Scaleable Audio for Collaborative Environments

By Milena Radenkovic, Dipl. Ing. (Hons), BA (Hons)

Thesis submitted to the University of Nottingham for the degree of Doctor of Philosophy, February 2002

To my parents

Acknowledgements

I would like to offer my deepest gratitude to Prof. Steve Benford and Dr. Chris Greenhalgh for all their help, support and kindness during the course of my PhD. Our collaboration and discussions were very important for me and for my work, and most enjoyable. I would especially like to thank Dr. Chris Greenhalgh for his invaluable input and feedback, which challenged me and encouraged me from the early stages of my PhD.

Many thanks go to my colleagues in Communications Research Group for their friendship over the years.

Contents

List of Figures

List of Tables

Abstract

This thesis is concerned with supporting natural audio communication in collaborative environments across the Internet. Recent experience with Collaborative Virtual Environments, for example, to support large on-line communities and highly interactive social events, suggest that in the future there will be applications in which many users speak at the same time. Such applications will generate large and dynamically changing volumes of audio traffic that can cause congestion and hence packet loss in the network and so seriously impair audio quality. This thesis reveals that no current approach to audio distribution can combine support for large number of simultaneous speakers with TCP-fair responsiveness to congestion.

A model for audio distribution called Distributed Partial Mixing (DPM) is proposed that dynamically adapts both to varying numbers of active audio streams in collaborative environments and to congestion in the network. Each DPM component adaptively mixes subsets of its input audio streams into one or more mixed streams, which it then forwards to the other components along with any unmixed streams. DPM minimises the amount of mixing performed so that end users receive as many separate audio streams as possible within prevailing network resource constraints. This is important in order to allow maximum flexibility of audio presentation (especially spatialisation) to the end user. A distributed partial mixing prototype is realised as part of the audio service in MASSIVE-3. A series of experiments over a single network link demonstrate that DPM gracefully manages the tradeoff between preserving stable audio quality and being responsive to congestion and achieving fairness towards competing TCP traffic.

The problem of large scale deployment of DPM over heterogeneous networks is also addressed. The thesis proposes that a shared tree of DPM servers and clients, where the nodes of the tree can perform distributed partial mixing, is an effective basis for wide area deployment. Two models for realising this in two contrasting situations are then explored in more detail: a static, centralised, subscription-based DPM service suitable for fully managed networks, and a fully distributed self-organising DPM service suitable for unmanaged networks (such as the current Internet).

1. Introduction

There are many situations in everyday life in which large groups of people “speak” at the same time. Audiences and crowds of spectators at performances and sports events provide the most extreme examples, with thousands of people engaging in activities such as cheering, chanting and singing together. However, any relaxed social setting that involves sizeable groups of people is also likely to involve significant numbers of simultaneous speakers; for example, interruptions, non-verbal speech cues [Goodwin, C., 1984], shouting out answers to questions, dynamically forming conversing sub-groups within a large space [Kendon, A., 1990].

Over the past several years the Internet has been experiencing the emergence of large-scale collaborative networked applications that aim to support such real world scenarios and the associated patterns of audio activity. For example, collaborative virtual environment technology has been used for applications that support large on-line communities and highly interactive social events such as multi-player games and inhabited television [Greenhalgh, C., Benford, S., Taylor, I., et al, 1999]. Experiments have shown that these applications involve potentially significant numbers of simultaneous speakers [Greenhalgh, C., Benford, S., Craven, M., 1999]. The deployment of such applications is expected to increase and pattern of simultaneous speaking to continue.

The current Internet is not well suited to support such applications. Real-time audio communication is bandwidth intensive, and has strict requirements for minimum throughput, network delay and jitter. Today’s Internet operates on a best-effort basis and cannot guarantee an upper bound on end-to-end delay nor lower bound on available bandwidth. These applications can easily cause network congestion, and hence packet loss and increased delays that can significantly impair audio quality. As such applications become more widespread, large number of audio streams may form a considerable portion of the Internet load. Therefore, the overall behaviour of the applications that include large number of streams will have a significant impact on the Internet traffic and the quality of delivered audio.

This thesis aims to design, build and validate an audio service that efficiently supports applications that allow simultaneous speaking as a natural pattern of audio activity among participants, and can adapt to network conditions as well as smooth out the effects of the network on the end user audio quality. Such an audio service is essential for both the health of the Internet and the quality of delivered audio.

This chapter is organised as follows. Section 1.1 introduces some aspects of collaborative audio and the underlying networks used for communication that are of direct relevance to the work presented in this thesis. It gives an overview of traditional and newly emerging applications of collaborative audio, their underlying requirements and constraints and provides the motivation for this thesis. Section 1.2 describes focus of the thesis. Section 1.3 gives an overview of the thesis and identifies its main contributions.

1.1 Collaborative Audio

Audio is arguably the most important component of a real-time collaborative application and perhaps the most complex one. It often carries the core content of our interaction and serves as the baseline for successful collaboration. For example, recent studies have noted how participants fall back on audio to resolve difficulties with other aspects of collaborative applications such as negotiating a shared perspective [Hindmarsh, J., Fraser, M., Heath, C., et al, 1998].

However, the design of audio for collaborative interfaces is complicated by: constraints imposed by the environments in which the audio service must operate, potentially large numbers of simultaneous participants, as well as a wide range of interpersonal communication models.

The underlying network requirements for any collaborative audio are very strict (stricter than for non-interactive audio). Collaborative audio is sensitive to network packet loss, latency and jitter, and to gaps in the audio caused by the lack of the real-time support on general purpose hosts. These factors affect intelligibility of audio and speech.

Digitised audio is transmitted over a network in a series of packets. In order for the receiver(s) to reproduce the original audio signal, they need to receive all the packets with preserved timing relationships among them. However the Internet does not guarantee when or whether a packet will be delivered to the receivers. Buffering of the packets in the network nodes introduces variable delays and can distort the original timing relationship among the packets. In addition to the delay, jitter that results from the variable buffering times when the buffer limit is exceeded, can further impair the audio quality. This is the primary cause of packet loss in the Internet [Kouvelas, I., 1998].

Different models of communication among participants will have different bandwidth requirements and architectural needs. After a brief overview of the traditional applications and their needs, a newly emerging style of applications with different needs is introduced.

The most common scenarios for the traditional use of networked audio include public presentations, on-line lectures and small-scale interactive conferencing.

In public demonstration applications only a small number of participants produce audio while the majority of the attendees can only receive. The active participants are statically determined and do not change over the lifetime of the application. These applications are therefore not interactive and in general have less strict requirements on the audio latency but stricter requirements for audio quality.

In an on-line lecture set-up, the subset of participants allowed to send audio data can change during the evolution of the application. Such changes are, however, conditioned to the occurrence of conditions or on rules depending on the application itself e.g. the lecturer can occasionally allow the students to ask questions. In this scheme the active participants are not predetermined but there is a control over who will speak at any time. Such scenarios are also not very interactive and have relatively similar requirements to completely static ones i.e. low packet loss but higher delays are tolerable.

In interactive small scale conferencing (such as some telephony and more traditional CSCW applications) more than one (potentially all) participant(s) can transmit audio at the same time. They do indeed support several simultaneous speakers as there is no control over who will speak at any time. But they focus mostly on small groups and a slow pace of interaction.

The distribution of audio streams in the network for such applications is either one-to-many or few-to-many (M-to-N where usually M<<N). Various techniques have been proposed for reducing the bandwidth required for such broadcasts, most notably network multicasting [Macedonia, M., Brutzman, D., 1994] (see the next chapter).

Due to the nature of these applications, some protocol developers have explicitly disregarded the possibility of multiple simultaneous speakers and have focused on supporting static, controlled, small scale and slow pace dynamic models of interaction. The Internet Standard Real Time Protocol (RTP) is the standard packet format used for continuous media traffic – such as audio – on the Internet [Schulzrinne, H., Casuer, S., Frederich, R., et al, 1996]. RTP includes sophisticated algorithms to control the amount of management traffic placed on the network, but assumes that audio traffic will not be a problem:

> "For example, in an audio conference the data [audio] traffic is inherently self-limiting because only one or two people will speak at a time…" [[Schulzrinne, H., Casuer, S., Frederich, R., et al, 1996], section 6.1]

Similarly, the multicast backbone network (MBone) [Macedonia, M., Brutzman, D., 1994] provides the Internet's wide-area multicast capabilities and supports wide-area multimedia sessions. The MBone's guidelines for use, and the available resources (at least historically) assume that each audio session will not have more than one active speaker at a time (indeed, many of the larger MBone sessions have been effectively broadcasts rather than conferences, e.g. feeds from the NASA shuttle launches).

However, newly emerging large scale networked applications such as collaborative virtual environments (CVEs) can support large on-line communities and highly interactive social events such as multi-player games and inhabited television [Greenhalgh, C., Benford, S., Taylor, I., et al, 1999]. These applications have a fast pace of interaction, and have M-to-N distribution of audio streams where M is almost equal to N. The work of Communications Research Group (CRG) from Nottingham University on inhabited television has focused on enabling public participation in on-line TV shows within shared virtual worlds [Greenhalgh, C., Benford, S., Taylor, I., et al, 1999]. As part of a public experiment in inhabited TV called Out of This World (OOTW), patterns of user activity, including audio activity, were studied by statistically analysing system logs [Greenhalgh, C., Benford, S., Craven, M., 1999]. The results showed that overlapping audio transmissions from several participants were common for this event. Indeed, during a 45 minute show, there were several minutes when all 10 of the participants were generating audio traffic at the same time. Periods of high activity included teams shouting instructions to one another during action games and also shouting out answers to questions (OOTW was a gameshow).

Similar analyses of patterns of audio activity in other CVE applications and platforms, for example virtual teleconferencing in the DIVE system [Frécon, E., Greenhalgh, C., Stenius, M., 1999], have also revealed significant periods when several participants are simultaneously generating audio traffic. Indeed, previous experiences suggest that, even for relatively focused applications such as teleconferencing, audio activity is best approximated by a model of people transmitting audio at random, rather than deliberately avoiding overlapping speech. For more socially dynamic kinds of events such as OOTW, audio activity appears to be even more strongly positively correlated among participants.

Not all of this activity is necessarily verbal; it may also include non-verbal utterances and background noise. However, as noted above, non-verbal utterances form a significant part of human communication. In addition, studies of social interaction in CVEs have noted how communication in a virtual world can be influenced by events in participants' local physical environments [Bowers, J., Pycock, J., O'Brien, J., 1996]. Obtaining awareness of these events through overheard background audio might help participants account for actions in the virtual world. Media spaces are one class of collaborative application that employ open audio connections to provide this kind of awareness. For example, when discussing patterns of usage for the Thunderwire audio media-space, Hindus *et al*. note that it might be better to consider the number of live microphones, rather than the number of active participants [Hindus, D., Ackerman, M., Mainwaring, S., et al, 1996].

Therefore, overlapping audio, including talk and other noise, is likely to be the norm for some CSCW applications. This thesis anticipates that as audio-enabled applications become more widespread, and are used to support larger virtual communities, so designers of audio services will see increasing numbers of simultaneous speakers.

The presence of many simultaneous speakers in these applications has very profound implications for the bandwidth requirements of an audio service. Each speaker independently introduces a new audio stream that has to be accommodated by the network and that also has to be processed by each recipient's local computer. This co-ordinated vocal activity results in significant bursts (peaks) in the audio traffic that could bring instability to the whole application if the audio service was not built to support it.

If the bandwidth requirements of such bursts are lower than the available bandwidth, then all of the packets will be delivered to the end user successfully. However, during loaded times when there is insufficient bandwidth for these peaks, the audio will suffer severe packet loss and delays and may be unusable to the end users. From the users' point of view, the audio services should "smooth out" these effects of the network and deliver usable audio.

From the network point of view, (assuming the network does not micromanage the utilisation of its own resources, e.g. Internet) large scale deployment of such applications can result in severe unfairness against TCP-based traffic (and other responsive traffic) and possibly cause congestion collapse [Mahdavi, J., Floyd, S., 1997], [Braden, B., Clark, D., Crowcroft, J., et al, 1998]. Unfairness refers to bandwidth starvation that unresponsive flows can inflict on well-behaved responsive flows such as TCP flows. TCP is the dominant transport protocol in the Internet and the current stability of the Internet is mainly due to TCP's congestion control which fairly shares link bandwidth between multiple connections and maintains load at the network on useful levels [Jacobson, V., 1988], [Stevens, W., 1997]. If the multiple real-time streams introduced to the network do not obey the laws of existing traffic on the Internet, and do not respond to signals of congestion as TCP does, then that can lead to network collapse [Mahdavi, J., Floyd, S., 1997], [Braden, B., Clark, D., Crowcroft, J., et al, 1998].

This means that if we want to assure quality of experience for users of very dynamic, large-scale, interactive applications, as well as other network users, the design of the audio service must include some sort of congestion control mechanism that enables them to match their transmission rates to the current available bandwidth and be TCP fair even during peak times.

1.2 The Specific Focus of This Thesis

The modeling approach in this thesis has arisen largely because of the work done in virtual collaborative environments (Inhabited TV) but it also offers more a generic framework for scaling audio use over heterogeneous networks.

The precise focus of this thesis is the following: *How can we establish a scaleable audio service for network applications such that:*

- *This audio service can support situations in which large numbers of mutually aware users (potentially tens or even hundreds) are speaking simultaneously,*
- *This audio service exhibits "network-friendly", and in particular "TCP-friendly" behaviour where available resources, especially bandwidth and local processing, are limited and dynamically changing.*

A novel technique called Distributed Partial Mixing (DPM) is proposed as the basis for this service. DPM can dynamically adapt both to varying numbers of active audio streams in the collaborative network application and to congestion in the network. Each distributed partial mixing component adaptively mixes subsets of its input audio streams into one or more mixed streams, which it can forward to the other components along with any unmixed streams. DPM minimises the amount of mixing performed so that end user recipients receive as many separate audio streams as possible within prevailing network resource constraints. Delivering large numbers of independent streams to the end user is important in order to allow maximum flexibility of audio control to the end users. Results of a series of experiments over a single network link demonstrate the effectiveness of congestion control performed by distributed partial mixing. Distributed partial mixing manages the tradeoff between preserving stable audio quality, being responsive to congestion and achieving fairness towards TCP traffic.

More generally, distributed partial mixing supports dynamic load balancing between different distributed components of the audio service; as it avoids concentrating audio traffic an any single point in the network (which a single server approach would). It can also support heterogeneity of networks and user machines in a graceful manner.

The topology of DPM components and clients can be static or dynamic. A shared tree of DPM servers and end users, where the nodes of the tree can perform distributed partial mixing, is proposed as an effective basis for wide area deployment. Two models for realising this in two contrasting situations are then explored in more detail: a static, centralised, subscription-based DPM service suitable for fully managed networks, and a fully distributed self-organising DPM service suitable for unmanaged networks (such as the current Internet).

1.3 Thesis Overview and Contributions

In the process of designing and evaluating a scaleable audio service to support large numbers of mutually aware speakers over heterogeneous wide area networks the following has been done:

Chapter 2 identifies a series of criteria important for any audio service that aims to support large scale collaborative applications that potentially involve large number of simultaneous speakers. The chapter then reviews related work in CSCW and networked multimedia in terms of these criteria.

Chapter 3 presents a new model, called Distributed Partial Mixing (DPM), for designing scaleable and adaptable audio service for large scale collaborative applications in the Internet. Distributed partial mixing performs flexible management of audio streams at multiple points in the network in order to decrease network requirements as well as provide different levels of audio fidelity to the end users. The chapter presents a functional model of distributed partial mixing and describes the individual functional stages. The purpose of this model is to clarify key design issues, possibilities and tradeoffs in the complex design space of distributed partial mixing. The choice of streams to be mixed can affect the effectiveness of distributed partial mixing. The chapter gives examples of mixing criteria that might be used to dynamically determine how many and which streams should be mixed. Various factors that might affect mixing decisions in distributed partial mixing are discussed. Examples of how each factor could be applied to a certain application and network situation are provided.

Chapter 4 presents a distributed partial mixing prototype implemented in the MASSIVE-3 CVE platform [Geenhalgh, C., Purbrick, J., Snowdon, D., 2000] and an evaluation of its effectiveness in terms of the criteria from Chapter 2. The chapter begins with a brief description of the scenario and infrastructure for building a small-scale demonstration system, which incorporates distributed partial mixing. The chapter then describes the design choices for the various functional components of distributed partial mixing (only the network driven mixing criteria) and then discusses particular implementation details of the proposed example system built into the MASSIVE-3 platform. The chapter then describes experiments that show the steady state behaviour of the built system against both non-adaptive traffic and adaptive traffic. Detailed discussion of the results is also included.

Chapter 5 discusses the constraints and feasibility of large scale deployment of Distributed Partial Mixing (DPM) in wide area networks and over heterogeneous deployment domains. The chapter is also concerned with the dynamics of the topology of DPM servers and clients. The chapter proposes a basic general shared tree solution for the topology of DPM servers and clients. Two contrasting proposals, for fully managed and fully unmanaged networks, are also described in detail. The chapter includes a detailed discussion of the proposed approaches as well as alternative approaches.

Chapter 6 concludes the work presented in this thesis, presents main contributions of the thesis and discusses possible future work.

2. Related Work

The scale of the network and variations in available bandwidth complicate the design of network adaptive audio services that support many simultaneous speakers. Even though there is no work that has explicitly addressed this problem, there are a wide range of relevant existing techniques that may be useful to building such a service.

This chapter is organised as follows. We first introduce, define and motivate the set of criteria used for reviewing existing work. In the second part, we move to detailed description and discussion of the most relevant approaches according to this set of criteria. The third part discusses the methods which could be integrated in the design of a scaleable audio service and also identify the outstanding challenges.

2.1 Criteria

2.1.1 Introduction

The following set of criteria is chosen to be applied when reviewing related existing techniques to building scaleable audio service (and to evaluate the work presented in this thesis):

1. Support for the most natural audio communication between users
2. Support for maintaining acceptable audio quality (especially minimising packet loss but also bounding delays, even with network and host heterogeneity and congestion)
3. Support for tailored audio presentations for each user (i.e. individual tailoring of mixing and independent audio streams at the end systems)
4. Support for heterogeneous unmanaged large scale networks, end systems and wide area user distribution (e.g. Internet).

5. Support for efficient distribution of audio streams in the network
6. Support for adaptability and responsiveness of the audio traffic in the network including inter-protocol fairness

The first three criteria address user issues whereas the last three criteria are network oriented.

The following sections define and describe each criterion in turn.

2.1.2 Support for the most natural audio communication between users

Supporting the most natural audio communication between users refers to not limiting who can speak, or what can be said and heard, at all times. More specifically, this refers to:

- not limiting the number of simultaneous speakers,
- allowing spontaneity in the way people react in the audio medium,
- including non-verbal sounds as well as verbal sounds,

e.g. open microphones transmit all the sounds produced by users at all times.

This is extremely important for collaborative applications for which the audio component is undoubtedly the most significant media stream because it not only carries the critical content for group discussions but also important intonation cues with regard to people's reactions as well [Schooler, E., 1996].

Work cited in chapter 1 shows that the number of simultaneous speakers in some applications can be very high. This chapter argues that it is important to provide a technical infrastructure that can support such situations in order to ensure the stability of these applications and the network, as well as quality of user experience.

2.1.3 Support for maintaining acceptable audio quality

There are different approaches to dealing with maintaining acceptable audio quality for different user and application requirements. Many researchers suggest adding a new service model to accommodate the QoS requirements of real-time multimedia applications. This approach is usually referred to as *integrated services* because it incorporates QoS requirements for streaming applications and provides performance guarantees for this class of applications. However, this thesis focuses on limiting packet loss and minimizing delays over heterogeneous, unmanaged, best-effort networks such as Internet. In such networks there can be no guarantees about the quality of service when physical limitations are present, and packet loss can be limited only through adaptation. Adaptation can be useful even for integrated networks in case of interactive applications. Having constant guarantees for interactive applications is usually not cost effective because it is often not possible to predict the maximum bandwidth and QoS requirements to be reserved for them.

This criterion is important because the level of packet loss experienced is the primary determinant of whether a network audio stream will be intelligible to the user and therefore of any use at all (while other factors such as delay or jitter play a secondary role and should also be minimised). Experiments by Hardman *et al.* and others have explored the relationship between packet loss and intelligibility (e.g. [Hardman, V., Sasse, A., Handley, M., et al, 1995]). They argue that this is a complex relationship, and that it depends upon the proportion of packets lost, the size of the packets, and the repair strategy that is used to compensate for missing audio information. Moreover ergonomic studies and experience gained from the Internet demonstrate that people can use audio data as long as the information content is above some minimum level [Wilson, F., Wakeman, I., Smith, W., 1993]. This level depends on media content and the task in hand. For instance, a foreign language is more difficult to understand than a native language. However, some simple guidelines form a useful starting point e.g. some typical values found in literature [Hardman, V., Sasse, A., Handley, M., et al, 1995] suggest that audio will become unintelligible if packet loss is higher than 15%.

2.1.4 Support for tailored audio presentations for each user

Supporting tailored audio presentations for each user refers to supporting individual tailoring of mixing and independent audio streams at the end systems according to user's requirements (and network constraints). More specifically, this refers to allowing support for individual spatialiastion and other forms of receiver's control over independent audio streams.

Previous research has shown that providing spatialised audio is a significant factor in engendering a sense of presence in a virtual environment. For example, Hendrix and Barfield conducted experiments to compare the presence or absence of spatialised sound alongside the stereoscopic display of a virtual environment [Hendrix, C., Barfield, W., 1996]. They found that the addition of spatialised sound did significantly increase the sense of presence in the virtual environment.

Moreover, experiments described in [Watson, A., Sasse, A., 2000] demonstrated that the perceived quality of network audio is not affected only by the level of packet loss observed in the large-scale field trials (provided that a packet loss repair method such as packet repetition is in use). Volume discrepancies (e.g. poor quality microphones) and echo have a greater impact on the user's experience meaning that it is possible to have perfect transmission from a network viewpoint, but still have poor quality audio from a user's viewpoint. Support for individual tailoring of audio at the end systems could allow users to improve the audio streams that are of more importance to them (e.g. by increasing the volume of only these streams) and to remove those which are disturbing or not important (e.g. by decreasing their volume).

2.1.5 Support for heterogeneous networks and end systems

This criterion refers mainly to the character of the situation where the service is deployed.

Designing large scale services is fundamentally challenged by the heterogeneity that is inherent in the disparate technologies that comprise the Internet both within the end systems and across the network infrastructure. More specifically, the following two types of diversities have to be taken into account:

- End-systems diversity: With more home-users becoming connected to new networked services, the disparity in end-user platforms has increased dramatically. End-systems could differ significantly in their CPUs, I/O devices, storage capabilities, codec support (dedicated board/software), communication protocol support and network interfaces. These issues place limits on the end-systems capabilities to process, consume and generate multimedia data. For example, limited resources in end-systems can result in buffer overflow, lateness in processing data and inability to process data. This manifests itself to the user as (one or more of) unacceptable playout delays, lost audio segments and also poor user interaction etc.
- Network diversity: End-systems are likely to be connected to different networks with different bandwidth capabilities but also varying access delay characteristics. For example: round trip times (propagation delays), packet loss rates, congestion levels, service types, as well as minimum and maximum transmission unit (MTU) can be very different for different networks. In general we can segregate the types of networks into high-speed (FDDI. DQDB, ATM, Fast Ethernet), medium-speed (Ethernet, Token Ring, Local Area Radio) and low-speed mobile networks (Modem link, Local Area Infra Red, Wide Area Radio data rates).

In order to be widely usable and scalable, these services should be bandwidth efficient irrespective of the underlying network, and similarly should not rely on platform-specific features (e.g. different software and hardware audio codecs in case of audio) or network-specific characteristics (e.g.. particular bandwidths or delays).

Another fundamental aspect of managing heterogeneity over unmanaged, large scale networks is having distributed rather than centralized solutions. Although it is often simpler to take a centralized approach, in a geographically distributed environment the centralized choice may result in unacceptable communication delays. For example, centralised architectures might provide poor response for time sensitive operations as well as imposing heavier levels of network traffic in some areas, because all communications must be dealt with by the central unit. Distributed solutions that deal with local parts of the network are more suited to the large scale setting such as Internet.

2.1.6 Support for efficient distribution of audio streams in the network

Efficient distribution of audio streams in the network primarily refers to minimising the network traffic even for very large numbers of simultaneous audio sources as well as for large numbers of simultaneous audio receivers. This means that by increasing the number of senders and/or receivers, the actual network traffic (the number of streams and/or packets) should not increase excessively (e.g. exponential or even proportional increase in audio traffic may not be acceptable). More specifically, network traffic optimisation techniques can include:

- Techniques that affect individual streams (i.e. reducing the bandwidth of a single stream) and/or
- Techniques that affect multiple streams (i.e. reducing the bandwidth of multiple streams).

Because this thesis especially deals with situations with large number of speakers, we mainly focus on reducing the total number of audio streams. Similarly to section 2.1.5, we are also interested in distributed solutions that are more scaleable (compared to those that are centralised) and able to operate across large numbers of networks and very large numbers of users.

Minimising the number of audio streams across the network is important because the cost (e.g. network bandwidth and overheads) of maintaining and distributing audio streams for extremely large numbers of simultaneous speakers and listeners can increase drastically compared to small-scale scenarios (typically O(N) or O(N**2)).

2.1.7 Support for adaptation and fairness

Adaptation refers to a mechanism that enables the source to moderate its transmission rate in response to the currently available bandwidth e.g. reduce its sending rate as a response to increased packet loss and round trip time. Adaptation mechanisms have been critical to the robustness and stability of the Internet. Applications that react to congestion by adapting their transmission rates, not only avoid congestion collapse but also keep network utilization high, make more efficient use of the network and generally see better performance because of it. Adaptive applications are capable of running over a much wider range of bandwidths and hence are more useful and scaleable in the Internet. Contrary to this, applications that do not perform any congestion control could be detrimental to the Internet and cause congestion collapse. Adaptation mechanisms can be broadly divided into two large categories:

- Single stream adaptation, which refers to mechanisms that dynamically change a single stream bandwidth, usually by changing its encoding schemes in the case of audio streams.
- Multiple stream adaptation, which refers to all mechanisms that do not change the bandwidth of individual streams but operate across multiple streams.

In particular, we are interested in how scaleable these mechanisms are when there are very large numbers of active audio streams as well as very large numbers of audio receivers.

As already introduced in chapter 1, fairness of these approaches towards TCP is also considered. As networked multimedia applications become widespread, it becomes increasingly important that these applications can coexist with (i.e. can be fair towards) the current TCP-based applications that form the majority of Internet traffic today. The TCP protocol is designed to reduce its sending rate when congestion is detected. Networked multimedia applications should exhibit similar behaviour. By this we mean that if a non-TCP connection shares a bottleneck link with TCP connections (travelling over the same network path under the same conditions of delay and packet loss) then the non-TCP connection should receive the same share of bandwidth (i.e. achieve the same throughput) as a TCP connection. Fairness is important because it prevents bandwidth starvation of responsive and adaptive streams.

2.1.8 Overview of the Rest of the Chapter

This section presents an overview of the rest of this chapter. The related work is reviewed in three categories and it covers all of the six criteria by combining some where appropriate.

Section 2.2 reviews techniques that deal with simultaneous speakers. The evaluation is mainly done in terms of the support for the most natural audio communication among users provided by these techniques (criterion 1).

Section 2.3 reviews techniques to deal with distribution of audio streams in the network in terms of maintaining acceptable audio quality even with heterogeneity and congestion (criterion 3) and efficient distribution of audio streams in the network (criterion 5). These two criteria are combined together because they effectively address the same issue: that of having and managing many simultaneous speakers but addressed from different and potentially conflicting network and user perspectives. Section 2.3 explains why spatial separation and efficient distribution of the audio streams are in conflict and shows examples of the solutions which are best according to criterion 3 but are least effective in terms of the criterion 5 (and vice versa).

Section 2.4 reviews various adaptation techniques in terms of support for heterogeneous unmanaged large scale networks and end systems (criterion 4), adaptability and fairness of the audio traffic in the network (criterion 6) as well as maintaining acceptable audio quality (criterion 2). These three criteria are combined together because successful and efficient adaptation is necessary to limit the packet loss in non-QoS capable networks as well as spanning wide range of heterogeneous networks.

At the end of each section there is a short summary and discussion subsection that emphasises which of the reviewed methods could (and could not) be potentially useful and integrated to a scaleable audio service to support many simultaneous speakers. Section 2.5 summaries all of these discussions and presents the core challenges which the model presented in chapter 3 must satisfy.

2.2 Techniques that deal with many simultaneous speakers

2.2.1 Introduction

Techniques that explicitly recognise the existence of many simultaneous speakers differ mainly in the way they determine which audio streams are transmitted over the network i.e. which users can be heard and what part of their audio activity can be heard. The following sections give brief overview of the limitations and capabilities of some of the most relevant existing techniques.

2.2.2 Floor-control

Floor control can be employed within shared workspaces to control access to the shared workspace. Each system must decide the level of simultaneity to support (i.e. numbers of active users at any time) and a granularity at which to enforce access control [Dommel, H.-P., Garcia-Luna-Aceves, J., 1997]. In the simplest form of floor control, only one participant has the floor at any given time and the floor is handed off when requested. In the case of audio, floor control introduces a management framework around the audio session, that (in this context) enforces turn-taking, thereby removing any potential simultaneity. This may be suitable for some structured applications, but may not be suitable for many others.

There are several types of floor control policy available for use by collaborative environments including explicit release, free floor, round-robin. This is addressed in greater detail in [Greenberg, S., 1991]. The principal difficulty is in achieving spontaneity of the communication between the users, but scaling to large groups is also difficult. In a conference where all members can request and be granted the floor, there is no global reason for giving one person the floor and not another.

2.2.3 Push-to-talk

"Push to talk" requires that users perform an explicit interface action such as pushing a button in order to be heard by others. This is the default approach used by the current MBone tools such vat [Jacobson, V., McCanne, S., 1999] and RAT [Hardman, V., Sasse, A., Handley, M., et al, 1995]. These tools enable every participant to hear everybody else in the conference simultaneously. The system interface usually comprises a window showing a list of the participants in the audio conference, gain and volume control etc. In order to talk, the pointer has to be placed in the window, and left mouse button pressed.

This allows users to make very careful choices over whether to "speak" or not, avoiding significant amounts of simultaneous speaking in many (more restrained) contexts. However, it is also liable to remove subtle non-verbal cues and sounds (which the user does not think to or chooses not to send). Also, the conscious activity required in order to be heard may also make conversation and interaction slower and more stilted. This reduces the spontaneity and interactivity of the conversations.

2.2.4 Silence Suppression

Silence detection (suppression) is commonly used in packet network audio systems to reduce the bandwidth consumed. Silence suppression continuously monitors the audio signal to be sent, and only sends audio packets when it detects significant audio activity (typically a minimum energy threshold). This technique reduces the audio traffic provided that the silence detection mechanism is effective and that each speaker's audio input is free of significant background noise (including echo). However, even with an ideal implementation, if several participants speak simultaneously then there will still be that number of audio streams. Therefore, even though silence suppression can decrease average levels of audio traffic, it cannot affect peak levels of audio traffic in the network. This is considered in more detail in the analysis of audio traffic in collaborative virtual environments (CVEs) described in chapter [Frécon, E., Greenhalgh, C., Stenius, M., 1999], [Greenhalgh, C., Benford, S., Craven, M., 1999].

2.2.5 Hardware Support - implicit floor control

[Kyeong-yeol, Y., Jong-hoon, P., Jong-hyeong, L., 1998] proposed an Audio Processing Unit (APU) that deals with linear Pulse Code Modulation (PCM) audio signals in a Multipoint Control Unit (MCU) that is used for multipont audio-video conferencing systems. This is a centralized unit which takes a fixed number of audio sources and distributes them echo free to specific clients. This unit therefore performs total mixing on a dynamic set of input streams.

The particular proposal for the APU in the paper deals with twelve linear PCM signals as input and makes four mixed PCM outputs. The APU has a speaker detection block and audio mixing block. In the speaker detection block power values of all channels are compared. The channel with biggest power value is considered to be the current speaker channel. Two other channels are selected as active speaker channels. The speaker information is transferred to the mixing block for selecting the channels which will be mixed. If S1, S2 and S3 are three current speakers and D1, D2 and D3 their data streams, the mixing block will always make the same four mixes: D1+D2, D2+D3, D1+D3 and D1+D2+D3. Those mixes will be transmitted to specific users: D1+D2 to S3, D1+D3 to S2, D2+D3 to S1 and D1+D2+D3 to all the others. This architecture supports only three simultaneous speakers and produces pre-determined mixes to do echo cancellation. There is no dynamics in the mixing policy.

2.2.6 Open Microphones

This technique allows all the microphones to be open at all times. This means that all audio data, produced by a user, is transmitted over the network to the rest of the users without any filtering (or any limitation).

2.2.7 Discussion

The primary criteria used when reviewing existing techniques that deal with multiple simultaneous speakers is the extent to which these techniques allow natural audio communication between the users. More specifically, various reviewed techniques have different ways of determining which users can be heard by the rest of the users as well as what part of their audio activity is transmitted to the other users.

Four major types of control imposed on selecting the audio streams to be transmitted can be identified in the reviewed techniques:

- *Explicit-select* refers to techniques where the application explicitly selects the speakers (and thus the audio streams to be transmitted), e.g. floor control,
- *Self-select* refers to techniques that allow the users to select themselves as speakers by, for example, pressing a button as in push-to-talk or by having audio energy threshold as in silence suppression,
- *Central-select* refers to techniques where a central unit chooses *n* audio streams based on current energies and thus determines the speakers e.g. APU, and

- *All-select* refers to techniques that do not explicitly limit the number of simultaneous speakers in any way (i.e. audio streams coming from all the users are directly transmitted) e.g. open-microphones.

Support for large numbers of concurrent speakers (to be heard) is provided only in open microphone and silence suppression techniques but not in floor control, push to talk and APU techniques.

Support for transmission of non-verbal as well as verbal sounds is very limited in push to talk and silence suppression techniques because only part of the audio streams for the current speakers is transmitted (usually targeted at verbal sounds). Floor control and APU techniques allow full transmission of the streams but only for the selected speakers. Only open microphones allow transmission of both verbal and non verbal sounds at all times for all speakers.

Support for spontaneity of the speakers is limited in case of floor control (since it exists only for the chosen speakers) and is prevented in push to talk techniques (because push-to-talk imposes an unnatural mechanism for requesting the permission to talk. Spontaneous conversations are allowed in silence suppression, APU and open microphone techniques.

The resulting number of simultaneous audio streams transmitted in the network differs for all the techniques but most often it is larger than 1. This number of streams is explicitly bound in floor control and APU techniques but not in the other techniques. Push to talk and silence suppression techniques result in the number of streams equal to the number of active speakers, while the open microphone technique results in the number of streams equal to the total number of open microphones. This means that there are techniques that allow potentially very large numbers of concurrent audio streams transmitted in the network whenever applications have large numbers of users.

Table 2.1 summarises this classification and evaluation of all of the approaches in terms of the type of control they impose on audio communication, their support for large number of speakers, support for spontaneous communication, and support for verbal and non-verbal sound transmission.

The table also shows the number of streams offered to the network by each technique. Floor control and APU offer a strictly bound number of streams that is predetermined in advance (before the session starts). Push to talk and silence suppression offer dynamic number of audio streams to the network that is equal to the number of currently considered speakers. And open microphones offer the number of streams that is equal to the total number of participants in the application.

	Floor control	**Push-to-talk**	**Silence Suppression**	**APU**	**Open Microphone**
Policy of determining what could be admitted in the network	Explicit select	Self-select (button)	Self-select (audio energy)	Central select (N*audio energy)	Select-all (continuous self-select)
Support for spontaneity	Current speaker only	No	Yes	N loudest speakers	Yes
Support for verbal sounds	Current speaker only	Yes	Yes	N loudest speakers	Yes
Support for non-verbal sounds	Current speaker only	Limited	Limited	N loudest speakers	Yes
Allow for large No of Sim. Speakers	Current speaker only	Yes	Yes	No	Yes
Bandwidth/Streams	Nbound	Nsp	Nsp	Nbound	Ntotal

Table 2.1 Types of control and level of support for natural audio communication in different techniques that deal with multiple users

None of the techniques satisfies all of our goals listed in the section 2.1.2, some of the techniques could still be included in the design for a truly scaleable audio service. For example, even though silence suppression does not in itself address the issue of simultaneous speakers, the scaleable audio service can be designed with or without it. Open microphones can also be a part of a new model for a scalable audio service.

Other techniques explicitly break one or more of the goals listed in the section 2.1.2 and therefore cannot be used in a scaleable audio service that allows natural audio communication.

The techniques reviewed in this section do not in themselves address distribution issues, which are considered in the next section. Note, however, that the acceptable techniques can potentially result in large numbers of audio streams. This means that we have to deal with simultaneous speakers in the network. The next section presents approaches to dealing with this problem.

2.3 Techniques to deal with efficient distribution of audio streams in network

2.3.1 Introduction

This section reviews techniques that deal with efficient distribution of audio streams across the network (criterion 5) and explores the effect they have on the support for individual tailoring of mixing and independent audio streams at the end systems (criterion 3). Any change in the number of streams in the network will have a direct impact on the flexibility the end users have when creating their own mixes. A brief discussion below shows how the solutions that are the most effective for the users' satisfaction, are not necessarily optimal for the network, and vice versa.

There are two basic types of distribution techniques that could be defined in the following way:

- Peer-to-peer distribution where audio mixing is done in the clients and packets are sent either via unicast or multicast.
- Server mixing where server components mix all of the incoming streams and transmit them either via unicast or multicast.

Each of these techniques will result in a different number of audio streams in the network for the same number of sender and receivers. More specifically peer-to-peer and server forwarding distribution techniques do not change the number of audio streams coming from a certain number of senders, while sever mixing techniques do.

From a user's point of view, the ideal multi-party audio distribution would present them with an independent audio stream from each individual speaker (criterion 3) at a satisfactory quality (criterion 2), with minimum latency and with no jitter. Receiving independent streams would allow each listener to create their own personal audio mix. This can be tailored to their own display equipment; can be spatialised according to their location within a shared virtual space; and can be under their own control, allowing them to raise or lower the volume of particular speakers. In networking terms this is a "peer-to-peer" solution. It assumes the existence of a number of speakers (audio sources), each producing a distinct audio stream, and a number of listeners (audio sinks), each receiving the audio stream from each source. This is illustrated in Figure 2.1 (a). It should be noted that this is a logical model of inter-process communication; it can be realised using underlying unicast or multicast protocols.

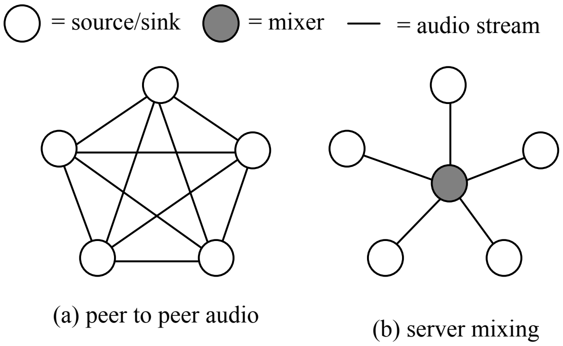


Figure 2.1 Logical model of peer-to-peer vs. server based approaches for distribution of audio streams

However, this ideal, even with multicast mechanisms, is very demanding in terms of network resources, particularly bandwidth: the peer-to-peer approach can easily flood the network with traffic for large number of simultaneous audio sources. With underlying unicast protocols – the norm today for wide-area communication – the resulting number of audio streams is of the order $O(n^2)$, where n is the number of simultaneous users (for the case in which all users are sending audio data simultaneously). With underlying multicast protocols – which are still experimental over many wide-area networks – this reduces to $O(n)$, but no lower.

Introducing server mixing components in the network can also reduce the number of audio streams in the network compared to peer-to-peer uincast, but it does not allow for any individual tailoring of mixing at the end clients (Figure. 2.1 (b)).

This section continues by briefly reviewing the techniques that make the distribution of a single flow more efficient. Section 2.3.2 reviews the approaches that affect (reduce) the bandwidth requirements of a single stream imposed on the network. We then focus especially on the approaches that affect the bandwidth of multiple streams by combining them because that is the only approach which can be efficient when dealing with large numbers of simultaneous speakers. Section 2.3.3 reviews relevant techniques based on mixing. Some techniques belong partially to both classifications and their relevant characteristics will be discussed in both sections.

2.3.2 Techniques that affect single stream bandwidth

2.3.2.1 Multicast

Network multicasting offers an efficient multipoint delivery mechanism: a single packet is sent to an arbitrary number of receivers by replicating the packet within the network at fan-out points along a distribution tree rooted at the packet source. This means that transmissions one to many are done without packet duplication on network links. Multicast is based around the formation of groups whereby a number of processes may join a multicast group and will then receive all data sent to that group address.

Multicast is still not widely supported in commercial routers and is undergoing incremental deployment in the Internet. However IP multicast has been heavily exploited in the design of a number of end-to-end protocols and multicast applications such as real-time media transport of audio and video with accompanying control information (e.g. vat [Jacobson, V., McCanne, S., 1999], RAT [Hardman, V., Sasse, A., Handley, M., et al, 1995], vic [McCanne, S., Jacobson, V., 1995], nv [Frederick, R., 1993] and ivs [Turletti, T., 1994]).

