available that can provide channels with deterministic guarantees [132]. A control mechanism suitable for such networks is described in [142].

Feedback control mechanisms have also been proposed for networks with VBR channels such as the Internet. There, feedback information about the changing state of the channels is used by a controller to adjust the output rate of the video coders [7] (feedback controllers have also been developed for audio coders, e.g. [143]. Examples of such controllers are described in [99,144]. However, none of the mechanisms described in these references handles multicast distribution. The scheme in [99] requires that switches send their buffer occupancies and service rates back to the source. The scheme in [144] describes a mechanism in which the time at which video packets are sent (and hence the rate at which the image is refreshed at the destination) is controlled, rather than the coding process of the video frames (and hence the quality of the image received at the destination). This is similar to our control scheme when in the PQ mode. Other schemes are designed for specific applications, typically to control the delivery of video streams from a server to client workstations [145,146].

6.7 Conclusion

Video applications are thought to fall in the class of adaptive applications [112], which can be shown to benefit from sharing rather than reserving the resources of the network [147]. However, sharing might lead to over-consumption of a resource by a greedy application. Our work provides a way to control video applications in a way that scales up well with the number of receivers in the multicast tree. The control mechanism makes it possible to hold video-conferences with reasonable quality even across congested links without requiring special support from the network such as admission control or resource reservation mechanisms. Furthermore, our mechanism prevents video sources from swamping the resources of the Internet.

The algorithms and techniques we described can be used in other tools such as NV or VAT. Specifically, our mechanism can be used to control the output rate of MPEG or JPEG-based coders. Also note that the scalable feedback mechanism can be viewed as a distributed searching technique. As such, it can be used in other applications where we are attempting to elicit the worst or best service from a large number of servers, such as an attempt to find the least loaded server for a load balancing system amongst a large number of servers. By using RTP options and offering a standard application programming interface to the code, the probing mechanism can be incorporated easily within the code of these applications.

The scalable feedback mechanism and the video control algorithm described in the chapter are included in release 3.3 of IVS.

Chapter 7

Conclusion - Whither Video on the Internet?

In conclusion, we re-iterate the contributions of this thesis, and then establish some historical context in which we discuss future work.

The contributions are:

- We have presented a combined admission and congestion control algorithm for unicast video and other real-time streams using the minimum quality threshold, and proven its stability under certain assumptions and shown it working in simulation.

- For multicast congestion control, we have argued that the response to congestion is an application policy issue, dependent both upon the network provider and the application user group. We have shown a novel probing mechanism for deriving the state of the network that scales well with the size of the group. The mechanism has been implemented and works in the popular IVS video conferencing tool.

This work started in late 1991[1], when there was a resurgence in interest in using the Internet for wide area voice and video services [131]. Workstations appeared with cheap audio input and output devices, using consumer chips based on telephone subscriber line devices. Affordable video frame grabbers were appearing, which made possible the use of conventional workstations for video conferencing experiments. Given that the Internet was being used for voice and video anyway, the obvious question to be asked was whether

[1] After the initial literature survey, the unicast scheme described in Chapter 5 was designed and simulated in the Summer and Autumn of 1992. The multicast scheme from Chapter 6 was designed and implemented in 1993 and was released in IVS in the Spring of 1994.

the Internet could be turned into a true multi-service network. There were a number of advantages that could be leveraged if this was possible (borrowing heavily from [148]):

- The Internet already has large global connectivity, which is growing fast (over the deployed telephone network).

- The trend in the world telecommunications network is towards greater De-regulation, with more providers needing to inter-operate. A larger number of providers means more and smaller autonomous domains, with more domain boundaries to traverse. The routing of IP has been designed to cope with multiple autonomous domains.

- With the introduction of the next generation IP [149], and the widespread deployment of multicast, IP will scale better for large distribution groups.

- The IP service abstractions protects the user from the details of the subnetworks' technology, and thus allows technology to evolve whilst maintaining connectivity.

As part of this effort, the principal motivation behind this work was to see if the end-system based congestion control techniques already in use on the Internet, such as the TCP Slow Start [1], could work for real-time services. From the above contributions, the answer is *"yes"*.

However, there is one major caveat. Both schemes described require the end-systems to conform and co-operate about reducing bandwidth when there is congestion. For single enterprise networks, or for networks where everyone is in a single community, it may be possible to assume this. But is it reasonable to assume that all users will be altruistic in a global multi-service network, especially when they pay for services? Probably not.

This does not invalidate the approach. A global multi-service network will require traffic accounting and charging, and this will require the switches to provide more control over the packets carried. If policing mechanisms must be deployed so that the contracts between carriers and users are honoured, then the policing can also be used to protect co-operative network citizens from misbehaving citizens.

Since the early 1990s there has been continued work on producing a commercial IP multi-service architecture that builds upon the strengths of IP. One approach is the class based queuing strategy described in more detail in Appendix A and in [150]. The protection and quarantining provided in this approach are very suitable for the co-operative schemes in this thesis. Another approach is described in [71]. However, these strategies still only contain pointers towards how the problems of congestion management can be solved. A number of areas related to the work within this thesis still need to be fully investigated, besides those detailed in individual chapters.

For the practical implementations in this work, the H.261 compression scheme has been chosen, since it is a standard, but it does have limitations as a network codec. It is hoped that the next generation of video compression standards will take into consideration the requirements of packetised video, as in Chapters 2 and 3. MPEG 2 [151] appears to be a step in the right direction.

The traditional approach to traffic engineering for congestion management has been able to ignore considerations of human factors to a large extent. However, the transport of audio and video information for direct human consumption requires us to incorporate concerns for ergonomics into the design of the traffic control mechanisms. The flexibility afforded by the use of sophisticated computers as terminals allows us to simplify aspects of traffic control - the minimum quality threshold allowed admission control in Chapter 5 to be very simple. Indeed, the re-examination of all aspects of the constraints on telecommunications traditionally imposed by human communications will prove valuable in simplifying the multi-service networks. The tolerable round trip delay for echoless voice communication is quoted as being between 600 and 1200 milliseconds, based on research at Bell Labs [152]. But if visual cues are present indicating that a speaker is about speak, such as in VAT, then much longer delays are tolerable.

Whilst there is some evidence for the minimum quality threshold, it remains to be proven that it will be acceptable to all users. Technically-aware users may accept variability as a result of congestion, but the naive consumer may not be so happy. Work is needed to investigate how people react to variable quality video (and audio) in telecommunications, and how it affects task performance. Other areas of human factors that need to be investigated of relevance to this facet of traffic engineering include:-

- The multi-service network of the future will (hopefully) have a continuum of services available, at a variety of costs to the user and to the network. Currently, networked applications either try to hide the existence of the network from the user, or present a complicated array of measurements and parameters to the user that cannot be easily related to the task in hand. The design of network abstractions that can be used and understood by the network user and related to their use of the network is becoming very necessary.

- How should the minimum quality threshold for a particular application be determined? One can envision the use of bandwidth controls in the user interface of applications, but these may overburden the user. Techniques for automating this process need to be investigated.

- The necessary bandwidth will depend heavily upon how good the compression of the scene is, and this will vary from scene to scene. To what extent can it be quantified, and what granularity is needed for use in traffic control algorithms?

The congestion control schemes described in Chapters 5 and 6 present a number of problems that have yet to be fully investigated, described briefly below.

Data Traffic In chapter 5 we have concentrated on continuous flows, where we have a minimum required bandwidth for the flow to be useful, and where we attempt to constrain the maximum delay. By contrast, traditional data traffic, such as file transfer and remote terminal access, have no bandwidth constraints, beyond moving all the data from place to place as quickly as possible. An obvious extension to our work would be to integrate the continuous media with data traffic. To protect the minimum quality of the video, the data would be transmitted at lowered precedence (since it could be recovered by retransmission). However, it remains an unresolved problem as to what congestion control mechanism to use for the data. A first order mechanism such as slow start would make the queue process oscillate between its target length and overflowing, forcing the continuous flows to suffer jitter, whereas a second order process would suffer from being unable to provide the data traffic with proportional constants that would be stable.

A solution to this problem is to provide a degree of isolation between the flows, such as the CBQ mechanism from Floyd and Jacobson [125] and in Appendix A, in which the flows sharing a link can borrow bandwidth from other classes when there is excess, but are limited to their own bandwidth when there is not.

Video Traffic In the simulation experiments we have modeled the sources by generating traffic with inter-packet gaps drawn from a normal distribution, and assuming that their mean data rate is infinitely adjustable. However, this is not realistic. Real compressed video sources will have different distributions for their output. In addition, codecs are unlikely to be infinitely adjustable. Instead, they are likely to be have only a set number of rates available. The effect of this granularity needs to be investigated, as well as how to decouple the codec from the network interface.