Layered multicast is based on layered encodings [McCanne, S., Jacobson, V., Vetterli, M., 1996] that can provide several different versions of the same signal at a range of different bandwidths. The media stream is encoded into a number of layers that can be incrementally combined into a number to provide a refined versions of varying quality of the encoded signal. The individual layers are transmitted on separate multicast addresses. Receivers adapt to network conditions by adjusting the number of layers they subscribe to and thus maximising perceived quality by trading off average signal quality against packet loss. McCanne at al propose multicast video as an application for this scheme.

Simulcast, another approach to layered transmission, simply transmits the same data stream encoded with different quality levels on different multicast sessions [Furht, B., Westwater, R., Ice, J., 1998]. This scheme is often called *simulcast* because the source transmits multiple copies of the same signal simultaneously at different rates resulting in different qualities. The problem with this approach is the cost of sending multiple replicated steams and thus possibly congesting the network on links close to the source.

2.3.2.2 Speech compression –static (basic compression)

Speech compression algorithms are used to reduce the data and/or packet rate. For example, the ITU has recommended a series of compression schemes for speech such as the following:

- **G.721 –** is a simple ADPCM technique where the sampling rate is 8KHz and the bit rate is 32 Kbps.

- **G.722 –** is a more sophisticated standard that targets improved quality. It ranges from 48Kbps to 64Kbps.

- **G.711** – is the international standard for encoding telephone audio on an 64 kbps channel. It is a pulse code modulation (PCM) scheme operating at a 8 kHz sample rate, with 8 bits per sample. It uses logarithmic compression that matches the way the human ear works. It only loses information which the brain would not process, and gives good quality results for speech. There are two standard forms: A-law and m-law.

- **G.723** – is another lossy standard that operates at 2.4Kbps. The resulting sound quality is inferior to that of the uncompressed PCM or that of G.722. The G.723 standard seems to be little used.

- **G.728** – targets low bandwidth, 16Kps only, but the resulting quality is inferior to that of the other standards.

The most highly performing schemes use vector quantisation. US Federal Standards exist which operate at very low bit rate. The *code excited linear prediction* (CELP) technique operating at only 4.8 Kbps for a quality slightly inferior to conventional G.711.

There are also other compression algorithms that target much lower rates than conventional PCM encoding such as:

- **GSM-** GSM is a compression standard for mobile telephony. The version 6.1 operates at 13.2 Kbps. The sampling rate is 8KHz. The resulting sound quality of a GSM compression scheme is inferior to that of G.711 or G.722

Higher quality audio compression algorithms are also possible but they are used mainly for sound compression and not speech:

- **MPEG-** The family of MPEG-Audio standards is designed to operate at compressed bit rates ranging form 32Kps to 448 Kbps per monophonic channel. MPEG compression schemes are lossy but they achieve perceptually lossless quality.

2.3.2.3 Self-Organised Transcoding Scheme

Kouvelas et al in [Kouvelas, I., Hardman, V., Crowcroft, J., 1998] present a self-organised scheme to form groups out of co-located receivers with bad reception. A representative of the group is responsible for locating a suitably positioned receiver with better reception that is willing to provide a customised transcoded version of the session stream. In this way, the transcoding site provides local repair to the congestion problem of the group with minimal increase in stream delay. The data rate and level of redundancy of the transcoded stream are continuously modified to adapt to the presumed bottleneck link characteristics using reception quality feedback from a member of the formed loss group. This approach also achieves network friendly congestion control of the real-time multicast stream.

2.3.2.4 Filters

Pasquale et al in [Pasquale, G., Polyzos, E., Kompella, V., 1993] propose the use of self-propagating filters over a dissemination tree. Leaf nodes specify to the node above them filters that can convert an incoming stream to match their requirements. When a non-leaf node has multiple output links with similar filters, the filter is propagated to a node higher up in the tree. This scheme can achieve optimal network utilisation with minimal processing but requires full knowledge of the distribution tree topology and processing capabilities at each node.

Pasquale's Multiparty Multimedia Channel (MMC) proposed in [Pasquale, G., Polyzos, E., Anderson, E., et al, 1994] is designed specifically for heterogeneity by imposing only a loose coupling between the receivers and senders. In their architecture the receivers connect to a multicast channel though a port. By attaching a filter to the port, a receiver instructs the network to transform a flow to a lower rate. In turn the network optimises the delivery of heterogeneous flows by propagating the receiver-specified filters up the multicast tree. At merge points in the tree, filters of the same types are combined and further propagated to the tree. While this architecture supports heterogeneous transmission, it does not provide a mechanism for end-systems to adapt to available capacity in the network. Instead MMC relies on admission control to explicitly allocate available bandwidth to heterogeneous receivers.

The loose coupling between source and receivers supported by MMC is well suited to a broadcasting or general mass media distribution, but it is perhaps not as appropriate for more interactive applications where tighter coupling between group members are required. Pasquale's filters have been controversial because of the heavy load certain filter operations may place on the network objects. The filter propagation protocol may be over complicated to implement in networks where switches have low processing capabilities.

[Yeadon, N., 1996] proposes using filters primarily to reduce data transmitted to receivers that either cannot use it or do not wish to use it: for example sending stereo audio to a receiver with a single speaker is wasteful. Yeadon employed filters on data streams, which can tolerate data loss to provide individual QoS for individual clients. The proposed model involves placing filters at strategic points, such as network nodes or specialised servers around a multicast network tree. The designated source may send a stream at the quality required by the highest capability receiver while low capability receivers acquire filtered version of a media stream. However, even though filters are distributed, all the decisions about resource management (admission control and filter allocation) are made in a single central point (session manager):

"Knowledge about network load and present QoS levels being supplied to all group members is held by the session manger." The bandwidth between the session manager and the filter is assumed to be high enough and the link stable (i.e. there is no mechanism to handle failure of any of these links). Filters themselves do not perform any monitoring of the current bandwidth (there is no congestion control or fairness).

2.3.2.5 Proxy services

In the proxy model proposed by [Fox, A., Gribble, S., Brewer, E., et al, 1996], media gateways are situated at strategic points within the network and actively transform media streams to mitigate bandwidth heterogeneity and client diversity. By placing a proxy between the source and the sink of data, network bandwidth variation can be accommodated through format 'distillation', and the allocation of bandwidth across flows optimised using intelligent rate adaptation. Moreover, the proxy can translate the underlying media representations to enable communication among otherwise incompatible clients.

Proxy services can be very effective in adjusting the QoS for the end systems (e.g. they usually have fine selection of coding algorithm) but they are not equally effective for the entire network since they do not monitor the core network and they only adjust the traffic locally for the network edges.

2.3.2.6 Discussion

Various multicast techniques, compression, transcoding filter and proxy services were discussed. Since these techniques affect only the bandwidth of a single stream and do not operate across multiple streams, they deliver all the audio streams separately to the end user. Thus, these techniques provide the best support for individual tailoring of mixing and spatialisation at the end user.

These techniques are also quite effective in reducing the bandwidth requirements of a single stream as well as in optimising distribution of audio streams in the network from a single source to many listeners.

However, none of these techniques addresses the issue of many simultaneously active sources (speakers). Therefore, the amount of audio data and processing for these techniques rises proportionally with the number of simultaneous speakers. (e.g. ten simultaneous speakers will produce ten times as much audio data and require ten times as much processing as one speaker).

Even in case of single stream distribution, compression (and techniques based on compression) can sometimes cause additional problems. Considerable processing is required by most compression/decompression schemes, and this is proportional to the number of different streams being received. For example, it may not be realistic to expect every end user machine to have hardware codec boards on the same machine nor software capable for decoding the majority of international standards already installed. Also, some very low bit-rate compression schemes do not handle mixed signals well, since they are modelled on a single human voice.

[Kouvelas, I., Hardman, V., Crowcroft, J., 1998] point out difficulties imposed by layered encoding for speech. They argue that although layered encoding in layered multicast is possible for sampled speech, the transmission bandwidth range is significantly more restricted than for video. Currently available speech codecs do not render themselves naturally to layering. Hardman et al. point out yet another disadvantage of layering that it makes playout calculation more difficult [Hardman, V., Sasse, A., Handley, M., et al, 1995]. They argue in the following way. Different layers and different multicast groups may be routed differently in the network. Reconstructing the data segment consisting of data packets sent over different layers the receiver needs to wait until all the different parts are received. This incurs additional delays and might reduce perceptual quality. Finally, multiple simultaneous speakers will always produce multiple layers to be distributed and processed, and there is always at least one layer per source.

This section discussed the techniques that affect the bandwidth of a single stream. Even though this thesis focuses on multi-stream techniques, single stream techniques can still be beneficial for creating a scaleable audio service, and thus employed as an integral part of it.

2.3.3 Techniques that affect multiple streams

Multi-stream techniques are based on some form of mixing to affect (decrease) the number of audio streams in the network.

2.3.3.1 Mixing Architectures

The techniques for mixing depend upon the type of the medium as well as the application. In the case of audio, mixing multiple streams involves digitally summing the audio samples and then normalising the result. In case of a point-to-point unicast and multicast architectures supporting virtual meeting of multiple users, each user will get a number of streams equal to the number of other simultaneous speakers and mix them before playing them out on their machine. This approach does not affect the number of audio streams in the network. However, mixing of audio streams does not have to happen only in the end systems but could also happen in server components in the network. Server mixing components can drastically reduce the number of audio streams in (some parts of) the network in the cases of very large numbers of simultaneous audio sources. In order to eliminate one’s own audio feedback, each participant, on receiving the mix, must remove their own contribution before playing it back. The need to continuously mix many audio streams, coupled with the real-time nature of the medium poses special requirements for media transmission protocols. Positive and negative aspects of mixing are discussed in detail in the discussion section 2.3.3.7.

Mixing can be incorporated within several different communication architectures.

2.3.3.2 Centralised (total) mixing

This is an extreme alternative to the peer-to-peer approach. In the centralised architecture every audio stream is sent to a centralised audio mixer that mixes multiple streams into a single stream that is then redistributed to each listener. Total mixing requires that each listener (and their nearby network) handle only one audio stream. However, this prevents each listener from creating their own personal mix. Also, the server becomes a potential bottleneck in the system, since it (and its nearby network) has to deal with, mix and distribute *O(n)* streams.

2.3.3.3 Client-specific Mixing

Client-specific mixing has been proposed in the SPLINE system to allow users with low bandwidth connections to access an audio-graphical multi-user virtual world [Waters, R., Anderson, D., Barrus, J., et al, 1997]. One or more low-bandwidth users connect to a specialised access server that interfaces to the main system (which uses peer-to-peer multicast audio). This access server provides a customised audio mix for each connected user, which is sent directly to them. This approach still requires (like total mixing) that the access servers deal with and mix all of the available audio streams. Unlike the previous total mixing solution it does provide a separate mix to each listener, giving greater flexibility, though at the cost of more work for the access servers (since sending two copies of the same mix is easier than computing and sending two different mixes).

2.3.3.4 Hierarchical Mixing

In order to support applications such as tele-orchestra, which involve large numbers of simultaneously active audio streams, Rangan in [Rangan, P., Harrick, M., Ramanathan, V., 1993] proposed a hierarchical mixing architecture, and showed that it is an order of magnitude more scaleable to purely centralised or distributed architectures. In this mixing hierarchy, participants constitute leaf nodes, and the mixers are non-leaf nodes. During the transient phase, the root mixer computes the *fusion set* (since it is the only node that receives packet information from all participants), and propagates it to each of the intermediate mixers. The fusion set is determined as the set of packets from different sources that are to be mixed to form a composite packet. During the steady phase, each mixer receives media packets from its children, mixes them, and sends the composite packet to its parent. The mixer that is at the root of the hierarchy forwards the final mixed packet to each of the leaf nodes. The bandwidth required for packet reception at each mixer is proportional to the number of its children, whereas the bandwidth for packet transmission is that of sending to just one parent. By increasing only the height of the hierarchy while bounding the number of children of each mixer, the hierarchical architectures can be made highly scaleable.

Like centralised mixing, this approach prevents each listener from creating their own mix(es) since all the streams will be mixed all the time. This is not very well suited for interactive and collaborative applications where each user should have the flexibility of controlling and mixing independent audio streams. This approach also has no awareness of the underlying network conditions, and relies only on statically configured mixers. The root node is a potential bottleneck as it receives the packets from all the nodes.

2.3.3.5 Filtering and proxy services which perform mixing

Filter and proxy services mentioned in sections 2.3.2.4 and 2.3.2.5 respectively could also perform total mixing in addition to the different compression and encoding schemes. This mixing is always done in the same way: all the incoming streams are mixed together in order to produce single output stream. They can be distributed within the network and therefore can be very efficient in terms of bandwidth reduction (criterion 5) and heterogeneity (criterion 4), but do not allow flexibility in managing independent audio streams in the end user (criterion 3).

2.3.3.6 Real-time Protocol (RTP)

RTP [Schulzrinne, H., Casner, S., Frederick, R., et al, 1996] has been standardised for real-time delivery of multimedia data over multicast (and unicast) networks by the IETF Audio/video Transport Working Group. RTCP is the control protocol embedded in RTP. RTCP requires receivers to periodically multicast reception reports that include loss rate and throughput to the entire group to provide better scaling properties. As the number of members in the group increases, each receiver transmits reception reports less frequently such that aggregate bandwidth for all reception reports remain below a small percentage of the session bandwidth. RTP does not address congestion control but reception reports provide sufficient information to all members to identify receivers that are experiencing congestion. Thus a source might be able to exploit this information to adapt its transmission rate. The main problem here is that as the size of the multicast groups increases, the resolution of reception reports decreases. Thus the source becomes less responsive and congestion lasts for longer period of time.

RTP also has the concept of application/RTP level mixers and translators. These are intended to support heterogeneity. A mixer takes RTP packets from a number of sources, mixes them and outputs a single RTP stream, thus taking a fraction of the bandwidth to transmit all constituent frames. A translator may alter the encoding format or act as a bridge between different styles of network, e.g. a translator could copy each multicast packet to a number of unicast packets if the network on one side of the translator could not support multicast communication whereas the other could.

2.3.3.7 Discussion

This section summarises the techniques for distributing audio streams in the network that affect multiple audio streams, and discusses how effective they are in terms of decreasing the network traffic for large numbers of simultaneous senders as set in the criterion 5. These techniques are also evaluated in terms of the support they provide for individual tailoring of mixing and independent audio streams (criterion 3).

Performing mixing (e.g. in the server, filter, proxy or RTP application-level component) in the network is the only approach that can decrease the number of audio streams in the network. Mixing can thus drastically reduce network traffic for many simultaneous speakers. This makes it efficient according to its support for efficient distribution of audio streams in the network (criterion 5). Mixing also imposes no constraints on individual speakers (compared with floor control's "gagging" of users). This means that the system's view of "speaking" is the same as the user's. This makes it effective according to supporting the most natural audio communication (criterion 1).

Besides these benefits, there are also negative aspects to mixing which make it a less popular and less used approach compared to multicast for designing scalable audio services. These negative aspects mainly concern the user's point of view:

- The mixed signal is noisier than the separate streams (and/or has reduced dynamic range) since it includes the noise from all of the separate streams in fewer bits.
- Mixing may not work well with some ultra-low-bandwidth CODECs due to their compression algorithms;
- Mixing introduces additional delay compared to direct peer-to-peer communication, as audio streams have to be processed by mixers en route from speakers to listeners; and
- Mixing components introduce more components (servers) in the network, with corresponding hardware, organisational and real-time control requirements.

The major disadvantage of mixing considered in this thesis is that mixing introduces loss of spatialisation and other aspects of individual listener control over individual audio streams. The mixed stream does not retain any distinction between the component signals (the effect is similar to that when the "mono" button on a stereo hi-fi or radio is pressed, which mixes the left and right audio channels together). Note that the mixing process is in most cases irreversible and the individual streams cannot be split at a later stage.

However, despite these negative aspects, integrating mixing within an audio services that attempts to support large numbers of simultaneous senders (and not only large numbers of concurrent receivers) still seems to be the most natural approach.

Another guideline for designing an audio service for efficient distribution of audio is that such a service should be distributed: the more distributed any chosen approach for the audio service is, the more potentially scalable that service is. This means that such a service might cope with more audio streams, and the CPU load and the number of input/out streams (I/O load) could be evenly distributed across the distributed components in the network. For example hierarchical and filter-based mixing approaches should be more scaleable than centralized or proxy-based mixing.

RTP can also be integrated when developing scaleable audio services as it provides necessary information for potential increase/decrease of the audio traffic in the network as well as application-level mixers. In other words, RTP does not perform any decrease/increase of the bandwidth on its own, but could be used as one component for a system that aims to do it.

Table 2.2 summarises this discussion in terms of the efficient distribution of audio streams (criterion 5) but also how it affects individual tailoring of mixing and independent audio streams in the end systems (criterion 3). Note that none of the mixing approaches has the possibility of delivering separate streams to the end user since the corresponding mixing components (i.e. server, filter, proxy) perform total mixing on all the input audio streams.

	RTP	Centralised Mixing /Proxy Mixing	Client-specific Mixing	Hierarchical Mixing	Filter mixing
Decrease in the per stream bandwidth	Provides information	No	No	No	No
Decrease in the num. of streams	Potentially (provides information)	Yes	Yes	Yes	Yes
Distribution	Distributed	Centralised	Distributed	Distributed but only for LANs	Distributed functionality/ centralized decision unit
Allowing tailoring of mixes and spatialisation	Only if mixing is not performed	No	Yes	No	No

Table 2.2 Overview of the discussion of the techniques that deal with efficient distribution of audio streams

2.4 Rate adaptation techniques

2.4.1 Introduction

This section reviews various rate adaptation techniques as possible ways of dealing with congestion in the network (criterion 6), supporting heterogeneous networks (criterion 4) and managing packet loss (criterion 2). More specifically, rate adaptation mechanisms of various well-known congestion control protocols are discussed. These mechanisms are evaluated in terms of how fair they are towards TCP and how scaleable they are for large numbers of simultaneous audio senders.

As already discussed in section 2.1.7, rate adaptation techniques might be broadly divided into those that perform adaptation of a single stream and those that perform adaptation across multiple streams.

However, there are no techniques that explicitly adapt multiple streams, so this section reviews well-known adaptation mechanisms which affect bandwidth of a single stream. Before reviewing individual mechanisms, basic terminology and classifications of any congesting control mechanism are introduced.

2.4.1.1 Classification of congestion control mechanisms

There are various ways for detecting, signalling and reacting to congestion. However, there are only two basic resource management models to regulate the offered load to a network: *open loop* and *closed loop* [Rejaie, R., 1999].

In open loop congestion control, the burden of traffic management is in the network. A source describes its traffic to the network with several parameters, and the network reserves some resources along the path during call establishment. If resources are not available, the request for new connection is rejected. The source should wait for the flow specification and shape its traffic to stay within that profile. This approach is well suited for reservation-based networks. The main drawback of this approach it that it might result in suboptimal overall utilisation of the network in cases when a resource allocated to a connection remain unutilised by the requested source while the network rejects requests from other client. The ratio of peak and average load on the particular link is quite high and provisioning for peak load is not economical [Rejaie, R., 1999].

This thesis focuses mainly on closed loop congestion control, in which a source receives feedback about the state of the network and reacts to that feedback by adjusting its transmission rate accordingly. The source might utilize explicit feedback provided by the network that carries some information about the state of the network e.g. ECN [Ramakrishnan, K., Floyd, S., 1999], DEC-bit protocol [Ramakrishnan, K., Jain, R., 1990]. Alternatively, the source may exploit implicit feedback by inferring the state of the network from parameters such as round trip times and loss rates. The accuracy of the rate adaptation strategy when adjusting the transmission rate depends on the accuracy of the feedback signal. The more information the feedback signal provides, the more effective the congestion control mechanism can be. This thesis is mainly interested in mechanisms based on implicit feedback, as schemes with explicit feedback cannot easily be deployed over today's Internet since the required feedback components are not implemented (e.g. most of the current routers in the Internet implement only FIFO queuing).

Closed loop congestion control mechanisms can further be classified, based on the manner in which transmission rate is adjusted, into window-based and rate-based congestion control. In a window-based scheme, a source directly controls the number of packets in transit by adjusting a window that is the upper bound for the number of packets in flight. This is the mechanism that TCP uses for its congestion control. In a rate-based scheme, the source directly adjusts its transmission rate by controlling the gap between consecutive packets. This thesis is concerned mainly with rate-based congestion control because it is more useful for applications that are inherently rate-based such as streaming multimedia [Handley, M., Padhye, J., Floyd, S., et al, 2000]. It is relatively easy to modify the sending rate to adhere to application constraints such as timing constraints or delay jitter. Reliability and congestion control are decoupled in the rate-based scheme. This provides more flexibility than TCP's window mechanism which combines reliability and congestion control.

Congestion control mechanisms can also be classified, based on the point where congestion control functionality is implemented, into end-to-end and hop-by-hop congestion control. Hop-by-hop congestion control [Mishra, P., Kanakia, H., Tripathi, S., 1996] refers to the cases where congestion control is deployed between every two elements (or hops) in the network. Alternatively congestion control can be deployed only between two end points, this is referred to as end-to-end congestion control. Even though the hop-by-hop approach has much smaller delays and is potentially more effective, it adds to the complexity of the intermediate nodes in the network and assumes greater homogeneity. This thesis is concerned mainly with end-to-end congestion control and assumes that the network is passive. This approach is chosen in order to provide better support for heterogeneous networks.

The following sections review end-to-end rate adaptation mechanisms that receive explicit feedback about the state of the network from the network and server components, and then those that receive implicit feedback from the other end systems.

2.4.2 Rate-adaptation techniques with explicit feedback

2.4.2.1 Adaptive Load Service (ALS)

ALS [Sisalem, D., Schulzrinne, H., 2000] is basically similar to the available bit rate (ABR) service for ATM [Shirish, S., 1995]. With ALS, the sender transmits control messages indicating its desired transmission rate to use. The intermediate routers adjust this value in accordance with their available resources and forward the control messages to the next network node until they reach the receiver. The receiver in turn transmits the updated information back to the sender which adjusts its transmission behaviour in accordance with the received information.

While ALS is rather similar to ABR service, it is much simpler in its specification and it is based on the IP protocol. ALS was designed to accommodate the needs of heterogeneous receivers by supporting the notion of layered data transmission whereby each layer is sent to a different multicast group. ALS provides the receivers with the exact information about the layers to join in order to receive the QoS level that corresponds to their capacities.

The performance of ALS in environments with heavier loads and more members joining and leaving was not investigated. The effects of choosing the length of the adaptation intervals on the achieved utilization was not considered. Finally, the performance of ALS was not tested under a real network situation.

2.4.2.2 Bandwidth Adjustment Server (BWAS)

BWAS [European Patent 1 024 638] is a centralized server which monitors the bandwidth in a shared broadcast medium such as a LAN, decides on the coding schemes that the end terminals should use to fit the available bandwidth and instructs each of them to increase or decrease the current codec algorithms in use. The single server keeps information about all on-going connections in the form of look-up tables: how much bandwidth they are using and which codec hierarchies they have and can use. BWAS uses two pre-determined bandwidth thresholds to decide the upper and lower bandwidth allowed for the connections. The terminals must use H.323 and they use H.245 signalling to renegotiate their coding capabilities when instructed by the BWAS to change their current codec algorithms.

2.4.3 Unicast-based rate adaptation techniques with implicit feedback

2.4.3.1 Rate Adaptation Protocol (RAP)

Rate Adaptation Protocol (RAP) [Rejaie, R., Handley, M., Estrin, D., 1999] is an end-to-end congestion control mechanism suited for unicast delivery of multimedia streams, and it reacts to congestion on very short time-scales. The authors propose an additive increase/multiplicative decrease (AIMD) rate control protocol [Bolot, J., Turletti, T., Wakamn, I., 1994] [Busse, I., Deffner, B., Schulzrinne, H., 1996] that uses acknowledgements (in a manner similar to TCP) to estimate round trip times and detect packet loss. The rate adjustment is done every round trip time: linear rate increase on no packet loss and halving the rate on packet loss. The authors also propose to use the ratio of long-term and short term average round trip times to further fine tune the sending rate on a per-packet basis.

RAP is just a core component for an end-to-end transport protocol targeted at playback of real-time streams. RAP machinery is mainly implemented at the source.

However, there has been no work done towards using and testing RAP for interactive applications.

It achieves rates similar to TCP in environments where TCP experiences no or few timeouts since RAP's rate reductions resemble TCP's reaction to triple duplicate ACKs. However, RAP does not take timeouts into account and is therefore more aggressive when TCP's throughput is dominated by timeout events.

2.4.3.2 Direct Adjustment Algorithm (DAA)

DAA [Sisalem, D., Emanuel, F., Schulzrinne, H., 1997] is based on TCP congestion control mechanism and relies on end-to-end Real Time Transport Protocol (RTP) for feedback information. DAA uses a combination of the two approaches: additive increase/multiplicative decrease proposed in [Bolot, J., Turletti, T., Wakamn, I., 1994] [Busse, I., Deffner, B., Schulzrinne, H., 1996] and an enhancement of the throughput model described in [Floyd, S., Fall, K., 1997]. It results in TCP-friendly behaviour as long as TCP remains in its AIMD regime. Scalability of DAA has been evaluated only in multicast sessions with several hundreds of receivers (but not senders). One of the main issues when evaluating scalability is that increasing the number of session members increases the number of received RTCP packets at the receiver.

This scheme does not mimic TCP behaviour in timer-driven mode because it does not take timeouts into consideration. The timer-driven mode in TCP plays a major role in stability of the network under heavy load, and all other end-adjustment strategies should incorporate such a conservative mechanism as well.

2.4.3.3 SCP

Cen et al [Cen, S., Pu, C., Walpole, J., 1998] presented the SCP protocol for media streaming. SCP deploys a modified version of TCP's congestion control mechanism that performs Vegas-TCP-like rate adjustment in steady state. With Vegas-TCP

Brakmo et al [Brakmo, L., Peterson, L., 1995] describe an improved approach for determining widow sizes using variations in the measured round trip times. Their results show that SCP is not TCP-friendly. [Rejaie, R., 1999] points out that this may be due to the use of the shortest round trip time (RTT) that has been measured since this may vary widely for different flows. This approach has also not examined issues such as stability or responsiveness.

2.4.3.4 Multimedia Transport Protocol (MTP)

Jeffy et al [Jeffy, K., Stone, D., Talley, T., et al, 1992] designed an unreliable connection oriented transport protocol on top of UDP/IP called Multimedia Transport Protocol (MTP). MTP monitors the local packet transmission buffer to detect congestion. Once a packet is discarded due to buffer overflow, the protocol signals the application to reduce the data rate. This scheme is only suited for LANs where congestion could result in increased media access latencies at the local adapter.

2.4.3.5 TFRCP

Using the formula proposed by Padhye, J., Firoiu, V., Towsley, D., et al, 1998], Padhye et al in [Padhye, J., Kurose, J., Towsley, D., et al, 1999] present a scheme in which the sender estimates the round trip time and losses based on receiver's acknowledgments. In case of losses, the sender restricts transmission rate to the equivalent TCP rate calculated using the formula, otherwise the transmission rate is doubled. The scheme behaves in a TCP-friendly manner during loss phases. However, the increase behaviour during underloaded situations is rather arbitrarily chosen and might result in severe unfairness as the adapting end systems might increase its transmission rate much faster than competing TCP connections.

2.4.3.6 TFRC

TRFC [Floyd, S., Handley, M., Padhye, J., et al, 2000a] is evolved from the TFRCP protocol [Padhye, J., Kurose, J., Towsley, D., et al, 1999]. Similar to TFRCP it adjusts its sending rate based on the complex TCP equation [Floyd, S., Handley, M., Padhye, J., 2000] but uses more sophisticated methods to gather necessary parameters. Immediately after start-up, the sender goes into a slow-start phase similar to TCP slow start to quickly increase the sending rate to a fair share of the bandwidth. TFRC slow start is terminated with the first loss event. Once per round trip time the TFRC receiver updates its parameters and sends a state report to the sender. The sender then computers new fair rate from these parameters and adjust the sending rate accordingly. A major advantage of TFRC is that it has a relatively stable sending rate while still providing sufficient responsiveness to competing traffic.

2.4.3.7 LDA+

Unlike many of the other schemes, the Loss-Delay Based adaptation algorithm LDA+ [Sisalem, D., Wolisz, A., 2000] does not devise its own feedback mechanism to control the sending rate but relies solely on the RTCP feedback messages provided by the RTP. LDA+ is an AIMD congestion control. The increase and decrease factors for AIMD are dynamically adjusted to the network conditions. An estimate of bottleneck bandwidth is obtained using packet pairs. With packet pairs the time interval between the receipt of the two packets that were sent back-to-back is used as a hint of current maximum rate of a flow.

The problem with LDA+ is that RTCP reports are generated infrequently (usually within several seconds). This makes LDA+ slow to react to changes in the network conditions.

2.4.4 Multicast-based rate adaptation techniques with implicit feedback

Multicast schemes can be classified based on the entity that actively changes the performance of the entire system. Rate adaptation schemes for multicast can be divided into three main categories: sender-based, receiver-based and hybrid adaptation schemes.

A multicast flow is defined as TCP-friendly when for each sender-receiver pair, the multicast flow has the property of being unicast TCP-friendly (i.e. when it does not reduce the long-term throughput of any co-existing TCP flows more than another TCP flow on the same path would do under the same network condition).

2.4.4.1 Sender-based

In sender-based approaches the sender adapts its transmission behaviour based on feedback information generated from its receivers. However, having each receiver frequently reporting feedback information would result in *feedback implosion* at the sender. To avoid this situation, congestion control schemes for multicast communication usually use special mechanisms to reduce the flow of feedback information from the receiver to the sender and still provide the sender with enough information about the congestion state of the receivers. Such mechanisms can be roughly divided into five categories [Hoffmann, M., Nonnenmacher, J., Rosenberg, J., et al, 1999]: distributed [Handley, M., 1996], suppression [Sisalem, D., Wolisz, A., 2000], representatives [DeLucia, D., Obraczka, K., 1997], polling [Bolot, J., Turletti, T., Wakamn, I., 1994] [Sisalem, D., Wolisz, A., 2000] and hybrid [Rhee, I., Balaguru, N., Rouskas, G., 1999]. Detailed analysis of these approaches could be found in [Sisalem, D., Wolisz, A., 2000].

2.4.4.2 Receiver-driven Layered Multicast Congestion Control

RLM [McCanne, S., Jacobson, V., Vetterli, M., 1996] is a framework that was developed to enable efficient utilization of network bandwidth and to allow application to receive data at the highest quality possible without causing network congestion. It does this by having a number of multicast groups to which data at different qualities is sent. RLM uses a cumulative model whereby each multicast group, or layer, provides refinement information to the previous layers.

The receiver can control the quality of the data it receives by joining and leaving groups. It does this according to a simple control loop: if the network is congested then it drops a layer and if there is spare capacity on the network it adds a layer. To do the former, the receiver simply adds layers until congestion occurs and then backs off to an operating point below this bottleneck. The latter is more complicated and here the receiver makes use of a “join-experiment”. This involves the receiver carrying out active experiments by spontaneously adding layers at “well chosen times”. If this causes congestion then the receiver drops the offending layer. A learning algorithm is used for this so that the receiver will, over time, determine the level of subscription that will cause congestion.

However, Joerg Widmer at al in [Widmer, J., Denda, R., Mauve, M., 2001] argue that the use of RLM to control congestion is problematic since RLM’s mechanism of adding and dropping a single layer based on detection of packet loss is not TCP-friendly and thus can result in unfair distribution of bandwidth among concurrent RLM sessions. Furthermore, they point out that leaving a multicast group can take a significant time, usually on the order of several seconds. Failed join experiments (e.g. a receiver joining a layer has to immediately leave again because the necessary bandwidth is not available) are therefore very costly because of the additional congestion they may cause.

RLC is another TCP-like congestion control mechanism for layered multicast described by Vicisano et al in [Vicisano, L., Rizzo, L., Crowcroft, J., 1997]. They propose to dimension the layers so that the bandwidth consumed by each new layer increases exponentially. In this approach, layer 1 carries twice as much data in the same amount of time as layer 0. The time the receiver has to wait before being allowed to join a new layer also increases exponentially with each additional layer. On the other hand, a layer is dropped immediately when congestion becomes apparent from a packet loss. This emulates behaviour of TCP since increase in bandwidth is proportional to the amount of time required to pass without packet loss before being allowed to join the layer. At the same time the reaction to congestion is a multiplicative decrease, since dropping one layer results in halving the overall receive rate. To improve synchronisation between receivers, RLC receivers may join a layer only at so-called synchronisation points (SP) (receivers that share the same bottleneck should be joining and leaving layers synchronously). In order to decrease the likelihood that a join experiment fails, the RLC senders create short burst periods before a SP (during these periods the data rate in each layer is doubled). Only if a receiver does nor experience any congestion during the bursts it is allowed to join the next higher layer.

Despite the improvements in the congestion control mechanism over RLM, RLC still has some drawbacks. These are discussed in more detail in [Widmer, J., Denda, R., Mauve, M., 2001], but here we address only few aspects of these relevant to the rest of the thesis. The granularity at which the bandwidth in RLC can be adapted to the network conditions is very course and may cause unfair behaviour. They argue that this is due to the exponential distribution of the layers that only allows doubling or halving of the received rate. The second problem identified by Joerg Widmer at al is that the transmitted data must support layering. While this is true for bulk data transmission, streams that are more interactive (such as speech) cannot easily be separated into multiple layers. Also, RLC does not take the round trip time into account when determining the sending rate. This can lead to unfairness toward TCP since TCP is biased against connections with high round trip time. [Widmer, J., Denda, R., Mauve, M., 2001] also argue that artificial bursts of packets introduced by RLC may not be acceptable for a broad range of applications that support layered transmission.

Fair Layered Increase/Decrease with Dynamic Layering is proposed by [Byers, J., Frumin, M., Horn, G., et al, 2000] to address some of the deficiencies of RLC. The protocol uses a Digital Fountain at the source. With Digital Fountain encoding, the sender encodes the original data and redundancy information such that receivers can decode the original data once they have received a fixed number of arbitrary but distinct packets. Since it is not necessary to ensure delivery of specific packets, the layering scheme is much more flexible. FLID-DL includes dynamic layering with which the bandwidth consumed by a layer decreases over time. Thus the receiver has to periodically join additional layers to maintain its receive rate. The receive rate is reduced simply by not joining multiple layers to reduce the total number of layers required by the mechanism. Each layer is reused after a quiet period when no data has been transmitted over it. This scheme provides an elegant solution to avoiding the effect of long leave latencies, provided that the quiet period is sufficient for the normal leave operation to take effect.

All three mechanisms, RLM, RLC and FLID-DL, are “feedback-free” mechanisms. This means that they do not take into account round-tip-time and therefore cannot exhibit fair behaviour towards TCP under many network conditions. Many multicast congestion control schemes do not determine round trip times as it is a difficult task: for example, different layers might actually take different routes to the receiver, and so experiencing different delays. Determining round trip times for each layer can result in a major overhead for the underlying multicast routing protocol as join and leave decisions occur much more frequently.

2.4.5 Discussion

This section summarizes the adaptation approaches that were reviewed in section 2.4. The discussion is structured in terms of TCP-fairness, heterogeneity and support for multiple simultaneous speakers.

There exists no technique that performs adaptation across multiple flows and which is thus particularly suitable for large numbers of simultaneously active audio sources. However, this section tries to identify the most useful points which could potentially be borrowed from the adaptation mechanisms that affect single streams and integrated when constructing a new audio service to support many simultaneous speakers.

A common approach for rate adaptation is adaptive encoding through the adjustment of codec quantisation parameters based on the state of the network. Many of these studies have not addressed inter-protocol fairness; instead they strive to improve the perceptual quality of the received traffic.

Protocols that do not address TCP-fairness are BWAS, RLM and MTP. Protocols that do take TCP-fairness into account, but do not take round trip time into account are: RLC and FLID-DL. These protocols thus result in unfairness towards TCP in situations of heavier network load. Protocols like RAP and DAA aim to be TCP-fair but do not fully achieve it since they do not take TCP timeouts into consideration. TCP timeout has a significant impact for loss rates higher that 5%. Hence RAP and DAA exhibit TCP unfair behaviour for loss rates above 5%. [Mathis, M., Semke, J., Madhavi, J., 1997]. Other TCP-friendly congestion control protocols, such as DAA and LDA+, do not have their own feedback mechanisms for estimating loss rates and delays, but rely on RTCP messages which are not frequent enough to allow sufficiently timely response to congestion.

TFRCP does not manage TCP-friendliness during underloaded situations as its increase behaviour is arbitrarily chosen. TFRC and ALS manage TCP-friendly behaviour in the largest variety of network situations. Therefore, ways of detecting and measuring packet loss, delays and timeouts in TFRC could be very beneficial if integrated in a new solution for scaleable TCP-friendly audio service for large numbers of simultaneous speakers.

All the reviewed protocols manage packet loss as they estimate it and react to it. But not all of the protocols manage heterogeneity: BWAS and MTP are only suitable for LANs, for example.

All these approaches are based on layered audio and they suffer from an inherent limitation concerning the lack of scope for layered audio adaptation already discussed in section 2.3.2.6. Moreover, in the case of very large numbers of simultaneous sources congestion and packet loss may happen on certain links even if all the streams are already at their base layer. Even highly compressed multiple audio streams (at their base layer) can require a significant percentage of the overall bandwidth available in current networks, in cases when there are many simultaneous senders. If an already compressed streams should be pushed down a low speed or congested link, and further compression is either not possible or nor practical, a way of combining the flows to cope with these severe congestion levels is necessary.

The main challenge is whether it is possible to achieve adaptive aggregation of multiple flows. There has been no attempt yet to investigate if audio streams could be mixed in a way that is adaptive (therefore support heterogeneity) and friendly towards the other traffic in the network.

This discussion is summarised in Table 2.3.

	ALS	**BWAS /MTP**	**RAP/ DAA**	**SCP**	**TFRCP**	**TFRC**	**Sender-based Multicast**	**RLM/ RLC FLID-DL**
Adaptation of a single streams	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Adaptation over multiple streams	No	No	No	No	No	No	No	No
TCP Fairness	Yes	No	Yes, but not for all cases	No	No	Yes	Some	Problem-atic/ Yes/Yes
Managing packet loss for the user	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Supporting heterogeneity (suitability for Internet)	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes

Table 2.3 Overview of the discussion of some of the rate adaptation techniques in terms of TCP fairness, packet loss management and support for heterogeneity

Table 2.3 also provides information on how these approaches affect the packet loss and the number of independently delivered streams to the end user. Since none of the adaptation mechanisms includes mixing components, all sent audio streams are delivered to the end user. Likewise, since all the adaptation techniques change their transmission rates when subject to packet loss, the packet loss at the end user is managed for all the approaches.

2.5 Discussion

This chapter has reviewed some of the most relevant techniques and approaches according to the criteria given in the section 2.1. This section summarises all the approaches reviewed in this chapter according to the list of selected criteria give in the section 2.1.1. We mainly concentrate on discussing the benefits and challenges of the approaches already highlighted in the discussion subsections at the end of the each of the previous three sections. We argue that there is no single approach that satisfies all the criteria and we need some kind of combined approach that would provide the best support for all the criteria.

In order to meet criterion 1, we choose silence suppression and open microphone strategies as the only two strategies that do not artificially limit the number of simultaneous speakers but allow transmission of the audio streams from all simultaneous speakers in the system (although silence suppression may cut off transmission of non-verbal sounds).

In order to deal with potentially very large numbers of audio streams in the network, the supporting audio service must efficiently distribute audio streams in the network (criterion 4) We recognize the usefulness of efficient distribution of a single flow, i.e. multicast and compression techniques, but we focus on the multi-flow dimension of efficiency which is especially suited for cases of very large numbers of simultaneous speakers. Accordingly we choose mixing as the only approach that affects the number of streams in the network.

Efficient distribution of audio streams is in direct conflict with criterion 3: it aims to minimize the number of audio streams in the network (by mixing) whereas criterion 3 aims to maximize the number of separate streams. The challenge is to investigate if it is possible to find a tradeoff between these two criteria.

Maintaining audio quality (criterion 2) over heterogeneous networks and end systems (criterion 4) could be achieved only through a highly adaptive scheme since in the current Internet we cannot rely on reservation services to guarantee quality of service. There are many techniques which adapt individual streams but none which can do the adaptation across multiple streams. In order to design a cross-stream adaptation scheme some mechanisms from per-flow adaptation techniques could potentially be integrated in new multi-stream adaptive solutions: e.g. TFRC's estimation of TCP rate that uses TCP-like timeout estimation, TFRC's loss and delay calculations and RAP's packet loss detection.

The model proposed in the next chapter attempts to satisfy all of the six criteria and to reconcile the challenges and conflicts in some criteria. More specifically, the model proposes a way of answering the following questions:

- How can the audio streams be mixed in a way that it is adaptive and friendly towards other traffic in the network (criterion 6), supports heterogeneous networks (criterion 4) and limits the packet loss experienced (criterion 2)?
- How can we do no more mixing than absolutely necessary, i.e. keeping the maximum possible number of independent streams (criterion 3) while having an efficient aggregation of audio streams (criterion 5)?

The next chapter proposes a technique called distributed partial mixing as a way to make the mixing of audio streams flexible, adaptive, dynamic so that it can change in response to application requirements and network conditions while maintaining the best possible trade-off between audio quality and volume of traffic.

3. Distributed Partial Mixing

This chapter presents a model for designing a scaleable and adaptable audio service for large scale collaborative applications on the Internet. An example target is a large scale collaborative application in which large numbers of audio sources and audio sinks are active at the same time such as a Collaborative Virtual Environments (CVE) deployed over heterogeneous networks and end systems.

This chapter is divided into four sections. The first section introduces the proposed approach in a very general form as it might be applied to the target environment. The second section provides functional model of the approach. The third section explores the design space and discusses examples of applying the model in various application and network scenarios. The fourth section gives conclusions and a short summary of the chapter.

3.1 Distributed Partial Mixing (DPM)

This section introduces distributed partial mixing in two steps: by first introducing (non-distributed) partial mixing, and then by giving a general overview of distributed partial mixing. In order to introduce the partial mixing a simple small-scale scenario is assumed comprising two LANs connected via a WAN connection with unknown properties. A collaborative virtual environment (CVE) application similar to those introduced in the first chapter is introduced and runs in the following scenario: a number of people from both LANs are connected to the CVE, each person having an open microphone and being able to hear the audio from the rest of the people at any time. A large-scale scenario is then considered (e.g. an Internet scenario), to introduce the concept of distributed partial mixing.

3.1.1 Partial Mixing – An Overview

Partial mixing extends the traditional concept of mixing to be more dynamic and flexible. Unlike total mixing that is based on mixing the whole set of received streams into a single output stream, partial mixing dynamically chooses to mix only a subset of the available audio streams at any given time and forwards this along with the rest of the un-mixed streams. In this way instead of producing the single output stream in all cases, partial mixing produces varying numbers of streams in different situations.

An example of how this can support adaptation to various bandwidth conditions on the neighbouring link is given in Figure 3.1. Figure 3.1 shows various resulting audio stream distributions ranging from "all forward" (peer to peer), where speakers' audio streams are received by every listener in their original form, to "all mix" (server total mix), where the streams are received as part of a new mixed stream. The figure depicts LAN_1 with a partial mixer and N users, and LAN_2 with P users. The two LANs are connected by a network link with unknown properties.

- Fig 3.1 (a) shows the "all forward" scenario in which the partial mixer receives *N* audio streams from the sources in LAN_1 and forwards all *N* received audio streams to the receivers in LAN_2. This happens when there is enough bandwidth available for extreme peer-to-peer communication on the link connecting the two LANs i.e. non-congested or a high speed link.
- Fig 3.1 (b) shows the "all mix" scenario, in which the partial mixer receives *N* audio streams from the sources in LAN_1, mixes all the received N audio streams, and transmits the resulting stream to the receivers in LAN_2. This happens when the bandwidth is scarce i.e. the link connecting the two LANs is low speed or severely congested.
- Fig 3.1 (c) shows a partial mixer that receives *N* audio streams from the sources in LAN_1, mixes a subset of *M streams* out of all *N* received audio streams into a single stream, and transmits this mixed stream along with the rest of the non-mixed streams to the receivers in LAN_2. Thus the total number

of audio data streams transmitted from LAN$_1$ to LAN$_2$ is equal to ($N - M + 1$). This happens when more than one but less than the total number of received- streams can be accommodated within the available bandwidth on the link joining the two LANs.

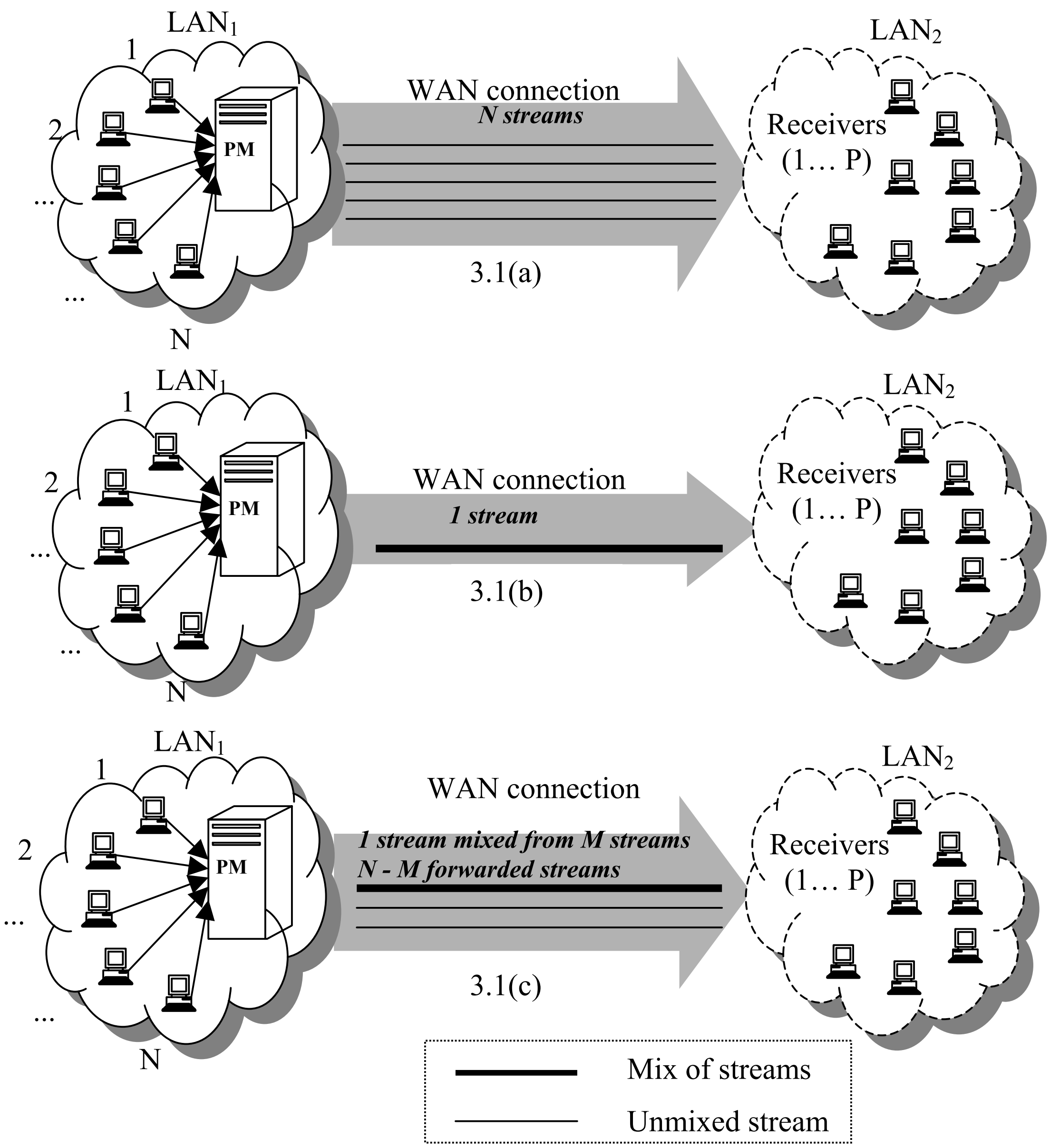


Figure 3.1 Small-scale partial mixing example

Note that a partial mixer is not limited to producing only one mixed stream. It can also produce multiple mixed streams, where each stream contains a different mix of the received streams.

The fundamental idea shown in the Figure 3.1 is simple: the transmission rate of the partial mixer in LAN_1 over the WAN link is increased during underload situations and reduced otherwise. However, as already discussed in Chapter 2, mixing reduces not only the traffic imposed on the neighbouring link but also the future flexibility of the mixed audio streams. In particular, with regard to spatialisation, once mixed the streams cannot be unmixed. The aim of partial mixing is to limit the amount of mixing performed, so that end users receive as many separate audio streams as possible within prevailing network resource constraints. In order to satisfy user and application requirements, as well as network requirements and constraints, partial mixing has to be driven by both the underlying network and end systems requirements, as well as by end users and application preferences.

Figure 3.2 illustrates that partial mixing system acts like a mechanism that produces various network distributions of audio streams as its output and has three kinds of input: the received audio streams, and application and end systems requirements and preferences and network conditions.

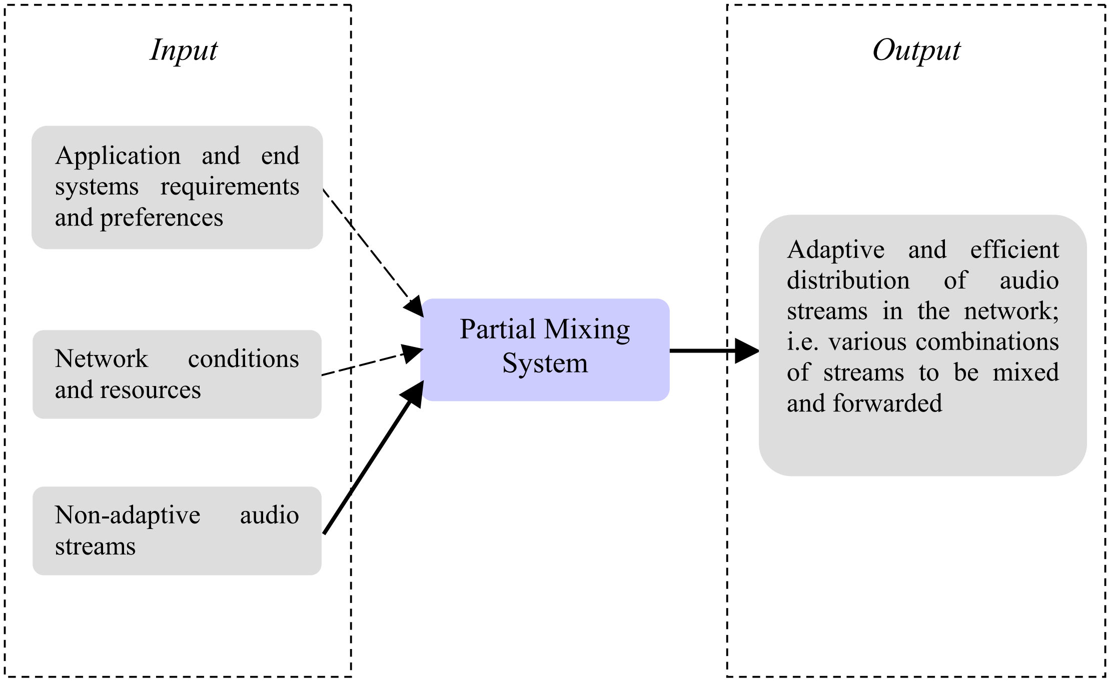

Input
Application and end systems requirements and preferences
Network conditions and resources
Non-adaptive audio streams
Partial Mixing System
Output
Adaptive and efficient distribution of audio streams in the network; i.e. various combinations of streams to be mixed and forwarded

Figure 3.2 Input and output of a partial mixing system

The received audio streams can be either non-adaptive audio streams (referring to streams that do not change their encoding in response to congestion) or adaptive audio streams (referring to streams that have reached their minimum encoding layers and cannot further decrease their bandwidth consumption).

The other two kinds of input are the controlling signals to indicate network conditions as well as application and user preferences. Since partial mixing dynamically responds to both kinds of requirements, the resulting audio stream distribution is adaptive and more efficient than the input one.

3.1.2 Distributed Partial Mixing - An Overview

Distributed partial mixing performs partial mixing at many points in the network. Distributed partial mixing is based on a distributed audio service that comprises an arbitrary graph of audio processes, interconnected by varying numbers of audio streams, as well as feedback and control data flows. These audio processes include sources, sinks and partial-mixing components, and in general there may be any number of each. As described previously each source typically transmits one or more audio streams to a “nearby” or “parent” partial mixer node. Each partial mixer may create multiple mixes from arbitrary subsets of the audio streams that it receives, and send these mixes instead of – or as well as – any of the audio streams which it might otherwise pass on to other parts of the system. At one extreme (plentiful resources, few speakers) the system performs no mixing, i.e., giving a peer-to-peer solution. At the other extreme (many speakers, very few resources) system performs total mixing.

Two kinds of graph topologies for the partial mixer placement can be distinguished: static and dynamic. Static topologies assume that the processes in the network are statically located and that there is a fixed number of them in fixed positions in the network. In contrast, dynamic topologies assume that neither the number nor the locations of the processes are static, i.e. both can potentially vary with the number of users and other network and application conditions. In this and the following chapter a static topology of partial mixers is assumed because they focus on the functional model of distributed partial mixing. Various issues of more dynamic distributed partial mixing deployment will be discussed in chapter 5.

Figure 3.3 illustrates a larger scale scenario in which there are multiple partial mixing stages between the end systems. The figure shows the streams that get mixed at different stages in the system. In order to decide which and how many streams to mix each partial mixer needs to exchange control and feedback information (describing network conditions and application preferences and end systems bottlenecks) with other partial mixers, end systems or network components. Therefore there are always two channels between the neighbouring partial mixers: the channel that contains the audio streams as well as the channel that contains the control and/or feedback information.

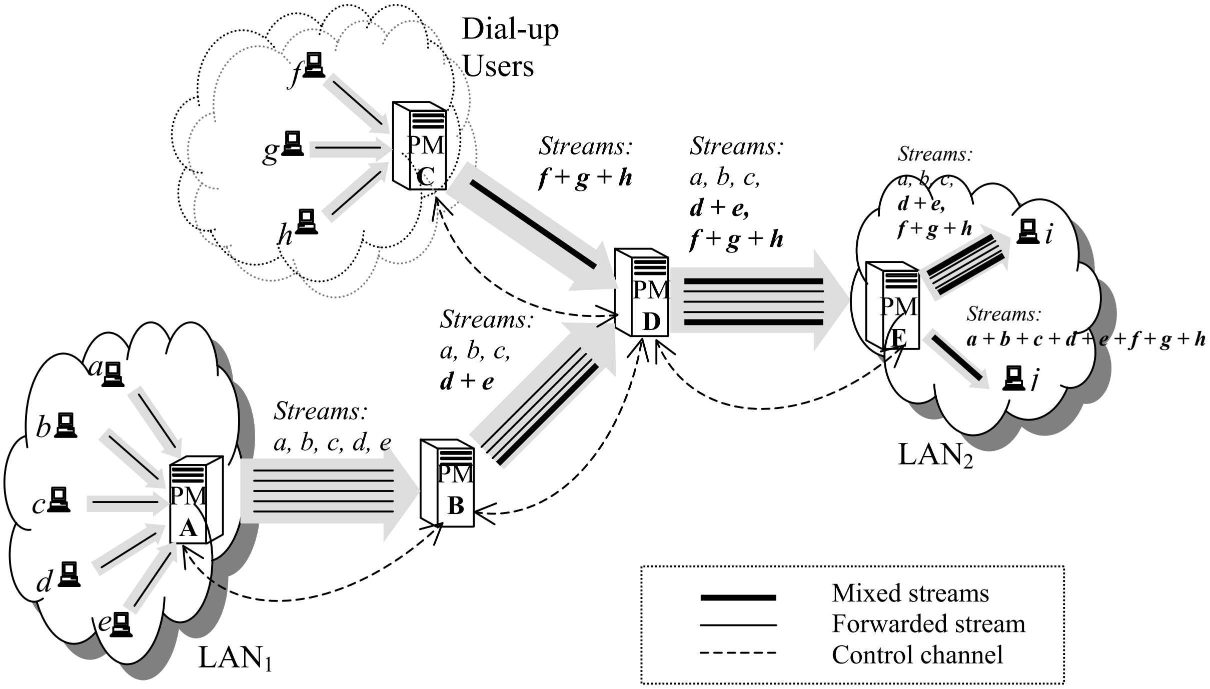


Figure 3.3 A tree of sources, sinks and (partial) mixers

The figure shows only half duplex communication of audio streams and controlling data for reasons of clarity and simplicity. However, the reader should assume that the communication is full duplex.

Figure 3.3 shows a network which contains a number of sites (LAN$_1$, LAN$_2$ and dial-up users), sources (*a, b, c, d, e, f, g, h)*, receivers (*i, j*) and partial mixers (PMs) (**A, B, C, D, E**). Senders *a, b, c d, e* are situated in LAN$_1$ and transmit audio streams to PM **A**. In this example, PM **A** receives the audio streams from senders in LAN$_1$, determines that network resources are sufficient and chooses to forward all streams to PM **B** without mixing them. PM **B** receives the audio streams from PM **A**, determines that at the current time network resources are not sufficient for total forwarding, chooses to mix streams *d* and *e*, and transmits the resulting streams to PM **D**.

Senders *f, g, h* are dial-up users and transmit audio streams to PM **C**. PM **C** determines that network resources are currently not sufficient, chooses to mix all of the incoming streams and transmits the resulting single stream to PM **D**. PM **D** receives the audio streams from PM **B** and **C**, determines that network resources are sufficient, and chooses to forward all streams to PM **E** without mixing them. PM **E** receives the audio streams from PM **D**, determines that network resources are sufficient and chooses to forward all streams to receiver *i* without mixing them. However PM **E** determines that the hardware capabilities of receiver *j* are limited to playing out only mono and chooses to mix all incoming streams and transmits the resulting single stream to receiver *j*. Of course, the decisions at PM **A**, PM **B**, PM **C** and PM **D** about which streams to mix and which to forward can change dynamically.

Being distributed can aid scalability. By distributing partial mixing nodes throughout the network as described overall audio mixing task can be shared, and mixing bottlenecks can be avoided. For example, CPU load and the number of input and output streams of a partial mixer will be reduced by serving only a subset of all users.

In this way, distributed processing provides not only support for very large numbers of simultaneous speakers in the audio application, but also enables each partial mixer to monitor and respond to changing local network conditions at critical points in heterogeneous environments such as Internet. Thus each partial mixer can cope with regional variations and transitory congestion very efficiently and limit the effect of traffic peaks from one region on the rest of the system.

Optimising local parts of the network (i.e. not treating the network as a single shared communication medium) is more appropriate than having global decision making about the whole network. While global monitoring is suited for small networks, it is not realistic to expect that a single global monitor can acquire all of the necessary information for a large network, monitor and respond to the current network conditions on all existing links in a timely manner.

3.1.3 Summary

This section introduced partial mixing and distributed partial mixing, which flexibly manage the number of audio streams according to the network limitations and user requirements in small and large scale target application environment respectively.

The next section describes the functional model of distributed partial mixing.

3.2 Functional Model

3.2.1 Introduction

This section describes a functional overview of distributed partial mixing**.** The purpose of this model is to clarify key design issues, possibilities and tradeoffs in the complex design space of distributed partial mixing. The functional model comprises the following stages:

- receiving multiple audio streams transmitted from one or more audio sources distributed in the network and monitoring network resource parameters on the neighbouring links;

- processing resource parameters to determine available resources for subsequent transmission of received audio streams to one or more nodes in the network;
- comparing available resources with resource requirements necessary for further transmission and determining the number of audio data streams to be mixed prior to transmission in response to said comparison;
- selecting the streams to be mixed based on application and user requirements and capabilities;
- mixing or forwarding audio streams towards the next network node or end system; and
- transmitting audio streams to the next node and sending feedback and control signals.

Figure 3.4 overviews the functional model of distributed partial mixing.

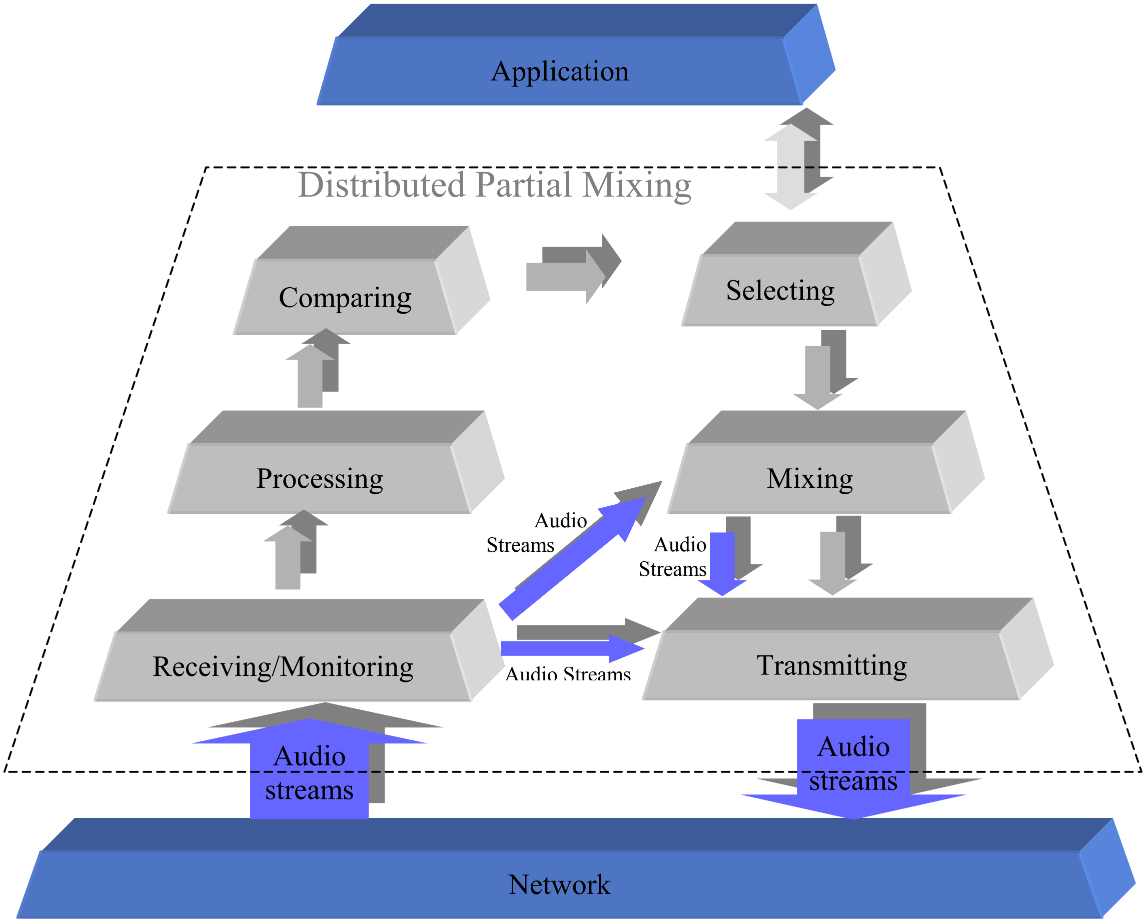


Figure 3.4 Functional model of Distributed Partial Mixing

The next section describes in more detail each of these functional stages.

3.2.2 Functional Stages

This section describes each of the functional stages of the model, and for each stage discusses a number of factors that can be addressed and possible mechanisms that can be used.

3.2.2.1 Receiving audio streams and monitoring network resources

Audio streams can be received using various mechanisms, i.e. unicast, multicast or layered multicast.

Monitoring network resources refers to monitoring network parameters of the links connecting the partial mixing unit to the other parts of the network. The parameters monitored could include:

- available bandwidth,
- packet loss rates,
- delays (round trip times).

Depending on the type of the network where distributed partial mixing is deployed, network monitoring can utilise various mechanisms.

In best effort networks, network monitoring includes exchanging control messages with neighbouring partial mixing units and end systems, or utilising network protocols like Real Time Control Protocol (RTCP) or Simple Network Management Protocol (SNMP) in order to deduce network parameters.

In integrated services or intelligent networks, network monitoring includes polling the network components for these network quality parameters.

3.2.2.2 Processing monitored resource parameters

Processing monitored parameters aims to provide an estimate of the available resources for transmitting the received data streams to the next relevant node or nodes in the network. A variety of criteria can be used, depending on the monitored parameter(s). The processing algorithms differ depending on the parameter(s) chosen and depend on the current network situation. For example, availability of the resources could be:

- predetermined, e.g. administratively set number of multicast channels on a certain link, with guaranteed QoS based on resource reservation,
- calculated to satisfy certain network criteria e.g. responsiveness to network congestion, fairness towards TCP traffic.

These examples are discussed in greater detail in section 3.3.1.

3.2.2.3 Comparing available and required resources

The difference between the available and the required resources determines the number of streams that must be mixed. Ideal resources are determined from the bandwidth consumption of the received audio streams. If the available resources are sufficient, all of the received streams are transmitted onwards as received. If the available resources are not sufficient, then the number of streams to be mixed is determined to fit in the available resources.

3.2.2.4 Selecting the streams to be mixed

If the resources are not sufficient and a certain number of streams needs to be mixed, then both application and user preferences can be considered when selecting which streams to mix in order to maximize user experience. Some of the factors that might affect the choice of streams to be mixed (or forwarded) include:

- user-specific mixing preferences,
- receiver and sender requirements,

- end system resource capabilities i.e. audio data processing and playback capabilities,
- predetermined-criteria, e.g. current tariff data,
- patterns of users' activity,
- content of audio streams,
- stability considerations.

The definition of some of these factors and the corresponding mixing criteria for choosing streams, is highly dependent on the application type. Section 3.3.3 discusses in more detail some of these factors and gives some examples of the choices that could be made for the case of collaborative virtual environments and CSCW applications (e.g. Inhabited TV).

3.2.2.5 Mixing and/or Forwarding

Once the total number of streams to be sent is determined and the appropriate streams to be mixed are selected, mixing and/or forwarding of the corresponding streams is performed.

3.2.2.6 Transmitting and sending feedback and/or control information

Audio streams can be transmitted using various mechanisms e.g. unicast, multicast or layered multicast. The feedback and/or control information can be sent using unicast or multicast and include packet loss reports, packet acknowledgements, RTT reports.

3.2.3 Summary

This section has presented a functional model of distributed partial mixing. A summary of functional stages and some key factors that can be considered and mechanisms that can be used is given in Table 3.1.

Functional stage	**Factors**
Receiving audio streams, Monitoring network	Unicast, multicast, layered multicast Packet loss, RTT using RTCP, RSVP, SNMP
Processing resource parameters	Fairness, responsiveness, predetermined conditions
Comparing	Comparing available and required resources
Selecting	Application and end system preferences and requirements: predetermined criteria, receiver requirements, source requirements, patterns of activity, content of audio streams, voice characteristics, stability considerations. End –systems processing capabilities
Mixing	Mixing, Forwarding
Transmitting audio streams, feedback and control data	Unicast, multicast, layered multicast, packet loss, RTT, packet acknowledgements

Table 3.1 Factors to be considered for each functional stage

3.3 Mixing Criteria Example Space

The single most distinctive aspect of the design is the mixing of a subset of incoming streams. The choice of streams to be mixed – which and how many –determines the effectiveness of distributed partial mixing. This section gives examples of mixing criteria that might be used to dynamically determine how many and which streams should be mixed. Mixing criteria can be divided into three main groups:

- network-driven,
- end-system-capability driven
- application-driven,

A number of examples for each kind of criterion are given in the following three sections.

3.3.1 Mixing criteria driven by network

3.3.1.1 Acceptable packet loss

This criterion considers packet loss rate experienced as a network resource parameter. It determines the number of streams to be mixed so as to maintain packet loss below some application dependent threshold.

3.3.1.2 Congestion Control

This criterion considers available network bandwidth for transmission of audio data streams to one or more receivers as a network resource parameter. It determines the maximum number of audio streams that can be transferred without causing excessive congestion in the network.

3.3.1.3 Fairness

This criterion also considers available network bandwidth as a network resource parameter. It determines what portion of the link bandwidth (number of audio streams) DPM should use (transmit), when coexisting on the same link with other applications in order to be fair, and restricts it to that fair share only.

3.3.1.4 Network link limitation

This criterion also concerns available network bandwidth as a network resource parameter. It determines the available bandwidth according to network limitations that could be administratively imposed on a network link.

An example is a partial mixer that receives four audio streams but has a preset limit of only three multicast groups for sending on a certain network link. At least one pair of audio streams will be mixed prior to subsequent transmission by the respective partial mixer.

3.3.2 Mixing criteria driven by end system capabilities

3.3.2.1 Receiver processing characteristics

Mixing decisions can be determined according to the characteristics of the respective receivers. In this way separate audio streams might otherwise be sent to a receiver with low processing capability or capacity can be mixed in the network so that the number of audio streams to be processed by receiver is reduced. For example, a receiver may comprise a full 3-D audio system capable of re-creating fully spatialised studio quality audio where mixing considerations are important for recreating spatialised audio. In this case, the amount of mixing streams must be minimised. Alternatively a receiver may comprise a simple mono audio system where mixing considerations are much less important and all of the streams can be mixed.

3.3.3 Mixing criteria driven by application and user

3.3.3.1 Pre-determined criteria

Audio data streams can be selected for mixing according to predetermined criteria. For example, the allocation or reservation of bandwidth may be controlled by different charging tariffs associated with the quality of service required. In this way a user may specify a quality of service requirement of say 3 x 64kb/s audio channels in which case selected audio streams will be mixed in the network when more than three separate audio streams are to be transmitted by the network. In this case available bandwidth may be considered as allocated or reserved bandwidth for use according to user specified quality of service requirements**.**

3.3.3.2 Content of audio streams

Audio streams can be mixed according to audio stream content. For instance, in a virtual environment one or more audio streams may be more significant in terms of audio content than the others. The user’s experience might be better maintained by mixing less significant audio streams in preference to the more significant ones.

Some real-time CSCW applications assign participants different roles within an event. For example, early experiments in inhabited television have differentiated between performers, inhabitants and viewers [Greenhalgh, C., Benford, S., Taylor, I., et al, 1999]. Performers are part of the core content of an on-line TV show.

Inhabitants are active within the virtual world, but are typically less central to the content, for example, forming an on-line audience. Viewers are more passive; they typically receive a broadcast mix created by a director. These roles are complemented by differences in the technologies used to access the event. Performers may use professional studio-quality equipment, with fully spatialised 3D audio. Inhabitants may use commodity PCs, equipped with headphones. Viewers may use conventional television sets, equipped with surround-sound systems. Inhabited television has coined the term “layers of participation” to describe this fusion of role and technical capability. Roles or layers of participation can drive mixing policy. It may be appropriate to ensure that performers are heard with the maximum possible audio quality. As a result, as congestion increases, the audio streams for inhabitants might be mixed together first, with the performers streams being kept separate for as long as possible. Other CSCW applications might also benefit from defining layers of participation and using these to help prioritise audio sources during distributed partial mixing.

3.3.3.3 Voice characteristics

The timbre of voices (or other audio sources) can be taken into account when selecting the streams to be mixed. It may be better to mix a high and a low voice into a single stream, so that a listener can separate them in the overall mix even though they are no longer spatially separated.

3.3.3.4 Patterns of activity

Statistical analysis of participants’ activities could inform the selection process of the streams to be mixed. For example, participants whose speech is observed to rarely overlap could be mixed together (this assumes that silence suppression is not used because this mixing would yield no reduction in bandwidth if it were used). The information required to support this can be provided by techniques for logging and analysing patterns of activity, as are already being used as part of the evaluation of

CVEs (e.g., [Frecon, E., Greenhalgh, C., Stenius, M., 1999], [Greenhalgh, C., Benford, S., Craven, M., 1999]).

3.3.3.5 Receiver requirements

The selection of streams to be mixed can be determined by the receiver's own requirements, for instance the extent of audio spatialisation required.

For example, an active inhabitant in an Inhabited TV show may benefit from and require fully spatialised audio that provides cues to support navigation and conversation management. A passive viewer with a surround-sound system may benefit from and require a mix that clearly separates the key performers, but where their accurate location in the world is less important. In the first case it may be important to maintain the separation of streams from nearby participants, whereas in the latter, it may be appropriate to maintain the separation of key performers.

3.3.3.6 Source requirements

Audio streams can be mixed according to the audio stream sources requirements. Hence, audio streams from related sources could be mixed such as a particular group of participants in an audio conference or virtual environment.

CSCW applications often group participants in some way. For example, shared windowing systems can group people according to the window or shared-tool that they are currently using, and CVEs typically group people according to the world or a region that they are currently in. It will often make sense to mix together the audio streams from one coherent group. For example, avatars in a CVE may have clustered into definable and separate groups. Each group could be mixed to a single stream that could be spatialised to the average position of the group as a whole, maintaining an approximate spatialisation of the world. This would minimise the loss of spatialisation due to mixing. Some CSCW applications calculate levels of mutual awareness among participants, and these might provide a more dynamic basis for grouping them as part of the mixing process (e.g. Rodden's proposal for a computational framework for awareness that can be applied to a wide range of CSCW applications [Rodden, T., 1996]).

3.3.3.7 Stability considerations

The current and past states of the system can be taken into consideration when selecting streams for mixing. For example, frequent transition from one choice of mixed streams to another may be noticeable to users, and potentially undesirable. In this case, the system needs to be optimised in order to minimise transitions frequency and produce a relatively stable mix.

3.3.4 Summary

This section has discussed various factors that might affect mixing decisions in distributed partial mixing. Examples of how each factor could be applied to certain application and network situation have been provided. Mixing criteria and factors considered are summarized in the classification in Figure 3.5.

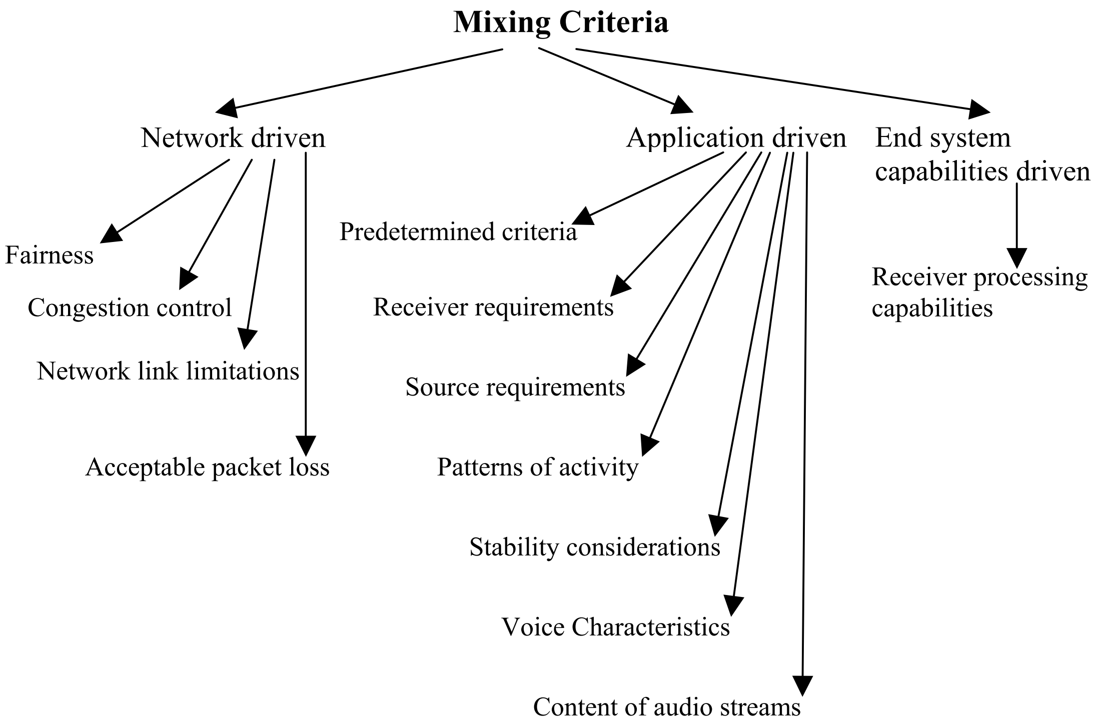


Figure 3.5 Scheme of various types of mixing criteria

3.4 Summary

This chapter has introduced distributed partial mixing as a novel method for efficient distribution of audio streams through flexible management of the number of the audio streams in the network. In the first part of the chapter, a large scale collaborative virtual environment where large numbers of users could be simultaneous active in the audio medium was proposed as a target environment where the proposed method could be utilised.

A simple example scenario was provided to illustrate partial mixing functionality in a small scale scenario where multiple users from two different LANs communicate over a single link with unknown properties that connects the two LANs. Then, a larger scale scenario with multiple heterogeneous networks and users was provided to illustrate the high level architectural overview of distributed partial mixing. In distributed partial mixing, partial mixing can be performed at various points in the networks, therefore enabling rate and quality adaptation between each two partial mixing points.

The second part of the chapter described the functional model of distributed partial mixing and identified the parameters and factors to be considered for each of its functional stages. Various network parameters may be monitored and processed in order to estimate available network resources. Comparing the available and required resources determines the number of forwarded and mixed audio streams that will fit in the available bandwidth on the link(s). In this way the number of data streams transmitted can be controlled so that the network traffic can be optimised according to available network resources. This aspect of the distributed partial mixing is particularly relevant for dynamic applications involving varying numbers of active participants engaged in various activities and running over dynamic networks where congestion and delay may change quite rapidly.

As the choice of streams to be mixed can affect the effectiveness of distributed partial mixing, the third part of the chapter provided an example design space for multiple algorithms that could be used for determining which and how many streams to mix. Mixing criteria were divided into three main categories depending on what they are based on: various aspects of the underlying networks, various application and user quality requirements, or end system capabilities, and examples provided for each of them. Selecting the streams is highly specific to the application and the remainder of the thesis focuses mainly on the networking aspects of distributed partial mixing.

The next chapter describes a particular implementation of network-driven distributed partial mixing, and demonstrates of its effectiveness in terms of congestion control, TCP fairness, managing packet loss and maximising the number of independent audio streams delivered to the end user.

4. Implementation and Evaluation

4.1 Introduction

Chapter 3 introduced the model that forms the central part of the work presented in this thesis. It detailed the functional structure of the model and provided examples of a variety of criteria that could drive the mixing decision making process. This chapter describes an implementation and demonstration of the model driven by two network criteria: congestion control and fairness towards TCP.

This chapter discuses and verifies the adaptation and fairness of network-driven distributed partial mixing in a variety of scenarios on an unmanaged single network link. The following chapter talks about issues concerning large-scale deployment of distributed partial mixing and proposes a model solution for placing and setting up partial mixers in various networks.

Section 4.2 briefly describes a scenario and infrastructure for building a small-scale demonstration system that incorporates distributed partial mixing. Section 4.3 explains the design choices for the various functional components of distributed partial mixing concerned with network driven mixing criteria. For each functional component the chosen approach is justified and alternative choices discussed. Section 4.4 describes in greater detail MASSIVE-3 and its audio service, within which DPM has been prototyped. It gives an overview of a small scale DPM system as built in MASSIVE-3. Sections 4.5 and 4.6 describe experiments that evaluate DPM against a wide range of network conditions. These include observing steady state characteristics and dynamics of the built DPM system against both non-adaptive traffic and adaptive traffic. Section 4.7 summarises the work presented in this chapter.

4.2 Scenario and Infrastructure

4.2.1 General scenario

This section outlines the general scenario and infrastructure of the experiments used to demonstrate the effectiveness of distributed partial mixing with respect to network friendliness. A single unmanaged network link shared by both adaptive and non-adaptive traffic is chosen to validate adaptation of distributed partial mixing. Even though simple, this scenario is representative of the most common scenarios in today's Internet (where the traffic conditions themselves can be quite complicated and subtle [Paxson, V., Floyd, S., 1997]). It is therefore sufficient to test and validate the proposed approach in terms of all six criteria set out in the section 2.1.

The scenario assumes two LANs (generally congestion-free and high bandwidth networks) connected via a lower-bandwidth shared unmanaged WAN link (typically an order of magnitude smaller network capacity), which is therefore prone to congestion. A number of users are introduced on each LAN. A single distributed partial mixer is started (when required) on each LAN. Additional non-adaptive traffic is offered to the WAN link and multiple experiments are run to observe the steady state behaviour of distributed partial mixing. Adaptive traffic is also offered to the WAN link, in order to observe the dynamic behaviour of DPM in multiple experiments. This scenario is illustrated in Figure 4.1.

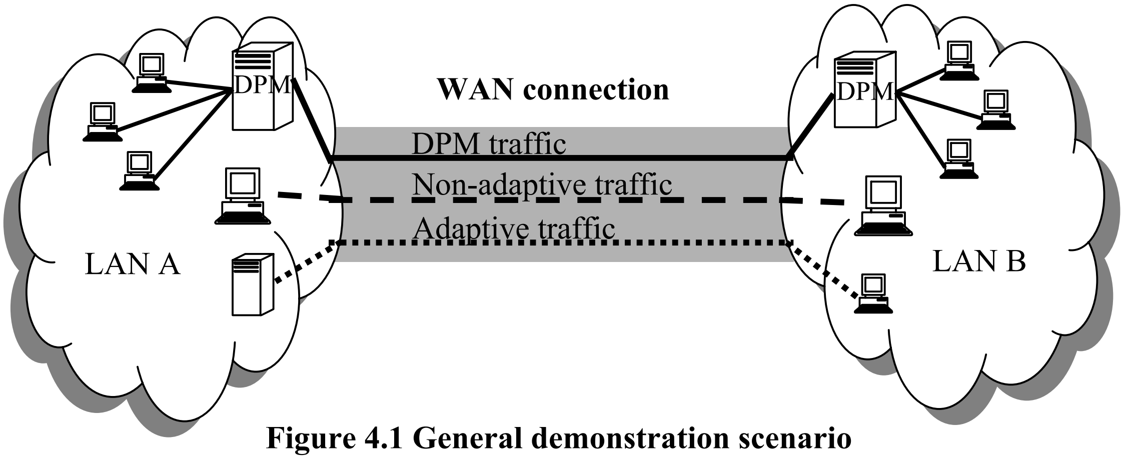


Figure 4.1 General demonstration scenario

4.2.2 WAN Emulation

The '*Dummynet'* tool [Rizzo, L., 1997] is used to emulate a bandwidth-limited WAN connection within the local network. It was originally designed for testing networking protocols, and since then has been used for bandwidth management.

It simulates and enforces queue and bandwidth limitations, delays, packet losses, and multipath effects. It can also implement a variant of Weighted Fair Queueing called WF2Q+. It can be used on users' workstations, or on FreeBSD machines acting as routers or bridges.