Misbehaving Sources The schemes described require all sources to co-operate in setting their transmitting rate to the level described by the algorithm. This is a feature of all end-system based congestion control algorithms. However, a misbehaving source may well be able to occupy the entire bandwidth and prevent other sources from transmitting. The only defence against this is for the switches in the network to detect this misbehaviour and to indicate to the source that it is misbehaving, and then to penalise it by denying service. It is a major challenge in engineering the multi-service network to design the necessary policing mechanisms.

Heterogeneous congestion signals In Chapter 6, the receivers are free to choose the congestion signal most suitable for their applications. Whilst this provides greater flexibility for the application, the consequences of this are not yet fully explored. Since future multi-service networks will be based upon a variety of technologies, some of which will provide explicit congestion notifications, a number of questions must be answered:

- Is it fair?
- Is it stable?
- Is it efficient?

Integration with other control schemes The approach taken to congestion control in the LBL suite of tools [153] - vat, vic and wb - is based upon the multicast dissemination of reception reports at a fraction of the bandwidth in delivering data, and then taking decisions based upon a statistical analysis of the reports. Integration of this mode of operation with the scheme in Chapter 6 is currently being worked upon.

Charging Model Bandwidth on data networks is currently paid for by large agencies as a recurring lump sum cost. However, this may change as the PTTs and other commercial companies enter more fully into the computer communications market. New economic models of the telecommunications market are required to guide how best to encourage the take up of the new offered services.

The models described guarantee the video flow a minimum bandwidth, but are able to soak up any excess. A charging model where the minimum bandwidth is the basis for the charge will simplify metering and accounting, and may encourage users to use the network out of peak hours, since they will be more likely to obtain excess bandwidth, thus spreading load. This is part of the general problem of how to charge for a *best effort* delivery model.

The charging model for multicast presents additional difficulties, since it is uncertain how the cost should be divided between the sources and the receivers. Depending upon the routing model, in addition to those resources required to deliver packets, network resources are used in constructing and maintaining the delivery tree and these resources will probably have to be paid for. It is hoped that the charges reflect the actual costs and are not a *premium* charge.

Global Synchronisation phenomena Signals for congestion such as lost packets can act as synchronisation points, and push the network into cyclic modes of operation. Since this is unlikely to be good for efficiency, the *emergent behaviour* of large scale networks must be studied.

Evolutionary introduction of technology Large scale service interruptions are no longer feasible in the Internet, since it is integral part of the communications fabric for many organisations. Like the telephone network before it, the evolution of technology must happen transparently for the user. As routers are designed which provide policing and appropriate scheduling for multi-service traffic, they will be introduced as replacements for bottleneck and other strategic routers. Gradual evolution through the release of software for end-systems can only work through incentives to the user to upgrade. One possible incentive would be to give better service for co-operative applications in new routers. The problems of transition need further study.

The last question to be asked is perhaps the most important; what is the most cost-effective, fair and efficient combination of admission control, scheduling and end-system feedback control? No single answer is likely, although it is hoped that the multi-service network of the future will be close to the optimum for all users.

Appendix A

Implementing Real Time Packet Forwarding Policies using Streams

A.1 Introduction

The Internet is fast approaching a period of revolutionary change in the services provided and how they are paid for. New architectures and protocols are being designed and implemented to extend the Internet to support Integrated Services, based on the work of the INT-SERV working group of the IETF [71]. It is envisioned that audio, video and other real-time services will be sent over the Internet. The ongoing commercialisation of the Internet necessitates a new model of service, where the users and providers exchange money for a guarantee of a basic level of service. Since the delivered service for both real time applications and contractual guarantees is dependent upon the mix of packets on the links and in the switches, a key component of the new Internet will be the packet forwarding scheduler within the switch. This must provide both a method for sharing bandwidth amongst the organisations who pay for the link and provide appropriate levels of Quality of Service for flows with real-time requirements.

One vision of how to design this building block has been offered by Sally Floyd and Van Jacobson [150] [125]. They start from the premise of link sharing, where links are leased by multiple organisations, or agencies, who then require a guarantee of a share of the bandwidth when they need it, but if the bandwidth is not used, then other users can

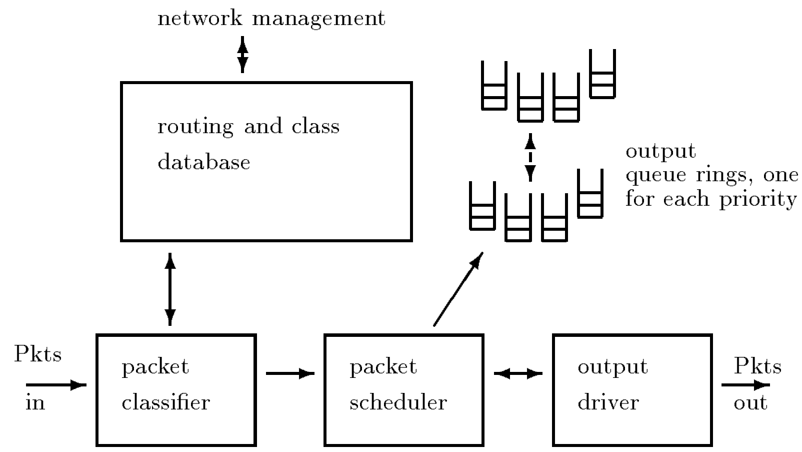


Figure A.1: Conceptual Breakdown of the CBQ filter Code

send packets. These requirements can only be satisfied in any sensible manner through the scheduling of packets to be forwarded. Each of the agencies are guaranteed a minimum amount of the bandwidth, with the proviso that any instantaneously unused bandwidth is shared amongst the agencies in some previously agreed upon manner. This technique for allocating bandwidth can be naturally extended to provide the allocation mechanisms for real-time traffic, providing the required Quality of Service (QoS) through guaranteeing bandwidth and low delay. The unifying abstraction for link-sharing and real-time traffic is the use of a class hierarchy; each class is provided with a share of the bandwidth and a delay characteristic, and the hierarchy enables mechanisms and policies for "borrowing" of bandwidth from other classes.

The conceptual breakdown of how CBQ should be implemented has been proposed by Van Jacobson and is illustrated in Figure A.1. The classifier interprets the header information of an incoming packet to determine the class of service that the packet should receive from the scheduler. The classifier returns a pointer to the class structure that holds the queues and associated information. The packet is placed on the appropriate queue by the scheduler if the queue is not full. The output driver works asynchronously, invoking the scheduler to determine from which queue to send the next packet, according to the current utilisation of the queues and their priorities.

Classifying a packet is very similar to the problem of determining the route matching a destination address in a packet. In both cases, fields in the header are used to look up information in a table. However, the classification problem is more complex because the patterns upon which the packet may match a class must be more general and can match any part of the header. For instance packets may be from one of multiple agencies, requiring the examination of destination and *source* addresses, classified on transport or other protocols such as TCP, UDP or ICMP, or applications such as ftp or telnet, requiring the examination of the port numbers. For video streams we may look even further within the packet to determine the level and "droppability" of a packet within a hierarchically encoded video stream [15].

It is a point of some controversy as to how widely the above mechanisms need to be fielded within the Internet. It could be argued that they need only be used where the links are heavily utilised, and thus subject to congestion. Links which have low utilisation can supply the necessary Quality of Service for all types of stream with normal FIFO packet scheduling, since queues on the links are very small or non-existent. The motivation for the work described in this paper came from attempting to make the allocation of bandwidth more efficient on the Trans-Atlantic link (known as the FAT pipe) used to connect the data networks of the UK Ministry of Defence (MoD) and the Department of Defence (DoD), of the European Space Agency (ESA) and NASA, and of the UK academic IP network and the US academic network. Currently, the link is hard-multiplexed into three parts, one for each agency pair[1]. However, large cost savings could be achieved on an expensive link if we guarantee each of the agencies a minimum share of the bandwidth for their mission, yet could allow any unused bandwidth to be used by agencies with excess traffic. The current version of the CBQ filter is fielded on the FAT pipe.

In this paper we describe the implementation of a programmable CBQ mechanism within the Streams implementation of IP forwarding. The classes and the patterns that determine the membership of a particular class are compiled in user space, and lookup engines are chosen to optimise the per-packet lookup code. The classes are then down-loaded to the CBQ filter module, which schedules the packets using the down-loaded information. The scheduling code is largely based upon the work of Lawrence Berkeley Laboratory (LBL), whilst the classifier and the surrounding infra-structure are the work of University College London (UCL). The particular choices made in the design of the classes and the classifier are described in Section A.2, the design of the Streams module

[1] Currently split into an E1 and a T1 link, with the T1 link split into two equal channels. For details on the usage see [11]

and its virtual interface are described in Section A.3 and performance measurements reported in Section A.4. We conclude in Section A.5 with ruminations on the feasibility of this design and the limitations of the design from the Streams mechanism and the pointers to future work.

A.2 Link Sharing Policies and the Classifier

The problem is to map the high level specification of policies onto a classifier that maps a packet into a particular class so that the necessary information can be found for scheduling the packet for output.