Dummynet works by intercepting packets (selected by *ipfw* rules - *ipfw* is one of the FreeBSD firewalls) on their way through the network protocol stack, and passing them through one or more queues and pipes, which simulate the effects of bandwidth limitations, propagation delays, bounded-size queues, packet losses, and multipath. Pipes are fixed-bandwidth channels. Queues represent queues of packets, associated with a weight, which share the bandwidth of the pipe they are connected to in proportion to their weight. Each pipe and queue can be configured separately, so that different limitations and delays can be applied to different traffic according to the ipfw rules (e.g. selecting on protocols, addresses and ports ranges, interfaces, etc.).

4.2.3 Processing results

All network traffic is captured, and records of all network activity at end systems and distributed partial mixing hosts is logged to files using the *tcpdump* tool. Packet loss and round trip times estimated by each DPM server are dumped to log files. Additional application and user information such as users' port numbers, and times when certain application and user events occur are also dumped to log files.

Tcpdump files are first processed to identify and classify individual traffic flows (i.e. tuples featuring DPM, adaptive and non-adaptive traffic going from a certain IP address and port number to another IP address and port number) over a given link.

The amount of traffic per second is measured for each flow. For the DPM flows, the number of audio streams in transit is also calculated. These files are then processed together with the log files to produce averaged performance information about DPM behaviour (levels of packet loss and delays experienced by the audio streams in transit) over different stages of traffic conditions on the link. All processing tools were built by the author in form of *perl* scripts.

4.3 Design Choices

This section describes the particular design choices made in this implementation. First the partial mixer architecture is specified. Then the design choices made for each partial mixer component are described and justified.

4.3.1 Overview of Distributed Partial Mixing Realisation

A process that performs partial mixing as described in chapter 3 is called a partial mixer. Each partial mixer acts both as a source i.e. sending the audio streams (partial mixer source), and as a receiver, i.e. sending feedback information (partial mixer receiver).

In order to achieve congestion control, fairness and to deliver maximum number of streams on an unmanaged network link, each partial mixer must contain four components:

- Congestion monitor
- Rate adaptor
- Selector and database
- Mixer and forwarder

These components are illustrated in Figure 4.2.

The congestion monitor monitors congestion on the adjacent network links and sends control messages to other partial mixers in the system. In this prototype, a partial mixing source sends audio data packets with sequence numbers, and a partial mixing sink acknowledges each packet that it receives, providing end-to-end feedback. Distributed partial mixing is source-driven: the source uses the feedback information to detect losses, calculate loss rates over, estimate the round-trip-time (RTT), and perform adequate rate adaptation. Increased packet loss rates and RTTs are considered to be main indications of congestion (and thus available bandwidth). Control messages comprise feedback information concerning congestion on the neighbouring link. The monitored values are fed to the rate adaptor of the partial mixer.

The rate adaptor processes the monitored parameters in order to determine the available bandwidth on a link and the appropriate transmission rate to accommodate the maximum number of streams within that bandwidth. The number of streams allowed is then fed to the selector, and these to the mixer and forwarder.

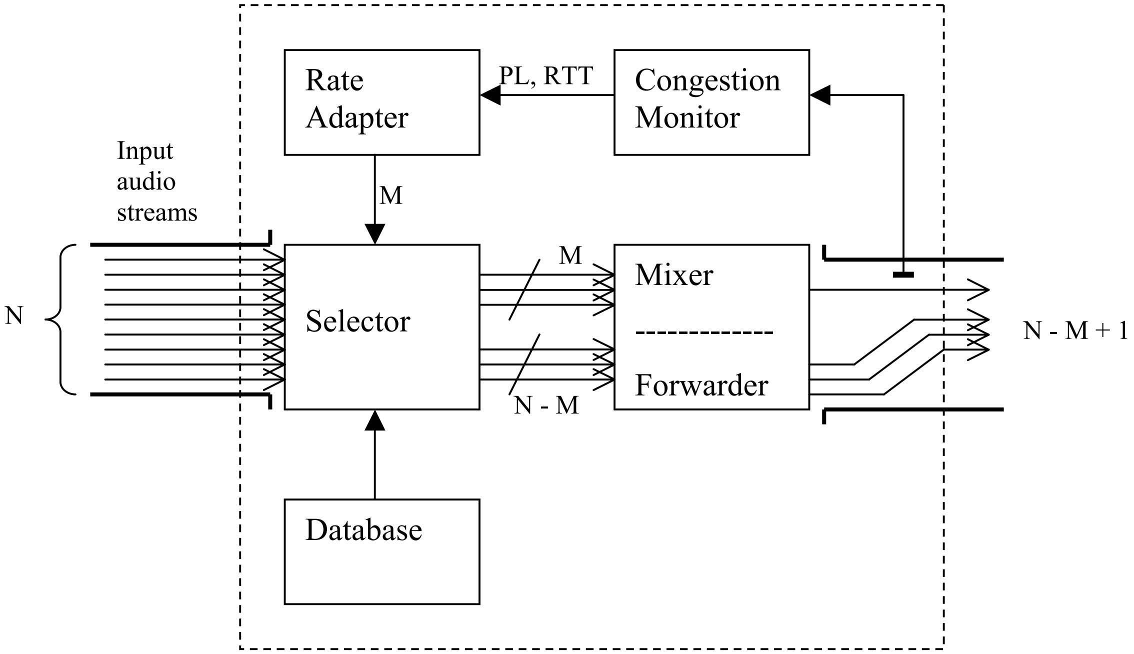


Figure 4.2 Architectural overview of distributed partial mixing

In this design, the selector uses a random policy to choose exactly which streams should be mixed. For distributed partial mixing implementations that also want to maximise user experience within the available resources more sophisticated selection policies would be needed in the selector. The selector would also use the database information to choose which streams should be mixed, according to application and end system criteria as described in chapter 3.

Since the problems of rate adaptation and the selection process in distributed partial mixing are orthogonal, the design and implementation of the two processes can carry on independently. As already discussed in chapter 3, the work of the selector and database is highly application specific and since this thesis is concerned mainly with the networking aspects of distributed partial mixing, a more sophisticated design of the selector and database is not included in this implementation.

The mixer/forwarder mixes/forwards the incoming audio streams as required and transmits any of the unmixed streams, together with the mixed stream, to the next partial mixer or client.

The following sections describe the chosen design for the congestion monitor (which monitors packet loss rates, RTTs and sends feedback information to the neighbouring nodes), and the rate adaptor (which determines the maximum sending rate according to the congestion control and fairness criteria).

4.3.2 Monitoring and estimating packet loss rates

4.3.2.1 Introduction

When choosing the method for packet loss detection, it is important to choose a method that detects packet losses as early as possible and as accurately as possible. Incorrect detection of failure to deliver a packet or late packet delivery can lead to incorrect packet loss estimation, poor rate adaptation, and therefore unresponsive and unfair behaviour.

Calculating packet loss rates can be done over various lengths of measurement intervals. In general, shorter intervals result in more responsive behaviour but they are more susceptible to noise in the packet loss signal. Longer intervals result in smoother packet loss signal but less responsive behaviour. It is important that the interval achieves reasonable balance between resilience to noise and responding quickly to real changes in network conditions.

In order to guarantee sufficient responsiveness to congestion and preserve smoothness, methods for detecting and calculating packet loss must be carefully chosen. Questions posed and answered in this section are:

a) *What mechanism will be used for packet loss detection?*
b) *What algorithm will be used for packet loss rate calculation?*
c) *Where will packet loss detection and calculation happen?*

4.3.2.2 What mechanism will be used for packet loss detection?

Chosen approach

Distributed partial mixing uses a TCP-like timeout-based mechanism to detect packet loss. All sent packets are marked with consecutive sequence numbers. When a packet is sent a timeout value for this packet is computed and an entry containing the sequence number and the timeout value is inserted into a list and kept there until the packet delivery is acknowledged or the timeout expires. If the timeout expires before the packet is acknowledged, the corresponding packet is considered to be lost.

In order to adapt to varying and unpredictable network conditions, the timeout is not fixed, but computed based on the algorithm for TCP timeout computation [Paxson, V., Allman, M., 2000]. This approach can be summarized in the following steps:

Before the first packet is acknowledged and RTT measurement is made, the sender sets the TIMEOUT to a certain initial value. This value is usually 2.5 – 3 seconds for TCP. For DPM, the timeout value should be set to the tolerable delay for interactive conferencing of approximately 0.5 seconds (as recommended by [Brady, P., 1971]).

When the first RTT measurement is taken the sender sets the smoothed RTT (SRTT), RTT variance (RTTVAR) and TIMEOUT in the following way:

$$SRTT = RTT$$

$$RTTVAR = RTT/2$$

$$TIMEOUT = Mu*SRTT + 4*RTTVAR,$$

Where *Mu* is a constant, which in this implementation is 1.08 (determined after extensive experimentation).

When subsequent *RTT* measurements are made the sender sets the *RTTVAR, SRTT, TIMEOUT* in the following way:

$$RTTVAR = (1 - ¼) * RTTVAR + ¼ * |SRTT - RTT|$$

$$SRTT = (1 - 1/8)* SRTT + 1/8 * RTT$$

$$TIMEOUT =Mu*SRTT + 4*RTTVAR$$

Other approaches

Besides timeout-based techniques, there are also gap-based techniques that can be used for packet loss detection.

Gap-based techniques are based on identifying a gap in the sequence numbers of the received packets (i.e. the packet here refers to audio packets or acknowledgement packets). When a gap in the sequence numbers of the packets is detected, it is assumed that the packets with missing sequence numbers are lost. For example, the TFRC receiver [Floyd, S., Handley, M., Padhye, J., et al, 2000a] can notice lost packets by detecting errors in the order of sequence numbers. The protocol waits for three more packets to arrive after receiving an out of order sequence number. If all the sequence numbers are higher than the missing sequence number, the packet with the missing sequence number is assumed lost and the current estimate of the loss rate is adjusted accordingly. Variations of this technique allow different levels of packet reordering. However, extreme packet reordering (out of order packet delivery) is uncommon for most computer networks, and it is considered safer to be conservative and reduce the sending rate under these circumstances.

Justification

Timeout-based packet loss detection allows more consistent and timely packet loss detection compared to identifying gaps in the packet sequence numbers. Identifying a gap in packet sequence numbers requires that one or more subsequent packets be delivered successfully to detect a packet loss. The timeout-based approach relies only on the pre-computed timeout value for each single packet. This enables timer-based packet loss detection to precisely detect the losses even at 100% packet loss rates. In these cases, the gap-based techniques cannot determine the packet loss as they will forever continue to wait for the following packets.

4.3.2.3 What algorithm will be used for packet loss rate calculation?

Chosen approach

Distributed partial mixing uses the Weighted Loss Interval Average (WLIA) approach for computing the packet loss. This approach was first introduced in [Floyd, S., Handley, M., Padhye, J., et al, 2000]. It relies on using loss events and loss intervals for correct computation of packet loss rate and is in accordance with how TCP performs packet loss calculation. A loss event is defined as a number of packets lost within a single RTT. The number of packets between two consecutive loss events defines a loss interval. The more detailed proof and analysis of this method can be found in [Floyd, S., Handley, M., Padhye, J., et al, 2000a], but here we describe several aspects of it that are of direct relevance to the performance of distributed partial mixing.

The method takes a weighted average of the last *n* loss intervals, with equal weights for the most reset *n/2* intervals and smaller weights for the older intervals. The average loss interval ŝ is calculated as follows:

$$\hat{s} = \frac{\sum_{i=1}^{n} w_{\mathrm{i}} s_i}{\sum_{i=1}^{n} w_i}$$

where weights w_i are defined as follows:

$$w_i = 1,\ 1 \le i \le n/2 \text{, and } w_i = 1 - \frac{i - n/2}{n/2 + 1},\ n/2 < i \le n$$

and s_i is the number of packets in the *i*-th most resent loss interval.

The most recent interval (s_0) contains the packets that have arrived since the last loss. It is different from all other intervals since it is not terminated by a loss, and it is important to determine whether or not to include it in the calculations of packet loss. The authors of this method recommend that this interval should be ignored in calculating the average loss interval unless it is large enough that including it would increase the average. They argue that this allows the calculated loss interval to track smoothly in an environment with a stable loss event rate. In order to formally determine whether to include s_0, the method also calculates $\hat{s}_{\mathrm{new}}$, which is the average loss interval over intervals s_0 to s_{n-1}, rather than over s_1 to s_n.

$$\hat{s}_{\mathrm{new}} = \frac{\sum_{i=0}^{n-1} w_{\mathrm{i+1}} s_i}{\sum_{i=1}^{n} w_i}$$

In order to include s_0 only at the correct times, the value used for the average loss interval is

$$\max(\hat{s}, \hat{s}_{\mathrm{new}})$$

The value *n* for the number of loss intervals used in calculating the loss event rate determines the speed in responding to changes in the level of congestion i.e. the sensitivity to noise of the calculated loss rate depends directly on the choice of *n*. [Floyd, S., Handley, M., Padhye, J., et al, 2000a] argue that values of *n* significantly greater than 8 should not be used for traffic that might compete in the global Internet with TCP, and that a value of 8, with the most recent four samples equally weighted, is the lower bound that still achieves a reasonable balance between resilience to noise and responding quickly to real changes in network conditions [Floyd, S., Handley, M., Padhye, J., et al, 2000]. For $n = 8$, the calculated weights are: w_1, w_2, w_3, $w_4 = 1$; $w_5 = 0.8$, $w_6 = 0.6$, $w_7 = 0.4$, and $w_8 = 0.2$.

The Weighted Loss Interval Average approach also employs history discounting to allow a more timely response to a sudden decrease in congestion. History discounting is used after the identification of a particularly long interval since the last dropped packet in order to smoothly discount the weight given to older loss intervals. History discounting is described in more detail in [Floyd, S., Handley, M., Padhye, J., et al, 2000a]. Note that without some form of history discounting the method would have rapid response only to increases in the congestion but be slow to respond when congestion decreases.

If $s_0 > \hat{s}_{i\geq 1}$ then the most recent loss interval s_0 is considerably longer than the recent average, and the weights for the older loss intervals are discounted correspondingly using the following formula also proposed by Floyd et al in [Floyd, S., Handley, M., Padhye, J., et al, 2000a].

$$d_i = \max\left(0.5, \frac{2\hat{s}_{(i\geq 1)}}{s_0}\right), \text{ for i} > 0,$$

$d_0=1$.

The discount factor has a lower bound of 0.5 in order to ensure that past losses will never be completely forgotten, regardless of the number of packet arrivals since the last loss.

History discounting gives the estimated loss interval of:

$$\hat{s} = \frac{\sum_{i=0}^{n-1} d_i w_{i+1} s_i}{\sum_{i=1}^{n} d_{i-1} w_i}$$

When loss occurs and the old interval s_0 is shifted to s_1, then the discount factors are also shifted, so that once an interval is discounted, it is never un-discounted, and its discount factor never increased. In normal operation, in the absence of history discounting, $d_i=1$ for all values of *i*.

Other approaches

The simplest approach to estimating packet loss is to use a static measurement interval. These methods estimate packet loss rates by calculating the number of lost packets over a fixed number of received packets using a sliding window technique. More specifically, packet loss rate refers to the rate loss fraction calculated by dividing the number of packets that were lost by the number of packets transmitted.

Other more dynamic approaches include: the Dynamic History Window method, that uses a history window of packet, with window length determined by the current transmission rate; the EWMA Loss Interval method, that uses an exponentially moving average of the number of packets between loss events; the Average Loss Interval method, that computes a weighted average of the loss rate over the last *n* loss intervals, with equal weights on each of the most recent *n/2* intervals.

Justification

The primary reason for measuring loss event rates rather than loss rates is that this is much more consistent with the way TCP responds to loss (therefore it models the best behaviour of conformant TCP implementations) while at the same time being relatively stable and resilient to noise. As with loss event rates, losses that follow an initial loss within a round trip time are explicitly ignored, this models a TCP implementation, which reduces its window at most once for congestion notification in one window of data. Most TCP versions halve the window at most once when multiple packets are lost within one window i.e. Tahoe, NewReno, Sack TCP. Only Reno TCP reduces its congestion widow twice. [Floyd, S., Handley, M., Padhye, J., 2000] argue, that depending on the router mechanism in use, the difference between loss event rates and loss rates can be more or less significant: if routers in the network use RED queue management, where multiple packet drops in a window of data are not very common, then the difference between loss rates and loss event rates is not very significant. They also argue that with Drop-Tail queue management, which is predominant in today's networks and where it is common for multiple packets to be lost when the queue overflows, the difference between the loss fraction and the loss event rate of a flow can be significant. The implementation's use of loss event rates can better model TCP's behaviour under these circumstances.

In [Floyd, S., Handley, M., Padhye, J., et al, 2000] the difference between the loss-event fraction and the regular loss rate in the presence of random packet loss is explored in detail. It shows that for high and low loss environments the difference between the two approaches is very small. However, for a stable steady-state moderate packet loss rate this difference is significant. Loss event rates can allow for more fine-grained changes in the loss estimate compared to the loss rates. Fine-grained changes are very important for stability and smoothness of any audio application that uses distributed partial mixing.

A static-window approach is not flexible enough for a wide range of network conditions because it does not change its window size. Having a large window means it reacts more slowly to the changes in the network, while having a smaller window means it reacts more rapidly to the changes, and depending on the congestion levels in the network different window sizes might be needed. Therefore, it is highly recommended that the size of the loss window should be dynamically adjusted.

Having a more dynamic method (than static window) does not necessarily produce much better results. [Floyd, S., Handley, M., Padhye, J., et al, 2000] identified some of the major flaws that can be summarised as follows. The Dynamic History Window method suffers from the effect that even with a perfectly periodic loss pattern, loss events entering and leaving the window cause changes to the measured loss rate, and hence add unnecessary noise to the loss signal. The EWMA Loss Interval method performs better than the Dynamic History Window method, However it is difficult to choose an EWMA weight that responds sufficiently promptly to loss events in several successive round-trip times, and at the same time does not over-emphasise the most recent loss interval. The Average Loss Interval (ALI) method has the best performance, while giving equal weights to the most recent loss intervals. Its disadvantage is that, while it responds reasonably rapidly to a sudden increase in congestion, it is slow to response to a sudden decrease in congestion. This is due to the fact that the ALI method averages over a number of loss intervals rather than over a number of packet arrivals and does not take into consideration the decrease in loss represented by a large interval since the last loss event. More detailed analysis and comparison of the performance of the three methods can be found in [Padhye, J., Kurose, J., Towsley, D., et al, 1999] which showed that Average Loss Interval method results in much smoother throughput. The use of the Weighted Loss Intervals method reduces the sudden changes in the calculated loss rate that could result from unrepresentative loss intervals leaving the set of loss intervals used to calculate the loss rate.

4.3.2.3 Where will packet loss detection and calculation happen?

Chosen Approach

The distributed partial mixing prototype performs packet loss detection and loss rate calculation in the sender. The receiver explicitly acknowledges every packet received by returning the acknowledgements (potentially piggybacked in the audio packets as part of the audio streams going in the reverse direction) to the sender.

Other Approaches

Packet loss can be detected and/or packet loss rates calculated in the receivers. These techniques do not need acknowledgments. They are very suitable for purely receiver-driven approaches in which the receivers perform the adaptation e.g. receiver-driven layered multicast. However, if the sender is responsible for rate adaptation, then the receivers need to report all the loss detection and loss rates calculation explicitly back to the sender.

Justification

A sender-based approach to packet loss detection and calculation is chosen since the sender is responsible for adjusting the transmission rates. There are two more general problems with sender-based approaches. Explicit acknowledgment of each packet can increase the amount of traffic on a certain network link and contribute towards congestion. This is usually resolved in any of the following ways: having the receivers report summaries of losses, having the receivers acknowledge every *n*th packet or every *n*th RTT, or piggybacking the acknowledgments within the audio streams coming from the direction of the receiver. Also, sender-based approaches have no way of determining whether the acknowledgement of the packet or the packet itself was lost e.g. it may assume that a successfully delivered packet got lost when only its acknowledgment was lost. Therefore sender-based approaches can sometimes overestimate packet loss rates.

However, receiver-based packet detection and calculation used in sender-based rate adaptation mechanisms suffers from similar problems as well. Packets with explicit reports of already detected lost packets and calculated packet loss rates can also get lost in times of congestion. If the sender had no mechanism to detect losses itself and the feedback from the receiver is lost, then the sender would continue sending at the rates previously calculated and therefore have totally non-responsive behaviour.

Having a timeout mechanism in the sender enables distributed partial mixing to react even in times of high congestion and broken connections.

4.3.3 Monitoring and estimating round trip times (RTTs)

4.3.3.1 Introduction

Round trip time indicates the time a data packet requires to go from one end system to the other and back. This time includes propagation delays over the physical links and time spent in the buffers of the routers as well as the transmission and processing time in the end systems. This section poses and answers the question:

What mechanism will be used for calculating RTT?

4.3.3.2 What mechanism will be used for calculating RTT?

Chosen approach

When estimating the round trip times for the purpose of adapting to network conditions, the processing and buffering times at end systems needs to be subtracted from the total round trip time, and only network distance calculated. Distributed partial mixing performs network distance estimation based on a simplified version of the NTP [Mills, D., 1992] algorithm. Figure 4.3 shows one round of message exchange used to estimate the distance between two partial mixers.

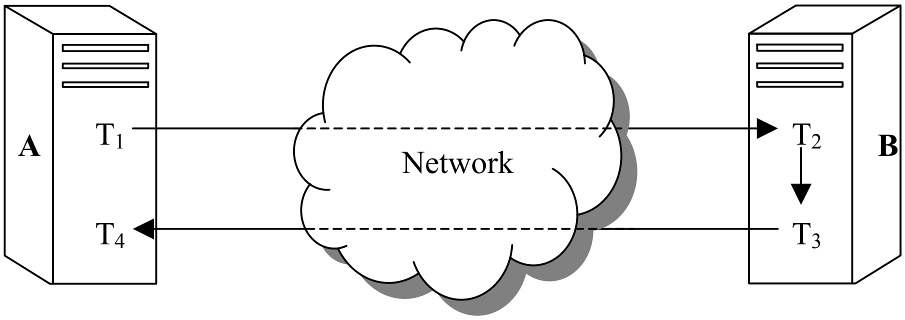


Figure 4.3 Calculating network distance

Each message includes the local timestamp recording when that message was sent. The information about the times can be piggybacked on the audio packets travelling from partial mixer A to partial mixer B in order to save the bandwidth. The packet sent by partial mixer A at time T_1 contains T_1. Partial mixer B records time T_2 when this packet arrives at B. At time T_3 partial mixer B piggybacks times T_1, T_2 and T_3 onto an audio packet travelling from B to A. Upon receiving the packet that carries the tuple (T_1, T_2, T_3) or (T_1, T_3-T_2) at time T_4, partial mixer A can estimate the two-way distance to partial mixer B using the following expression and without assuming synchronised clocks:

$$RTT = T_4 – T_3 + T_2 – T_1$$

In today’s networks and with high congestion levels, round trip times can vary significantly. In order to smooth those variations, rather than using the current measured round trip time value, an exponential moving average of the round trip times is used as a parameter for estimating congestion levels and deciding on the appropriate sending rates. Exponential moving average is defined in the following way:

$$AvRTT = c*AvRTT + (1-c)*measRTT,$$

where measRTT is the last measured RTT, AvRTT is the averaged RTT. The value of the constant *c* can vary depending on how much influence a single measurement is allowed to have on the overall estimate of RTT. For the implementation it was chosen to be 7/8 (as in TCP). Distributed partial mixing uses the most recent value of AvRTT.

Alternative approaches

A straightforward approach for estimating the total round trip time between two points (A and B) in the Internet is to send a data packet from A to point B with the packet indicating the time the packet was transmitted, $T_{1,}$ as measured by A. After receiving this measurement packet, B sends the received packet back to A which receives it at time T_4. The total round trip time T_{RTT} can then be estimated simply as:

$$T_{RTT} = T_1 - T_4$$

Justification

The reason for taking network distance rather than total RTT is that if end systems are taking too much time for processing there is no need to "punish" the network, and decrease the sending rate as if the network is congested. Similarly, the RTT estimate should not reflect any delay in acknowledging a packet caused by waiting for a return packet to piggyback.

The choice of smoothed version of RTT was taken because measured RTT has a random nature [Bolot, J., 1993], and using the last measured RTT can result in poor behaviour. The smoothed version of RTT reflects low frequency variation of RTT and reduces transient (i.e. high frequency) changes.

4.3.4 Rate adaptation

4.3.4.1 Introduction

Once the parameters of a given link are measured (packet loss and round trip times), there are a range of approaches that can be followed when choosing a rate adaptation scheme(s) to be integrated into distributed partial mixing.

As with any congestion control mechanism for multimedia applications, designing a network friendly rate adaptor in distributed partial mixing should manage the trade-off between the following properties (as identified by [Yang, Y., Kim, M., Lam, S., 2001] and more formally defined in [Floyd, S., Handley, M., Padhye, J., et al, 2000]):

- *Responsiveness* – identified as fast deceleration of protocol sending rate when there is a step increase of network congestion.
- *Stability and smoothness* – identified as small sending rate variations over time for a particular flow in a stationary environment.
- *Fairness* – identified as small variations over sending rates of competing flows.

The requirements for combining these three properties are in conflict and this complicates the design of the rate adaptor in distributed partial mixing. Mimicking TCP behaviour in distributed partial mixing results in fairness towards TCP but also in very significant oscillations in bandwidth (despite the form of packet loss measurement used in DPM, that contributes significantly to both stability and fairness). However, distributed partial mixing, where relatively smooth sending rate is of importance to the end-user perceived quality, needs to have much lower variation in throughput over time than TCP, so that it is suitable for streaming media. The penalty for having smoother throughput than TCP is slower response to changes in available bandwidth.

Achieving a trade-off between these three goals for partial mixing gets more difficult when more bandwidth consuming audio encodings are in use. Depending on the audio encoding used by a partial mixer, achieving this trade-off can be at least as challenging as for approaches that are based on changing the audio stream encodings as a response to congestion. The granularity of every congestion response step in distributed partial mixing is one audio stream and therefore at least as large as one layer in an adaptive layered encoding. A single increase/decrease step when changing sending rate is therefore large, and can decrease the smoothness and stability of partial mixing. In the demonstration part of this chapter, it is shown that partial mixing with an appropriate rate adaptation mechanism can achieve a trade-off between fairness, stability and responsiveness even when streams are encoded at a relatively high bandwidth (64Kbits/s).

This section poses and answers the following questions:

a) *What rate adaptation approach is used for distributed partial mixing?*
b) *What is the increase/decrease policy of distributed partial mixing?*
c) *What is the decision frequency of distributed partial mixing?*
d) *(How) is self-limitation achieved for distributed partial mixing?*

4.3.4.2 What rate adaptation approach is used for distributed partial mixing?

Chosen rate adaptation approach

For applications that compete in the best-effort Internet with TCP and require relatively smooth changes in the sending rate, being responsive to network congestion over longer time period (seconds, as opposed to fractions of a second) is more important than the opportunistic use of increases in the available bandwidth [Floyd, S., Handley, M., Padhye, J., et al, 2000]. The particular rate adaptation approach chosen and implemented in distributed partial mixing is an equation-based approach based on the model first proposed in [Floyd, S., Handley, M., Padhye, J., et al, 2000]. Equation-based rate control mechanisms generally use a control equation that explicitly gives the maximum acceptable sending rate as a function of the packet loss and round drip times in response to the feedback from the receiving partial mixer. Hence, the sending partial mixer directly adjusts its transmission rate guided by this control equation in response to the measured RTT and packet loss rates. For distributed partial mixing that competes in the best effort Internet with TCP, the appropriate control equation is the TCP response function characterizing the steady-state sending rate of a TCP flow as a function of a round-trip time and steady-state loss event rate. This equation is formulated as follows:

$$T = \frac{s}{R\sqrt{\frac{2p}{3}} + t_{RTO}\left(3\sqrt{\frac{3p}{8}}\right)p\left(1+32p^2\right)}$$

where T is the upper bound on the sending rate, *s* is the packet size, *R* is round-trip time estimated as described in 4.3.3.2, *p* is steady-state loss event rate estimated as described in 4.3.2.3, and the t_{RTO} is the TCP retransmit timeout value.

Implementing this TCP response function when distributing UDP audio streams ensures that UDP-based distributed partial mixing competes fairly with TCP over long time scales.

Alternative Approaches

Equation-based rate adaptation was first proposed informally in [Mahdavi, J., Floyd, S., 1997]. A simpler version of the formula that determines the maximum bandwidth share of a TCP connection having the same steady-state loss ratio (PL), maximum transmission unit (MTU) and minimum round trip time of the connection (RTT) was also suggested by Floyd in [Floyd, S., Fall, K., 1997], and is given below:

$$T = \frac{1.22 \cdot MTU}{RTT \cdot \sqrt{PL}}$$

where S is given in bytes per second, MTU in bytes, RTT in seconds and PL is the packet loss rate. While this model is simpler than the chosen one, it is only partially correct. The authors have verified this formula by simulations only for the rates up to 5%. Since it does not consider timeout cases or delayed acknowledgments, the formula typically results in overestimated throughput of the connection as the loss rate increases. Simulations in [Floyd, S., Fall, K., 1999] show that it can only be applied up to a loss ratio of 15% and overestimates the bandwidth for packet loss above 5%.

Additive Increase Multiplicative Decrease (AIMD) [Bolot, J., Turletti, T., Wakamn, I., 1994], [Busse, I., Deffner, B., Schulzrinne, H., 1996] is a yet another approach for rate adaptation, and TCP is the best known example of AIMD. In the absence of congestion, TCP increases its congestion window linearly with time, and on congestion reduces it multiplicatively by a factor of 2. It is usually known in literature as AIMD(1,1/2) where 1 and 1/2 denote increase and decrease parameters. The important feature of AIMD is that it backs off in response to a single congestion indication.

There are many variants of AIMD depending on their increase and decrease parameters. Most general denotation for all variants of AIMD is AIMD(a,b), where a and b are increase/decrease parameters respectively i.e. AIMD(1/5,1/8), AIMD(2/5,1/8). For flows desiring smoother changes in the sending rate, congestion control mechanism should reduce its rate with the decrease parameter less than ½ in response to congestion.

Justification

The primary reason for choosing equation-based congestion control rather than AIMD-based approaches is that it is more appropriate for applications that need to maintain a slowly-changing sending rate, while still being TCP fair and responsive to network congestion over longer time periods (seconds as opposed to fractions of a second). To accomplish this, equation-based congestion control finds and uses available bandwidth in a less-aggressive manner compared to AIMD-based congestion control mechanisms and TCP, and maintains a relatively steady sending rate while being responsive to congestion.

However, given that the stability of the current Internet rests on AIMD congestion control mechanisms in general, and on TCP in particular, a proposal for non-AIMD congestion control requires more thorough justification in terms of suitability for the global Internet. This problem has already been posed by [Floyd, S., Handley, M., Padhye, J., et al, 2000], and analysed in detail in [Floyd, S., Handley, M., Padhye, J., et al, 2000a]. Some of the most significant issues for DPM that [Floyd, S., Handley, M., Padhye, J., et al, 2000a] emphasise include the following.

The most obvious advantage of AIMD-based mechanisms compared to equation-based congestion control is that they are familiar and reasonably-well understood in terms of fairness, stability, oscillations and other properties. However, there are two major general arguments in favour of non-AIMD approach that also have to be taken into account.

The first argument in favour of a non-AIMD approach is that the principle threat to the stability of end-to-end congestion control in the Internet comes not from flows using alternate forms of TCP compatible congestion control, but from flows that do not use any congestion control at all. For much current traffic, the choice has been between TCP, with its reduction of the sending rate to half in response to a single packet drop, and no congestion control at all. Therefore both AIMD and equation-based approaches are beneficial for reducing congestion in the Internet.

The second argument in favour of a non-AIMD approach is that preserving the stability of the Internet does not require that flows reduce their sending rate by half in response to a single congestion indication. In particular, the prevention of congestion collapse simply requires that flows use some sort of end-to-end congestion control to avoid high sending rate in the presence of a high packet drop rate. [Floyd, S., Handley, M., Padhye, J., et al, 2000] show that even preserving some degree of fairness against competing TCP traffic does not require such a drastic reaction to a single congestion indication. Moreover, [Tan, D., Zakhor, A., 1999] argue that halving the sending rate of a flow is too severe a response to a congestion indication for some applications, as it can noticeably reduce the user-perceived quality.

Moreover, the most obvious disadvantage of AIMD-based mechanisms compared to equation-based congestion control is that equation-based congestion control has less abrupt changes in the sending rate; any congestion control based on AIMD inherently includes oscillation in the sending rate. The development of congestion control mechanisms with smoother changes will increase incentives for applications to use end-to-end congestion control, thus contributing to the overall stability of the Internet.

A more detailed argument in favour of the particular approach chosen is that pure AIMD schemes, as well as rate-based (equation-based) schemes based on the simple TCP model, do not take TCP timeouts into account. Consequently they are unfair to TCP in environments with a high loss rate, where TCP behaviour is dominated by timeout mechanisms.

4.3.4.3 What is the increase/decrease policy of DPM?

This section discusses how the control algorithm adjusts sending bandwidth once the allowed sending rate is calculated. The increase phase happens in situations when the allowed rate is higher than the current sending rate. The increase in the sending rate is allowed during underload periods. The decrease phase refers to the period when the sending rate should be decreased to be equal or smaller than the expected rate. The decrease happens during overloaded periods. There are a number of possibilities for increase as well as decrease policy. Different policies can affect the responsiveness, stability and fairness of the rate adaptation approach in different ways. For example, a quick increase in the rate can result in a very aggressive adaptation approach that is not very stable. This would also achieve higher bandwidth on average than other rate adaptation approaches, and thus is not fair towards them.

Chosen policy for increase/decrease phase:

Constant additive increase rate was chosen for the increase phase and straight jump down to the allowed value (calculated by the formula) for the decrease phase.

If there are potentially more audio streams to be sent, and sending an additional stream would not exceed the current allowed (TCP-fair-calculated) bandwidth, an additional stream is sent. If allowed expected bandwidth falls below the current sending level, then the sending rate is reduced below that.

If the allowed rate results in bandwidth less than that required for one stream at the tolerable packet loss rate for its encoding, the packet transmission does not stop; rather dummy packets are transmitted at a fixed probing rate. This is termed probing mode.

Alternative Approaches

Apart from linear increase, the increase phase can be:

- a multiplicative increase rate.
- a straight jump to the allowed value calculated by the formula.

Apart from a direct decrease, the decrease factor can also be:

- a multiplicative decrease factor, e.g. 1/2, 7/8, 1/8, of the previous value,
- a linear decrease.

Justification

Constant linear increase during non-congested periods is the default for the Internet. One could argue that a loss estimate of zero indicates that there is no congestion and thus the sending rate should be increased with the maximum possible increase factor until a loss event occurs. However, this approach causes instabilities in the sending rate and is very susceptible to a noisy packet drop rate. Therefore, DPM takes a more conservative approach of linear increase even during the times of zero packet loss. Constant linear increase also happens when there is packet loss, but the formula calculates that TCP would increase its rate under the same conditions. It is not advisable to use the new calculated sending rate directly, since the altered bitrate might cause changes in other network parameters. For instance, an increased sending rate can cause buffers to fill up, leading to an increased round-trip time. This in turn results in the decrease in the expected bitrate [Rejaie, R., 1999]. To prevent such oscillations, the bitrate has to be adjusted gradually so that the receiver has the time to report the new network conditions back to the sender.

The decrease should always be multiplicative rather than linear because congestion recovery must be exponential in order to remain stable. (The default decrease factor for the Internet is multiplicative decrease e.g. halving). Since TCP-fairness is an important requirement for the rate adaptor in this particular implementation of distributed partial mixing, and the equation gives the expected rate that TCP would have, it is important for distributed partial mixing not to exceed the rate calculated by the formula: the simplest option is to go straight down to what the formula calculates.

In this way, distributed partial mixing tracks nominal TCP bandwidth usage, but increases linearly to give stable behaviour.

4.3.4.4 What is the decision frequency of DPM?

The decision frequency specifies how often the sending rate is changed. Based on system control theory, optimal adjustment frequency depends on the feedback delay.

The feedback delay is the time between changing the rate and detecting the network reaction to that change.

Chosen frequency

The chosen decision frequency is 6-8 RTTs.

Alternative options

It is suggested that rate-based schemes adjust their rates not more than once per RTT [Mahdavi, J., Floyd, S., 1997] (but it could be any multiple of RTT). Changing the rate too often results in oscillation whereas infrequent change leads to an unresponsive behaviour.

Justification

The rate adaptation equation is applied every several RTTs rather then every single RTT in order to minimise the frequency of rate changes. Frequent rate changes in DPM are undesirable for the end user because they have a direct impact on their perceived QoS in terms of the levels of spatialisation. In addition to this, [Floyd, S., Fall, K., 1999] recommends the equation to be applied every several RTTs and not every single RTT if long-term TCP behaviour is being considered. The exact values were determined through extensive experimentation to a achieve trade-off between TCP fairness, smoothness and responsiveness of distributed partial mixing.

4.3.4.5 (How) is self-limitation achieved?

Self-limiting behaviour is the classic problem with rate-based schemes. In window-based schemes the source stops once it has a full window worth of data on the fly. This property makes window-based schemes intrinsically stable. However, if the source allows retransmission beyond the current window stability is lost, and so the number of retransmitted packets must also be limited. Rate-based schemes need to find some approach analogous to a finite window to bound the volume of outstanding data in the network. One way of achieving this is to use correctly implemented timers. In the absence of any feedback, the expired timer forces a source to drop its rate.

Chosen Approach

Distributed partial mixing achieves self-limitation by using a timeout mechanism for loss detection. In the extreme case, when no acknowledgements are received (e.g. when the connection goes down, the network gets partitioned, the receiving end crashes, or during very severe congestion times) or when the packet loss rates are above 20% (and therefore unusable for audio), the transmission rate drops to the probing mode rate using the decrease policy described in section 4.3.4.3. Distributed partial mixing sends one dummy packet per second while it is in the probing mode. Sending dummy packets is needed in order to allow distributed partial mixing to resume sending when the connection or the end system recovers (when these dummy packets begin to be acknowledged).

Other approaches

As already discussed, any window-based mechanism or rate-based schemes with timeout can achieve self-limitation. Different sending rates can be used during the probing mode, e.g. TFRC sends one packet each 64 seconds.

Justification

One packet per second is a faster probing rate than that of some other protocols. This is chosen to suit the nature of increase/decrease policy and decision frequency of distributed partial mixing. Note that distributed partial mixing goes directly to the probing mode from sending one stream, needs to have a bandwidth of one or more streams to resume sending audio, and tries to do this not more often than every 6-8 RTTs. Consequently, it is relatively easy for distributed partial mixing to go to probing mode and very difficult for it to recover. More frequent probing packets allow faster receipt of acknowledgments and thus quicker sending rate recovery. A slower rate for the probing mode delays distributed partial mixing recovery from the probing mode.

Other protocols (such as TFRC) that react much more frequently (every single RTT) and that are able to increase/decrease the sending rate with finer granularity (i.e. send few packets per RTT) go to probing mode less frequently and recover faster than distributed partial mixing. Such protocols can therefore have slower sending rates in the probing mode.

4.4 Implementation in MASSIVE-3

This section introduces the MASSIVE-3 system within which distributed partial mixing has been prototyped, and the audio process that has been extended by the author of this thesis to support distributed partial mixing. The second part of this section focuses on the additional features that the author has added and discusses additional features implemented in the MASSIVE audio process. In particular, the details of the congestion monitor and the comparator responsible for the network part of the DPM unit are presented.

4.4.1 Platforms

Distributed partial mixing was developed as an extension of the audio service in MASSIVE-3.

The primary reason for choosing MASSIVE-3 for the development platform and testbed application was because it is a collaborative virtual environment (CVE) platform that enables interaction among multiple simultaneous distributed users in a 3-D virtual world where all users have real time audio communication [Greenhalgh, C., Purbrick, J., Snowdon, D., 2000]. CVE platforms can have very challenging multimedia communications and are relatively complicated to implement. These features make MASSIVE-3 a natural and appropriate testbed for developing and testing the network and user features of an audio service based on distributed partial mixing.

Another equally important reason for choosing MASSIVE-3 for the development platform is that its source code was available to the author and it allowed modifications and extensions at the level of its network infrastructure.

The model in chapter 3 has been implemented in C. The distributed partial mixing implementation and experiments with it have been done using Silicon Graphics workstations. None of the code requires special hardware or platform-dependent software. The code is modular and extensible.

4.4.2 MASSIVE-3 audio service

Each MASSIVE-3 user has their own MASSIVE-3 client program to access the shared virtual world generated by the MASSIVE-3 world server. Each user's client program gives them a 3D view of the virtual world and allows them to move around within it. Each user also has their own local audio server that interfaces to the audio hardware on their computer, allowing them to talk to and be heard by the other users. Each user's MASSIVE-3 client controls their local audio server, using information in the virtual world to determine how it should send, receive and render audio streams (e.g. according to other users' positions within the virtual world). This audio server is a general-purpose audio server process developed by Greenhalgh for MASSIVE-2. The original audio features of the server are listed below:

- Each audio server can be controlled remotely by several processes at the same time.
- It can communicate audio data using both unicast and multicast protocols (both using UDP/IP).

- It can interface to the audio hardware on the computer on which it is running, acting as the audio client or peer for that user.
- Independently of any local user, it can support multiple mixing sessions that re-send their mixed audio streams on to other audio processes.

This audio server component can therefore act as a source, sink or mixer. Using this component several different audio services for MASSIVE-3 have been implemented by the author including distributed total mixing unicast [Radenkovic, M., Greenhalgh, C., Benford, S., 1999], [Radenkovic, M., Greenhalgh, C., Benford, S., 2000], distributed total mixing multicast [Radenkovic, M., Greenhalgh, C., Benford, S., 1999], [Radenkovic, M., Greenhalgh, C., Benford, S., 2000] and distributed partial mixing [Radenkovic, M., Greenhalgh, C., Benford, S., 2001].

The particular features the author has added to this audio process to provide for distributed partial mixing capabilities are:

- Support for direct forwarding of audio streams to other processes with no mixing. Each forwarded packet keeps the IP address and port number of the real sender.
- Support for monitoring loss rates on all the neighbouring links. Since the loss rates are being calculated for a given link, and not for a given stream, the sequence numbers are sequentially assigned to all of the packets in that link, irrespective of the streams the packets are coming from. Sequencing each stream individually would result in significant additional calculation when a partial mixer began to forward streams that were mixed.
- Support for monitoring delays on all the neighbouring links. Since a partial mixer acts both like a server (adapting its sending rate) and a client (sending the feedback information) the feedback information is piggybacked onto the audio packets. These contain both the time when the packet was sent and acknowledgment of a received time-stamped packet.
- Support for rate adaptation and TCP fairness. The data structures associated with each link contain all of the information necessary for determining the appropriate rate at which the application should send audio streams in order to perform

congestion control in a TCP fair manner (packet loss and RTT associated with that link).

An audio process which has these additional capabilities is termed distributed partial mixing unit (DPM unit).

4.4.3 Overview of distributed partial mixing in MASSIVE-3

Figure 4.4 shows a minimal distributed partial mixing system, as realised in MASSIVE-3 with this extended audio server. This example system has four concurrent users: A and B on one Local Area Network (LAN 1) and C and D on another LAN (LAN 2). The MASSIVE-3 world server is shown in this case on LAN 2.

For a peer-to-peer audio service, each user's audio process simply sends audio directly to every other user's audio process (using unicast or multicast protocols, as selected). However, for distributed partial mixing, there are also additional audio mixer processes not associated with any single user. In this example, there is one for each LAN. The resulting audio distribution graph is shown using dashed arrows, with each user (e.g. A) sending their audio stream to the local audio mixer process (LAN 1), which in turn sends it to other users on that LAN (B) and to the remote audio mixer process (LAN 2), which distributes it to the users on the other LAN (C and D). Distributed partial mixing processes can dynamically change between forwarding audio streams (for example, so that separate audio streams from A, B and C are all received by D), or mixing any subset of audio streams (for example, so that C and D receive a mixed stream combining sources A and B, while receiving separate streams from each other) to satisfy the goals set out at the beginning of this chapter, i.e. to be responsive, TCP fair and stable.

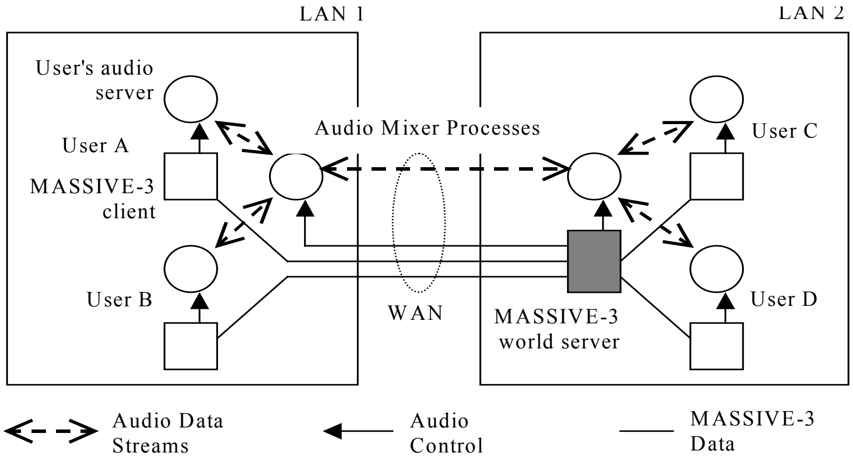


Figure 4.4 An example of distributed partial mixing in MASSIVE-3

The next two sections discuss a set of experiments that validate DPM behaviour across wide range of network conditions.

4.5 Demonstration scenario 1: distributed partial mixing and non-adaptive traffic

This section demonstrates the responsiveness of distributed partial mixing in the face of step-wise increases in competing non-adaptive traffic. It focuses on the steady state and demonstrates the effectives of distributed partial mixing in terms of the levels of audio quality that it can provide, given varying levels of network congestion.

This section begins by introducing the approach chosen to test distributed partial mixing in a steady state against various levels of congestion. It then moves to describing the demonstration scenario, as well as the infrastructure created for this demonstration. This section concludes with the results from the demonstration and discussion of each experiment.

4.5.1 Approach

The responsiveness of distributed partial mixing against increasing levels of congestion is tested when a step-wise increase in non-adaptive traffic is introduced on a shared unmanaged link. As well as responsiveness, two quantifiable aspects of audio quality are also considered: the level of packet loss experienced, and the degree of spatialisation available to the end user, as described in the chapter 2. The degree of spatialisation is determined by the number of independent audio streams that are delivered to the listener, and that can therefore be independently localised within their own subjective audio mix. Quantitative evaluation based on these two criteria was chosen because these two criteria are both clearly related to the end-user's experience of the system, and can also be objectively determined from measurements of the system in use (e.g. numbers of packets per second).

4.5.2 Scenario and Infrastructure

An additional application is used to introduce increasing and controlled levels of competing non-adaptive traffic onto the emulated WAN so as to create congestion. Measurements are made of the levels of audio packet loss and the numbers of independent audio streams delivered to the listener (degree of spatialisation) that are achieved by distributed partial mixing.

The author has constructed an emulation of a MASIVE-3 user client that interacts with the system in exactly the same way as a normal user, producing and consuming audio streams as would a normal user client. The complete experimental set-up is as shown in the Figure 4.5.

The particular scenario that has been chosen is as follows:

- Eight (emulated) users on host A, each continuously send audio data, to give a total of eight distinct audio streams heading towards host B.

- A 635000 bits/s bandwidth limit is introduced on the virtual WAN using Dummynet. This corresponds to just over 8 audio streams with the audio encoding that we are using (8KHz, 8bit, Ulaw, mono). The WAN connection also has 70 milliseconds delay. This means that with no traffic introduced by the additional process, there should be no congestion.
- Three audio distribution strategies are compared to demonstrate the effectiveness of distributed partial mixing: forward all audio streams without mixing (equivalent to peer-to-peer multicast), mix all audio streams before forwarding (equivalent to total mixing at LAN 1), and mix a dynamic subset of audio streams.

In this demonstration scenario we wish to show the utility of distributed partial mixing. We introduce nine levels of competing (congestion-inducing) UDP traffic: 0, 78400, 156800, 235200, 313600, 392000, 470400, 548800 and 627200 bits/s. Each congestion stage lasts for about five minutes. This competing traffic is non-responsive (i.e. does not change in the face of packet loss).

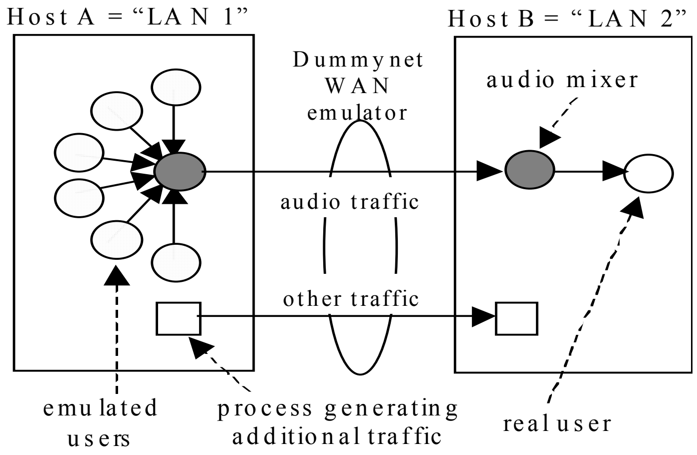


Figure 4.5 Experimental set-up

4.5.3 Results against non-adaptive traffic

The experimental results are shown in Figures 4.6, 4.7, 4.8 and 4.9. These show the steady state characteristics of distributed partial mixing (dynamics are considered in section 4.6). Figure 4.6 shows the effect that increasing levels of additional traffic have on the packet loss rate experienced by the three audio distribution strategies: distributed partial mixing, peer-to-peer (total forwarding) and total mixing.

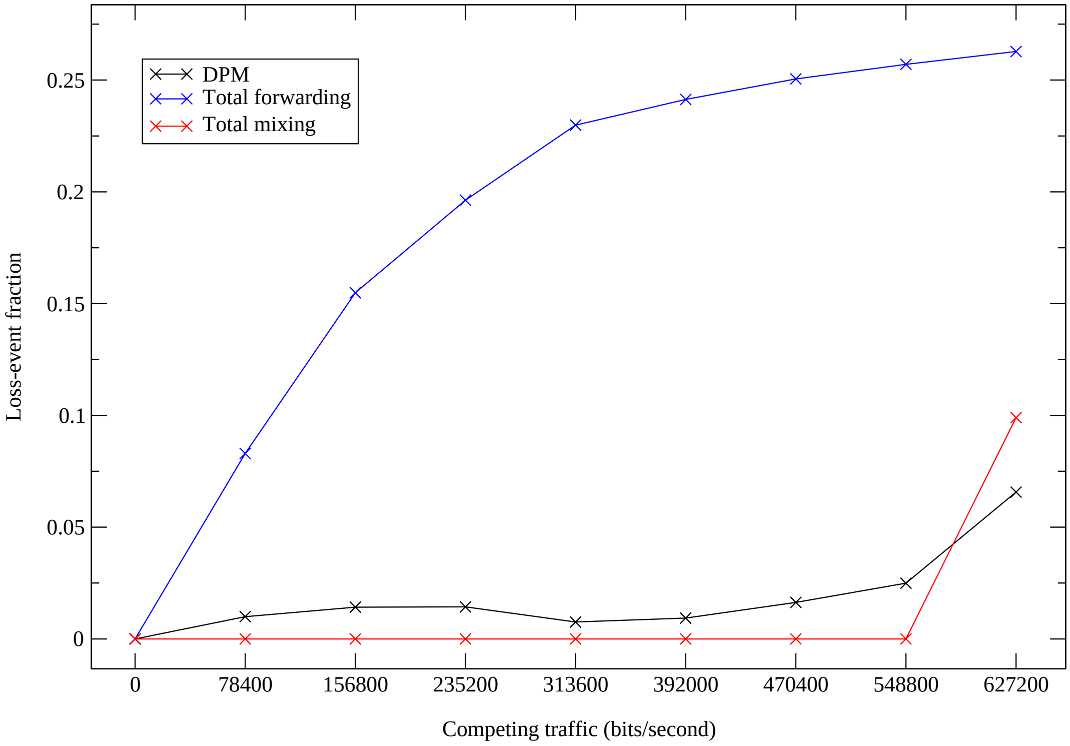


Figure 4.6 Rates of packet loss against competing (additional) traffic

The total forwarding approach experiences increasing levels of packet loss as the competing traffic increases. The packet loss event rate exceeds 15% with only 156800 bits/s of additional traffic**.** The total mixing approach (that uses the minimum bandwidth throughout) starts to experience congestion only when the competing traffic reaches more than 548800bits/s or the full bandwidth of the link. Distributed partial mixing gives much lower loss event rates than total forwarding, and maintains its loss event rate below 5% even with 548800bits/s of competing traffic (as full mixing does). Distributed partial mixing gives higher loss event rates than full mixing does for congestion levels up to (approximately) 600000 bits/s. For congestion level higher than (approximately) 600000bits/s, distributed partial mixing results in the lowest loss event rates (lower even than for full-mixing) because it oscillates between sending a single stream and probing.

Figure 4.7 shows the number of streams being transmitted to the listener on host B (LAN 2). For total forwarding, 8 separate streams are always sent, filling up the whole link. However, none of these streams arrives in a useful form with competing traffic of more than approximately 156800bits/s because the packet loss exceeds 15%. For total mixing, 1 stream is always sent, filling approximately one eighth of the channel at all times, and therefore underutilising the channel. Distributed partial mixing naturally lies between these two extremes. With no congestion, it still sends 8 distinct streams over the WAN connection. As competing traffic, and hence congestion increases, it reduces the number of distinct streams by mixing more audio streams together. With approximately 548800 bits/s of competing traffic, distributed partial mixing has fallen back to total mixing, with only a single stream sent over the WAN. For higher congestion levels, the graph shows that distributed partial mixing oscillates between sending a single stream and being in the probing mode. The graph shows not only the mean value of the number of independent streams sent by distributed partial mixing source within each congestion level but also all the variations in bandwidth within two standard deviations of that mean value. Such a representation shows, with 95% correctness, that maximum deviation from the mean value is approximately one stream for the test period of 5 minutes. This shows that distributed partial mixing has very small oscillations (due to adaptation) and therefore has reasonably stable behaviour.

Figure 4.8 demonstrates temporal behaviour of distributed partial mixing as congestion levels on the link increase from zero to the full link. This figure shows the responsive and stable behaviour of distributed partial mixing while still preserving relatively high numbers of independent audio streams for each congestion level.

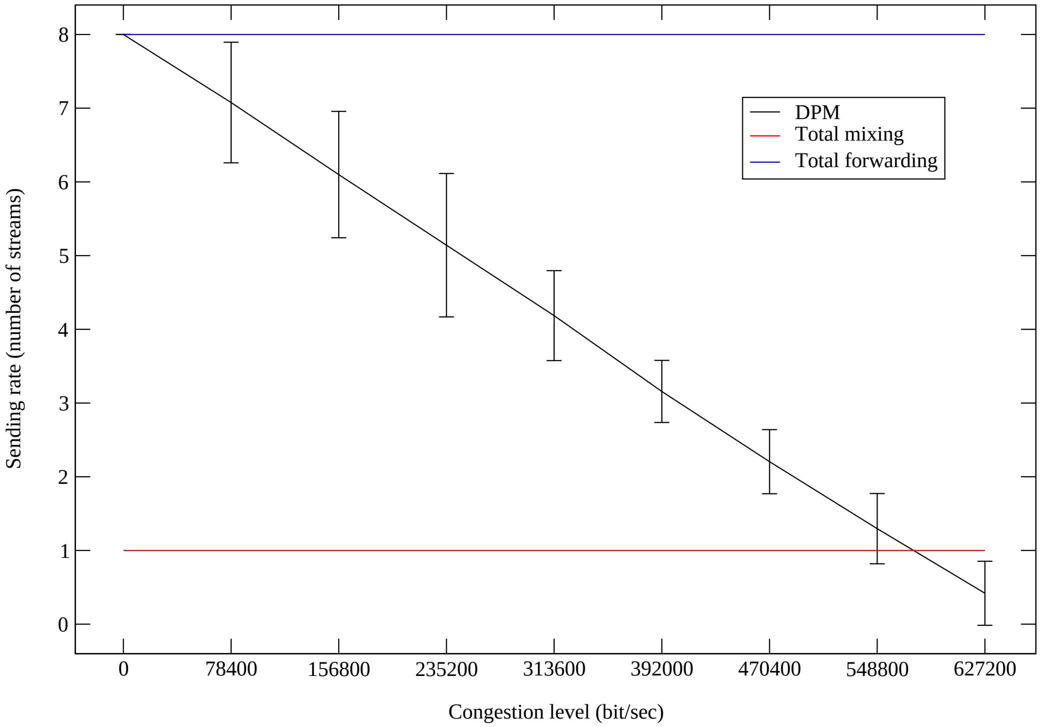


Figure 4.7 Degrees of spatialisation (number of independent audio streams) within two standard deviations against competing traffic

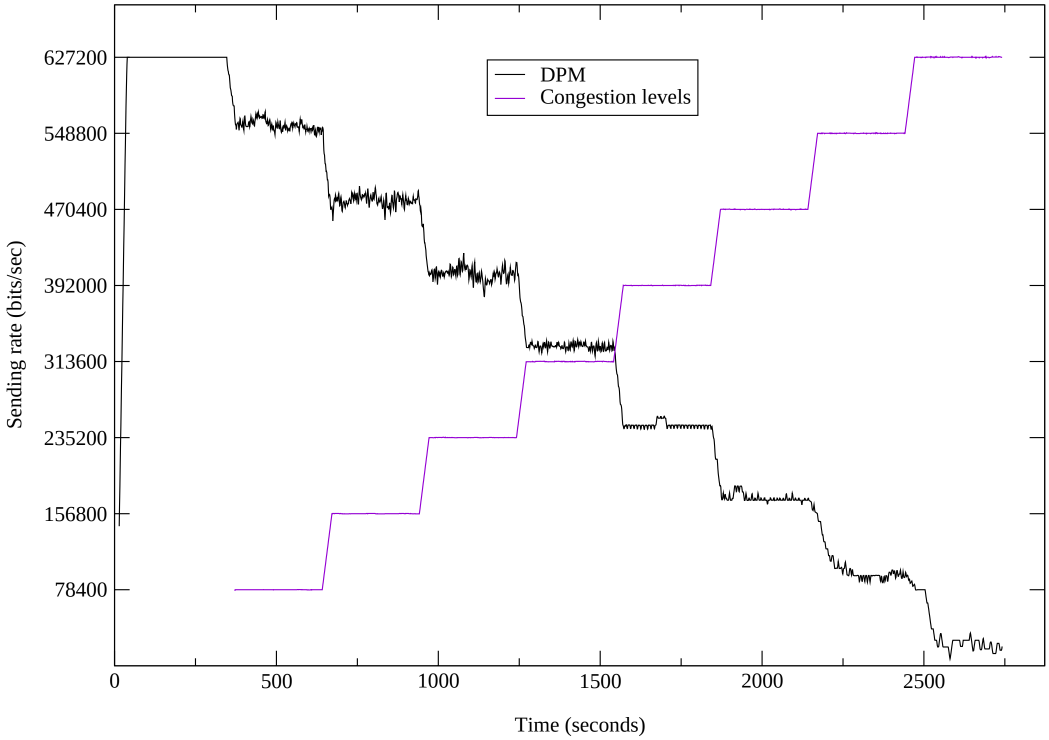


Figure 4.8 Temporal behavior of DPM: Responsive and stable DPM behavior

Figure 4.9 demonstrates the self- limiting behaviour of DPM based on the timer mechanism on a given link. Initially the sending distributed partial mixer sends at full rate (0 – 71 seconds). At around 72 seconds the receiving distributed partial mixer that provides the packet acknowledgements, is stopped for several seconds (72 – 92) to emulate loss of connectivity due to heavy congestion, and then restarted again (92 - 142). While the receiving DPM unit is stopped, the sending DPM unit detects increasing packet loss due to lack of acknowledgements and falls into network probing mode.

When the acknowledgements start being received, the sending DPM unit linearly regains its initial sending rate.

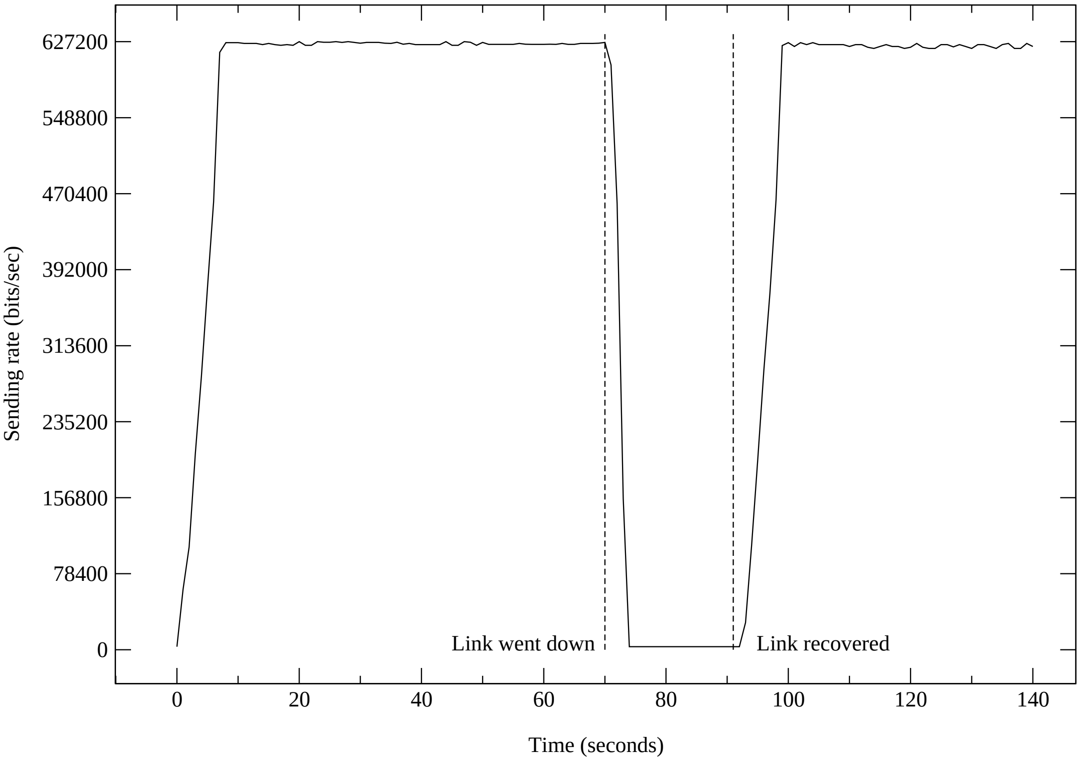


Figure 4.9 Self-limiting behaviour of DPM

4.6 Demonstration scenario 2: distributed partial mixing against adaptive traffic

This section focuses on the dynamics of the distributed partial mixing implementation and provides results from experiments to demonstrate its fairness towards TCP. It also shows the fairness between two DPM capable applications sharing the same unmanaged link.

4.6.1 Scenario and Infrastructure

A similar scenario to that presented in the section 4.5.1 is adopted for testing the fairness of distributed partial mixing towards TCP traffic and other distributed partial mixing traffic. Instead of introducing increasing levels of non-adaptive traffic, in this section increasing numbers of TCP flows are introduced. The results demonstrate that distributed partial mixing gets approximately the same share of bandwidth as any of the other TCP flows. Then an additional distributed partial mixing application with a large numbers of users is introduced. The results demonstrate that the two DPM applications achieve fairness between each other as well as towards TCP.

4.6.2 Results: TCP fairness

To demonstrate the fairness of the distributed partial mixing approach, Figure 4.10 shows the bandwidth consumed by the audio traffic sent by the partial mixer as well as competing TCP traffic as a function of time. Initially – with no competing traffic – the number of audio streams sent rises linearly and rapidly. After just a few seconds, it fills the link. At roughly 72 seconds a competing TCP flow starts. After 10 seconds of slow start behaviour this enters its own rapid steady state oscillation, continually probing available bandwidth. The offered traffic exceeds the capacity of the link, causing an increase in round trip time (due to buffering delays) and resulting in increased packet losses. The TCP-fair bandwidth calculated by DPM then falls, and the number of streams moderates accordingly. Distributed partial mixing takes its fair share of the bandwidth (one half or 3-4 streams) oscillating much less than TCP.

Figure 4.11 shows 20-second average bandwidths for distributed partial mixing and for the competing TCP traffic (so that the TCP oscillations are smoothed out). The graph shows how DPM moderates its bandwidth usage as first one, then a second, and finally a third TCP flow is started.

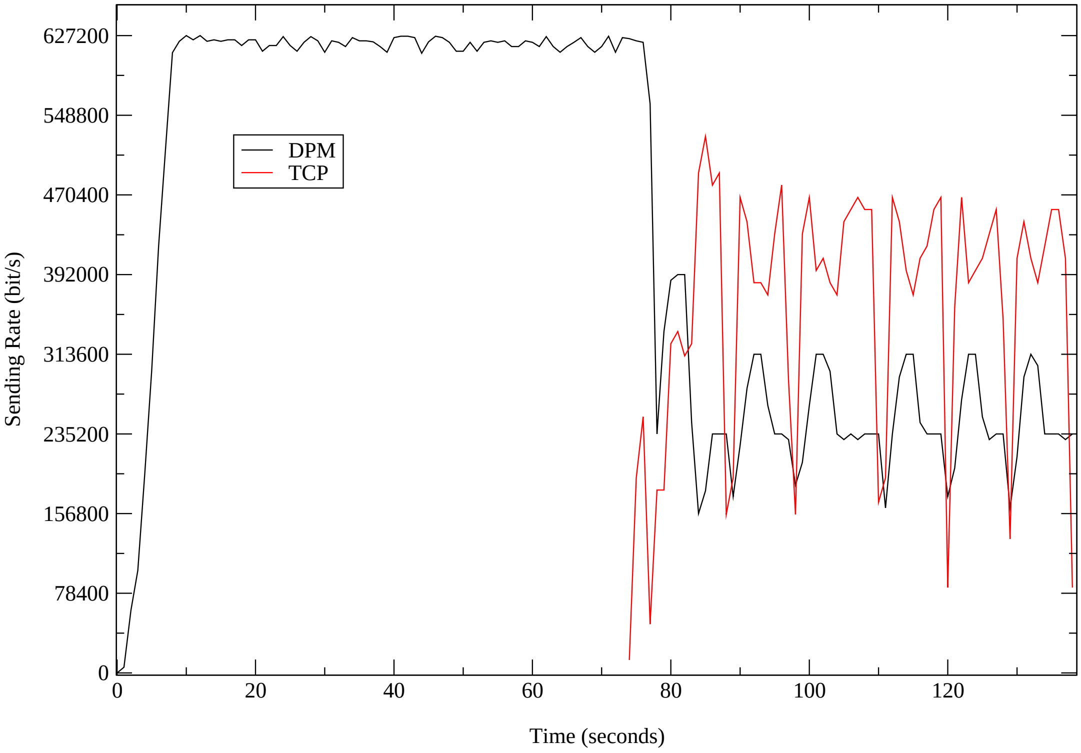


Figure 4.10 TCP-fair audio distribution showing detailed behaviour

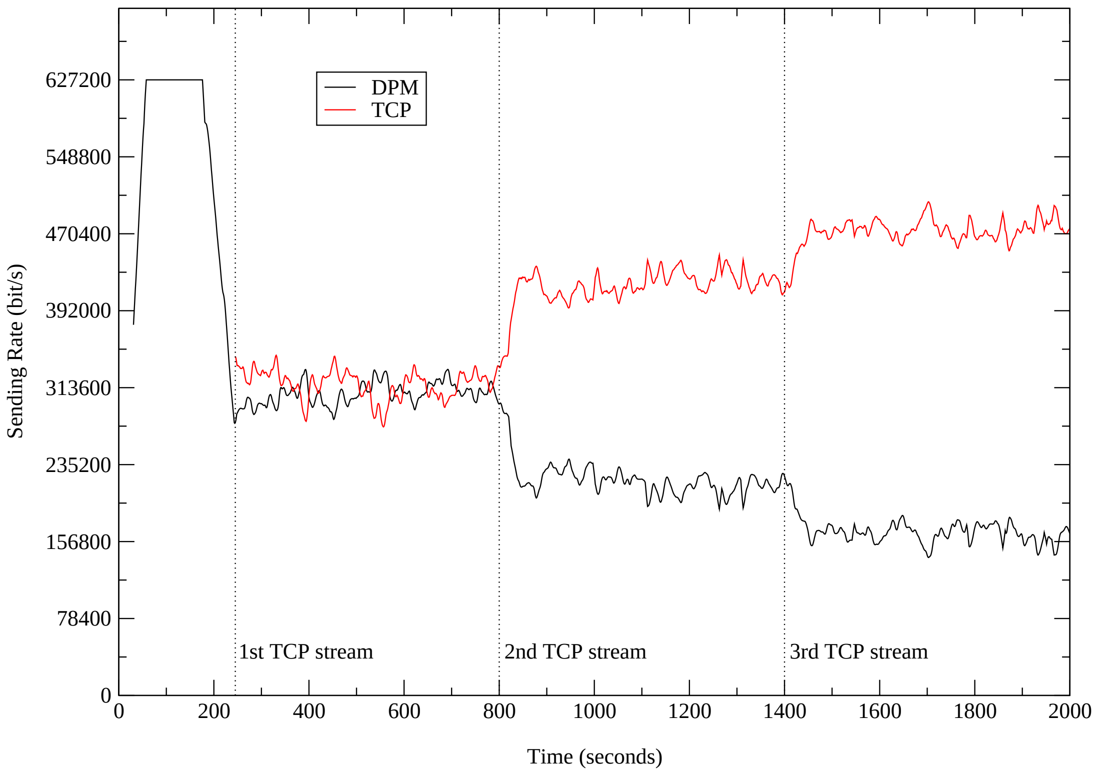


Figure 4.11 TCP-fair audio distribution showing medium term (smoothed) behaviour with competing TCP flows

4.6.3 Results: Demonstrating fairness between distributed partial mixing systems

To demonstrate fairness between multiple distributed partial mixing-capable applications, two such applications are started in the same LAN, each of them with eight users.

Figure 4.12 demonstrates this scenario. The first DPM application is started in the beginning, and it fills in the whole link. At times 260sec, the second DPM application is started (with eight players, all simultaneously active in the audio medium). The second application tries to send all the eight audio streams, but since the link is already full with the first application sending its eight streams the second application performs partial mixing and very quickly stabilizes at half of the link bandwidth. The first application also decreases its sending rate and stabilizes at half of the link bandwidth (at around 4 streams or 313600bits/s).

At time 820sec, a TCP flow is added to the link already taken fully by the two DPM applications. The graph shows that the two DPM applications and the TCP flow each adapt their rates to stabilise quickly at their fair share of the link by taking approximately one third of the link (each 2-3 streams or 192000bits/sec).

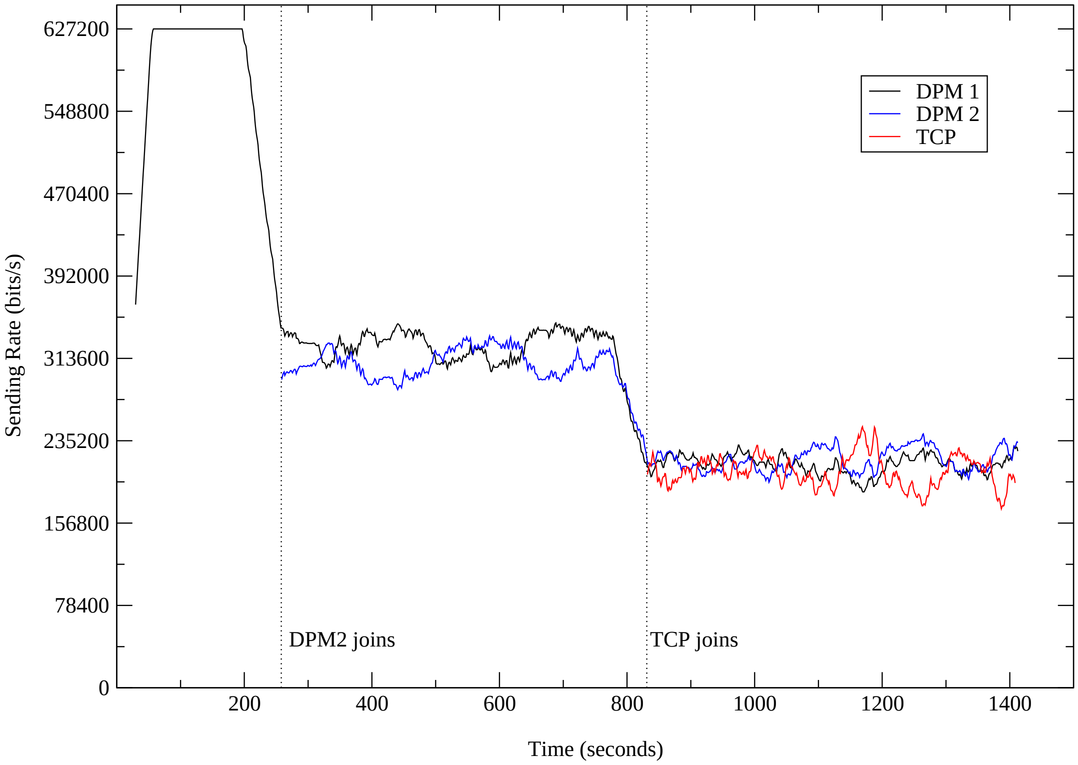


Figure 4.12 Fair audio distribution showing medium term (smoothed) behaviour with competing DPM and TCP flow

4.7 Summary

This chapter has demonstrated the responsive and TCP-fair behaviour of distributed partial mixing proposed in chapter 3. It has also showed that distributed partial mixing manages to maintain levels of end user's quality of service that can be quantitatively measured: high levels of spatalisation and low packet loss rates.

The chapter first introduced the design space and justified the design choices for implementing the components of distributed partial mixing that monitor and adapt to network conditions. Packet loss detection based on the TCP timeout mechanism enables timely and consistent packet loss discovery. DPM uses the Weighted Loss Interval Average approach to packet loss calculation. This approach is based on loss event rates (and not loss rates), and it explicitly ignores losses that follow an initial loss within a single round trip time. This allows for the sending rate to be reduced at most once for a congestion notification within one RTT. This mechanism allows for fine-grained changes in packet loss and copes gracefully with dynamically changing network conditions, including rapid increases and decreases in congestion, while preserving a stable packet loss estimate. The rate adaptation in DPM is based on the TCP response function that models long term TCP behaviour proposed by [Floyd, S., Handley, M., Padhye, J., et al, 2000]. Using this approach for distributing UDP audio streams ensures that UDP-based distributed partial mixing competes fairly with TCP over long time scales. Stable packet loss estimation, a decision frequency of 6 RTT, and a non-opportunistic linear increase policy towards the expected value for the sending rate result in stable and smooth sending rates for DPM that are suitable for multimedia applications.

The implementation and experiments were done in the MASSIVE-3 system which was chosen as a suitable development platform because it enables collaboration between participants that can transmit live audio streams at all times without restriction, and also because its source code was available to the author. The chapter then described the experimental set-up and gave results from the set of experiments done to demonstrate the behaviour of distributed partial mixing against non-adaptive traffic, TCP traffic and other distributed partial mixing traffic.

Experiments against various levels of non-adaptive traffic aimed to demonstrate the steady state behaviour of DPM on heterogeneous links. Experiments showed that DPM is responsive to congestion while maximising the quality of audio received by the end users. The quality was measured in terms of controlling packet loss and maximising levels of spatialisation in the end systems. At low, moderate and high steady state congestion levels, distributed partial mixing has a relatively high mean throughput and small variations of throughput, while preserving the packet loss under 10%.

Experiments against adaptive traffic demonstrate the dynamic behaviour of DPM. The adaptive flows included both TCP flows and other DPM flows, and both long-term and short-term DPM behaviour was observed. The results demonstrated that distributed partial mixing co-exists acceptably well when sharing congested links with other adaptive traffic. In the long-term, distributed partial mixing consumes its fair share of the bandwidth when competing with multiple TCP flows, as well as other distributed partial mixing flows. In the short-term, DPM also consumes its fair share while oscillating significantly less than TCP.

In the case of persistent high congestion levels or connection loss, distributed partial mixing goes to probing mode that allows it to probe for the available bandwidth with dummy packets. It resumes sending audio data when congestion decreases or the connection recovers because the sending partial mixer starts to receive acknowledgments from the receiving partial mixer.

In conclusion, this chapter has demonstrated that the chosen design for DPM can sustain high numbers of independent audio streams at acceptably low levels of packet loss while being responsive and fair towards TCP and other distributed partial mixing traffic.

5. Deployment

5.1. Introduction

This chapter focuses on the large-scale deployment of distributed partial mixing over wide area networks. The goal of this chapter is to examine the issues raised when deploying DPM within the context of large dynamic environments. The fundamental question is whether or not the DPM paradigm remains desirable or even feasible in such environments. This chapter is divided into seven sections.

Section 5.2 begins with a brief description of a steady-state scenario and infrastructure for large scale DPM system deployment over wide area networks that accommodates very large numbers of geographically distributed users. It identifies the particular problems and constraints that arise in such WAN deployment of DPM. It then suggests a shared tree as one possible minimal solution for interconnecting DPM servers in a topology that avoids, or at least minimises, the impact of these constraints. Some fundamental algorithms and heuristics that can be used for constructing such a tree topology are given. Other possible topologies are also discussed.

Section 5.3 introduces and motivates a new set of issues concerned with the realisation of a large scale DPM system over WANs.

Sections 5.4 and 5.5 then describe in more detail two contrasting proposals for possible DPM deployment over managed and unmanaged deployment domains. Section 5.4 proposes static, fully centralised DPM service with user subscription for the fully managed scenarios. Section 5.5 proposes a self-organising, fully distributed

DPM service that copes with dynamic networks and user membership changes for the unmanaged scenarios.

Section 5.6 begins by summarising the two proposals and then discuses other approaches and their advantages and disadvantages according to the issues raised in Section 5.3. Section 5.7 summarises the issues involved in the steady state and dynamic realisation of large scale DPM over wide area networks. It summarises a proposed solution for the ideal topology of DPM servers and clients as well as the two more specific proposed schemes for its realisation in contrasting deployment domains.

5.2 Steady-state large scale DPM deployment

5.2.1 Scenario and Infrastructure

This section discusses constraints concerning deployment of DPM over wide area networks that support very large numbers of geographically distributed users. The wide area network scenario assumes a collection of autonomous unmanaged and managed networks such as the Internet, and a large number of dispersed users that can be served by large number of DPM servers distributed throughout the network. It is assumed that there are potentially large numbers of DPM servers located in various networks, and large number of users connected to various of the DPM servers. This is illustrated in Figure 5.1. Therefore, wide area deployment has to address issues of many more links, users and DPM servers all potentially communicating with one another. If the DPM servers are not designed properly for these scenarios, several issues can degrade the performance and scalability of the distributed partial mixing method.

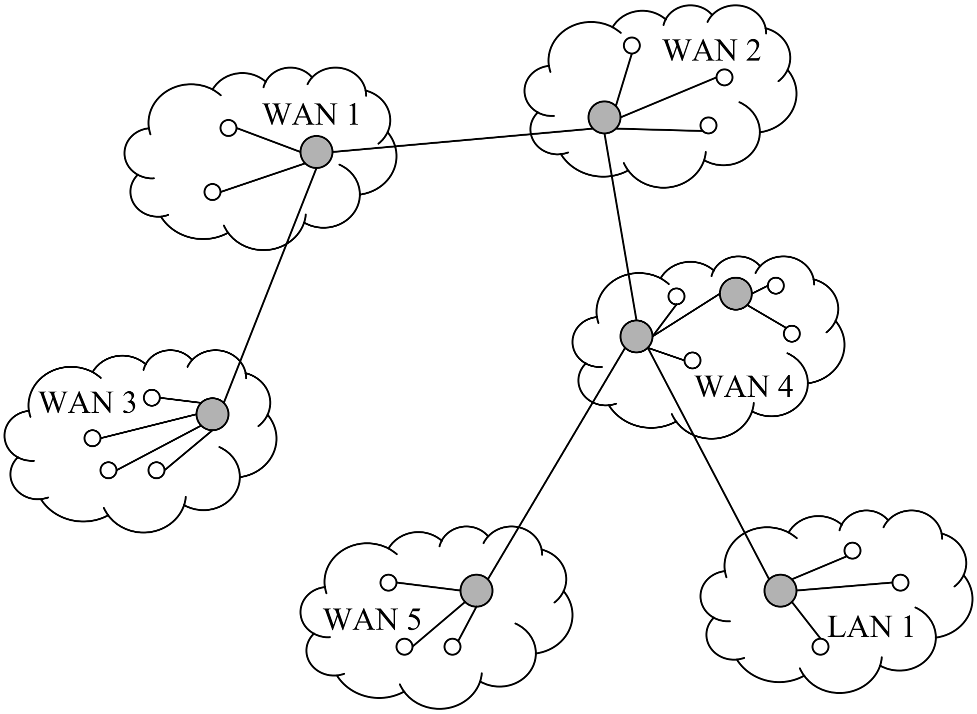


Figure 5.1 Wide area DPM deployment

The current Internet is served by an established wired backbone infrastructure of routers that provide communication among the Internet nodes. The routers are assumed to be special purpose nodes that are dedicated to perform only routing and forwarding, and would not perform distributed partial mixing functionality.

Suppose that an application that can have very large numbers of geographically distributed simultaneous users is started in part of the network. In order to be network friendly, the application requires that the participating users make use of distributed partial mixing servers for audio communication.

WAN deployment raises some significant issues that did not exist or were of lower importance when considering the single link scenario in Chapter 4.

5.2.2 Issues in the steady-state large-scale DPM deployment

In this section we assume that a centralised authority has a complete knowledge of all members and links since the beginning of time. This allows us to explore fundamental issues and limitations to any large scale DPM system:

- DPM topology
- Existence of echoes and loops in the topology
- Delays imposed by the topology
- Fan in and fan out of the DPM servers

These issues are introduced in the following sections. The assumptions of centralisation and complete knowledge are relaxed in later sections.

5.2.2.1 DPM Topologies

DPM topologies refer to different ways of interconnecting DPM servers and clients, i.e. establishing routes between all clients. The main goal of a DPM topology is to allow support for many-to-many communications i.e. multiple simultaneous senders and receivers.

This problem scenario is naturally modelled in terms of graph theory. A graph contains a set of vertices and edges. In this case each vertex represents a DPM server or a client, and each edge represents connection between them. The graph’s edges can have weights associated with them, where weights represent some cost or distance metric in connection with the problem that is being modelled; such a graph is called weighted graph. If the graph also contains ordered pairs of vertices it is called weighted directed graph.

This issue is considered in section 5.2.3, which illustrates a tree-like minimal solution to connecting all of the nodes in a graph as a basic approach to establishing DPM topology. It also discusses other useful techniques and topologies that can be used in combination with or as an alternative to the basic tree approach.