The high level specification of policies should be at a level of abstraction that can be used in the negotiation of legally binding sharing of the link. For instance, the agencies should be named as NASA or UK academic organisations, packet types should be specified by the services they provide - interactive data, bulk data, video data, audio data, specific sites, protocol suites (DECnet, IP etc.), and specific protocols.

Our initial class structure definition has the following hierarchical ordering when we are allocating the shares of the bandwidth:-

1. Agency

2. Protocol Suite

3. Protocol

4. Service

This partitioning first allows the bandwidth to be divided up by organisation, allowing arbitrary levels in this level. The next level is a suggestion that bandwidth should be given to protocol suites separately, so, for instance, OSI and IP packets are separated out in treatment. However, an actual implementation would reverse the process in classifying the packets, since the agency could only be discovered after knowing the structure of the packet. In this paper, we consider only IP. The partitioning by protocols allows protection against various forms of link sharing at the congestion control level - e.g. none versus slow start [1] versus responsible second order sharing [44] [43], [98] and to specify appropriate actions when the classes are exceeding their bandwidth allocation when the link is loaded. The final division by service allows us to divide up bandwidth amongst the applications. By adding a priority level at this point and implementing sensible borrowing policies in the scheduler, we can ensure that when there is congestion the service offered to the applications is degraded according to their importance, such as video packets being dropped before audio in a video conference.

A.2.1 What's a class?

Having derived the abstractions used to map a packet onto a class, we then want to consider the actual attributes that will be attached to a class. A class is :-

- A share of the bandwidth
- A priority
- A parent class
- A set of pattern tuples over a packet, $P_j = \{\{A_i, V_i\}\}$,where a pattern is:-
  - A pattern A_i.
  - A mask of significant bits V_i.

As happens in the solution to many computing problems, we define a hierarchy of classes. Rather than attempting to map a packet onto a class in one step, our pattern matching proceeds in order down the tree. We thus compare the packet against the patterns of the children of the current class. To ensure that a packet maps unambiguously only onto a single class, the patterns specified should follow the following constraint in the general case:-

- For each child of a given parent, each pattern of each child should be distinguishable from the patterns of all other children, ie for children i and j, there should exist no pattern tuples in i and j such that $A_i \wedge V_i \equiv A_j \wedge V_j$.

To simplify the implementation, and because the mapping abstractions correspond nicely to the packet headers we are using, the patterns are further constrained to map directly onto fields in the protocol header. For our initial IP implementation, the agency maps onto the destination and source addresses[2], the protocol maps onto the protocol field, whilst the application maps onto the port numbers in UDP or TCP [8]. The

[2] If a member of agency A is temporarily on walkabout, yet still wishes to use the share of the link, an additional pattern can be added to the class to cope with the temporary addresses.

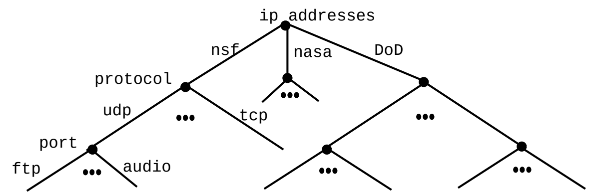


Figure A.2: Example Class tree hierarchy

abstraction of a class and the tree structure of the classes leads to the number of patterns we have to compare a packet against being exponential in the depth of the tree, using Figure A.2. In addition, the number of patterns at a particular level may be very large - e.g. at the agency level, the current design of protocols insists that we compare against the destination and source net numbers of the composite networks of the agency. For the initial case this can be as large as 10,000 or more (UK and US academic institutions). Furthermore, since the patterns will be repeated at each level (because each agency will want to partition traffic along similar lines), this will require a large number of patterns to be constructed if we attempt to classify the pattern in one step. Making a comparison at each level of the tree reduces the total number of patterns that need to be stored, but increases the number of comparison operations.

However, this is not a problem, since the class tree will be shallow - generally three or four levels - and we can select a lookup engine for a given class level according to the nature of the patterns to be compared against. The use of IP and simple classes allows the classification to proceed by a sequence of hash table or simple table lookups. Note that there is a default class as a child of the root of the tree into which a packet falls if it does not map to any other class. This default class gives the lowest quality of service, since it is not being used by any of the agencies paying for the link. It should be noted that the existence of a default class will encourage agencies to offer their link as a transit link for other agencies, and promote greater connectivity.

The use of sequenced lookups for the classes thus adds additional parameters to the class - a lookup engine function, and the data structures for the lookup engine.

Its then becomes natural for the network manager to specify the full set of parameters for the classes in a file, which is then compiled. Simple parenthesised definitions are used to define the tree. Patterns are defined very simply at present, just as a specification of the field within the IP packet and the pattern to compare it against.

The classifier compiler types the set of patterns that define the child classes of a class, checking that they can all be used to create a single engine. It then creates the data structures for the engine, placing the engine type in the class data, along with functions to lookup, insert and relocate the data, along with other parameters to describe a class.

Relocation of the data structures is necessary to allow the compiler to work in user space and then pass the data structures down to the kernel driver. The design is split in this manner so that we could experiment with reservation strategies - adjustments to the bandwidth and the priorities of classes can be worked out in user space and then passed down to the kernel. Additional management functionality, such as Management Information Base (MIB) creation etc., can be easily designed and added at a later date. An example configuration file is shown in Figure A.3, which is used in the tests described below. An alternative approach to packet classification is to use a generalised patricia lookup engine [154], [155]. Patricia constructs a tree which is traversed by testing only against the bits of the key which differentiate the patterns stored in the tree. The technique has been generalised to cope with masked keys by Halpern [156] and Tsuchiya [157]. However, because our patterns may have no bits in common, such as matching on one of either destination or source address, multiple passes are still required through the tables. In addition, tests against a generalised Patricia tree lookup engine show the sequenced table lookup as more efficient (Section A.4).

Another common mechanism for classifying packets is the automata used inside the Berkeley Packet Filter (BPF) and other similar entities [158]. These are designed to match a narrow range of patterns across a packet, and to do this very efficiently. However, the range of patterns we expect the classifier to handle is very wide, which would end up with a number of automata which would all need to be tried sequentially. Therefore we have generalised the framework of the classifier to use generalised engines. This does not preclude use of the a BPF automata as a particular instance of an engine, if that is the most suitable.

A.2.2 The scheduling properties of a Class

Each of the classes maintains an exponentially weighted moving average (EWMA) (avgidle in Figure A.4) of the idle period between packets, updated on every packet using the time it would take to send the same size packet using the percentage of the bandwidth allocated to the class. When the average is less than zero, a class is transmitting

```
# bandwidth of wire in
# nanoseconds per byte
linkspeed 20000
{ name  ucl-cs
# percentage of bandwidth
  bandwidth 90
  pattern [addr 128.16.0.0]
  { name ucl-cs-tcp
# percentage of total bandwidth
    bandwidth 50
# action when overlimit
    overlimit drop
    pattern [proto tcp]
    { name highTCP
# high number = high priority (max 7)
      priority 6
      bandwidth 29
      overlimit drop
      pattern [port 5001]
    }
    { name lowTCP
      bandwidth 19
      overlimit drop
      pattern [port 5002]
    }
  }
  { name ucl-cs-udp
    bandwidth 39
    overlimit drop
    pattern [proto udp]
    { name CBR
      priority 7
      bandwidth 38
      overlimit drop
      pattern [port 4579]
    }
  }
}
```

Figure A.3: Classes used in Testing the Streams Module more than its share of the bandwidth and is thus "overlimit". If the class of a packet is overlimit, the "borrow" class of the packet is examined to see if the original class can borrow bandwidth to transmit the packet, and so on up the hierarchy. If the class goes overlimit and can't borrow, then the overlimit action is invoked, such as dropping the packet, or delaying the packet. The maximum size of the avgidle estimator is limited by the maxidle parameter which prevents the class from building up credit when it isn't transmitting, and so limits the maximum burst size from the class.

```
struct rm_class {
  mbuf_t *tail; /* tail of circularly linked output q */
  struct timeval last; /* time last packet sent */
  struct timeval undertime; /* time can next send */
  int sleeping;      /* != 0 if delaying */
  int qcnt; /* # packets in queue */
  int avgidle; /* EWMA of idle time between pkts */
  struct rm_class *peer; /* Linked list of same priority classes */
  struct
  rm_class *borrow; /* Class to borrow bandwidth from */
  struct rm_class *parent; /* Parent class */
  struct rm_ifdat *ifdat; /* Output Device data structure */
  int priority; /* Class priority */
  int maxidle; /* Roof of avgidle */
  int offtime; /* Penalty added to class when overlimit */
  int qmax; /* Maximum queued pkts */
  void (*overlimit)(); /* Action to take when we can't borrow */
}
```

Figure A.4: The Class Structure related to Scheduling (after Sally Floyd)

The scheduling code maintains equal priority classes with traffic to send in circular queues. The packet to send is either from the highest priority class which is underlimit, or from the highest priority class which is overlimit but can borrow from classes above it in the borrowing hierarchy of classes. After a packet is sent from a particular class the queue pointer is advanced to the next position in the queue. Thus we implement prioritised round robin as long as classes are underlimit. When the class goes overlimit, it allows other underlimit classes to send first and so prevent starvation of any class. In this way, real time flows can ensure low delay by setting the priority of the class high and ensuring that the bandwidth share requested is sufficient such that they are always underlimit. When the queues are full or when the overlimit action specified is to drop a packet, the current implementation drops from the tail. However, other mechanisms such as dropping from a random position within the queue are possible [115]. The details of scheduler are described more fully in [150].