5.2.2.2 Echo and loops

A second issue is that of echo over wide area networks. Echo refers to situations when audio is sent over a wide area network and then returned to the sender. This is typically very disturbing for the sender because they may hear themselves delayed by a few hundred milliseconds. Therefore any distributed audio service needs to be designed to remove (or at least minimise) echoes.

Wide area loops in audio streams can also potentially happen whenever audio mixers are multiply connected, e.g. as in a peer-to-peer scenario. This refers to situations when an audio stream is returned indirectly to its originating node. This can result in indefinite loops of sending and receiving the same audio stream, and must be prevented by any distributed audio service.

This issue is considered in section 5.2.4, which describes how DPM servers could be designed to avoid loops and echoes.

5.2.2.3 Delays

The number of mixing stages between any two users is also an issue in WAN deployment of DPM because each mixing stage introduces additional delay in the audio communication between the two end points, (e.g. buffering to compensate for incoming network jitter and re-packetisation). This delay should be minimised or limited to a tolerable delay for the users (e.g. tolerable delay depends on the nature of the application, but typically it is between 50 to 150 milliseconds [Schooler, E., 1996]). In order to minimise the latency, the number of mixing stages between any two end points needs to be minimised. This can be achieved by minimising the diameter of the tree for example (see section 5.2.3.2). Also, real delays between the nodes can be measured and minimised using shortest path algorithms between the two nodes.

5.2.2.4 Fan in and fan out / CPU load

Fan in/fan out refers to the number of relationships a DPM server has, and includes connections to its children, peers and parent (if any). Fan in and fan out of clients and DPM servers needs to be limited because it directly affects the load imposed on the individual DPM servers and clients, and can cause loss, delay or even failure due to overload.

The CPU load of a DPM server increases as more users and other DPM servers are connected to it. The internal complexity of the DPM server can grow rapidly if each adjacent DPM server and/or client needs their own customised mix to prevent WAN loops and echoes. Memory and storage requirements of a DPM server also increase with the increase in fan in/out of that DPM server because the server must have information about all the nodes with which it is communicating, and monitor all the links to each of them. Similarly local bandwidth requirements will be higher with more DPM connections.

Therefore, an architecture designed to limit the connections handled by each DPM server and client also limits the CPU load, memory, storage, and bandwidth requirements of these DPMs.

5.2.2.5 Tradeoffs

While wide area network loops and echoes always need to be prevented, there is a tradeoff between minimising the number of stages between the end users and minimising fan ins and outs of each DPM server. More specifically, the smaller the fan in and out of each DPM server is, the less load it handles and the less users it can support. The only way to then grow the supported number of users is by allowing the diameter of the tree to increase, which in return increases the delays. Therefore, the number of users can be increased only at the expense of increased delays, and vice versa.

For example we can express mathematically the inescapable relationship between the height of the tree and fan in and fan out for the ideal case of a perfectly balanced *m*-ary tree topology of DPM servers and users.

If we suppose that *m* is the number of connections each DPM can have and *h* is the height of a perfectly balanced tree and *N* is the number of clients supported in such a tree (where clients are only in the leaf nodes of the tree), then formula (1) gives the number of clients supported in a tree of a height *h* and fan in/out *m*:

$$N = m(m-1)^{h-1} \qquad (1)$$

The height of the tree can then be expressed as a function of a fan in/out *m* by the formula (2):

$$h = 1 + \log_{m-1} \frac{N}{m} \qquad (2)$$

If we differentiate this function, we get (3) which is smaller than zero for any *m*>2 (4).

$$\frac{d}{dm}\left(1 + \log_{m-1} \frac{N}{m}\right) = \frac{d}{dm}\left(\frac{\ln \frac{N}{m}}{\ln(m-1)}\right) = -\frac{\frac{\ln(m-1)}{m} + \frac{\ln \frac{N}{m}}{m-1}}{\ln^2(m-1)} \qquad (3)$$

$$-\frac{\frac{\ln(m-1)}{m} + \frac{\ln \frac{N}{m}}{m-1}}{\ln^2(m-1)} > 0, m > 2, m \in \mathrm{I}^+ \qquad (4)$$

This shows that the height of the tree (*h*) decreases with the increase of fan in/out (*m*) of each individual DPM server for a given number of clients. Note that (3) is not defined for *m*=2; and for *m*=1 there is no tree.

Range of example values for fan in/out and the height of the tree for supporting 100, 1000, 10000 users are given in Table 5.1. The table shows that for approximately 100 supported users the increase in fan in/out limit (from 5, to 10, and finally 20) will result in the corresponding decrease in the height of the tree (from 4 to 3 and finally 2). Similarly for 1000 and 10000 supported users and an increase in fan in/out (5,10, 20), the height of the tree decreases (5, 4, 3) for 1000 users, (7, 5, 4) and 10000 users.

Approximate number of end nodes covered	Maximal fan in and fan out of DPM servers	Height of the tree
100	5	4
100	10	3
100	20	2
1000	5	5
1000	10	4
1000	20	3
10000	5	7
10000	10	5
10000	20	4

Table 5.1. Range of example values for maximal fan in/out and height of the tree for various numbers of clients

Having introduced these issues for steady state wide area deployment the remainder of this section proposes a minimal tree topology of DPM servers (that also addresses fan in/out of DPM servers and the height of the tree) and the design of DPM servers to avoid loops and echoes.

5.2.3 Overview of the proposed tree topology

This section proposes a shared tree of DPM servers and clients where the nodes of the tree can perform partial mixing as a very effective basis for large scale wide area network deployment of DPM. It also discusses other approaches for the DPM distribution topology and their respective advantages and disadvantages.

5.2.3.1 Introduction

This section describes a steady state spanning tree architecture for DPM servers as a minimal solution to interconnecting large numbers of DPM servers and clients. A spanning tree of a graph is defined as a subgraph that contains all of the vertices but only enough of the edges to form a tree. Any spanning tree is loop free. If an edge is added to such a tree, it must form a loop because there already exists a path between the two edges. Note that a spanning tree does not give a universal or optimal specification for wide area distributed partial mixing deployment. Rather, it highlights how an architecture can be designed to cope with some of the complex problems specific to WAN deployment of distributed partial mixing.

The problem of creating spanning trees in various kinds of graphs is an old and well-known problem and there are various algorithms for it.

Determining the Minimum Spanning Tree (MST) is a standard graph algorithm and is usually used for modelling minimum interconnection problems. The MST of a weighted graph is usually defined as a collection of edges connecting all the vertices such that the sum of weights of the edges is at least as small as the sum of the weights of any other collection of edges connecting all the vertices. A MST need not be unique. The weights assigned to the edges of a graph can model the number of mixing stages, the reliability or the financial cost between the two vertices. Popular algorithms for solving this problem are Kruskal’s algorithm, Prim’s algorithm, and Steiner minimum tree [Sedgewick, R., 1990], [Krumke, S., Noltemeier, H., Marathe, M., et al, 1996], [Zhu, Q., Parsa, M., Dai, W., 1994].

The Steiner tree and Minimum Steiner tree problem can be slightly distinguished from the minimum spanning tree problem because it permits the construction (or selection) of intermediate connection points to reduce the cost of the tree. Building such trees is more complicated when the possibility of *Steiner points* is added. Steiner points are like phantom nodes that can be added to the graph to shorten the connecting distance. Determining how and where to put these Steiner points in a non-trivial (NP hard) problem.

A Shortest Path Tree (SPT) gives minimum shortest paths between a given vertex to all the other vertices, and thus also create a spanning tree of that graph. The SPT problem is based on the basic algorithms for finding the shortest path between any two vertices. Various classical algorithms can be used for finding shortest paths between the two vertices in a weighted, directed graph such as Dijkstra's algorithm, Bellman-Ford algorithm, Johnson algorithm. More adaptive algorithms that cope with changes in the graph's weights and number of vertices and edges also exist such as [Humblet, P., 1991]. In [McDonald, A., 1997] the authors present a more complete survey of fundamental issues and advances in the area of adaptive shortest path routing for packet-switched networks, and examine them within the context of large, highly-dynamic, and re-configurable environments.

Given the nature of DPM - that adaptively mixes the audio from multiple sources - it is particularly important that the spanning tree being constructed is globally acceptable since it will be shared by multiple sources. Ideally such a topology would include DPM servers at the points where the audio streams coming from multiple sources join allowing efficient many-to-many casting (with multiple senders and multiple receivers).

Therefore, a form of a shared tree (shared among all the sources and receivers) approach is adopted for wide area DPM deployment. This is described in more detail in the next section. Such a tree has a single root that is the "topmost" node of the tree. Every other DPM node has a single parent and one or more children. Only the leaf nodes have no children. Shared trees are usually derived from minimum spanning trees and in particular minimum Steiner trees. If shared among all the sources, shortest path trees can also be used for DPM deployment.

Spanning trees that can be shared among all the sources and receivers can also be derived from Minimum Set Coverage problem. This is discussed in the next section in more detail. Alternatives to a tree are also discussed, including a hierarchical architecture of DPM servers (multiple trees connected with a mesh on the top or a mesh in each cluster), and broken hierarchies.

5.2.3.2 Shared Trees

Shared Trees and Minimal Shared Trees (MST) have their origin as approximations of Steiner Trees and Steiner Minimal Trees correspondingly. A shared tree is a single delivery tree that is shared by all the senders and rooted at a single node (usually called core point or rendezvous point (RP) that does not need to coincide with a sending/receiving node).

MST distribution algorithms have been introduced in order to address sparse distribution graphs (a graph with the number of edges much less than the possible number of edges, that is, the number of vertices squared) and to support many-to-many communication. Shared trees are likely to result in high concentration of traffic sent by multiple senders along the same path and near the core points. Depending on the location of the RP, a shared tree might not be optimal for all the sources (i.e. it does not necessarily provide the shortest path between the root and the other nodes as a corresponding shortest path tree does). These trees are in general best suited for situations where the bandwidth is a scarce resource and the senders do not send very large volumes of data.

The concept of shared trees has been extensively exploited in multicast. The main example of a shared tree protocol is Core Based Trees (CBT) [Ballardie, A., 1997], but also Protocol Independent Multicasting Sparse Mode (PIM-SM) [Estrin, D., Farinacci, D., Helmy, A., et al, 1997]. There are various approaches for creating shared trees such as [Carlberg, K., Crowcroft, J., 1997]. Within the general shared tree protocol framework there are several variations which can be implemented such as unicast sender mode, multicast sender mode and adaptive per source multicasting.

More detailed descriptions of these can be found in [Chiang, C-C., Gerla, M., 1998]. Traditionally, for multicast sources that start sending large volumes of traffic, source-specific (shortest path trees) are created in order to avoid very high traffic concentration on shared links and to ensure the optimal path from that source to its receivers. Source-based trees are best suited or environments with high bandwidth and densely distributed receivers. Examples of such protocols are DVMRP [Waitzman, D., Partridge, C., Deering, S., 1988] that is based on reverse path forwarding and is the main protocol currently used on the Mbone. Other examples are PIM and MOSPF [Moy, J., 1994]. However, with DPM allocated at the points of the shared multicast tree, DPM capable nodes will perform adaptive mixing and congestion control of the links with high traffic concentration. Shared trees provide much lower overheads compared to shortest path tress, as well as adequate stability even in the presence of fast moving sources [Chiang, C-C., Gerla, M., 1997]. More detailed trade-offs among shared tree types and source specific shortest path trees, by comparing performance over both individual multicast group and the whole network is given in [Wei, L., Estrin, D., 1994].

Basic algorithms for shared trees do not guarantee bounds on height and breadth of the tree but there has been some work emerging to address these problems. [Salama, H., Reeves, D., Viniotis, Y., 1996] studied the problem of constructing delay-constrained shared multicast trees for real time applications in connection-oriented high-speed networks. This problem can be formulated as a diameter-constrained minimum Steiner tree problem (NP-complete problem). They proposed distributed, dynamic heuristics for solving this problem that eliminates the need for a dedicated centre selection phase`. Other approaches have also proposed methods for constructing constrained shared trees such as [Krumke, S., Noltemeier, H., Marathe, M., et al, 1996] and [Zhu, Q., Parsa, M., Dai, W., 1994].

To the author’s knowledge, there has been no work to date that was explicitly designed to cope with a limited number of connections per node in the tree (so to limit fan in/out of the DPM servers). A simple heuristic based on Minimum Set

Coverage (MSC) problem can be used to build the trees with limited fan in/out. This is addressed below.

Using Minimum Set Coverage (MSC) to build shared trees

In order to build shared trees of DPM server and clients that do not exceed maximal fan in/out of a DPM server, minimum set coverage algorithm can be used. More specifically, MCS can be used to determine the set of DPM servers that cover the set of clients (thus building the first layer of the tree), then determine another set of DPM servers that cover the first layer of DPM servers (thus building the second layer of the tree). This process continues until there is a single DPM server that covers the previous layer of DPM servers.

An example of a bipartite graph that shows possible connections among a set of clients (C_1, C_2, C_3, C_4) and a set of DPM servers (DPM_1, DPM_2, DPM_3) is given in Figure 5.2. In order to find a subset of DPM servers to cover the complete set of clients, all the three DPM servers can be chosen (DPM_1 to cover C_1, DPM_2 to cover C_3, and DPM_3 to cover C_2 and C_4) while only two DPM servers can perform the same coverage (choosing only DPM_1 to cover C_1 and C_2, and DPM_2 to cover C_3 and C_4). Using a smaller number of DPM servers to cover all clients is more efficient because less DPM servers are used and more clients are aggregated together.

This problem is known in the literature as the minimum set coverage (MSC) problem. It is also a standard NP hard problem that can be solved using standard heuristic algorithms. Virtually every heuristic approach for solving general integer programming problems has been applied to it: greedy heuristics [Fisher, M., Wolsey, L., 1982], probabilistic search [Feo, A., Mauricio, G., Resende, A., 1989], simulated annealing [Johnson, D., Aragon, C., McGeoch, L., et al, 1989], neural networks [Aourid, M., Kaminska, B., 1994] and others have each been tried.

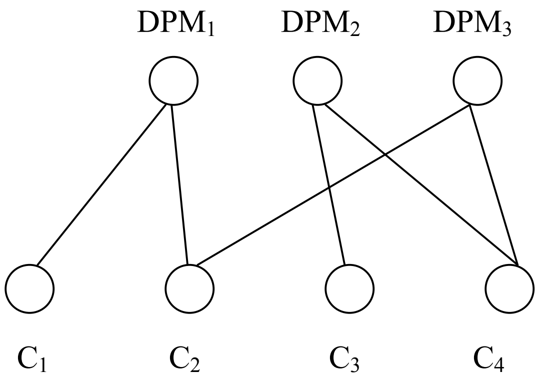


Figure 5.2 An example bipartite graph of clients and DPM servers

The greedy algorithm for approximating a minimum subcover chooses, at each step, the covering set with the maximum number of elements left, deletes these elements from the remaining covering sets and repeats this process until the ground set is covered. Even this simple algorithm is shown to have quite good quality solutions with low time complexity in [Slavik, P., 1995]. With simple extensions, this algorithm can be adapted in a way that each covering set is bound to M elements and no more. If the set has more than M elements, the M elements chosen first to be covered are the elements that have the smallest number of discovered DPM servers.

Some of these heuristics do not guarantee an optimal solution i.e. they might result in a slightly higher number of DPM servers to cover the set of users (or the lower level of DPM servers). Even though not minimal, the set of DPM servers chosen should still be significantly smaller than if the no heuristic was applied (e.g. if the chosen DPM server was always chosen by each client (or another DPM server) individually).

5.2.3.3 Overview of other topologies

Other (non-tree) approaches are also possible. Some options are mentioned below.

Multiple trees with mesh on the top – balanced hierarchies

Another approach to interconnecting DPM servers is to have multiple trees connected into a mesh on the top (instead of having a single root). This might be useful because it results in lower delays between end points than constructing a tree would.

Hierarchy with mesh in each cluster

This refers to the topology where all the nodes with the common parent in a tree (or multiple trees) are also connected to one another in a full mesh. This approach minimises the delays for members in the same cluster, but increases the fan in/out of the members of the same cluster (and therefore the overheads in terms of memory, CPU load, and storage requirements of any members in the same cluster).

Broken hierarchies

This approach refers to having some users bypass the hierarchy altogether by directly communicating to all (or some) of the other nodes in the system. In this way, these users are prioritised as they have much lower latency towards the others and also will never be mixed together with the other streams, however this introduces new complexity in configuring the whole system and additional overheads.

More general graphs

The DPM servers can be connected in a more general graphs, but the management of such a graph is expected to be much more complex than that of trees and hierarchies

Even though some of the non-tree approaches have benefits over tree topologies we adopt a tree topology as a minimal and the simplest solution to interconnecting DPM servers and clients.

5.2.4 Preventing loops and echoes

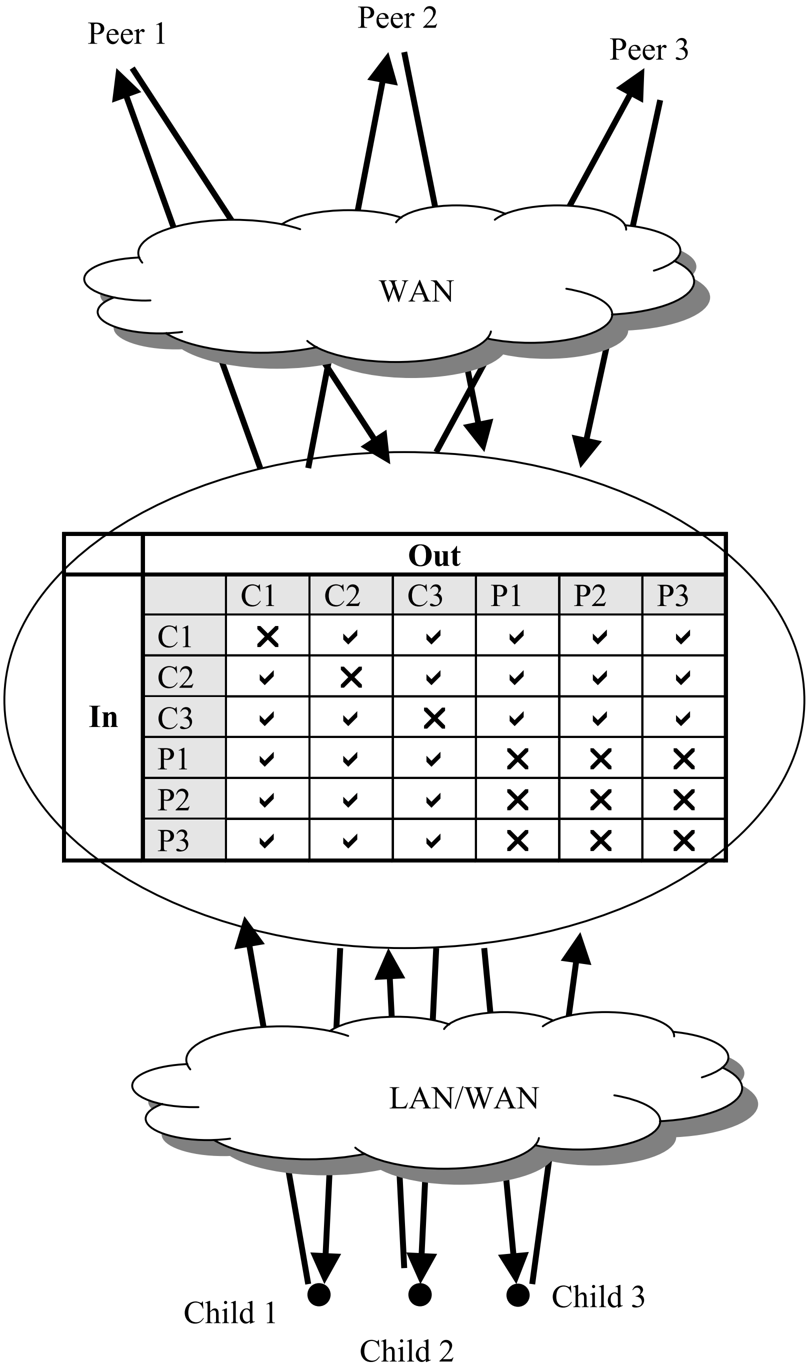
One goal of this particular deployment model is to avoid wide area network echoes and loops, i.e. audio should never be sent directly or indirectly to its original sender, because this creates disturbing echoes. Depending on the algorithm used and the topology created loops can be avoided in the construction of the topology, for example, creating a tree (rather than a more general graph). Also, when calculating shortest paths, using Humblet's algorithm (instead of the Ford-Bellman algorithm) would be loop free.

If there are loops in the topology, then loops and echoes can be prevented by defining how the input and output streams of a DPM server relate to each other. If this relationship is properly defined loops and echoes are avoided, since an audio stream is never returned to its sender (client or regional server).

The general internal logic (structure) of a regional DPM server with a number of children and peer connections is illustrated in the Figure 5.3. The table shows the relationships between all combinations of input and output streams within the DPM server.

The crosses in the leading diagonal of the table ensure echo-free DPM. Each cross denotes that an input audio streams from a client input c_i will not be contained in the mix that is sent to that client c_i. Similarly, any audio streams coming from a peer input p_i will not be contained in the mix for that peer p_i.

Crosses in the lower right quarter of the table ensure loop-free DPM: they ensure that audio received from any peer p_i will not be contained in the mix for any other peer p_j, $j \neq i$. This is particularly important if there is a mesh (peer-to-peer) relationship or any cross links in the topology of DPM servers and clients.



		Out					
		C1	C2	C3	P1	P2	P3
In	C1	✗	✓	✓	✓	✓	✓
	C2	✓	✗	✓	✓	✓	✓
	C3	✓	✓	✗	✓	✓	✓
	P1	✓	✓	✓	✗	✗	✗
	P2	✓	✓	✓	✗	✗	✗
	P3	✓	✓	✓	✗	✗	✗

Figure 5. 3 Internal logic (structure) of a DPM server when dealing with multiple links

Note that for each column there is potentially one or more independent partial mixers. Multiple partial mixers are necessary whenever each receiver needs an individually tailored partial mix. If the local clients and networks are sufficiently capable then all of the client streams can be forwarded to them directly, and each one can ignore its own stream(s) to avoid echo. For less capable local clients a dedicated partial mixer is needed for each one to generate a customised mix that specifically excludes its outgoing audio stream.

5.2.5 Summary

This section introduced and discussed some fundamental issues in large scale DPM deployment over WAN that might affect or constrain the performance of DPM. These issues are topology determination, echo and loop prevention, minimisation of delays between end clients and fan in and fan out of DPM servers. This section has pointed out the major underlying tradeoff between delays and fan ins and outs in a graph topology.

A shared tree of DPM servers and end users, where the nodes of the tree can perform distributed partial mixing, is proposed as an effective basis for wide area DPM deployment. The major benefits of shared trees are that they support many-to-many communication, low overheads and simplicity. However, any shared tree approach suffers from traffic concentration on the links shared by multiple senders. Using distributed partial mixing in the nodes enables the use of shared trees even for large number of senders that send large volumes of data without the risk of traffic concentration and congestion. This section also discussed other approaches such as trees based on minimum set coverage problem that can also be used to constructing spanning trees.

An additional (more general) advantage of using trees over more general graphs is the tree's simplicity, i.e. nodes are easily associated with a specific layer in a structured tree, and this results in simple decision making about connections and neighbouring nodes [Shoubridge, P., 1996] and is therefore simple to manage. A general disadvantage of any tree-like architecture is that it contains critical nodes and links that if removed partition the nodes. However this may be addressed, at least to some degree, by an adaptive design.

5.3 Realising a large scale DPM service over WANs

5.3.1 Issues in realising large scale DPM service

The last section considered an idealised scenario in which we assume global knowledge, sufficient number of available DPM servers distributed through the network and positioned in suitable locations, where they remain during the lifetime of the application. It also proposed a shared tree (and various algorithms for constructing it) as an effective topology for interconnecting DPM servers and clients that achieves global optimisations in terms of the issues addressed in the previous section.

However, in a practical DPM protocol, the assumption of global knowledge of weights between the nodes in a DPM system would lead to infeasible control traffic overheads [Ratnasamy, S., McCanne, S., 1999]. More specifically, IP unicast and multicast deliberately hide the underlying topology from the end systems which means that they have no easy way of building efficient topologies (i.e. topologies that are in accordance with the underlying network topology). Fortunately, the use of subgroups in multicast applications and protocols that can result in non-tree based topologies is an optimisation that need not be perfect at all times to dramatically enhance performance. A number of schemes proposed in the research literature rely on the coarse-grained clustering of co-located nodes without requiring detailed knowledge of the exact routing topology [Handley, M., 1997], [Kouvelas, I., Hardman, V., Crowcroft, J., 1998], [Liu, C., Estrin, D., Shenker, S., Zhang, L., 1997]. These approaches relax the need for information exchange at the cost of a decreased topological accuracy in a controlled manner.

This section moves beyond the idealised scenario discussed in section 5.2 and considers the practical issues of deploying a large scale DPM service over realistic deployment domains. These issues include:

- Deployment domain
- DPM placement
- DPM initiation
- DPM discovery and advertisement
- DPM topology determination
- Managing the process of topology determination
- Support for re-configuration

After introducing these issues the next two sections describe two contrasting proposals for DPM service realisation and summarise the choices made in relation to each of these issues for each proposal.

5.3.2 Deployment Domain

At a general level we can distinguish two main kinds of potential deployment domain: managed and unmanaged. Specifically, these domains differ in the kind of access rights they allow, the amount of available knowledge about the network and other systems, as well as the (assumed) stability and reliability of the network links. The target deployment domain will affect all of the other issues presented in this section. This chapter includes illustrations of deploying DPM in both kinds of domain, in section 5.4 and 5.5 respectively.

5.3.3 DPM Placement

DPM placement refers to where in the network DPM functionality is placed (how, and on which machines). General requirements for a DPM system are that DPM servers are highly available and easily accessible to the clients, but also positioned near to problematic links where they can perform effective congestion control.

However this is very challenging when no prior knowledge of the client distribution and network is available. For example, placing too many DPM servers in the network might result in under-utilisation and inefficient use of resources, and vice versa. Also, only placing DPM servers near links with no congestion will not result in efficient congestion control of the whole DPM system. Note that an even distribution of DPM servers in the network may be good for an even distribution of clients and network conditions, but not for an uneven one. The two example proposals, illustrated in sections 5.4 and 5.5, will show how DPM servers can be placed in critical places in the network, both on dedicated machines and on non-dedicated machines (including client machines), respectively. These proposals will also show how DPM servers can be placed automatically in advance, or ad-hoc during the use of the application, respectively.

5.3.4 DPM discovery and advertisement

This refers to the question of how a DPM server learns about (and becomes known to) the entire system (or the part(s) of the system that are relevant to it). This also includes the question how much knowledge a single DPM server should have about the entire system. This has a direct impact on the storage space and memory of each DPM, as the knowledge that each has about the system has to be stored and processed by it. The dynamics of DPM discovery and advertisement can influence the efficiency of the DPM system as a whole. The two examples in section 5.4 and 5.5 will illustrate a centralised approach, where a centralised authority with complete knowledge updates DPM servers (or clients) about the rest of the system, and a decentralised approach, where various discovery protocols and collaboration with other nodes is used to discover the nodes of interest.

5.3.5 DPM initiation

DPM initiation refers to activating a DPM server, e.g. to start monitoring the neighbouring links and to performing adaptive partial mixing. More specifically, a

DPM server can become active in various ways and at various times, e.g. on-demand or pre-configured. Ideally, DPM servers should be activated promptly when they are needed and by those who need them. If, on the other hand, DPM servers are not active when they are needed or this activation process takes a long time then the responsiveness of the whole DPM system may be compromised. The quantity of session initiation traffic can affect the congestion control of the whole system because it might cause additional congestion itself. The two examples in section 5.4 and 5.5 illustrate two possible contrasting DPM initiation schemes and discuss them in more detail.

5.3.6 DPM topology determination

DPM topology determination refers to deciding on a suitable topology of DPM servers (i.e. topology does not need to be based on trees). The problem is twofold: Firstly, how are the connections between DPM servers and clients and other DPM servers determined i.e. what are the metrics for determining the cost of a connection? This refers to the underlying aggregation metrics/techniques used by schemes that determine the topology**.** Secondly, are global or local optimisations of the topology made when determining it?

An overview of an idealised general shared tree topology derived from a weighted graph of DPM and clients was already addressed in section 5.2 for a steady-state scenario. This section pays more attention to the dynamics of the topology determination process i.e. to how such a topology of DPM servers can be realised in real networks and systems, e.g. taking into account the overheads, complexity and flexibility of the process. The main difference between this section and section 5.2 is that we recognise that if the realisation of an optimal steady-state topology is very complex, less flexible and with large overheads, then that topology becomes less attractive in real-world situations. We also recognise that the suitability of a topology might change over time and that the rules for suitability might not necessarily be deterministic or fixed during the lifetime of the application.

5.3.7 Topology Control

This refers to how the overall process of managing the topology is realised. There are two extreme approaches for realising this process: centralised and distributed. This is related to whether the topology is aimed to support global optimisations vs. local optimisations in the DPM system. Efficiency, scalability and robustness of the controlling process are of great importance for the stability and overall performance of the DPM system. The level of overheads (e.g. controlling messages, that include initiation messages and various update messages) in this process can also be an issue – ideally the controlling process should scale with the number of clients and DPM servers and the updates should be sufficiently frequent to allow accurate decision making. Robustness of the DPM controlling process to various faults in the system will directly impact the stability of the whole DPM system. The two illustrations given in sections 5.4 and 5.5 provide contrasting examples of managing the topology: completely centralised and completely distributed. Note that combinations of centralised and distributed control are also possible and will be included in the final discussion.

5.3.8 Support for re-configuration

This issue refers to whether and how the system can respond to changes in the network and user membership. At the most general level, the topologies of DPM servers can fall into two categories: non-adaptive and adaptive. This is important given that today's networks are not very stable, either in terms of user membership (session members can be expected to continually join and leave during the session) or in terms of network link reliability (network links can be expected to fluctuate in link quality).

Non-adaptive topologies are basically deterministic topologies that use fixed rules for a given network and user configuration. These rules are formulated in advance and the current state of the network and users are not taken into consideration. Adaptive topologies adapt to changes in traffic patterns, underlying network topologies and user membership. In order to support this the topology needs to be re-configurable, i.e. a centralised authority or the clients themselves need to be able to detect such events and re-configure the topology in a scalable manner. Two examples given in sections 5.4 and 5.5 provide more information on the differences between non-adaptive and adaptive topologies.

In the next two sections we describe two contrasting approaches of dealing with the issues described in this section. The goal of illustrating the two approaches is to reveal in more depth some of the problems and issues that occur in the static and dynamic establishment and configuration of a DPM system, and to show two possible solutions.

The next two sections (5.4 and 5.5) propose two contrasting solutions that make radically different choices for each of the issues introduced in this section.

5.4 Fully centralised DPM deployment over managed networks

This section illustrates one particular approach that can be used in a fully managed scenario with a centralised coordinator (manager) that assumes full knowledge of the network, DPM servers and clients, and has all the necessary access rights to configure them.

For example we might assume that an Inhabited TV show (a popular soap opera, fashion show, or football game) is shown over cable TV, and that only the subscribed viewers can view it and participate in it. Some viewers choose to become more active participants in the virtual world that they are seeing on their televisions (“inhabit” the TV show), and it is further assumed that all the network communication with the other participants will be done over fully managed corporate network. Similar assumptions can hold for a large teleconferencing application that has to be scheduled in advance, and where participants have to be registered (subscribed) before the beginning of the session.

The centralised controller plays the role of an ideal observer that has centralised global knowledge of the whole system at every point of time. This knowledge is available to the centralised controller via multiple databases including topological and user databases, and it enables the controller to create connections among the nodes and their topology. This is illustrated in Figure 5.4. The network topology database contains complete information about network topology including link reliability, link bandwidth, link cost, possibility of bottlenecks, time delays, topological delay, throughput. The users' database contains complete information about subscribed clients, their locations and other properties. The DPM server database contains the locations of the DPM servers and their connections. A fixed set of dedicated DPM servers is placed in well-suited positions in the network. The controller then creates a weighted graph with all this information and processes it into a spanning tree according to some minimisation cost (e.g. bounding latency, bounding fan in and fan out of the DPM servers) using one of the algorithms presented in section 5.2.3.2. Once all the interconnections between the nodes are determined, the centralised controller produces a set of rules (where each rule applies to one or more DPM server and clients), and disseminates these rules to each node for execution. These rules instruct each DPM server and client what to connect to and when to become active.

The basic assumption made here is that these rules (connections) remain fixed throughout the application lifetime for a given configuration. The set of rules changes only if a new user subscribes, an old user unsubscribes, the underlying network conditions change; the number of DPM servers in the system has to change. Each time the rules change, they are disseminated to the nodes affected by this change. In general, this approach is best suited for applications with fixed membership throughout the session (such as subscription-based applications).

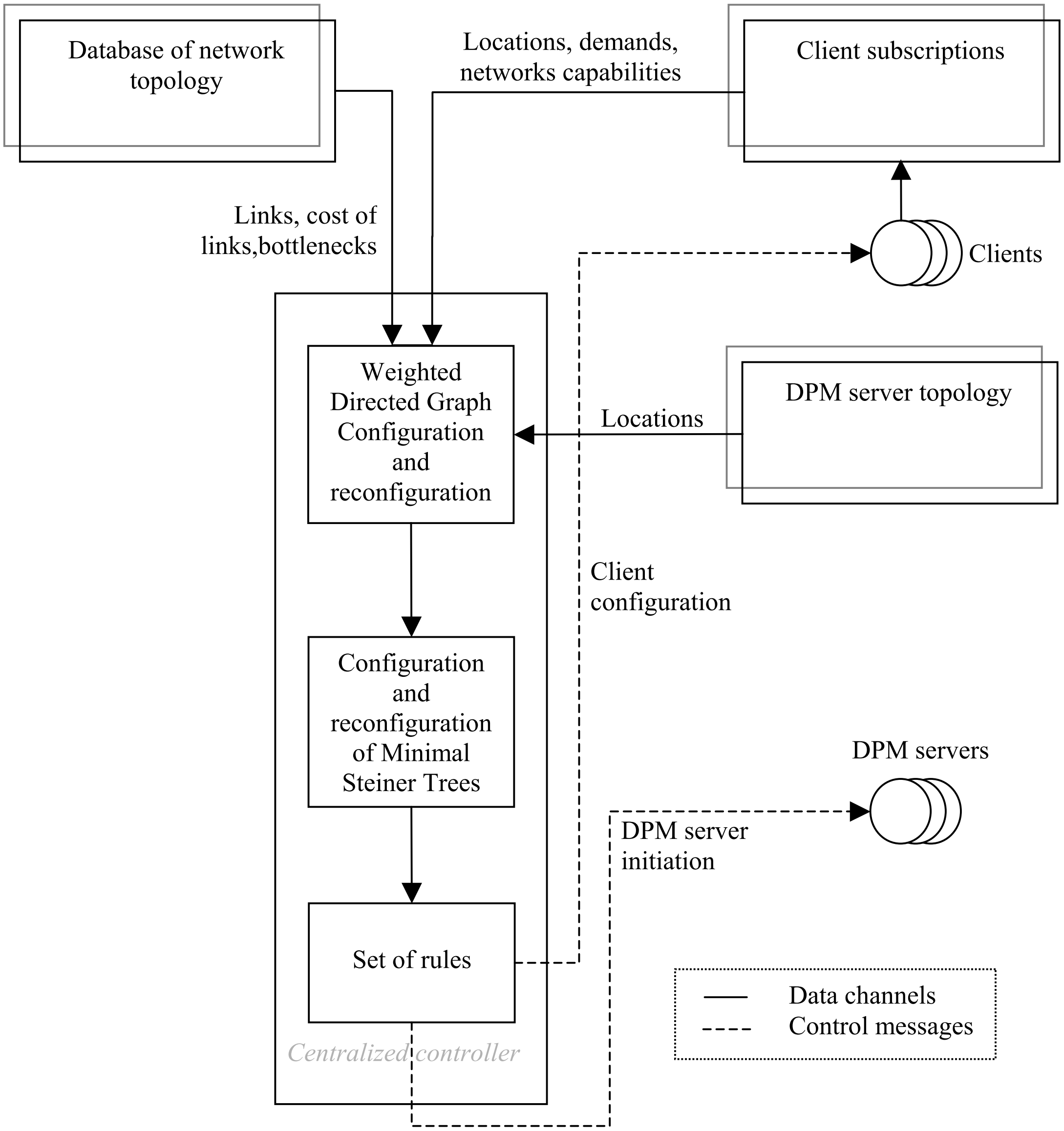

Database of network topology
Locations, demands, networks capabilities
Client subscriptions
Links, cost of links,bottlenecks
Clients
Weighted Directed Graph Configuration and reconfiguration
DPM server topology
Locations
Client configuration
Configuration and reconfiguration of Minimal Steiner Trees
DPM servers
DPM server initiation
Set of rules
Data channels
Control messages
Centralized controller

Figure 5.4 Static centralised DPM service

More specifically, when a subscribed user joins the application, they use pre-allocated DPM servers and they are instructed to communicate only with pre-determined DPM servers. This is all transparent to the user. Network robustness is decreased with a vulnerable central point of failure [Lin, F., Yee, J., 1989], [Frankel, M., 1987]. Since the knowledge of the users' locations is available before session configuration, and access rights to position DPM servers at the suitable locations are also available, this approach can work well for both even and uneven user distributions.

Storage and memory requirements for this approach can be very large. Propagation delays for re-configuring the users in response to changes are potentially high. The combination of high overheads and low robustness makes such centralised, relatively static scheme very unattractive for use in more dynamic networks.

5.5 Self-organised DPM over unmanaged networks

The example architecture illustrated in the previous section has several drawbacks, in particular it has very limited support for dynamic user membership and changes in the topology. Support for dynamic user membership refers to allowing users to join and leave the application at any time, and having the architecture respond dynamically to these events. Support for changes in topology refers to allowing changes in a distribution topology so that DPM system as a whole can respond gracefully to varying network conditions and/or user membership. This means that even a topology of DPM servers that was optimal for one set of network conditions (when it was constructed) might well be suboptimal when these network conditions change. For example, two nodes that were estimated to be topologically close at one time might not be considered close if the link between them became congested or went down. In the managed networks (assumed in the previous example) such unexpected and severe fluctuations in link quality are assumed to be infrequent, so that centralised fixed control is suitable. In order to accommodate more unexpected and frequent changes, the topology needs to be able to gracefully change over time, i.e. continuously evaluate the appropriateness and contribution of each distributed partial mixer to the whole system.

The example architecture given in this section aims to support user membership and topology changes, and be fully distributed and collaborative. In a collaborative distributed approach the group of nodes collaborates when determining their connections and configuring themselves. This means that they must take into account not only the optimal solutions for a single node but also for the other nodes as well. This is in contrast to individual distributed approaches, in which every node works in isolation from the other nodes and optimises the connections only according to its point of view. Collaborative approaches are usually more efficient because a collaborating node uses the knowledge of other collaborating nodes in order to make decisions and thus does not have to gather all the knowledge alone. More detailed discussion about collaborative and independent approaches is given in the section 5.6.2.5.

In the previous section distributed partial mixing servers were assumed to be started on pre-determined machines and be possibly idle before the establishment of a DPM session. In this example the distributed partial mixing service is started on demand, potentially at any time and in any part of the network. Moreover, the distributed partial mixing service does not depend on dedicated servers, as any client machine can potentially act as a DPM server. This second example is modelled on the self-organised transcoding (SOT) approach proposed by [Kouvelas, I., Hardman, V., Crowcroft, J., 1998]. The basic idea adopted is to use self-organisation to form groups out of co-located receivers, which share loss events, and to provide local adaptation though the use of a DPM service. The idea behind self-organisation usually is to have the end clients self-organise into a multi-level hierarchy of multicast groups, where each individual group corresponds to the homogeneous regions within the heterogeneous multicast tree regions. In this way the resulting topology is congruent with the underlying multicast tree, and one large heterogeneous and difficult problem is reduced to many small, homogeneous and simple sub-problems. The metric used to identify groups in this particular proposal is based on shared loss patterns among end clients. This and other similar metrics are discussed in greater detail in [Handley, M., 1997], [Kouvelas, I., Hardman, V., Crowcroft, J., 1998], [Liu, C., Estrin, D., Shenker, S., Zhang, L., 1997], [Ratnasamy S., McCanne, S., 1999].

Self-organised DPM can be described in the following way. It is assumed that all of the clients joining the session belong to a common (prearranged) multicast group (G_0) over which they can exchange various session messages and audio streams. When a client starts detecting packet loss, it requests a DPM server by multicasting a query to the whole multicast group. In this request message, the requesting client (requestor) gives its loss pattern (i.e. packet loss history from each source) and its own identity. All of the clients that share the same loss pattern as the requestor, should decide to join the requestor and use the same distributed partial mixing server. Members of the same loss group are likely to be behind the same bottleneck link(s) and therefore likely to need the same rate adaptation server (the same DPM server).

All of the other clients that receive this query determine whether they have better reception by comparing the received loss pattern with their own loss pattern. If they have better reception, they send a response (via unicast) to the requestor, indicating that they could act as a DPM server for the requesting client. Their response message contains their loss pattern as well as their distance to the requestor (e.g. TTL, time).

Once the requestor has gathered all of the available responses, it determines the best one and selects that client as its distributed partial mixer. It then multicasts the address of the selected distributed partial mixer in order to both to notify the offering client that is going to provide the distributed partial mixing service, and to instruct all of the members in the same loss group to switch to the new multicast group (G_1) to which this new distributed partial mixer will send. The newly elected DPM server can multicast the audio streams to the group it is serving if all of the streams are forwarded or if some of the streams are mixed but the mix does not include audio from any of the group members. This is important in order to allow the receiving clients to do echo cancellation. If a DPM server needs to send a mixed stream that includes audio from a group member(s), the DPM server will need to have an independent partial mixer for each group member and unicast a custom mix to that particular client (that will avoid echo). Note that the new DPM server should send its audio with all the other audio streams that it receives from other clients. Figure 5.5 illustrates this process in the case when the problematic link is congested only in one direction and there is only one group of clients affected by the same problem ($C_{1\text{-}3}$). Clients $C_{1\text{-}3}$ receive the audio streams from clients $C_{4\text{-}8}$ via C_{4DPM} on the new multicast group G_1. This new multicast group is also used for local communication (sending and receiving audio streams) among clients C_{1-3}.

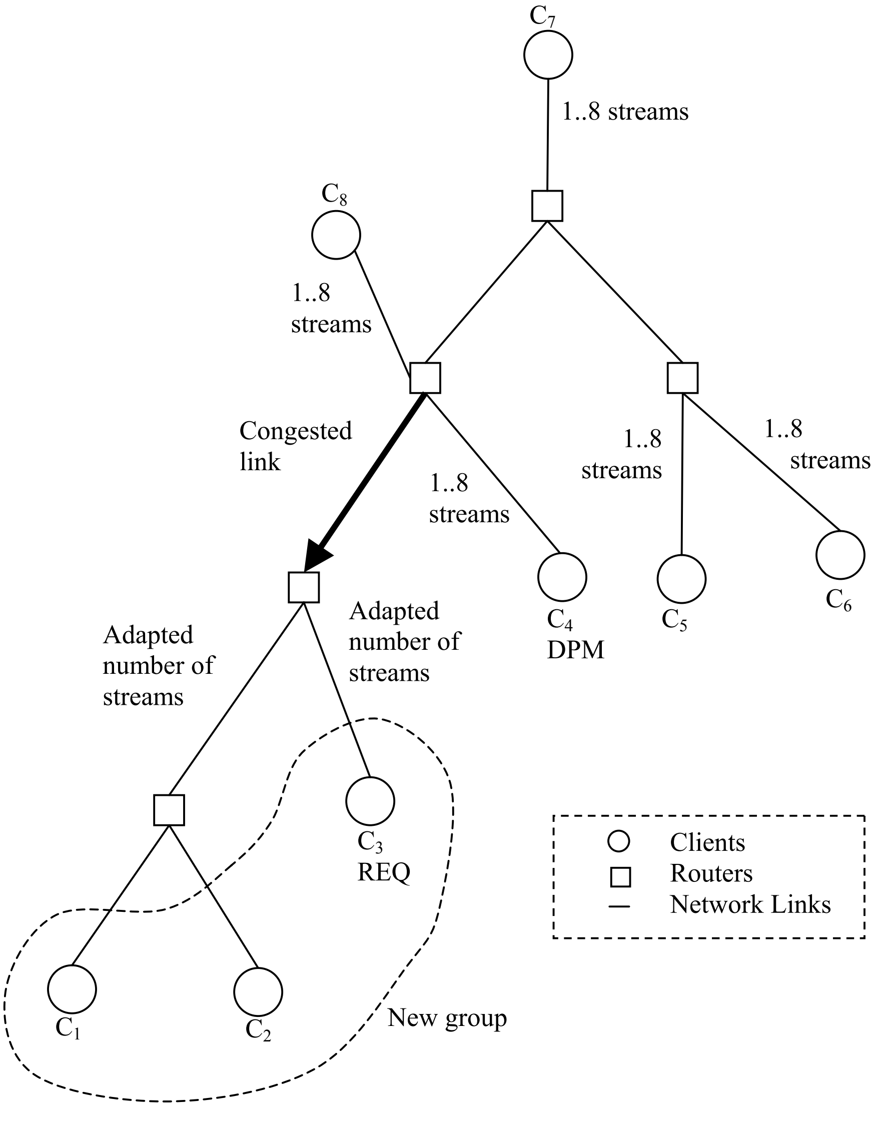


Figure 5.5 Self-organised DPM service

Figure 5.6 shows the scenario when the link is congested in both directions. We assume that C_3 is elected to perform DPM functionality for the group of clients on the other side of the link C_{4-8}. We also assume that C_4 is elected to be C_{4DPM} server for C_{1-3}. Two new multicast groups are used for achieving this communication. Clients C_{1-3} use G_1 for receiving audio from C_{4-8} via C_{4DPM}, and for communicating among each other. Similarly, clients $C_{4\text{-}8}$ use G_2 for receiving audio from $C_{1\text{-}3}$ via $G_{3DPM,}$ and for communicating among each other.

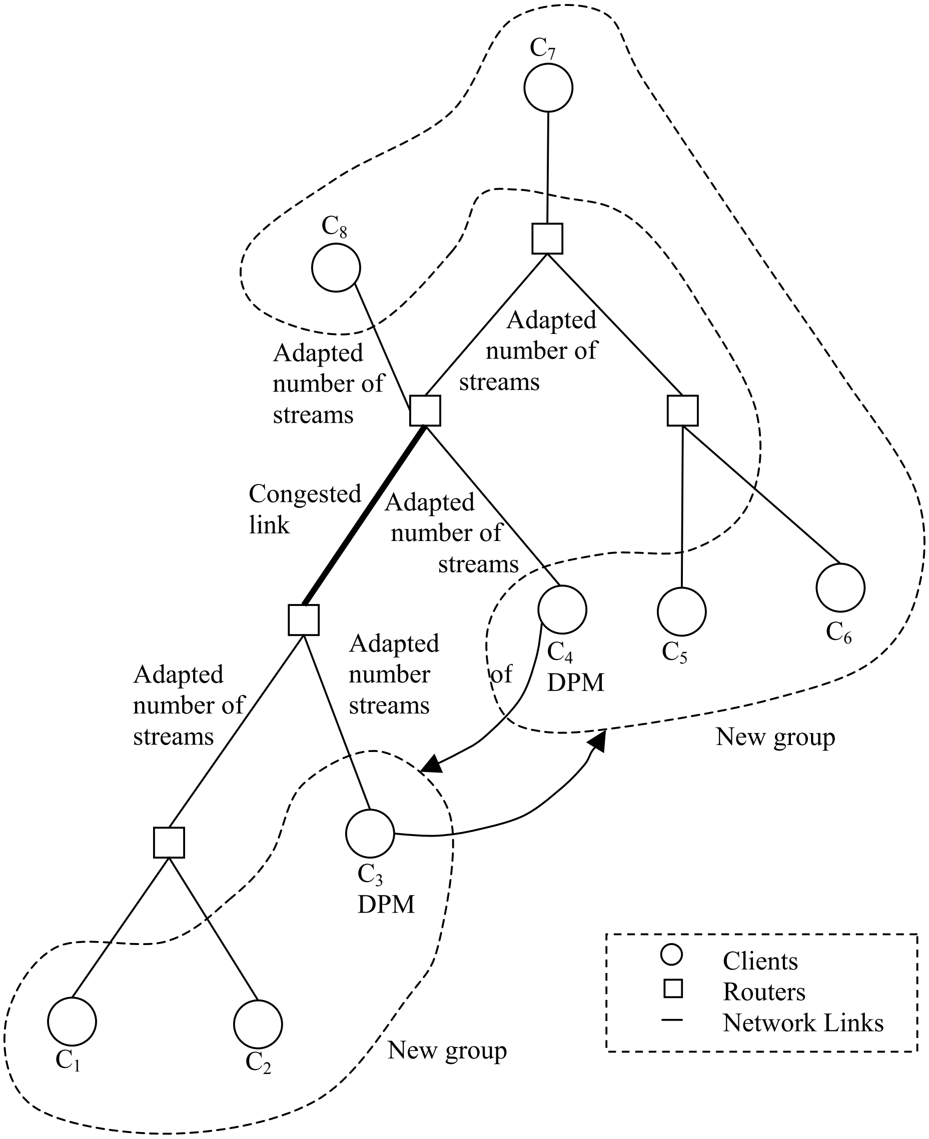


Figure 5.6 Partitioning the original multicast group into multicast islands when the problematic link is congested in both directions

This communication is shown in Table 5.2. The table also shows systematically how the original multicast group (G_0) gets partitioned into multicast islands (G_1 and G_2) each with their own DPM server (C_{3DPM} and C_{4DPM}). Each member in an island sends audio to its local multicast group, and one member forwards the audio to the other multicast island. Multicast partitioning is important because a new member joining the original multicast group would always pull down all traffic across the congested link.

	Receive from/via										
Send to		C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_{3dpm}	C_{4dpm}
	C	G_1	G_1	G_1	×	×	×	×	×	G_1	×
	C_2	G_1	G_1	G_1	×	×	×	×	×	G_1	×
	C_3	G_1	G_1	G_1	×	×	×	×	×	G_1	×
	C_4	×	×	×	G_2	G_2	G_2	G_2	G_2	×	G_2
	C_5	×	×	×	G_2	G_2	G_2	G_2	G_2	×	G_2
	C_6	×	×	×	G_2	G_2	G_2	G_2	G_2	×	G_2
	C_7	×	×	×	G_2	G_2	G_2	G_2	G_2	×	G_2
	C_8	×	×	×	G_2	G_2	G_2	G_2	G_2	×	G_2
	C_{3dpm}	G_1	G_1	G_1	G_2	G_2	G_2	G_2	G_2	G_1	G_2
	C_{4dpm}	G_1	G_1	G_1	G_1	G_1	G_1	G_1	G_1	G_1	G_2

Table 5.2 Multicast groups used for communication between the clients divided into two loss groups on the opposite sides of the congested link

This self-organisation approach gracefully accommodates new clients joining and old clients leavening. When a new client joins, it can probe for existing loss groups and existing distributed partial mixers in its network neighbourhood before it joins one. Potential implosion of responses to a query can be avoided in a number of ways that can limit the number of responses such as: [Bolot, J., Turletti, T., Wakamn, I., 1994], [Barcellos, M., Ezhilchelvan, P., 1998], [DeLucia, D., Obraczka, K., 1997]. If the requesting client leaves, some of the other clients in the groups must schedule a message to take over the role of the requestor. A distributed partial mixer can also detect the departure of the last member and stop sending to the group (using self-limitation when no acknowledgements are received as presented in chapter 4).

This approach also copes gracefully with new bottlenecks, resolution of the old ones and even link and router downtime. For example, introduction of new bottleneck links will cause some members of the loss group to detect different loss patterns to those of the requestors, and start the process of discovering a new distributed mixer again. In order to prevent too many simultaneous requests for a new DPM server, each client uses randomised timer for scheduling its request. The other clients with the same request, cancel their request and wait until the requestor locates the DPM server.

5.6 Discussion

5.6.1. Summary of the two proposals

This section summarises these two contrasting examples of DPM deployment in terms of the issues raised in section 5.3. The table 5.3 summarises how each of these two approaches copes with each of the issues.

Static centralised approach for DPM deployment

The deployment domain in the first example was strongly managed: client locations were known in advance (before session configuration) and distributed partial mixing servers were positioned in the suitable places across the network by the centralised authority.

Placement of DPM servers was done manually by a centralised authority before the beginning of the session. The advantage of this approach is that total machine resources are used for the distributed partial mixing.

The discovery and advertisement of DPM servers was not needed because the centralised authority had global knowledge about the system and informed all of the nodes about the rest of the nodes.

DPM initiation was done also by the centralised authority before the session and did not change throughout the session.

Topology was determined by static algorithms for finding minimum shared trees and limiting the height of the tree using some of the mentioned heuristics.

The process of topology management is fully centralised, thus resulting in large demands in terms of memory, storage and CPU requirements. It is also a single point of failure for the whole system.

Support for changes in the topologies and clients was limited since the fixed set of rules was used independently of the changes in the topology and user membership over time. This is however acceptable for fully managed scenarios and private networks where the traffic is controlled and thus stable.

Deployment domain	**Managed**	**Unmanaged**
DPM placement	-In-advance -On dedicated machines -On suitable places	-On client machines -On suitable places
DPM discovery	-Centralised authority with global knowledge updates all clients and DPMs	-Distributed discovery: based on advertisements and collaboration between the client
DPM initiation	-Centralised authority: pre-configured DPMs and client before the start of the session	-Demand-driven (election and self-evaluation) in order to improve the reception quality for a group of co-located clients
Topology determination	-Minimum shared trees (MST), shortest paths trees (SPT), Minimum set coverage (MCS), use of heuristics -Global optimisation	-Loose groupings of client based on shared loss and served by the clients with better reception -Local optimisaiton
Topology control	-Fully centralised -Static	-Fully distributed, self-organised, includes flooding
Support for re-configuration	-Limited support (depends on the algorithms used for topology determination and control)	-Highly re-configurable: any client can initiate re-configuration

Table 5.3 Summarising the choices in terms of issues for the two given illustrations

Self-organised DPM

The deployment domain of the second example architecture was unmanaged, ad-hoc, dynamic, and more suitable for the Internet deployment. No knowledge about the network, clients, DPM servers was assumed. No administrative and access rights were assumed.

The placement of DPM servers was done dynamically on the clients' machines. The placement was demand-driven and fully managed by collaborating clients. When requested by a client, multiple other clients might offer to be DPM servers, with the requesting client electing only the most suitable offer.

Discovery of DPM-capable nodes was based on flooding and collaboration among the nodes. Flooding included advertising the request for a suitable DPM (via multicasting loss pattern) and collecting unicast-based responses of potential DPMs that advertise their reception quality. Collaboration referred to discovering DPM–capable nodes to cover potentially a group of clients and not only a single client.

Initiation of DPM capability was done on-demand by a requesting client on a behalf of a group of clients. Many DPM-capable nodes could be initiated during the session and old DPM capable nodes could be deactivated when not needed.

Topology determination was based on making local optimisations, therefore ad-hoc and changing during the session. The main goal when determining the topology in this example is to optimise reception quality of a local group.

The process of topology management was fully distributed but based on finding optimal solutions for groups of clients affected by the same problem. Groups of clients were coordinating their efforts when locating the new DPM-capable servers. This scenario does not need any external or centralised control because each DPM server is responsible for its own configuration and dynamic control.

Topology re-configuration was dynamic, ad-hoc, fully controlled by the clients and provided graceful support for dynamic user membership and changes in the underlying network topology.

5.6.2 Discussion of other approaches to realizing DPM over WANs

This section discusses the issues introduced in section 5.3.1. For each issue, a range of approaches is discussed and compared in terms of how beneficial they are to realising large scale DPM service. This includes the two examples summarised in section 5.6.1, as well as other possible approaches.

5.6.2.1 Deployment domain

As well as strictly managed and strictly unmanaged deployment domains, it is likely that where will be mixed deployment domains in today's Internet. With the heterogeneity of users' equipment and networks, it is not difficult to imagine that large scale applications (such as Inhabited TV) may have some users using resource reservation protocols over integrated and differentiated services networks, while the others may run over best-effort unmanaged networks. Advantages and disadvantages of these two deployment domains are systematically presented in Table 5.4.

Deployment domain	**Advantages**	**Disadvantages**
Managed	-Full knowledge available (possible to determine optimal positions for the DPM servers) -Usually more stable networks -All administrative rights available	-Full knowledge required -Less dynamic - More expensive
Unmanaged	-Cheaper -Allows more ad-hoc service deployment -More dynamic	-No advance knowledge about the network topology and users -More difficult to learn about the network in order to control and configure the topologies

Table 5.4 Advantages and disadvantages of heterogeneous deployment domains

5.6.2.2 DPM Placement

The two examples provide radically different DPM placement mechanisms: a central authority placing DPM functionality on dedicated machines at (hopefully) optimal places in the network before the session starts; and electing a client to perform DPM functionality when requested by clients affected by a similar problem (during the session, when it is needed). There is also a range of other mechanisms for DPM placement which lie in between these two extremes. For instance, DPM functionality can be placed on dedicated machines but by local authorities so as to match administrative domains, e.g. every university system administrator could be responsible for allocating and dedicating a university machine to perform DPM functionality. Such a DPM server could later be configured by the centralized authority or application, or even self configured as will be better explained in the later sections on DPM discovery and DPM topology control. This approach needs simple administration and is very good for dial up connections because it is based on static (regional) connections.

Another strategy for placing DPM functionality can be on designated client machines. In principle, any client machine that runs the audio server is able to perform DPM functionality. Issues such as limited CPU time, memory and processing capabilities and security considerations can prevent some client machines from being willing to perform DPM functionality. Such client machines can be determined before the session starts (by the centralised authority or self-evaluation) or during the session (also by the centralised authority or through self-evaluation and election). Client machines performing DPM are generally less reliable (than dedicated machines), they allow less performance for the end user and more difficult join/leave handling.

There can also be a combination of these approaches, such as placing DPM functionality on some dedicated machines but allowing the client machines to perform DPM as well.

Note that DPM functionality need not be placed on any particular machine before there is demand for that service. In this case DPM placement and initiation coincide. These cases will be discussed in greater detail in the section on DPM initiation.

General advantages of using dedicated DPM servers is that they have full CPU load, memory and storage space dedicated to performing DPM and therefore resulting in each DPM server serving more clients than a non-dedicated one could. The disadvantage is that in unmanaged networks it is difficult to determine the total number of DPMs necessary to support some sessions (because it depends on the number of clients and that can vary significantly), find optimal positions for dedicated DPM servers (as there if no knowledge about the actual underlying network) and have access rights (and resources) to place and configure the servers at these locations. The advantage of having DPM on client machines is that the administrative problems are removed since the users themselves offer to perform DPM. General advantages and disadvantages of various placement schemes are systematically presented in Table 5.5.

Placement	**Advantages**	**Disadvantages**
Client machines	-No administrative rights needed -Very efficient use of resources -Cheaper -More dynamic (allows for ad-hoc service deployment)	-Possible low CPU, memory and storage space -Less reliable -Less performance for user -Difficult Join/leave handing
Dedicated machines	-Full CPU load, memory and storage space available for performing DPM -More reliable -More performance for user -Better join/leave handling	-Administrative rights required -Additional resources required
Optimal locations	- Efficient DPM functionality in terms of congestion control -High availability	-Explicit global knowledge required about the network and users (difficult for unmanaged networks) -All rights required
Regional DPMs	-Simple administration -Coincide with administrative domains -Static, regional connection is good for dial-up connections	-Sub-optimal number and placement of DPMs -Possibly not efficient congestion control of DPM

Table 5.5 Advantages and disadvantages of various placement schemes

5.6.2.3 DPM initiation

The two examples provided very different mechanisms for DPM initiation: DPM initiation on pre-determined machines before the start of the session by the centralised authority; and demand-driven fully distributed DPM initiation on client machines through election and collaboration of client nodes. There is a range of approaches that lies in between these two extremes. For example, centralised authority that is an ideal observer (has global knowledge of the system from the beginning of time) can direct (initiate) some of the client (or dedicated) machines to start performing DPM during the session if it estimates that there is need for it. Centralised schemes that perform global monitoring and use adaptive algorithms for determining shared trees and shortest path trees can initiate DPM functionality on-demand on suitable machines, or stop already-running DPM servers if they are no longer useful. The advantages of this approach are more efficient use of resources than static pre-determined initiation, the possibility of having global optimal solutions, and support for dynamic scenarios. The disadvantages are increased propagation delays (from the centralised authority to the local regions affected by the problem), and large overheads in terms of memory, storage space, CPU load and bandwidth at the centralised authority.

Decentralised initiation (done by clients only) is another possible approach for DPM initiation that can include decentralised self-initiation and election. Decentralised self-initiation is based on local self-evaluation of appropriateness and randomisation of local actions to avoid oscillations [Sharma, P., Estrin, D., Floyd, S., et al, 1998]. Election-based decentralised schemes are based on a group of clients that is able to detect the need for a DPM server and initiate it without any administrator rights [Kouvelas, I., Hardman, V., Crowcroft, J., 1998]. Each client (DPM) can also individually choose and initiate its own DPM server, e.g. by locating and connecting to its closest DPM server. However, given that clients and DPM servers do not have to be evenly distributed across the network, and clients make their choices in isolation from the others, such schemes can result in inefficient client coverage, e.g. having too many or too few DPM servers for the number of clients.

The disadvantages of decentralised client-based initiation schemes include implementation complexity in order to support communication between nodes, possible oscillations, implosion of responses, explosion of requests, and suboptimal client coverage. The underlying requirement for such schemes is that the clients in a group should collaborate among themselves in order to improve the client coverage and avoid oscillations. The advantages of these approaches include: efficient use of resources, dynamic and timely responses.

Advantages and disadvantages of various DPM initiation schemes are systematically presented in Table 5.6.

DPM initiation	Advantages	Disadvantages
Pre-determined	-Simple -Good for static scenarios	-Idle (inefficient use of resources) -Bad for dynamic scenarios
On-demand centralised	-Relatively efficient use of resources -Has global knowledge and can therefore make globally optimal solutions -Supports dynamic scenarios	-High overheads -Propagation delays (depends on the position of the central authority)
On-demand distributed: Election/ Self/election	-Efficient use of resources -Dynamic and timely responses: gracefully supports user and topology dynamics -Low overheads	-Possible oscillations, explosions and implosions of requests/replies if nodes do not collaborate -Can result in sub optimal client coverage if the nodes do not collaborate - more complex to implement

Table 5.6 Advantages and disadvantages of various DPM initiation schemes

5.6.2.4 DPM discovery

The two examples provided two very different approaches to DPM discovery mechanisms: central authority with global knowledge updating each DPM server and client about what their connections (children, parents, peers); and distributed discovery and evaluation of peer, child and parent connections. There is a range of approaches that can allow DPM servers and clients to learn about the system without being totally passive or self-organised. The basic techniques that can be used for discovery protocols are: multicast, expanding ring search, expanding ring advertisement, and use of directory services. Using multicast for sending queries and receiving queries is a very simple approach to locate the services and machines whose locations are not known. For example, if all clients (or DPM servers) belong to the same multicast group, it is easy to get information about every element. However, given that the number of members in the multicast group can be very high, such queries can result into response implosions and explosions of requests that might lead to congestion. The main advantage of this approach is that it is simple to realise.

Expanding ring search (ERS) refers to sending queries of interest to a multicast groups by limiting TTLs of the queries. A new joining client can use this approach to locate its nearby (regional) DPM service placed on the dedicated machine and being idle until clients (or other DPM servers) discover it and start using it. A very similar approach to ERS is advertisement of some nodes where the nodes send information about themselves to a certain multicast group or with increasing the TTL. Dedicated DPM servers that are idle can use this approach until they find their regional clients and other neighbouring DPM servers. Each DPM (client) can periodically list all of its neighbours and how far away they are. These messages can be flooded throughout the network so that every member can build up their own map of all the nodes and connections among them.

All three approaches are based on flooding principles and are fast and simple to implement, but may need a lot of bandwidth, memory and storage space. Even though these approaches limit TTL, they can still cause response implosions since the distribution of DPM services and clients need not be even or increase linearly with TTL (e.g. it may be the case that for initial TTL values there is no response, and then with one additional TTL increase a large number of nodes reply).

The major issue in discovery protocols is how much knowledge about the system one DPM server needs to have. For example, each DPM server can try to gain a complete knowledge of the system, but it can also limit its knowledge to only its neighbours and local clients. Global knowledge results in high overheads but also high robustness of the system. Localised knowledge results into lower overheads, but can also result in loops if nodes/links go down, and inability to perform global optimisation.

Server-based discovery can also be used for locating and discovering clients and servers. This can include directory services, that can be centralized or distributed data repositories that can be used over the Internet. A directory is a database that contains descriptive, attribute-based information. The information in a directory is generally read much more often than it is written (there is a general tradeoff between data replication and updating). As a consequence, directories do not usually cope well with the extensive and frequent updates that are needed in dynamic networks. Directories are usually intended to give quick responses to high-volume lookup or search operations. They may have the ability to replicate information widely in order to increase availability and reliability, while reducing response time. When directory information is replicated, temporary inconsistencies between the replicas are possible. The Lightweight Directory Access Protocol (LDAP) [Yeong, W., Howes, T., Kille, S., 1995] is a protocol for accessing online directory services. It runs directly over TCP and is based on a *client-server* model. One or more LDAP servers contain the data making up the LDAP directory tree. Some directory services are *local*, providing service to a restricted context (e.g., the finger service on a single machine). Other services are global, providing service to a much broader context (e.g., the entire Internet). Global services are usually *distributed*, meaning that the data they contain is spread across many machines, all of which cooperate to provide the directory service. Typically a global service defines a uniform *namespace* which gives the same view of the data no matter where you are in relation to the data itself.

The advantages and disadvantages of various discovery mechanisms are systematically presented in Table 5.7.

Discovery	Advantages	Disadvantages
Global knowledge	-Full knowledge: consistency and robustness	-Flooding -High overheads
Local knowledge	-Smaller overheads -Smaller bandwidth requirements	Possible inconsistencies when changes happen
Multicast	-Simple (can locate remote nodes)	-Flooding -Implosion of responses -Explosion of requests -Idle until they are discovered or they discover
Expanding ring search/TTL-based advertisement	-More limited scooping (smaller bandwidth requirements) -Simple to implement	
Directory Services (LDAP-based)	-Good for often reading -Fast response to high volume lookup or search operations -Information can be replicated to assure reliability and reduce response time	-Not good for often writing -Not fast enough for real-time multimedia (TCP-based) -High overheads

Table 5.7 Advantages and disadvantages of discovery mechanises

5.6.2.5 DPM topology determination

The two approaches provided very different measures of DPM topology suitability: a minimal shared tree with the use of heuristics to improve delays and fan ins and outs of DPM servers; and dynamic clustering of nodes into loss groups based on shared loss patterns to improve local reception for the groups. So the first approach was determining the suitability of the topology by trying to find a globally optimal solution for the whole system. Such approaches are based on centralised knowledge or on flooding techniques. Each DPM independently solves the optimisation problem from its point of view for the entire network. The main advantage of these approaches is that they sometimes respond faster and avoid loops after some changes. But the disadvantages are that they have substantially more information both in their internal structure and in messages the exchanged.

The second example showed a “divide and conquer” approach in which it was determining locally optimal solutions and building a global topology through topological localisation of problems. The fundamental feasibility of such approaches was studied thoroughly in [Ratnasamy, S., McCanne, S., 1999]. “Divide and conquer” approaches rely on each DPM server exchanging data and information only with its neighbouring DPM servers. In this way each DPM server solves only part of the problem and needs to receive only a portion of the total data (the impact of the control traffic is also localised). Disadvantages of these approaches are that they have to ignore data after some changes to prevent loops. Advantages of these approaches are that they have much lower overheads and improved accuracy. However, a range of other approaches can be used for topology determination. One approach is clustering together the nodes based on the latencies measured among them, i.e. only nodes with smaller than a certain threshold latency can be considered as a cluster of nodes, and one of them can look for their parent nodes (DPM server). A similar approach is to cluster the nodes according to their TTL distance so that a common DPM server can serve a sub tree of nodes where the TTL is not bigger than a certain value. Administratively-scoped zones can also be used to build a tree of clients and DPM servers where each administrative zone is served by a parent DPM. In this example, the founder of the group is its representative. An advertisement-based approach can also be used for dynamically adapting a hierarchy of servers [Hoffmann, M., 1997]. Another approach could be based on interest profiles and co-location of clients, whereby co-located clients with similar interest profiles can be collocated together and served by the same server.

Generally, the looser the grouping of servers and clients in the clusters is, the more dynamic and flexible the topology is. If members of a cluster are not loosely grouped, then it will be more difficult for the topology to support sessions where clients might join and leave at any time.

The general overview of advantages and disadvantages of various aspects of topology determination is shown in Table 5.8.

Topology determination	Advantages	Disadvantages
Global suitability approach	Globally optimal solutions	High overheads Delays
Divide and conquer approach	-Local optimisations -Low overhead, good accuracy -Timely response	Possible inconsistencies
Tight grouping: clustering based on TTL, delays, administrative-based approach, advertisement-based topology	-Simple, clear connections	-Flooding the network -Not flexible enough -May result in sub-optimal topologies (i.e. sub-optimal coverage of clients and DPM servers)
Loose grouping: loss groups	-More flexible -Groups only the nodes affected with the same problem	-More complex to implement -Danger of false sharing
Shared trees and the use of heuristics	Single tree/easier	-Traffic concentration -Not optimal for all nodes
Shortest Path Trees and use of heuristics	Lower delays between the nodes (more optimal trees)	-High overheads (CPU, memory, storage) -High bandwidth

Table 5.8 Advantages and disadvantages of various topology determination schemes

5.6.2.6 Managing the process of topology determination

The two examples illustrated radically different processes for managing the determination of topology: fully managed, static and centralised, and fully distributed, dynamic and self-organised.

Besides these two contrasting approaches, a range of other approaches for controlling topology determination is possible. For example, centralized approaches can include: an ideal observer that provides total information about the network and the nodes in the system, but where the nodes make the actual connections; a centralised controller that gathers all of the necessary information and measurements from all of the DPM servers and clients, determines the topology and instructs them with a set of rules; and a centralised controller that has total knowledge, but also instructs individual nodes what to do and where to connect to (second example).

In distributed approaches each node can solve only part of the problem and need only part of the information. This usually results in reasonable accuracy and low overheads but with the possibility of loops. Alternatively, each member can independently solve the optimisation problem from its point of view for the entire network. This is usually faster than the previous approach and results in significantly higher overheads, but no loops.

Distributed approaches can generally be divided into independent and collaborative approaches. In independent approaches each node works in isolation from the other nodes, gathering information alone and then determining its connections on its own. These approaches can be fast but they have high memory and storage requirements and also very high total bandwidth requirements because each node does all of the discovery alone.

In collaborative approaches on the other hand, even though the nodes take time to organise and synchronise among themselves, each node benefits from the computations done by its neighbours and thus reduces storage, memory and processing requirements as well as bandwidth requirements. If group dynamics is allowed, each group should monitor its own state and handle splitting and merging of groups. Even though the second example DPM deployment is distributed and collaborative, it is still based on global exchange of information about the packet loss patters among all participants, i.e. it is based on global flooding of information.

A hybrid approach of distributed and central control is also possible. Some schemes (e.g. the delta routing algorithm) take advantage of global updates from a centralised manager plus more accurate information from its neighbours to rapidly adapt to local changes [Rudin, H., 1976]. Table 5.9 systematically presents the advantages and disadvantages of various types of topology control.

Topology control	**Advantages**	**Disadvantages**
Centralised observer/controller	-Global knowledge (high robustness: no loops)	-High CPU load, memory, storage -Single point of failure -Propagation delays
Replicated observers/controllers	-No single point of failure	-Either high overheads (if global knowledge is replicated) or need to collaborate (if every controller keeps local information)
Distributed independent	-Fast	-Flooding (global information exchange) -High CPU, memory, storage
Distributed collaborative	-Benefits from calculation done by neighbouring nodes -Reduced memory, CPU, storage requirements -Localised info exchange -Many decisions are made and actions taken within the nodes that collaborate and without flooding the other non-affected nodes collaborate	-Nodes take time to organise -Some schemes can include flooding of the network in order to discover or locate some information
Hybrid (central and distributed)	-Faster response to local changes while having general overview of the system for non time-critical operations	-More complex to implement

Table 5.9 Advantages and disadvantages of various topology control mechanisms

5.6.2.7 Support for re-configuration

The two contrasting examples illustrate a static scheme with pre-determined globally optimal topology that does not support any changes, and a totally dynamic scheme that gracefully supports dynamics in terms of client membership and topology changes but does not necessarily provide optimal topology for all the nodes at all times. These two examples illustrated non-adaptive and highly adaptive topologies of DPM servers and clients.

Adaptive topologies are needed for today’s networks, which can have unexpected fluctuations in link quality, and because the initial topologies may not remain suitable for the entire session lifetime. With explicit knowledge of the topology, it is relatively easy to re-configure the topology with the variety of topologically sensitive protocol optimisations available [Jackobson, V., 1995], [Kouvelas, I., Hardman, V., Crowcroft, J., 1998], [Lin, J. C., Pauls, S., 1996]. However, DPM follows the core design principle of the Internet protocol family, that is based on the best effort delivery service of IP that deliberately hide the underlying network topology. It is not necessary to change any internal network equipment such as routers and switches nor to modify existing network layer protocols because end system can infer sufficient information about the underlying topology strictly from end-to-end observations. A common attribute of many proposed end-to-end multicast schemes is some method of exploiting the structure and topology of the underlying multicast distribution tree. Prior work [Ratnasamy, S., McCanne, S., 1999a], [Kouvelas, I., Hardman, V., Crowcroft, J., 1998] found that a relatively simple model based on shared loss patterns can work well. These approaches are based on measured net shared loss between the receivers but they suffer from shared loss pathology described in [Ratnasamy, S., McCanne, S, 1999a]. This can be briefly described as follows. Shared losses arise in two ways. True shared losses are losses in the end clients caused by the tree structure along the shared path from the root to their closest common ancestor. They are truly indicative of the underlying tree structure. In addition to these true shared losses, each client's loss pattern will also include the packets that are lost along the separate paths form the closest ancestor to each client. It is possible that two copies of the same packet are lost independently along these distinct paths on account of which a portion. These losses are called 'false shared losses' because they are random and not caused by the underlying tree structure. The failure modes that arise in the use of the net shared losses as selection criteria are due to this false sharing. [Ratnasamy, S., McCanne, S., 1999b] propose the use of real shared loss probabilities that combine shared loss patterns with a probabilistic model as a very efficient way of overcome false sharing.

Re-configuration can be initiated by the clients or by the central authority. In principle, it is most effective if the affected group of nodes can detect and initiate the re-configuration of the DPM topology.

A number of approaches have been proposed based on the assumption that shared loss can be used to imply spatial (topological) correlation among the clients. Clustering clients into groups through shared loss thresholding should result in topologies that model the underlying network topology and can therefore adapt to it efficiently. Even though a number of schemes rely on the use of TTL-based scope to cluster and organise the nodes in a topology, TTL is not a good measure of locality [Xu, R., Myers, A., Zhang, C., et al, 1997]. TTL by itself does not provide any information regarding losses, nor does it indicate where the bottleneck link is relative to the clients. However it is good to combine TTL and loss-based approaches for enhanced robustness and scalability. Table 5.10 systematically presents advantages and disadvantages various adaptive and non-adaptive topologies.

Support for re-configuration	**Advantages**	**Disadvantages**
Non-adaptive topologies (centralised or distributed)	-Good for stable networks -Simple to implement	-Do not support any changes in user membership and underlying topologies
Adaptive topologies: distributed based on loss sharing	-Support for unexpected changes in topology and user membership	-Possibility of false sharing but can be improved with probabilistic model
Adaptive topologies: distributed TTL-based	-Simple to implement -Support for user dynamics	- No support for network dynamics: TTL is not a good measure of locality, no information about the losses nor indications where bottleneck links are
Adaptive topologies: Centralised, explicit knowledge and use of topologically sensitive algorithms	-Global optimisations (consistency of the system) - Simple to implement	-Slow response time because of propagation delay -High concentration of traffic close to the centralised authority -High CPU load, memory, storage requirements

Table 5.10 Advantages and disadvantages of support for re-configuration of various topologies

5.7 Summary

This chapter identified and discussed some fundamental constraints and issues arising in large scale wide area network deployment of DPM and proposed ways of dealing with them. It proposed that a minimal shared tree of DPM servers and end users, where the nodes of the tree can perform distributed partial mixing, is an effective basis for wide area deployment. A set of fundamental protocols and heuristics needed for constructing it was also described. Shortest path trees and minimum set coverage trees were also described as alternative solutions for a DPM topology. Advantages and disadvantages of each of these approaches were also described.

The chapter then identified practical issues concerned with the realisation of a large scale DPM service over heterogeneous deployment domains that can significantly impact its efficiency. These issues include: type of deployment domain, DPM placement, DPM initiation, DPM discovery and advertisement, topology determination, service control and support for re-configuration.

Two models for realising a DPM service in two contrasting situations were then explored in more detail: a static, centralised, subscription-based DPM service suitable for a fully managed network, and a fully distributed self-organising DPM service suitable for an unmanaged network (such as the current Internet). The first model has very limited support for dynamic user membership and changes in the topology. The second model allows the users to join and leave the application at any time, and dynamically responds to changes in the underlying topology. A detailed discussion of alternative approaches and their advantages and disadvantages for each of the issues was also provided.

Work on further optimisation of large scale DPM topologies and their extensive evaluation is beyond the scope of this thesis and is a matter for future work.

6. Conclusions

This chapter is divided into three sections. Section 6.1 begins by summarising the work presented in this thesis. Section 6.2 then presents the main conclusions and contributions of this work organised in terms of its philosophy, theory, realisation and use. Section 6.3 reflects on the work presented here and identifies areas of future research.

6.1 Summary

This thesis is concerned with the design and realisation of large-scale adaptive audio services for environments that actively support collaboration between large numbers of simultaneous users.

Chapter 1 motivates the work in this thesis by introducing the problem of simultaneous speakers. With the rapid increase in the deployment of large scale collaborative applications involving real time audio there are severe consequences for ignoring the problem of multiple simultaneous speakers. It is therefore important to understand these issues and build audio services that can control and manage support for large numbers of audio sources as a necessary complement to those that support only large numbers of audio receivers (such as multicast).

Existing audio services and approaches can be used for supporting audio communication in collaborative environments. However they do not explicitly consider the existence of many simultaneous speakers, i.e. they do not address the possibility of very frequent and significant peaks in the audio traffic caused by correlated activity of the audio sources, and usually assume only average bandwidth requirements. Current approaches in CSCW and networking tend to deal with multiple simultaneous speakers in one of two ways: they either consciously disallow it, through floor control or push-to-talk mechanisms or, possibly worse, they refuse to acknowledge it at all. There are currently no schemes to perform control over multiple streams, which is the only way to deal gracefully with large numbers of audio senders.

Audio services that do not consider the peaks in the audio traffic that result from large numbers of simultaneous active senders exhibit a number of problems. These problems can include instability of the application, high packet loss, extreme unfairness towards competing TCP traffic, and even potential congestion collapse. Even mechanisms for single-stream compression and adaptation that deal with very large numbers of receivers are not enough to prevent these impacts when there are large numbers of audio sources.

The work in this thesis therefore focuses on the design of a new network audio service to support such applications, based on a new approach called distributed partial mixing. This thesis promotes the idea of congestion control over multiple audio streams as a necessary complement to existing techniques for single stream adaptation.

Chapter 2 gave an overview of work related to the problem of supporting simultaneous multiple speakers. A series of review criteria was chosen, described and motivated as important for designing scalable audio services: recognising and coping with multiple simultaneous speakers, minimising the packet loss experienced at the end user, maximising the number of independent audio streams received by the user, supporting heterogeneous networks and end systems, having efficient distribution of streams in the network, and being network friendly. Efficient distribution of audio streams is in direct conflict with maximising the number of independent audio streams because it aims to minimize the number of audio streams in the network (by mixing). The challenge is to investigate if it is possible to find a tradeoff between these two criteria. Maintaining certain audio quality (criterion 2) over heterogeneous networks and end systems (criterion 4) could be achieved only through a highly adaptive scheme since in the current Internet we cannot rely on reservation services to guarantee quality of service. These criteria were applied to various existing approaches in order to understand how well existing approaches could support them. Even though each of the existing approaches has its own significant contribution to multimedia networking research, none of them addresses all of the chosen criteria. For example, even though approaches exist that are efficient and adaptive in terms of a single stream (multicast and congesting control schemes), there is none that achieve efficiency and adaptation across multiple streams. Approaches that are efficient in terms of multiple streams (mixing) are not adaptive nor can they maximise the number of delivered streams to the end user. The chapter pinpointed particular features of interest from the reviewed work, such as the use of mixing for minimising the number of audio streams in the network, TFRCP estimation of TCP rate that uses TCP-like timeout estimation, TFRCP loss and delay calculations and RAP's packet loss detection method for achieving adaptive and TCP-fair aggregation of streams in the network. These ideas were selected to be brought forward and developed as part of the main novel approach presented in this thesis, namely distributed partial mixing. The chapter concluded by asking two questions: First, can the audio streams be mixed in a way that is adaptive and friendly towards other traffic in the network, supports heterogeneous networks and minimise the packet loss experienced? Second, how can we do no more mixing than is absolutely necessary i.e. keep the maximum possible number of independent audio streams, while maintaining an efficient aggregation of audio streams?

Chapter 3 introduced Distributed Partial Mixing (DPM) as a novel method for efficient distribution of audio streams through flexible management of the number of audio streams. Each distributed partial mixing component adaptively mixes subsets of its input audio streams into one or more mixed streams, which it can forward to the other components along with any unmixed streams. DPM minimises the amount of mixing performed so that end users receive as many separate audio streams as possible within prevailing network resource constraints. In the first part of the chapter, a large scale collaborative virtual environment, where large numbers of users could be simultaneously active in the audio medium, was proposed as a target environment where DPM can be utilised. A small scale scenario whereby multiple users from two different LANs communicate over a shared link with unknown properties was provided as an example scenario to illustrate DPM functionality. Then, a larger scale scenario with multiple heterogeneous networks and more users was provided to illustrate the high level architectural overview of distributed partial mixing. The chapter then described the functional model that any distributed partial mixing system should follow. The five main functional stages of the system were described and justified: receiving audio streams and monitoring the network links, processing monitored resource parameters, comparing the available and required resources, selecting the streams to be mixed to maximize user experience and having a database to keep user, application and system requirements (and preferences), mixing and forwarding, and transmitting audio streams, feedback and control messages to the other DPM components or clients. The parameters and factors to be considered at each of its functional stages were described and identified. One of the key principles of the proposed DPM functional structure is to keep rate adaptation and the selection process in distributed partial mixing orthogonal, so that the design and implementation of the two processes can carry on independently. This means that network-driven congestion control of DPM is separated from application-specific preferences and requirements. The rest of the thesis focuses primarily on the monitoring, comparing and mixing/forwarding stages of DPM. The last part of the chapter provided an overview of how distributed partial mixing can make mixing decisions based on multiple algorithms for dealing with various aspects of the underlying networks and end systems, as well as application and user preferences for adjusting the perceived quality of the received audio streams within available resources. The chapter included an example design space for different algorithms that can be used for various mixing criteria in DPM.