A.3 Implementation of a Streams-based Packet Forwarding Engine

The Streams "plumbing" used for the CBQ filter can be seen in Figure A.5. The CBQD module is used to communicate with the CBQ filter module and to download the class buffers and to obtain statistics on usage.

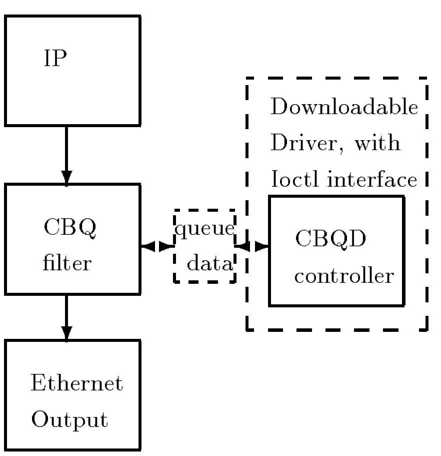


Figure A.5: The Streams Plumbing of the CBQ filter modules.

Our target scenario was for the workstation, a SparcClassic running Solaris 5.2, to sit transparently in front of the router whose output serial line we wished to protect[3]. The workstation and the router would be connected by Ethernet or some high bandwidth link. Thus we had to emulate the speed of the router interface in the CBQ software. This is done by tracking the time that the packet would be sent from the interface if it were the speed of the serial line in virtual time and suspending transmission if the virtual

[3]The obvious place to implement this code is in the router, but we had no way of modifying the router.

time gets too far in front of real time - ie the queue in the router builds up. We rely on incoming packets and a backstop timer to ensure that the transmission is continued at some point after being suspended. An additional advantage of running in "virtual" time is that the code becomes independent of output completion interrupts. The original LBL code used the completion interrupt to service another packet, but device drivers in Solaris do not provide this interrupt.

Implementing the CBQ filter as a Streams module and driver had some excellent benefits. We could introduce code into the protocol stack without requiring changes to the kernel source, with one major proviso. At the time the work was started the Streams plumbing code was built into a program ("ifconfig"); it was therefore necessary to take a copy of the "ifconfig" program and modify the source to enable us to insert the CBQ module at the correct point in the Stream. It would have been more convenient if the Streams implementations had used a configuration file to set up the plumbing, as some other implementations do.

As Solaris only supports dynamically loadable drivers we had hoped that it would be possible to write and debug the CBQ module without ever having to reboot the development machine. Unfortunately there was no obvious way of tearing down a Stream and creating a new one with a new CBQ module. Therefore it was necessary to reboot each time we wished to test a new module. If there were any serious bugs in the CBQ module the only way of recovering was to boot the machine from the boot prompt and edit the relevant startup files to not use the CBQ module. One way we attempted to circumvent this problem was to try and create a file before attempting to load the CBQ module - if this file already existed then the CBQ module would not be placed in the Streams stack. Unfortunately the network is configured so early in the Solaris boot sequence that the filestores are still readonly, so this strategy failed.

The scheduling code used was written at LBL and had been written for a BSD derived kernel. The first task was to change the code to use the Streams interface. This turned out to be quite easy; the BSD interface structure mapped quite simply onto a Streams queue and the "mbuf" structures mapped onto the Streams message buffers with the aid of some macros. The major change was the addition of some locks. The classification code was then integrated into the CBQ module.

A mechanism was required to control the CBQ module, to enable/disable the classification and to change the tables. The standard mechanism for sending information to drivers/modules is by obtaining a file descriptor to the driver/module and then issuing an "ioctl" which is understood by the driver. However the CBQ module is below a Streams multiplexor, and there is no way that the multiplexor layer can correctly deliver the ioctl message without rewriting the multiplexor code. The only way around this problem was to write a CBQ driver which accepts "ioctl" requests and makes a direct connection to the CBQ module, as in Figure A.5, using a shared piece of memory in the queue structure.

A.4 Performance measurements

A.4.1 Classifier performance

The classifier is designed to be flexible in the lookup engines used on the packets. The alternative design choice would have been to use a single engine, such as the modified Patricia code from Joel Halpern [156]. We compared the performance of the two approaches, using a classifier with multiple hash and normal table lookups with a modified Patricia engine. We specified the classes to consist of 1920 addresses culled from the NSFNet acceptable use database, with child classes of UDP and TCP traffic, and child classes of Telnet and FTP-data for the TCP class. We profiled the engines using the gprof profiler on a sparcStation10 under Solaris 2.3, and in all tests, the UCL classifier code was faster, even though the patricia code ignored the destination address and port fields, due to the limitations described in Section A.2. We present sample results from a test designed to exercise all paths of the engine in Table A.1.

Code	Lookups	Total Time spent /s	Mean ms/Call
UCL Classifier	15186	0.20	0.01
Patricia	15186	0.47	0.03

Table A.1: Classifier test results

A.4.2 Streams Module Performance

In the experiments described below, we use the setup described in Figure A.7. Both interfaces of the CBQ filter machine are on the same Ethernet, so we can see both input and output packets using a single tcpdump [159].

To measure the throughput of the Streams module, we measured the time taken to forward a series of packets well below the specified output rate of the filter for one of the connections on its own, and compared these with the performance of IP forwarding without the CBQ module. The results can be seen in Figure A.6. These suggest that a sparcClassic can forward 700 Kbit/s of minimum sized packets, or 7 Mbit/s of maximum sized packets on an Ethernet, and that traversing the CBQ module takes in the order of 300-450 microseconds. It should be noted that the current code is unoptimised - it does an extra copy to ensure alignment of the headers, and will pull up buffers without hesitation. The next version of the code will remove these inefficiencies.

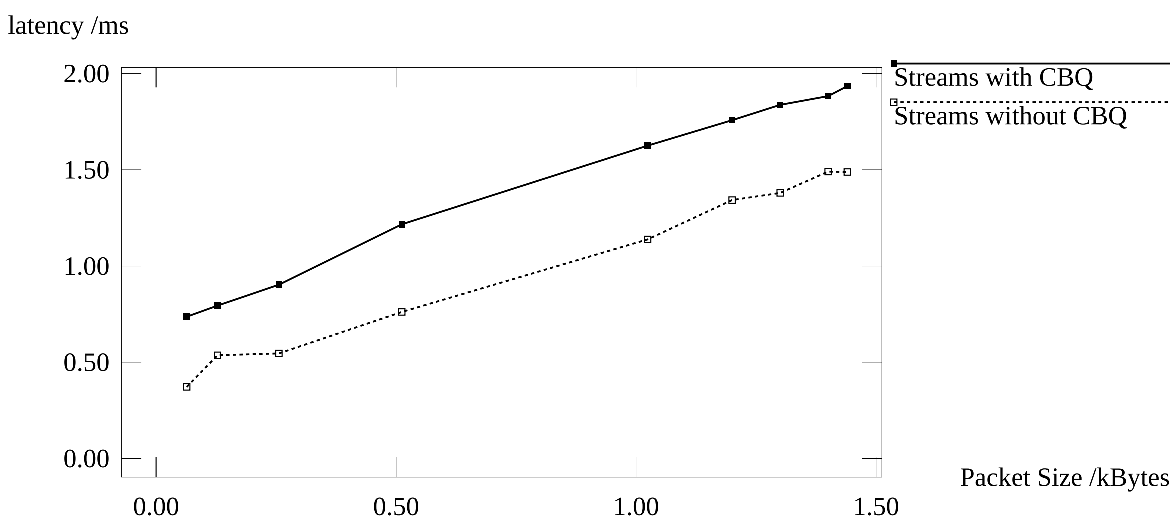


Figure A.6: Latency of Streams module against size of packet

To illustrate the performance of the Streams code in separating real packet streams, we set up an experiment using the classes in Figure A.3. The three flows are routed into the CBQ filter which exists on a dual-homed host with both ethernet outputs on the same physical segments. The flows are then routed to their target hosts via an ordinary cisco router. The return path does not pass through the CBQ filter. The medium priority TCP flow starts first, followed twenty seconds later by the low priority TCP flow. After a further twenty seconds, we get a minute of high priority constant bit rate UDP packets. The constant bit rate traffic is intended to simulate audio traffic.

The data gathered from the tcpdump were analysed and can be seen in Figure A.8.