Chapter 4 described the implementation and evaluation of the method introduced in chapter 3. It focused on network-driven distributed partial mixing, and demonstrated its effectiveness in terms of the criteria motivated in chapter 2. The chapter began by describing a general scenario and the experimental infrastructure used to demonstrate the effectiveness of distributed partial mixing in terms of these criteria. A single unmanaged network link shared by both adaptive and non-adaptive traffic was chosen to validate adaptation of distributed partial mixing. The chapter then specified the architecture of a partial mixer, and described and justified the particular design choices made for each partial mixer component. It then introduced the MASSIVE-3 system within which distributed partial mixing was prototyped, and its audio process which has been extended by the author of this thesis to support distributed partial mixing. The experimental set up and user emulation used to validate and demonstrate to DPM behaviour were described. The first set of experiments demonstrated the responsiveness and stability of distributed partial mixing against step-wise increase in competing non-adaptive traffic. It focused on the steady state and demonstrated the effectiveness of distributed partial mixing in terms of the levels of audio quality that it can provide, given varying levels of network congestion. Audio quality was measured in terms of unrestricted audio activity for each user, packet loss experienced and level of spatialisation (the first three criteria from chapter 2). The second set of experiments focused on the dynamics of distributed partial mixing, and provided results to demonstrate that DPM consumes its fair share of bandwidth when competing with TCP and other distributed partial mixing traffic on the same unmanaged congested link. The chapter concludes that the model proposed in chapter 3 does meet the initially chosen criteria and hence does offer natural support for large number of simultaneous speakers.

Chapter 5 motivated and discussed some fundamental issues to large scale DPM deployment over WANs that might affect and constrain the performance of DPM, and suggested ways of dealing with them. These issues concern topology determination, echo and loop prevention, minimisation of delays between clients, and fan in and fan out of DPM servers. A minimal shared tree topology of DPMs and clients was proposed as a basic approach for supporting many to many communication, and a set of fundamental protocols needed for constructing it was identified. The chapter then identified more practical issues concerning the realisation of a wide area DPM system. It also motivated a set of issues and requirements for a more dynamic deployment of DPM. These issues included: type of deployment domain, DPM placement, DPM discovery, DPM initiation, dynamic topology determination, controlling process, and support for re-configuration. The chapter proposed two contrasting illustrations of how it is possible to deploy large scale distributed partial mixing DPM in very different deployment domains. The managed DPM deployment example described one possible scenario for a subscription based DPM deployment in a future large scale Inhabited TV scenario. The fully unmanaged DPM deployment example described a proposal for ad-hoc deployment of self-organised DPM in the Internet based on [Kouvelas, I., Hardman, V., Crowcroft, J., 1998]. The chapter included extensive discussion of alternative approaches as well as advantages and disadvantages of the approaches described.

6.2 Contributions

This section draws out the main original contributions of this thesis. As noted at the beginning of the previous section the central interests of this thesis revolve around the scaleable and adaptable support for audio in large collaborative environments.

- The work contributes to the philosophy of supporting audio communication in large scale collaborative applications by recognizing the importance of allowing for large numbers of active audio streams. In this way, the applications are more robust and scalable to the traffic peaks that are likely result from large numbers of simultaneous users.

- This work presents a new concept, distributed partial mixing, that extends the traditional concept of mixing by making it adaptive and responsive both to the network and to the application requirements. In this way, congestion control and quality adaptation can be performed over multiple streams. The concept of distributed partial mixing is the subject of a patent application by BT, Patent Application Number, GB 01-03595, April 2000 (PhD studentship was funded by BT) [Radenkovic, M., Greenhalgh, C., 2000].

- The work includes a prototype of distributed partial mixing, realised as an extension of the audio service in MASSIVE-3, but is generic enough to be used in any other CVE or teleconferencing platforms. Distributed partial mixing operates transparently to the users.

- The work presents an evaluation of the design and implementation of the prototype across multiple user and network criteria. The design of the comparator and the congestion monitor was presented in detail, and extensively examined through a series of experiments against other adaptive and non adaptive traffic over the real network links controlled by *dummynet* [Rizzo, L., 1997]. This was published in the Proceedings of the Fifth International IEEE Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services (TELSIKS '2001), Nis, Yugoslavia [Radenkovic, M., Greenhalgh, C., Benford, S., 2001].

- The work demonstrates that distributed partial mixing can combine the benefits of both peer-to-peer and total mixing audio services. With plentiful bandwidth, the system can operate like a peer-to-peer scheme, delivering independent audio streams to each listener, giving maximum individual flexibility and control over what they hear. As bandwidth becomes restricted the distributed partial mixing scheme moves incrementally towards a totally mixed (minimum bandwidth) service, preserving a useful level of audio communication under the widest possible range of circumstances.

- The work successfully achieved this with very small oscillations in times of high congestion, while still being responsive and fair to TCP traffic. Using packet loss event rates and an equation-based approach to rate adaptation makes DPM reasonably stable in terms of the audio quality required by the end user. One strength of this approach is that it makes no assumptions about loss patterns and available bandwidth.

- The work proposes a shared tree solution and a particular design of the input/output of a DPM server as a basic solution for determining the topology of DPM servers and clients that copes with loops and echoes and can minimise the delays and fan in/out using a variety of heuristics.

- The work presented in this thesis describes a possible demand-driven self-organising mechanism for ad-hoc DPM deployment that improves its scalability, robustness, and support for dynamic membership and topology changes. This scheme allows clients with similar interest profiles concerning audio mixing, and/or similar network and system patterns, to be grouped and served by the same DPM server. The work in this thesis also described a fully centralised static approach for deploying DPM in fully managed network scenarios. One possible deployment scenario for DPM was published in [Radenkovic, M., Greenhalgh, C., Benford, S., 1999] and [Radenkovic, M., Greenhalgh, C., Benford, S., 2000].

6.3 Future Work and Reflections

The results from the experiments presented in this thesis are encouraging, validating the effectiveness of the DPM method in terms of network friendliness and preserving audio quality. However, there are still some issues that have not been fully addressed. These are identified below.

6.3.1 User and Application Modeling and Testing

The proposed model might be developed to explicitly address application and user control of distributed partial mixing, and ways might be proposed of representing application requirements and exploring techniques for implementing them in distributed partial mixing.

- For example more intelligent ways of mixing to cope with limitations of user equipment. More sophisticated user, application and end system mixing policies might be designed in a selector and database: various mixing preferences and machine-specific parameters suggested in chapter 3 can be exploited to improve the quality of the delivered audio. In general any available metric for measuring the delivered quality of audio should be fed into the database and the selector of a DPM. With regard to the users, more qualitative analysis can be done to evaluate the subjective effects that distributed partial mixing has, e.g. in terms of varying degree of spatialisation.

6.3.2 Real-world large scale experiments

While the motivational focus of this work has been on the support of potentially large numbers of simultaneous users it has not been possible to perform large-scale trials within the temporal and resource constraints of this work. This is a clear area for ongoing work and is necessary to address the effectiveness of distributed partial mixing and its philosophy in general as a congestion control method over the Internet.

A number of real-world experiments should be conducted to examine DPMs behaviour in real networks. These experiments should help to uncover some of the actual issues that exist in the Internet and that might not be seen in the controlled environments. For example:

- Stricter measurements of the delay and noise that every mixing stage introduces should be done in order to make stronger recommendations about the number of allowed mixing stages between the end users.
- Delays concerning DPM server activation and re-configuration should be measured in order to test the responsiveness of the topology to changes in the network.

- Other criteria as well as packet loss patterns could be used in addition for controlling the dynamics of DPM topology. For example clients with similar interest profiles might be connected to the same DPM server.

6.3.3 Differentiated and Integrated Services

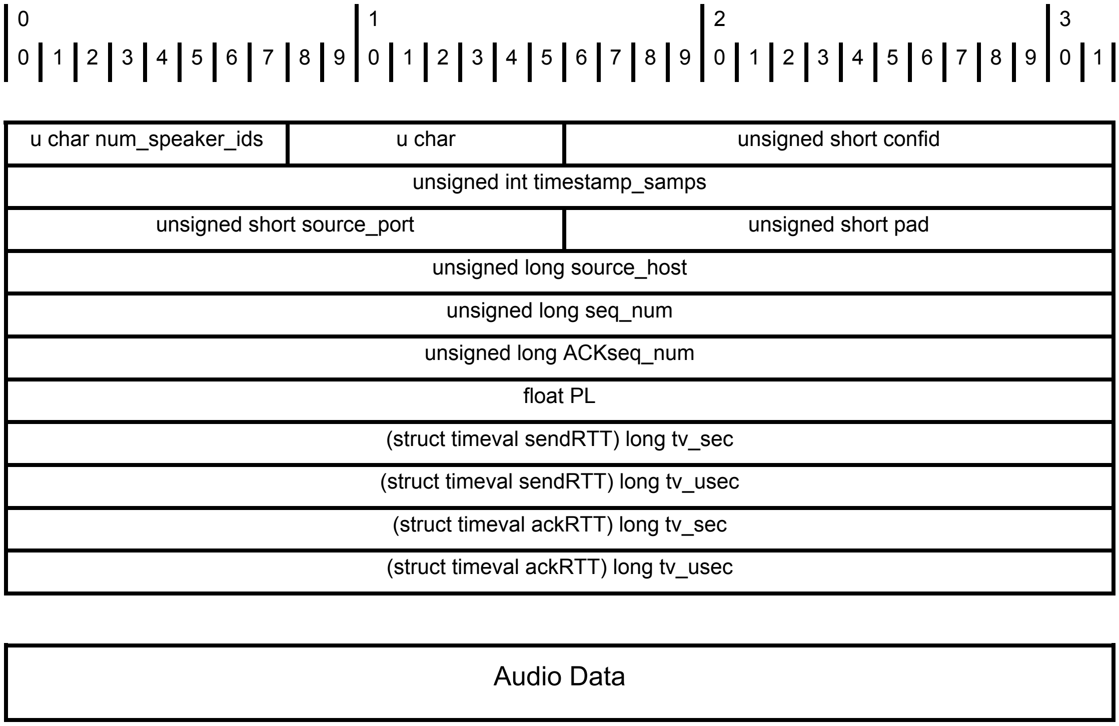
It is likely that Internet will support integrated and differentiated services in the future. Distributed partial mixing could exploit the knowledge that the end systems will have a range of expected qualities of service (a priori) based on the purchased service profile, in order to perform congestion control more smoothly and effectively. Stream differentiation enabled by DPM - due to the functionality of the selector that can choose which streams to mix and which to forward - can be potentially very useful for any interactive application across a network with a range of quality of services. However, this depends on the way that QoS profiles are defined in the future, which in turn depends on the economic model of the future services.

Future work plans bring many challenges, some of which are not straightforward extension of the basic model. However, we believe that the DPM design philosophy will remain applicable and effective both in terms of the health of the Internet, and the quality of the users' audio experience.

Appendix A

This appendix describes DPM audio packet format in more detail in order to clarify DPM protocol. Each DPM packet includes audio data, feedback and control information. Feedback information is piggybacked in the header of a returning DPM packet, and includes sequence number of the packet that is being acknowledged (unsigned long ACKseq_num), the difference between the time when it was received and time when it is being sent back to its sender (ackRTT). Packet loss rate estimated in the receiver is also sent back to the sender (even though the sender estimates packet loss event rates as well). Various control information such as speaker's identification number (e.g. num_speaker_ids can identify speaker's role) and conference ID number (e.g. the scope of speaker's interaction with the other users) are also piggybacked in the returning DPM packet header. The empty field (unsigned short pad) may be used to express the preferences of the current speaker. The selector in each DPM server can process these fields in order to choose which streams are to be mixed in order to maximise the end user experience. Each DPM packet audio header also included sequence number of the packet being sent (seq_num) and the times when it has been sent (sendRTT). Flag field (Flags) is set to different values depending on whether it marks the start of a new talkspurt, or audio format bits or packet type bits.

Exact DPM audio packet format (as used for the demonstration in Chapter 4) is graphically shown below.



<table>
<tr><td>u char num_speaker_ids</td><td>u char</td><td colspan="2">unsigned short confid</td></tr>
<tr><td colspan="4">unsigned int timestamp_samps</td></tr>
<tr><td colspan="2">unsigned short source_port</td><td colspan="2">unsigned short pad</td></tr>
<tr><td colspan="4">unsigned long source_host</td></tr>
<tr><td colspan="4">unsigned long seq_num</td></tr>
<tr><td colspan="4">unsigned long ACKseq_num</td></tr>
<tr><td colspan="4">float PL</td></tr>
<tr><td colspan="4">(struct timeval sendRTT) long tv_sec</td></tr>
<tr><td colspan="4">(struct timeval sendRTT) long tv_usec</td></tr>
<tr><td colspan="4">(struct timeval ackRTT) long tv_sec</td></tr>
<tr><td colspan="4">(struct timeval ackRTT) long tv_usec</td></tr>
</table>

Audio Data

References

[Aourid, M., Kaminska, B., 1994] Aourid, M., Kaminska, B., Neural Networks for the Set Covering Problem: An Application to the Test Vector Compaction, In Proceedings of IEEE International Conference on Neural Networks Conference, 7, 4645-4649, 1994.

[Ballardie, A., 1997] Ballardie, A., Core Based Trees (CBT Version 2) Multicast Routing -- Protocol Specification, RFC-2189, September 1997.

[Barcellos, M., Ezhilchelvan, P., 1998] Barcellos, M. P., Ezhilchelvan P. D., An End-to-End Reliable Multicast Protocol Using Polling for Scaleability, In Proceedings of the Conference on Computer Communications, IEEE Infocom, San Francisco, California, p. 1180, March/April 1998.

[Bolot, J., 1993] Bolot, J. C., Characterizing End-to-End Packet Delay and Loss in the Internet, Journal of High-Speed Networks, vol. 2, no 3, pp. 289-298, September 1993.

[Bolot, J., Turletti, T., Wakamn, I., 1994] Bolot, J., Turletti, T., Wakamn, I., Scaleable Feedback Control for Multicast Video Distribution in the Internet, In Proceedings of SIGCOMM Symposium on Communications Architectures and Protocols, pages 58-67, ACM, London, England, August 1994.

[Bowers, J., Pycock, J., O'Brien, J., 1996] Bowers, J., Pycock, J., O'Brien, J, Talk and Embodiment in Collaborative Virtual Environments, In Proceedings of CHI'96, 1996.

[Braden, B., Clark, D., Crowcroft, J., et al, 1998] Braden, B., Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K., Shenker, S., Wroclawski, J., Zhang, L., Recommendations on Queue Management and Congestion Avoidance in the Internet, RFC 2309, April 1998.

[Brady, P., 1971] Brady, P., Effects of Transmission Delay on Conversational Behavior on Echo-free Telephone Circuits, Bell System Technical Journal, 50:115--134, January 1971.

[Brakmo, L., Peterson, L., 1995] Brakmo, L. S., Peterson, L. L., TCP Vegas: End-to-End Congestion Avoidance on a Global Internet. IEEE Journal on Selected Areas in Communications, 13(8), October 1995.

[Burgin, J., 1988] Burgin, J., Routing and Resource Control in the Broadband ISDN, Australian Telecommunication Research, vol. 22, no. 1, pp. 3--19, 1988.

[Busse, I., Deffner, B., Schulzrinne, H., 1996] Busse, I., Deffner, B., Schulzrinne, H., Dynamic QoS Control of Multimedia Applications Based on RTP, Computer Communications, 19(1): 49-58, January 1996.

[Byers, J., Frumin, M., Horn, G., et al, 2000] Byers, J., Frumin, M., Horn, G., Luby, M., Mitzenmacher, M., Roetter, A., Shaver, W., FLID-DL: Congestion Control for Layered Multicast, In Proceedings of NGC 2000, November 2000.

[Carlberg, K., Crowcroft, J., 1997] Carlberg, K., Crowcroft, J., Building Shared Trees Using a One-to-many Joining Mechanism, ACM Computer Communication Review, pages 5--11, January 1997.

[Cen, S., Pu, C., Walpole, J., 1998] Cen, S., Pu, C., Walpole, J., Flow and Congestion Control for Internet Media Streaming Applications, In Proceedings of Multimedia Computing and Networking, January 1998.

[Chiang, C-C., Gerla, M., 1997] Chiang, C-C., Gerla, M., Routing and Multicast in Mulithop, Mobile Wireless Networks, In Proceedings of IEEE International Conference on Universal Personal Communications (ICUPC’97), pages 546-551, 1997.

[Chiang, C-C., Gerla, M., 1998] Chiang, C-C., Gerla, M., Adaptive Shared Tree Multicast in Mobile Wireless Networks, In Proceedings of IEEE GLOBECOM'98, 1998.

[DeLucia, D., Obraczka, K., 1997] DeLucia, D., Obraczka, K., A Multicast Congestion Control Mechanism for Reliable Multicast, Technical Report, 97--685, University of Southern California, Computer Science Dept., August 1997.

[Dommel, H-P., Garcia-Luna-Aceves, J., 1997] Dommel, H-P., Garcia-Luna-Aceves J., Floor Control for Multimedia Conferencing and Collaboration, Multimedia Systems, Vol. 5, p 23-38, 1997.

[Estrin, D., Farinacci, D., Helmy, A., et al, 1997] Estrin, D., Farinacci, D., Helmy, A., Thaler, D., Deering, S., Handley, M., Jacobson, V., Liu, C., Sharma, P., Wei, L., Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification, RFC 2117, June 1997.

[Feo, A., Mauricio, G., Resende, A., 1989] Feo, A., Mauricio, G.C., Resende, A., A Probabilistic Heuristic for a Computationally Difficult Set Covering Problem, Operations Research Letters, 8, 67-71, 1989.

[Fisher, M., Wolsey, L., 1982] Fisher, M.L., Wolsey, L. A., On the Greedy Heuristic for Continuous Covering and Packing Problems, SIAM J. Alg. Discr. Meth., Vol. 3: 584--591, 1982.

[Floyd, S., Fall, K., 1997] Floyd, S., Fall, K., Router Mechanisms to Support End-to-End Congestion Control, Technical Report, 1997.

[Floyd, S., Fall, K., 1999] Floyd, S., Fall, K., Promoting the Use of End-to-End Congestion Control in the Internet, IEEE/ACM Transactions on Networking, Volume 7, Number 4, IEANEP (ISSN 1063-6692), 458-472, August 1999.

[Floyd, S., Handley, M., Padhye, J., 2000] Floyd, S., Handley, M., Padhye, J., A Comparison of Equation-Based and AIMD Congestion Control, on-line draft http://www.aciri.org/tfrc/aimd.ps, May 2000.

[Floyd, S., Handley, M., Padhye, J., et al, 2000] Floyd, S., Handley, M., Padhye J., Widmer, J, Equation-Based Congestion Control for Unicast Applications, In Proceedings of SIGCOMM 2000, August 2000.

[Floyd, S., Handley, M., Padhye, J., et al, 2000a] Floyd, S., Handley, M., Padhye J., Widmer, J, Equation-based Congestion Control for Unicast Applications: the Extended Version, ICSI Technical Report TR-00-03, URL http://www.aciri.org/tfrc, March 2000.

[Fox, A., Gribble, S., Brewer, E., et al, 1996] Fox, A., Gribble, S.D., Brewer E.A, Amir, E., Adapting to Network and Client Variablility via On-Demand Dynamic Distillation, In Proceedings of the Seventh International Conference on Architectural Support for Programming Languages and Operating Systems, pages 160--170, Cambridge, MA, October 1996.

[Frankel, M., 1987] Frankel, M., Survivable Command, Control and Communication, in Proceedings of IEEE MILCOM, pp. 30.1.1--30.1.4, 1987.

[Frécon, E., Greenhalgh, C., Stenius, M., 1999] Frécon, E., Greenhalgh, C., Stenius, M., The DiveBone: An Application-Level Communication Infrastructure for Internet-Based CVEs, In Proceedings of VRST'99, 58-65, London, 1999.

[Frederick, R., 1993] Frederick, R., 1993, Network Video (nv), Unix Manual Page, Xerox Palo Alto Research Center, 1993.

[Furht, B., Westwater, R., Ice, J., 1998] Furht, B., Westwater, R., Ice, J., IP Simulcast: A New Technique for Multimedia Broadcasting over the Internet, Journal of Computing and Information Technology, Vol. 6, No. 3, pp. 245-254, September 1998.

[Goodwin, C., 1984] Goodwin, C., Notes on story structure and the organization of participation, in Atkinson, J., Heritage, J. (eds.), Structures of Social Action: Studies in Conversation Analysis, 225-46, CUP, 1984.

[Greenberg, S., 1991] Greenberg, S., Personalizable groupware: Accomodating Individual Roles and Group Differences, In Proceedings of the European Conference of Computer Supported Cooperative Work (ECSCW `91), pp. 17-32, Amsterdam, September 24-27, Kluwer Academic Press, 1991.

[Greenhalgh, C., Benford, S., Craven, M., 1999] Greenhalgh, C., Benford, S., Craven, M., Patterns of Network and User Activity in an Inhabited Television Event, In Proceedings of VRST'99, ACM, 58-65, London, Dec 20-22 1999.

[Greenhalgh, C., Benford, S., Taylor, I., et al, 1999] Greenhalgh, C., Benford, S., Taylor, I., Bowers, J., Walker, G., Wyver, J., Creating a Live Broadcast from a Virtual Environment, In Proceedings of SIGGRAPH'99, 375-384, 1999.

[Greenhalgh, C., Purbrick, J., Snowdon, D., 2000] Geenhalgh, C., Purbrick, J., Snowdon, D., Inside MASSIVE-3: Flexible Support for Data Consistency and World Structuring, in Proceedings of CVE'2000, ACM, 119-127, San Francisco, 2000.

[Handley, M., 1996] Handley, M., Session Announcement Protocol, Internet Draft, Internet Engineering Task Force, November 1996.

[Handley, M., 1997] Handley, M., A Congestion Control Architecture for Bulk Data Transfer, Presentation at the Reliable Multicast Research Group meeting, September 1997.

[Handley, M., Padhye, J., Floyd, S., et al, 2000] Handley, M., Padhye, J., Floyd, S., Widmer, J., TCP Friendly Rate Control (TFRC): Protocol Specification, Internet draft, draft-ietf-tsvwg-tfrc-00.txt, work in progress, November 2000.

[Hardman, V., Sasse, A., Handley, M., et al, 1995] Hardman, V., Sasse, A., Handley, M., Watson, A., Reliable Audio for Use over the Internet, In Proceedings of INET'95, Hawaii, 1995.

[Hayes, J., 1984] Hayes, J., Modelling and Analysis of Computer Communications Networks. Plenum Press, N.Y., 1984.

[Hendrix, C., Barfield, W., 1996] Hendrix, C., Barfield, W., Presence Within Virtual Environments as a Function of Visual Display Parameters, PRESENCE, 5(3), 274-289, MIT Press, 1996.

[Hindmarsh, J., Fraser, M., Heath, C., et al, 1998] Hindmarsh, J., Fraser, M., Heath, C., Benford, S., Greenhalgh, C., Fragmented Interaction: Establishing Mutual Orientation in Virtual Environments, In Proceedings of CSCW'98, ACM Press, 217-226, Seattle, WA, USA, November 1998.

[Hindus, D., Ackerman, M., Mainwaring, S., et al, 1996] Hindus, D., Ackerman, M., Mainwaring, S., Starr, S., Thunderwire: A Field Study of an Audio-Only Media Space, In Proceedings of CSCW'96, USA, ACM Press, 238-247, 1996.

[Hoffmann, M., 1997] Hofmann, M., Enabling Group Communication in Global Networks, In Proceedings of Global Networking, Calgary, Alberta, Canada, June 1997.

[Hoffmann, M., Nonnenmacher, J., Rosenberg, J., et al, 1999] Hoffmann, M., Nonnenmacher, J., Rosenberg, J., Schulzrinne, H., A Taxonomy of Feedback for Multicast, Internet Draft, Internet Engineering Task Force, June 1999.

[Humblet, P., 1991] Humblet, P.,Another Adaptive Shortest-Path Algorithm, IEEE Trans. Communications, June 1991.

[Jacobson, V., 1988] Jacobson, V., Congestion Avoidance and Control, In Proceedings of ACM SIGCOMM '88 Conference, pages 314--329, 1988.

[Jacobson, V., 1995] Jacobson, V., How to Kill the Internet, SIGCOMM’95 Middleware Workshop, August 1995.

[Jacobson, V., McCanne, S., 1999] Jacobson, V., McCanne, S., Vat - LBNL Audio Conferencing Tool, URL: http://www-nrg.ee.lbl.gov/vat/, 1999.

[Jeffy, K., Stone, D., Talley, T., et al, 1992] Jeffy, K., Stone, D. L., Talley, T., Smith, F.D., Adaptive Best-Effort Delivery of Digital Audio and Video Across Packet-Switched Networks, In Proceedings of Third International Workshop on Network and Operating System Support For Digital Audio and Video, pp. 1-11, San Diego, CA, November 1992.

[Johnson, D., Aragon, C., McGeoch, L., et al, 1989] Johnson, D. S., Aragon, C. R., McGeoch, L. A., and Schevon, C., Optimization by Simulated Annealing: an Experimental Evaluation. Part I, Graph Partitioning, Operations Research 37:865—892, 1989.

[Kasera, S., Hjlmtsson, G., Towsley, D., et al, 1999] Kasera, S., Hjlmtsson, G., Towsley, D., Kurose, J., Scalable Reliable Multicast Using Multiple Multicast Channels, Submitted to IEEE/ACM Transactions on Networking, January1999.

[Kendon, A., 1990] Kendon, A., A Description of Some Human Greetings, in Conducting Interaction: Patterns of Behavior in Focused Encounters, 153-207, 190, Cambridge University Press, Cambridge, UK, 1990.

[Kouvelas, I., 1998] Kouvelas, I., A Combined Network, System and User Based Approach to Improving the Quality of Multicast Audio, PhD Thesis, 1998.

[Kouvelas, I., Hardman, V., Crowcroft, J., 1998] Kouvelas, I., Hardman, V., Crowcroft, J., Network Adaptive Continuous-Media Applications through Self-Organised Transcoding, In Proceedings of Network and Operating Systems Support for Digital Audio and Video, 1998.

[Krumke, S., Noltemeier, H., Marathe, M., et al, 1996] Krumke, S., Noltemeier, H., Marathe, M., Ravi, R., Ravi, S. S., Improving Steiner trees of a network under multiple constraints, Technical Report LA-UR 96-1374, Los Alamos National Laboratory, Los Alamos, New Mexico, USA, 1996.

[Kyeong-yeol, Y., Jong-hoon, P., Jong-hyeong, L., 1998] Kyeong-yeol, Y., Jong-hoon, P., Jong-hyeong, L., Linear PCM Signal Processing for Audio Processing Unit in Mulitpoint Video Conferencing System, IEE, pages 549-553, XP002159070, Korea, 1998.

[Lin, F., Yee, J., 1989] Lin, F. Y. S., and Yee, J, R., A Distributed Routing Algorithm for Virtual Circuit Data Networks, 1989.

[Lin, J., Paul, S., 1996] Lin, J., Paul, S., RMTP: A Reliable Multicast Transport Protocol. In Proceedings IEEE Infocom'96, San Francisco, CA, March 1996

[Liu, C.-G., Estrin, D., Shenker, S., et al, 1997] Liu, C.-G., Estrin, D., Shenker, S., Zhang, L., Local Error Recovery in SRM: Comparison of Two Approaches, Technical Report USC-CS-97-648, Univesity of Southern California, 1997.

[Macedonia, M., Brutzman, D., 1994] Macedonia, M., Brutzman, D., Mbone Provides Audio and Vision Across the Internet, IEEE Computer, pp30-36, April 1994.

[Mahdavi, J., Floyd, S., 1997] Mahdavi, J., Floyd, S., TCP-Friendly Unicast Rate-Based Flow Control, Note sent to end2end-interest mailing list, January 1997.

[Mathis, M., Semke, J., Madhavi, J., 1997] Mathis, M., Semke, J., Madhavi, J., The Macroscopic Behaviour of the TCP Congestion Avoidance Algorithm, Computer Communications Review, 27, 3, July 1997.

[McCanne, S., Jacobson, V., 1995] McCanne, S., Jacobson, V., Vic: A Flexible Framework for Packet Video. In Proceedings of ACM Multimedia 95, pages 511--522, November 1995.

[McCanne, S., Jacobson, V., Vetterli, M., 1996] McCanne, S., Jacobson, V., Vetterli, M., Receiver-Driven Layered Multicast, In Proceedings of ACM SIGCOM'96, pp. 117—30, Palo Alto, CA., August 1996.

[McDonald, A., 1997] McDonald, A. B., Survey of Adaptive Shortest-Path Routing in Dynamic Packet-Switched Networks, University of Pittsburgh, April 24, 1997.

[Mills, D., 1992] Mills, D. I., Network Time Protocol (version 3) Specification, Implementation, Technical report, RFC 1305, Internet Engineering Task Force, March 1992.

[Mishra, P., Kanakia, H., Tripathi, S., 1996] Mishra, P., Kanakia, H., Tripathi, S., On Hop-by-Hop Rate Based Congestion Control, IEEE ACM Transactions on Networking, 4(2), 224-239, 1996.

[Moy, J., 1994] Moy, J., Multicast Routing Extensions for OSPF, Communications of the ACM, 37(8), August 1994.

[Padhye, J., 2000] Padhye, J., Model-based approach to TCP-friendly Congestion Control, PhD thesis, University of Massachusetts at Amherst, March 2000.

[Padhye, J., Firoiu, V., Towsley, D., et al, 1998] Padhye, J., Firoiu, V., Towsley, D., Kurose, J., Modeling TCP Throughput: A Simple Model and its Empirical Validation, In Proceedings of ACM SIGCOMM '98, Vancouver, BC, September 1998.

[Padhye, J., Kurose, J., Towsley, D., et al, 1999] Padhye, J., Kurose, J., Towsley, D., Koodli, R., A Model Based TCP-Friendly Rate Control Protocol. Network and Operating System Support for Digital Audio and Video (NOSSDAV), June 1999.

[Pasquale, G., Polyzos, E., Anderson, E., et al, 1994] Pasquale, G., Polyzos, Anderson, E.W., Kompella, V., The Multimedia Multicast Channel, Internetworking: Research and Experience, Vol. 5, No. 4, pp. 151-162, December 1994.

[Pasquale, G., Polyzos, E., Kompella, V., 1993] Pasquale, G., Polyzos, E., Kompella, V., Filter Propagation in Dissemination Trees, In Network and Operating Systems Support for Digital Audio and Video, November 1993.

[Paxson, V., Allman, M., 2000] Paxson, V., Allman, M., Computing TCP's Retransmission Timer, RFC 2988, November 2000.

[Paxson, V., Floyd, S., 1997] Paxson, V., Floyd, S., 1997, Why We Don’t Know How to Simulate the Internet, In Proceedings of the 1997 Winter Simulation Conference, 1997.

[Radenkovic, M., Greenhalgh, C., 2000] Radenkovic, M., Greenhalgh, C., Audio Data Processing, Patent Application Number GB 01-03595, April 2000.

[Radenkovic, M., Greenhalgh, C., Benford, S., 1999] Radenkovic, M., Greenhalgh, C, Benford, S, A Scaleable Audio Service for CVEs, In Proceedings of the Sixth conference of the UK_VRSIG, Salford, UK, September 1999.

[Radenkovic, M., Greenhalgh, C., Benford, S., 2000] Radenkovic, M., Greenhalgh, C., Benford, S., Inhabited TV: Multimedia Broadcast from a Large Scale Collaborative World, Facta Universitatis, Ser. Electronics and Energetics, 13 (1), IEEE, ISBN, 0-7803-5678-X, 2000.

[Radenkovic, M., Greenhalgh, C., Benford, S., 2001] Radenkovic, M., Greenhalgh, C, Benford, S., "A Scaleable Audio Service for the Internet", In Proceedings of Fifth International IEEE Conference on Telecommunications in Modern Satellite, Cable and Broadcsting Services (TELSIKS 2001), Nis, Yugoslavia, September 2001.

[Ramakrishnan, K., Floyd, S., 1999] Ramakrishnan, K., Floyd, S., A Proposal to add Explicit Congestion Notification (ECN) to IP, RFC 2481, Network Working Group, January 1999.

[Ramakrishnan, K., Jain, R., 1990]. Ramakrishnan, K., Jain, R., A binary feeback scheme for congestion avoidance in computer networks. ACM Transactions on Computer Systems, pages 158--181, May 1990.

[Rangan, P., Harrick, M., Ramanathan, V., 1993] Rangan, P. V., Harrick, M. Ramanathan, V. S., Communication Architectures and Algorithms for Media Mixing in Multimedia Conferences, IEEE/ACM Transactions on Networking, Vol.1, No.1, February 1993.

[Ratnasamy, S., McCanne, S., 1999] Ratnasamy, McCanne, S., Scaling End-to-end Multicast Transports with a Topologically-sensitive Group Formation Protocol, In Proceedings IEEE International Conference of Network Protocols, ICNP'99, 1999.

[Ratnasamy, S., McCanne, S., 1999a] Ratnasamy, S., McCanne, S., Inference of Multicast Routing Tree and Bottleneck Bandwidth Using End-to-end Measurements, in Proceedings IEEE Infocom. ‘99, New York, March 1999.

[Rejaie, R., 1999] Rejaie, R., An End-to-End Architecture for Quality Adaptive Streaming Applications in the Internet, Ph.D. thesis, University of Southern California, USC/CS- Tech Report-99-718, December 1999.

[Rejaie, R., Handley, M., Estrin, D., 1999] Rejaie, R., Handley, M., Estrin, D., RAP: An End to end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet. In Proceedings of IEEE Infocom'99, New York, NY, March 1999.

[Rhee, I., Balaguru, N., Rouskas, G., 1999] Rhee, I., Balaguru, N., Rouskas, G., MTP: Scaleable TCP-like Congestion Control for Reliable Multicast, In Proceedings of IEEE Infocom, New York, USA, March 1999.

[Rizzo, L., 1997] Rizzo, L., An Embedded Network Simulator to Support Network Protocol Development, In Proceedings of 9th International Conference on Computer Performance Evaluation: Modelling Techniques and Tools, St. Malo, France, LNCS-1245, pp 97-10 7, Springer-Verlag, June 1997.

[Rodden, T., 1996] Rodden, T., Populating the Application: A Model of Awareness for Cooperative Applications, In Proceedings of CSCW'96, ACM Press, 16-20, Boston, 1996.

[Rudin, H., 1976] Rudin, H., On Routing and Delta Routing: A Taxonomy and Performance Comparison of Techniques for Packet-Switched Networks, IEEE Transactions on Communications, COM-24(1):43--59, January 1976.

[Salama, H., Reeves, D., Viniotis, Y., 1996] Salama, H. F., Reeves, D. S., Viniotis, Y., Delay-Constrained Shared Multicast Trees, Technical Report 96/46, November 1996.

[Schooler, E., 1996] Schooler, E. M., Conferencing and Collaborative Computing, In Multimedia Systems 4, 5, 210-225, 1996.

[Schulzrinne, H., Casner, S., Frederick, R., et al, 1996] Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V., RTP: A Transport Protocol for Real-Time Applications, IETF RFC 1889, January 1996.

[Sedgewick, R., 1990] Sedgewick, R., Algorithms in C, Addison-Wesley Pub Co, ISBN: 0201514257, June 1990.

[Sharma, P., Estrin, D., Floyd, S., et al, 1998] Sharma, P., Estrin, D., Floyd, S., Zhang, L., Scalable Session Messages in SRM Using Self-Configuration, Submitted to Sigcomm '98, February 1998.

[Shirish, S., 1995] Shirish, S., ATM Forum Traffic Management Specification Version 4.0, Technical Report 94-0013R6, ATM Forum, June 1995.

[Shoubridge, P., 1996] Shoubridge, P. J., Adaptive Strategies for Routing in Dynamic Networks, PhD Thesis, University of South Australia, Australia, 1996.

[Sisalem, D., Emanuel, F., Schulzrinne, H., 1997] Sisalem, D., Emanuel, F., Schulzrinne, H., The Direct Adjustment Algorithm: A TCP-Friendly Adaptation Scheme. Technical Report, GMD-FOKUS, August 1997.

[Sisalem, D., Schulzrinne, H., 2000] Sisalem, D., Schulzrinne, H., The Adaptive Load Service: An ABR-Like Service for the Internet, In Proceedings of Fifth IEEE Symposium on Computers and Communications (ISCC'2000), (France), July 2000.

[Sisalem, D., Wolisz, A., 2000] Sisalem, D., Wolisz, A., LDA+ TCP-Friendly Adaptation: A Measurement and Comparison Study, In Proceedings of the 10th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'2000, Chapel Hill, NC, USA), June 25-28, 2000.

[Slavik, P., 1995] Slavik, P., A Tight Analysis of the Greedy Algorithm for Set Cover, November 1995.

[Stemm, M., Katz, R., Seshan, S., 2000] Stemm, M., Katz, R., Seshan, S., A Network Measurement Architecture for Adaptive Applications, In Proceedings of IEEE INFOCOM '00, pp. 2C--3, March 2000.

[Stevens, W., 1997] Stevens, W. R., TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Agorithms, RFC2001, Technical Report, Internet Assigned Numbers Authority, Jon Postel, USC/ISI, 4676 Admiralty Way, Marina del Rey, DA 90292, 1997.

[Tan, D., Zakhor, A., 1999] Tan, D., Zakhor, A. Real-Time Internet Video Using Error Resilient Scalable Compression and TCP-Friendly Transport Protocol, IEEE/ACM Trans. Multimedia, May 1999.

[Turletti, T., 1994] Turletti, T., The INRIA Videoconferencing System (IVS), ConneXions - The Interoperability Report Journal, 8(10):20--24, October 1994.

[Vicisano, L., Rizzo, L., Crowcroft, J., 1997] Vicisano, L., Rizzo, L., Crowcroft, J., TCP-Like Congestion Control for Layered Multicast Data Transfer, IRTF RM Workshop September 1997, Cannes, September 1997.

[Waitzman, D., Partridge, C., Deering, S., 1988] Waitzman, D., Partridge, C., Deering, S.E, Distance Vector Multicast Routing Protocol (DVMRP), Internet Engineering Task Force (IETF), RFC 1075, November 1988.

[Waters, R., Anderson, D., Barrus, J., et al, 1997] Waters, R., Anderson, D., Barrus, J., Brogan, D., Casey, M., McKeown, S., Nitta, T., Sterns, I., Yerazunis, W., Diamond Park and Spline: A Social Virtual Reality System with 3D Animation, Spoken Interaction, and Runtime Modiability, Presence, vol. 6, no. 4, pp. 461-481, August 1997.

[Watson, A., Sasse, A., 2000] Watson, A., Sasse, A., The Good, the Bad, and the Muffled: the Impact of Different Degradations on Internet Speech, In Proceedings of MM2000, Los Angeles, California, November 2000.

[Wei, L., Estrin, D., 1994] Wei, L., Estrin, D., The Trade-offs of Multicast Trees and Algorithms, In Proceedings of the 1994 International Conference on Computer Communications and Networks, San Francisco, CA, USA, September 1994.

[Widmer, J., 2000] Widmer, J., Equation-Based Congestion Control, Master's Thesis, University of Mannheim, February 2000.

[Widmer, J., Denda, R., Mauve, M., 2001] Widmer, J., Denda, R., Mauve, M., A Survey on TCP-Friendly Congestion Control, Technical Report TR-01-002, Department for Mathematics and Computer Science, University of Mannheim, 2001.

[Wilson, F., Wakeman, I., Smith, W., 1993] Wilson, F., Wakeman, I., Smith, W., Quality of Service Parameters for Commercial Application of Video Telephony, In Human Factors in Telecommunication symposium, Damstadt, Germany, March 1993.

[Xu, R., Myers, A., Zhang, C., et al, 1997] Xu, R., Myers, A., Zhang, C., Yavatkar, R., Reliable Multicast Support for Continuous Media Applications. In Proceedings of the Seventh International Workshop on Network and OS Support for Digital Audio and Video, ACM, St. Lois, MO, May 1997.

[Yang, Y., Kim, M., Lam, S., 2001] Yang, Y., Kim, M., Lam, S., Transient Behaviors of TCP-friendly Congestion Control Protocols. In Proceedings of the Conference on Computer Communications, IEEE Infocom, pp. 1716—1725, Anchorage, AK, April 2001.

[Yeadon, N., 1996] Yeadon, N., Quality of Service Filters for Multimedia Communications, Ph.D. Thesis, Lancaster University, Lancaster, U.K., May 1996.

[Yeong, W., Howes, T., Kille, S., 1995] Yeong, W., Howes, T., Kille, S., Lightweight Directory Access Protocol, RFC 1777, March 1995.

[Zhu, Q., Parsa, M., Dai, W., 1994] Zhu, Q., Parsa, M., Dai, W.W.M., An Iterative Approach for Delay-Bounded Minimum Steiner Tree Construction, Technical Report UCSC-CRL-94-39, UC Santa Cruz, 1994.