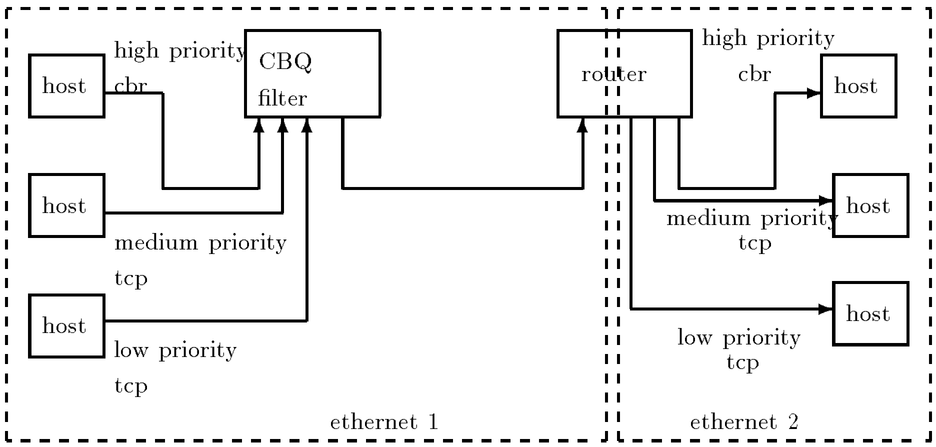


Figure A.7: Experimental Setup for Testing the Streams Module

The data are clumped in five second buckets for clarity. It is important to realise that the constant bit rate stream suffered no loss, whilst the TCP streams incurred loss when the congestion window was opened too large. In addition, the streams are kept to within some margin of their allocated shares of the 62.5 KByte/s that we have set the virtual Interface to.

The time series illustrating the delays suffered by the packets, measured by the time difference between the IP packets entering the filter and emerging on the wire again can be seen in Figure A.9. The data are bucketed in 1 second buckets. The high priority CBR stream suffers low delay, whilst the TCP streams suffer variable delays, dependent upon whether they are underlimit and thus transmitted in priority order, or overlimit and thus transmitted after any underlimit classes with traffic to send have been sent.

Non-intuitively the average delay experienced by the higher priority TCP class is larger than the average delay of the lower priority TCP class. This is an artifact of the particular borrowing strategy we've implemented. Packets that are sent in the higher priority TCP class when this class is underlimit are at higher priority and thus suffer low delay. When this class is overlimit, it has greater claim on unused bandwidth than any other overlimit class, but at lower priority than any underlimit class, so the packets sent using this bandwidth have a higher delay, contributing to an overall higher delay ($\bar{delay} = (delay_{low} \times packets_{low} + delay_{high} \times packets_{high})/packets_{total}$). Alternative borrowing schemes are detailed in [150].

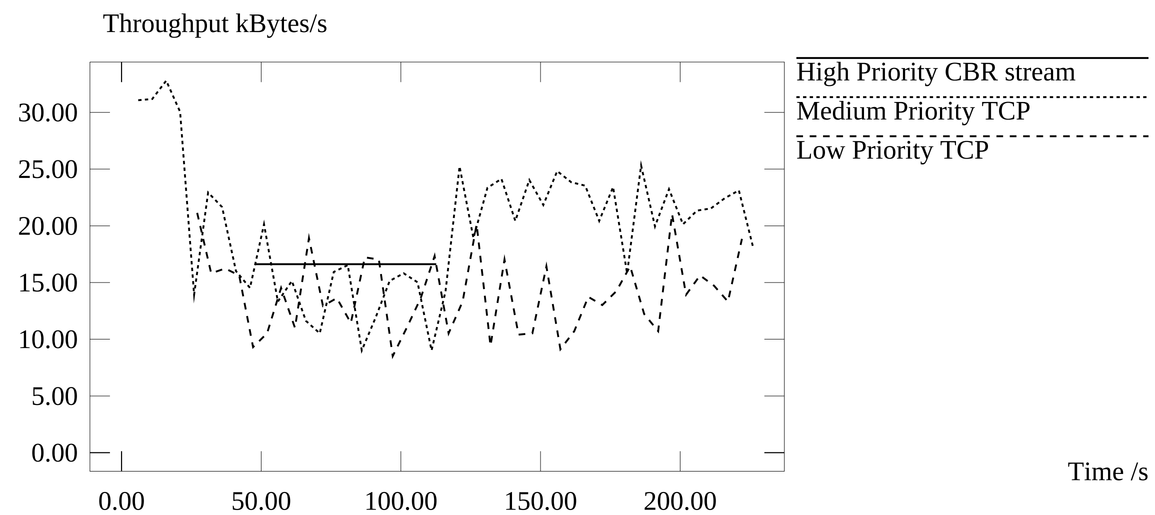


Figure A.8: Throughput for the illustrative Streams Experiment

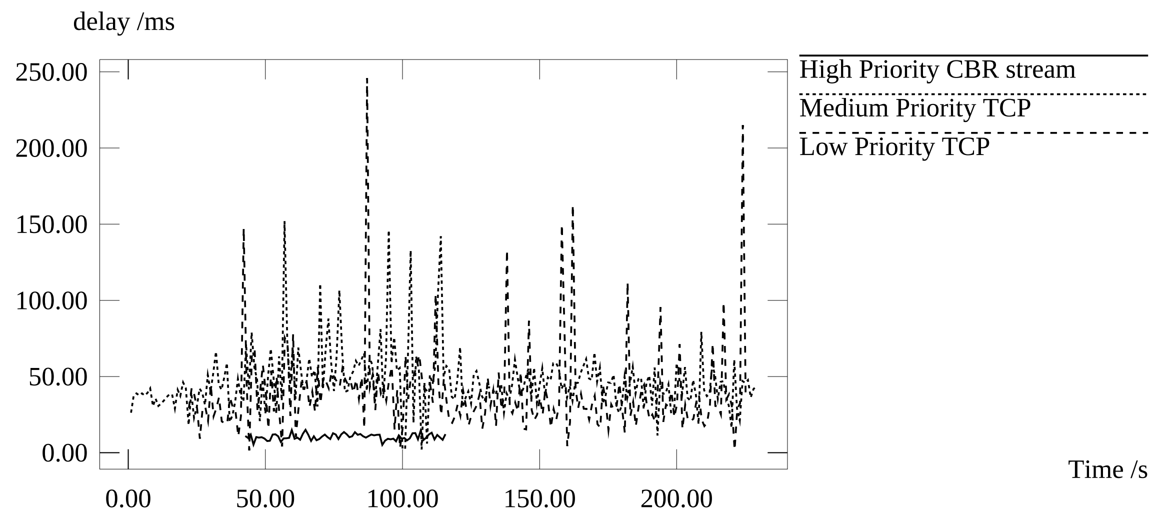


Figure A.9: Delay for the illustrative Streams Experiment:Average delays - CBR stream - 0.01s, high priority TCP - 0.045s, low priority TCP - 0.038s

A.5 Conclusion

We have described an implementation of the class based queuing strategy to provide link sharing, suggested by Sally Floyd and Van Jacobson. The Streams installation provided a modular environment in which to work. However, a number of factors proved a hindrance. The Solaris 2.3 release currently does not allow modules to be downloaded dynamically below the IP code. This slowed the development somewhat. In addition, some problems were encountered in determining when the packet had left the output driver.

The code is currently in use on part of the traffic passing over the UK-US FAT-pipe. This partitions traffic amongst a number of agencies, and allows the guaranteeing of multicast traffic for real-time applications a minimum of bandwidth and low delay scheduling.

We plan to extend this work to optimise the lookup engines and to allow more sophisticated pattern matching within the classifier. A remote management option will be added so that classes can be added, changed and deleted dynamically, without having to download all classes again. With this in place, we shall be able to experiment with real-time reservation, such as RSVP [160] and traffic management within the Internet.

References

[1] Van Jacobson, "Congestion Avoidance and Control," *Proceedings ACM SIGCOMM* (August 1988).

[2] W Sincoskie, "System Architecture for a Large Scale Video on Demand Service," *Computer Networks and ISDN Systems* 22,2 (September 1991).

[3] Thomas Little & Dinesh Venkatesh, "Prospects for Interactive Video-on-Demand," *IEEE Multimedia* 1,3 (Fall 1994), 14–24.

[4] C Huitema, "Software Codecs and Work-Station Video Conferences," *INET92*, Kobe, Japan (June 1992).

[5] Willem Verbiest, Luc Pinoo & Bart Voeten, "The Impact of the ATM Concept on Video Coding," *IEEE Journal on Selected Areas in Communications* 6,9 (December 1988), 1623–1632.

[6] Jeremy Bentham, "A Fragment on Government - the new authoritative edition," 1988.

[7] Michael Gilge & Riccardo Gusella, "Motion Video Coding for Packet Switching Networks - An Integrated approach," *SPIE Conference on Visual Communication 1991* (November 1991).

[8] D. Comer, *Interworking with TCP/IP, Principles, Protocols and Architecture*, Prentice Hall International, ISBN 0 13 468505 9, 1988.

[9] Mike Macedonia & Don Brutzman, "MBone provides Audio and Video across the Internet," *IEEE Computer* 27,4 (April 1994), 30–37.

[10] Martin De Prycker, *Asynchronous Transfer Mode - Solution for Broadband ISDN*, Ellis Horwood, 1993.

[11] Ian Wakeman, Dave Lewis & Jon Crowcroft, "Traffic Analysis of trans-Atlantic traffic," *Computer Communications* 16 (June 1993), 376,388.

[12] Will E. Leland, Walter Willinger, Murad S. Taqqu & Daniel V. Wilson, "On the Self Similar Nature of Ethernet Traffic," *Proceedings ACM SIGCOMM*, San Francisco Ca (August 1993).

[13] M. W. Garrett & W. Willinger, "Analysis, Modeling and Generation of Self-Similar VBR Video Traffic," *Proceedings ACM SIGCOMM*, London, UK (Aug. 31st - Sep. 2nd, 1994).

[14] Mark Hayter & Derek McAuley, "The Desk Area Network," *ACM SIGOPS Operating Systems Review* 25,4 (October 1991).

[15] Ian Wakeman, "Packetised Video: Options for interaction between the User, the Network and the Codec," *The Computer Journal* 36,1 (February 1993).

[16] Escobar, J, Deutsch, D, Partridge, C, "A Multi-Service Flow Synchronisation Protocol," *BBN STC Tech Report*, Cambridge, Mass (March 1991).

[17] Hui Zhang & Srinivasan Keshav, "Comparison of rate based service disciplines," *Proceedings ACM SIGCOMM*, Zurich (September 1991).

[18] ISO, "MPEG Systems standard," *ISO/IEC-11172-1* (August 1991).

[19] Wu-Hon F. Leung, Thomas J. Baumgartner, Yeou H. Hwang, Mike J. Morgan & Shi-Chiuan Tu, "A software architecture for workstations supporting multimedia conferencing in packet switching networks," *IEEE Journal on Selected Areas in Communication* 8,3 (April 1990), 380–390.

[20] Vicki Bruce & Patrick Green, *Visual Perception: Physiology, Psychology and Ecology*, Lawrence Erlbaum Associates, Hove, 1990.

[21] David Marr, *Vision: a computational investigation into the human representation and processing of visual information*, W. H. Freeman ad Co, San Francisco, CA, 1982.

[22] N. Jayant, J. Johnston & R. Safranek, "Signal compression based on models of human perception," *Proceedings of the IEEE* 81,10 (Oct. 1993), 1385–1422.

[23] Frank Wilson, Ian Wakeman & Walter Smith, "Quality of Service Parameters for Commercial Application of Video Telephony," *Human Factors in Telecommunication Symposium*, Darmstadt, Germany (March 1993).

[24] Leon Festinger, *A Theory of Cognitive Dissonance*, Harvard University Press, Stanford, CA, 1957.

[25] Valarie Zeithamd, "Consumer perceptions of price, quality and value: A Means-End Model and Synthesis of Evidence," in *Perspectives in Consumer Behaviour (fourth edition)*, Prentice Hall, 1991.

[26] H. W. Frowein, G. F. Smoorenburg, L. Pyfers & D. Schinkel, "Improved speech reception through videotelephony: an experiment with the hard of hearing," *IEEE Journal on Selected Areas in Communications,* 9 (1991), 611–616.

[27] J. Dowell & J. B. Long, "Towards a conception for an engineering discipline of human factors," *Ergonomics* 32(11), 1513–1535.

[28] Steve Whittaker, "Rethinking Video As a technology for interpersonal communications: theory and design implications," *Journal of Human-Computer Systems* (To be published).

[29] Judith S. Olson, Gary M. Olson & David K. Meader, "What Mix of Video and Audio is Useful for Small Groups Doing Remote Real-time Design Work?," *Proceedings of the 1995 ACM Conference on Computer Human Interface (CHI'95)*, Denver Co (May 1995).

[30] Paul Dourish & Sara Bly, "Portholes: Supporting awareness in a distributed working group," *Proceedings of the 1993 ACM Conference on Human Factors in Computer Systems (CHI'93)*, New York.

[31] Robert Fish, Robert Kraut, Robert Root & Ronald Rice, "Video as a technology for informal communication ," *Communications of the ACM* 36,1 (January 1993), 48–61.

[32] B Nardi, H Schwarz, A Kuchinsky, R Leichner, S Whittaker & R Sclabassi, "Turning away from talking heads: an analysis of "video-as-data"," *Proceedings of the 1993 ACM Conference on Human Factors in Computer Systems (CHI'93)*, New York.

[33] Bill Gaver, A Sellen, Christian Heath & Paul Luff, "One is not enough: Multiple views in a media space," *Proceedings of the 1993 ACM Conference on Human Factors in Computer Systems (CHI'93)*.

[34] Jim Hollan & Scott Stornetta, "Beyond Being There," *Proceedings of the 1992 ACM Conference on Human Factors in Computer Systems (CHI'92)* (May 1992).

[35] Henning Schulzrinne, Steve Casner, Ron Frederick & Van Jacobson, "RTP: A Transport Protocol for Real-Time Applications," *Internet Draft* (March 1995).

[36] CCITT H.221, "Frame structure for a 64 to 1920 kbit/s channel in audiovisual services," *CCITT White book* (1990).

[37] Robert J. McEliece, *The Theory of Information and Coding*, Encyclopedia of Mathematics and its Applications, Cambridge University Press, 1985.

[38] A. J. McAuley, "Reliable Broadband Communications using a Burst Erasure Correcting Code," *Proceedings ACM SIGCOMM*, Philadelphia (September 1990).

[39] Ernst W. Biersack, "Performance Evaluation of Forward Error Correction in ATM Networks," *Proceedings ACM SIGCOMM*, Baltimore (September 1992).

[40] Masahiro Wada, "Selective Recovery of Video Packet Loss using Error Concealment," *IEEE Journal on Selected Areas in Communications* 7,5 (June 1989), 807–814.

[41] J. Postel, "Internet Datagram Protocol," Network Information Centre, SRI International, RFC 791, September 1981.

[42] Nachum Shacham, "Multipoint Communication by Hierarchically Encoded Data," *Proceedings IEEE INFOCOM*, Florence, Italy (May 1992).

[43] Wang & Crowcroft, "A New Congestion Control Scheme: Slow Start and Search (Tri-S)," *Computer Communications Review* 21 (January 1991), 32–43.

[44] Ian Wakeman & Jon Crowcroft, "A Combined Admission and Congestion Control Scheme for Variable Bit Rate Video," *Journal of Distributed Systems Engineering* 1 (1994), 242–256.

[45] A. N. Netravali & J. D. Robbins, "Motion Compensated Television Coding - Part I," *Bell Systems Technical Journal* 58 (March 1979), 631–670.

[46] Ron Frederick, "Experiences with real-time software video compression," *Sixth International Packet Video Workshop* (July 1994).

[47] R. Frederick, "nv," *nv(1) Unix manual pages.*

[48] Edited by Don Pearson, *Image Processing*, McGraw-Hill, 1991.

[49] GEC-Plessey Semiconductors, "H.261 chipset advanced data sheet," September 1991.

[50] CCITT H.261, "Video Codec for audiovisual services at px64 kbit/s," *CCITT White book* (1990).

[51] ISO, "MPEG Video standard," *ISO/IEC-11172-2* (August 1991).

[52] Ya-Qin Zhang, William W. Wu, Kap S. Kim, Raymond L. Pickholtz & Jay Ramasastry, "Variable Bit-Rate Video Transmission in the Broadband Environment," *Proceedings of the IEEE* 79,2 (February 1991), 214–221.

[53] D G Morrison, "Variable Bit Rate Video Coding for Asynchronous Transfer Mode Networks," *British Telecom Technology Journal* 8,3 (July 1990), 70–80.

[54] M Ghanbari, "Two-Layer Coding of Video Signals for VBR Networks," *IEEE Journal on Selected Areas in Communications* 7,5 (June 1985), 771–781.

[55] Peter J. Burt & Edward H. Adelson, "The Laplacian Pyramid as a compact image code," *IEEE Transactions on Communications* COM-31,4 (April 1983), 532–540.

[56] Peter H. Westerink, Dick E. Boekee, Jan Biemond & John W. Woods, "Subband coding of images using vector quantisation," *IEEE Transactions on Communications* COM-36,6 (June 1988), 713–719.

[57] John C. Darragh & Richard L. Baker, "Fixed Distortion Subband Coding of Images for Packet-Switched Networks," *IEEE Journal on Selected Areas in Communications* 7,5 (June 1989), 789–800.

[58] Y. Linde, A. Buzo & R. M. Gray, "An algorithm for vector quantiser design," *IEEE Transactions on Communications* COM-28 (January 1980), 84–95.

[59] Nasser M. Nasrabadi & Robert A. King, "Image Coding using Vector Quantisation: A review," *IEEE Transactions on Communications* COM-36,8 (August 1988), 957–971.

[60] Morris Goldberg & Huifang Sun, "Image Sequence Coding using Vector Quantisation," *IEEE Transactions on Communications* COM-34,7 (July 1986), 703–710.

[61] John Bruder, "private communication," November 1991.

[62] Sun Microsystems, "Cell Image Compression Byte Stream Description," ftp://playground.sun.com:/pub/multimedia/video, March 1994.

[63] M Soryani & RJ Clarke, "Segmented Coding of Digital Image Sequences," *IEE Proceedings I Communications, Speech and Vision* 139,2 (April 1992).

[64] Q. Wang & RJ Clarke, "A New Motion Compensated Image Sequence Coding Scheme at 64 KBit/s," *IEE Proceedings I Communications, Speech and Vision* 139,2 (April 1992).

[65] Andrew Lippman, "Feature Sets for Interactive Images," *Communications of the ACM* 34 (April 1991), 93–102.

[66] Arch Luther, *Digital Video in the PC Environment*, McGraw-Hill, 1990.

[67] Kevin Harney, Mike Keith, Gary Lavelle, Lawrence D. Ryan & Daniel J. Stark, "The i750 Video Processor: A Total Multimedia Solution," *Communications of the ACM* 34 (April 1991), 65–78.

[68] Gunnar Karlsson & Martin Vetterli, "Packet Video and its Integration into the Network Architecture," *IEEE Journal on Selected Areas in Communications* 7,5 (June 1989), 739–751.

[69] Paulo Verissimo, "Real Time Communications," in *Distributed Systems (second edition)*, Sape Mullender, ed., ACM Press, 1992, 447–490.

[70] Van Jacobson & Steve McCanne, "Visual Audio Tool - vat," *vat(1) Unix Manual Page* (1992).

[71] R. Braden, D. Clark & S. Shenker, "Integrated Services in the Internet Architecture: an Overview," Network Information Centre, SRI International, RFC 1633, September 1994.

[72] Jeff Mackie-Mason & Hal Varian, "Pricing the Internet," *Proceedings of Public Access to the Internet*, Cambridge, Ma (1993).

[73] Jeff Mackie-Mason & Hal Varian, "Some Economics of the Internet," *Networks, Infrastructure and the new task of regulation*, Cambridge, Ma (1993).

[74] S. Shenker, "Making Greed Work in Networks: A Game-Theoretic Analysis of Switch Service Disciplines," *Proceedings ACM SIGCOMM*, London, UK (Aug. 31st - Sep. 2nd, 1994).

[75] Y. Rekhter & T. Li, "A Border Gateway Protocol 4 (BGP-4)," Network Information Centre, SRI International, RFC 1654, July 1994.

[76] M. Lepp & M. Steenstrup, "An Architecture for Inter-Domain Policy Routing," Network Information Centre, SRI International, RFC 1478, July 1993.

[77] Z. Wang & J. Crowcroft, "Quality of Service Routing: Complexity and Algorithms," *Submitted for publication* (April 1994).

[78] Craig Partridge, "A Proposed Flow Specification," Network Information Centre, SRI International, RFC 1363, September 1992.

[79] Frank Kelly, "Effective Bandwidth at multi-type queues," *Queueing Systems* 9 (1991), 5–15.

[80] Roch Guerin, Hamid Ahmadi & Mahmoud Naghshineh, "Equivalent Capacity and its application to bandwidth allocation in high-speed networks," *IEEE Journal on selected areas in communications* 9,7 (September 1991).

[81] Roch Guerin & Levent Gun, "A unified approach to bandwidth allocation and access control in fast packet-switched networks," *Proceedings IEEE INFOCOM*, Florence, Italy (April 1992).

[82] Viyadhar Kulkarni, Levent Gun & Phil Chimento, "Effective Bandwidth Vector for Two Priority ATM Traffic," *Proceedings IEEE INFOCOM*, Toronto, Canada.

[83] A. Parekh, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks," MIT Laboratory for Information and Decision Systems, PhD Thesis Report No. LIDS-TH-2089..

[84] Jay Hyman, Aurel Lazar & Giovanni Pacifici, "A Separation Principle Between Scheduling and Admission Control for Broadband Switching," *IEEE Journal Selected Areas in Communications* 11,4 (May 1993), 605–616.

[85] R. Braden, "Requirements for Internet Hosts – Communication Layers," Network Information Centre, SRI International, RFC 1192, October 1989.

[86] M Gerla & L Kleinrock, "Flow Control: A Comparative Survey," *IEEE Trans on Comms, Vol. COM-28, No. 4, pp. 553-574* (April 1980).

[87] DD Clark, "NETBLT: A Bulk Data Transfer Protocol," Network Information Centre, SRI International, RFC 969, December 1985.

[88] DR Cheriton, "VMTP: A Transport Protocol for the Next Generation of Communication Systems," *Proceedings ACM SIGCOMM*, Vermont (August 1986).

[89] Greg Chesson et al, *XTP Specification rev. 3.5*, Protocol Engines inc., September 1990.

[90] K. K. Ramakrishnan & Raj Jain, "A Binary Feedback Scheme for Congestion Avoidance in Computer Networks with a Connectionless Network Layer," *Proceedings ACM SIGCOMM*, San Francisco (August 1988).

[91] Scott Shenker, "A Theoretical Analysis of Feedback Flow Control," *Proceedings ACM SIGCOMM*, Philadelphia (1990).

[92] Amarnath Mukherjee & John C. Strikwerda, "Analysis of Dynamic Congestion Control Protocols - A Fokker Planck Approximation," *Proceedings ACM SIGCOMM*, Zurich, Switzerland (September 1991).

[93] Mitra D. & Seery J. B., "Dynamic Adaptive windows for High speed data networks: theory and simulation," *Proceedings ACM SIGCOMM*, Philadelphia (1990).

[94] Mitra D. & Seery J. B., "Asymmetrically Optimal Design of Congestion Control for High Speed Data Networks," *IEEE Transactions on Communications* COM-40 (1992), 301–311.

[95] Kerry W. Fendick, Manoel A. Rodrigues & Alan Weiss, "Analysis of a Rate Based Control Strategy with Delayed Feedback," *Proceedings ACM SIGCOMM*, Baltimore USA (August 1992).

[96] Raj Jain, "A Delay-Based Approach for Congestion Avoidance in Interconnected Heterogenous Computer Networks," *ACM Computer Communications Review* 19-5 (October 1989), 56–71.

[97] Ortwin Rose, "The CWR Scheme - Congestion Avoidance using Combined Window-Rate Adaptation," *INET92*, Kobe Japan (June 1992).

[98] L. S. Brakmo, S. W. O'Malley & L. L. Peterson, "TCP Vegas: New Techniques for Congestion Detection and Avoidance," *Proceedings ACM SIGCOMM*, London, UK (Aug. 31st - Sep. 2nd, 1994).

[99] Hemant Kanakia, P. Mishra & A. Reibman, "An adaptive congestion control scheme for real-time packet video transport," *Proceedings ACM SIGCOMM* (September 1993).

[100] Srinivasan Keshav, "A Control-Theoretic Approach to Flow Control," *Proceedings ACM SIGCOMM*, Zurich (September 1992).

[101] Lotfi Benmohamed & Semyon Meerkov, "Feedback Control of Cognestion in Packet Switching Networks: The case of a single congested node," *IEEE/ACM Transactions on Networking* 1,6 (December 1993), 693–708 .

[102] Kihong Park, "Warp Control: A dynamically Stable Congestion Control Protocol and its analysis," *Proceedings ACM SIGCOMM*, San Francisco USA (August 1993).

[103] Carey Williamson, "Loss-Load Curves: Support for rate based congestion control in high speed data networks," *Proceedings ACM SIGCOMM*, Zurich (September 1992).

[104] Carey Williamson, "Optimising File Transfer Response Time using the loss-load curve congestion control Mechanism," *Proceedings ACM SIGCOMM*, San Francisco Ca (August 1993).

[105] Raj Jain, "Congestion Control and Traffic Management in ATM Networks: Recent Advances and A Survey," *Submitted to Computer Networks and ISDN Systems* (January 1995).

[106] Partho P Mishra & Hemant Kanakia, "A Hop By Hop Rate-Based Congestion Control Scheme," *Proceedings ACM SIGCOMM*, Baltimore USA (August 1992).

[107] David Cheriton, "Sirpent: A High Performance internetworking approach," *Proceedings ACM SIGCOMM* (September 1992).

[108] M. Schroeder et al, "Autonet: A High Speed self-configuring local Area Network using point to point links," *Journal on Selected Areas in Communication* 9,8 (October 1991), 1318–1335.

[109] S. J. Golestani, "A Stop-and-Go queueing framework for congestion management," *Proceedings ACM SIGCOMM*, Philadelphia (September 1990).

[110] Zhang, L, "Virtual Clock: A New Traffic Control Algorithm for PSNs," *Proceedings ACM SIGCOMM*, Philadelphia (1990).

[111] A Demers, S Keshav & S Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," *Internetworking: Research and Experience* 1 (April 1990), 3–26.

[112] David D. Clark, Scott Schenker & Lixia Zhang, "Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism," *Proceedings ACM SIGCOMM*, Baltimore (August 1992).

[113] D. Ferrari & D. Verma, "A Scheme for real time channel establishment in Wide-Area Networks," *IEEE Journal on Selected Areas in Communications* 8,3 (April 1990), 368–379.

[114] Charles Kalmanek, Hemant Kanakia & Srinivasan Keshav, "Rate Controlled Servers for very High Speed Networks," *Proceedings of IEEE Global Telecommunications Conference*, San Diego, Ca (December 1990).

[115] Sally Floyd & Van Jacobson, "Random Early Drop Gateways," *IEEE/ACM Transactions on Networking* 1,1 (August 1993), 397,413.

[116] I. Leslie & D. McAuley, "Fairisle: An ATM Network for the Local Area," *Proceedings ACM SIGCOMM*, Zurich, Switzerland (September 1991).

[117] R. J. Black, I. Leslie & D. McAuley, "Experiences of Building an ATM Switch for the Local Area," *Proceedings ACM SIGCOMM*, London, UK (Aug. 31st - Sep. 2nd, 1994).

[118] A. Romanow & S. Floyd, "The Dynamics of TCP Traffic over ATM Networks," *Proceedings ACM SIGCOMM*, London, UK (Aug. 31st - Sep. 2nd, 1994).

[119] Lixia Zhang, Scott Schenker & David D. Clark, "Observations on the Dynamics of a Congestion control algorithm: the effects of two way traffic," *Proceedings ACM SIGCOMM*, Zurich (September 1991).

[120] Srinivasan Keshav, "Congestion Control in Computer Networks," UC Berkeley, PhD Thesis, Report UCB/CSD 91/649, September 1991.

[121] Walid Dabbous, "Analysis of a delay based congestion avoidance algorithm," *Proceedings of the IFIP TC6/WG6.4 Conference on High Performance Networking*, Liege Belgium (December 1992).

[122] J-C Bolot & A. Udaya Shankar, "A Discrete-Time Stochastic Approach to Flow Control Dynamics," *Proc. IEEE Globecom 92* Session on Dynamic Phenomena in Communication Networks (December 1992).

[123] Sally Floyd & Van Jacobson, "Traffic Phase Effects in Packet Switched Gateways," *Internetworking - Research and Experience* 3,3 (September 1992).

[124] Z Wang & J Crowcroft, "A New Approach to Congestion Control in Datagram Networks," *Computer Communications Review* (Jan 91).

[125] Sally Floyd & Van Jacobson, "Class based queueing for policy based resource sharing," 1993, Internal presentation and private email.

[126] Ian Wakeman, Jean Bolot, Thierry Turletti & Jon Crowcroft, "Multicast Congestion Control in the distribution of Variable Bit Rate Video in the Internet," *Proceedings ACM SIGCOMM*, London (August 1994).

[127] S. Deering, "Host extensions for IP multicasting," Network Information Centre, SRI International, RFC 1112, January 1989.

[128] U. Bilting, A. Sasse, C-D. Schulz & T. Turletti, "Remote seminars through multimedia conferencing: Experiences from the MICE project," *Proc. INET'94*, Prague (June 1994).

[129] Henning Schulzrinne, "NEVOT," *nevot(1) Unix manual pages*.

[130] T. Turletti, "H.261 software codec for videoconferencing over the Internet," INRIA Research Report 1834, June 1993.

[131] Steve Casner & Steve Deering, "First IETF Internet Audiocast," *Computer Communications Review* Vol 22,3 (July 1992).

[132] J. Kurose, "Open issues and challenges in providing QoS guarantees in high speed networks," *Proceedings ACM SIGCOMM* 23,1 (Jan. 1993), 6–15.

[133] J Crowcroft & K Paliwoda, ""A Multicast Transport Protocol",," *Proceedings ACM SIGCOMM* (1988).

[134] R. Yavatkar & L. Manoj, "Optimistic strategies for large-scale dissemination of multimedia information," *Proc. ACM Multimedia '93* (Aug. 1993).

[135] Waitzman, D., Partridge, C. & Deering, S.E., "Distance Vector Multicast Routing Protocol," Network Information Centre, SRI International, RFC 1075 , Nov 1988.

[136] M. Handley, P. Kirstein & A. Sasse, "Multimedia Integrated Conferencing for European Researchers (MICE): Piloting Activities and the Conference Management and Multiplexing Centre," *Computer Networks and ISDN Systems* 26 (November 1993), 275–290.

[137] R. Aravind et al., "Image and video coding standards," *AT&T Tech. Journal* (Jan/Feb. 1993).

[138] D. Sanghi, A. Agrawala & B. N. Jain, "Experimental assessment of end-to-end behavior on Internet," *Proceedings IEEE INFOCOM*, San Francisco, Ca (March 1993).

[139] D-M. Chiu & R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks and ISDN Systems* 17,1 (1989), 1–14.

[140] Sally Floyd & Van Jacobson, "The Synchronization of Periodic Routing Messages," *IEEE/ACM Transactions on Networking* 2,2 (April 1994), 122,137.

[141] C-T. Chen & A. Wong, "A self-governing rate buffer control strategy for pseudoconstant bit rate video coding," *IEEE Trans. Image Processing* 2,1 (Jan. 1993), 50–59.

[142] C. Elliott, "High quality multimedia conferencing through a long-haul packet network," *Proc. ACM Multimedia '93*, Los Angeles, CA (Aug 1993).

[143] N. Yin & M. Hluchyj, "A dynamic rate control mechanism for source coded traffic in a fast packet network," *IEEE Journal Selected Areas in Communications* 9,7 (Sept. 1991), 1003–1012.

[144] K. Jeffay, D. L. Stone, T. Talley & F. D. Smith, "Adaptive: best-effort delivery of digital audio and video across packet-switched networks," *Proc. 3rd Intl. Wkshp on Network and OS Support for Digital Audio and Video*, San Diego, CA (Nov. 1992).

[145] L. A. Rowe & B. C. Smith, "A continuous media player," *Proc. 3rd Intl. Wkshp on Network and OS Support for Digital Audio and Video* (Nov. 1992).

[146] S. Ramanathan & P. V. Rangan, "Adaptive feedback techniques for synchronized media retrieval over integrated networks," *IEEE/ACM Trans. Networking* 1,1 (Jan 1993).

[147] S. Shenker, "Resource reservation in the Internet," *Presentation at the November 1993 IETF meeting*, Houston, TX..

[148] Van Jacobson, "Learning from the present - things that IP got right and ATM got wrong," *Proceedings of the USENIX High-speed Networking Symposium*, Berkeley, Ca (1-3 August 1994).

[149] S. Bradner & A. Mankin, "The Recommendation for the IP Next Generation Protocol," Network Information Centre, SRI International, RFC 1752, 18 January 1995.

[150] Sally Floyd, "Link-sharing and Resource Management Models for Packet Networks," *Submitted to ACM/IEEE Transactions on Networking*.

[151] ISO, "MPEG 2 Video draft standard," *ISO/IEC-13818-1* (December 1993).

[152] P. T. Brady, "Effects of Transmission Delay on Conversational Behavior on Echo-Free Telephone Circuits," *Bell System Technical Journal* 50 (Jan 1971), 115–134.

[153] Van Jacobson, "Multimedia Conferencing on the Internet," *ACM SIGCOMM Tutorial*, London (September 1994).

[154] Donald R. Morrison, "PATRICIA - Practical Algorithm to Retrieve Information Coded In Alpha-numeric," *Journal of the ACM* 15,4 (October 1968).

[155] Robert Sedgewick, *Algorithms*, Addison-Wesley, 1988.

[156] Joel Halpern, "Modifications of Patricia Trees for Handling values with discontiguous masks with emphasis on internet routing applications," *Private Communication*.

[157] Paul Tsuchiya, "Cecilia: A Search Algorithm for Table Entries with Non-contiguous Wildcarding," *ftp://thumper.bellcore.com/pub/tsuchiya /cecilia.tar.Z* (January 1992).

[158] Steven McCanne & Van Jacobson, "The BSD Packet Filter: A New Architecture for User-Level Packet Capture," *Proceedings of Winter Usenix Conference*, San Diego Ca. (January 1993).

[159] Van Jacobson & Steve McCanne, "TCPDUMP," *ftp://ftp.ee.lbl.gov/tcpdump-*.tar.Z* (1990).

[160] L. Zhang, S. Deering, D. Estrin, S. Shenker & D. Zappala, "RSVP: A New Resource ReSerVation Protocol," *IEEE Network* (September 1993).