Network Adaptive Continuous-Media Applications Through Self Organised Transcoding

Isidor Kouvelas, Vicky Hardman and Jon Crowcroft
Department of Computer Science
University College London
{I.Kouvelas, V.Hardman, J.Crowcroft}@cs.ucl.ac.uk

Abstract

With the deployment of the Mbone, multimedia conferencing is becoming a common practice on the Internet. In order to coexist with traditional services like email and file transfer, mechanisms to fairly share available bandwidth have to be developed for real-time audio and video media streams. The network scale and heterogeneity in available bandwidth complicate the design of network adaptive multicast applications. This paper presents a new scalable architecture for congestion controlled multicast real-time communication. The proposed scheme uses self-organisation to form groups out of co-located receivers with bad reception and provides local repair through the use of transcoders. The receiver driven nature of the protocol ensures high scalability and applicability to large Mbone sessions. The viability of the proposed protocol is demonstrated through simulation.

1 Introduction

With the deployment of the Mbone, multipoint multimedia conferencing has become a common practice on the Internet. The network scale and heterogeneity in available bandwidth complicate the design of network adaptive real-time multipoint applications.

Audio and video conferencing applications can tolerate a certain amount of packet loss and delay jitter from the network. They have been designed to adapt to network conditions and minimise the perceived signal degradation by trading off reliability for interactivity [1, 2].

The UCL Robust-Audio Tool (RAT) [3] and Freephone developed at INRIA use forward error correction (FEC) techniques [4] and successfully address the loss problem with minimal increase in stream delay. FEC used is in the form of highly compressed low quality audio that is piggybacked on normal audio packets. The decision on the level of FEC to use is made per source based on receiver loss reports and is tailored to cover for the average or highest requirements of the receiver group. This strategy is only good for a group observing similar loss rates. In a diverse group receivers observing low loss are forced to receive useless redundant information whereas receivers with very bad loss may not be covered.

The variable network loss rates and perceived quality in different areas of a multicast distribution tree are a result of different link bandwidth availability and link load. The extent of this problem is best illustrated by the work of Handley [5] in figure 1. The graph represents the packet loss rates experienced by different receivers of a popular Mbone session over the period of the session. It is clear that although most receivers are seeing low to moderate loss, there are a small number of sites suffering.

This indicates that a single stream addressed to the whole group can not possibly cover the needs of all receivers. Instead the data rate and amount of redundancy has to be customised and separately distributed to problematic areas. It has been shown that sender driven schemes that try to address the receiver heterogeneity problem do not scale to large groups. Any scheme that attempts to separately cover for the different needs of problematic receiver subgroups has to be receiver driven to scale [6]. Subgroups of co-located receivers in a multicast delivery tree suffering from similar problems should co-ordinate their efforts in improving reception quality. Furthermore their attempts should not affect reception for the remaining participants in the multicast session.

Strict low delay requirements of real-time data distribution preclude solutions using retransmissions to achieve required reliability. The dynamic nature of the Mbone delivery and membership model does not allow for manually configured static schemes that work around congested links.

The solution we present in this paper uses a self-organisation 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. The transcoding site thus provides local repair to the congestion problem of the group with minimal increase in stream delay. The data rate and level redundancy of the transcoded stream are continuously modified to adapt to the bottleneck link characteristics using reception quality feedback from a member of the formed loss group. Network friendly congestion control of the real time multicast stream can thus be achieved.

The rest of this paper is structured as follows. Section 2 describes related work on congestion control for multicast distribution and attempts to solve the group reception diversity problem. In section 3 we describe our self-organised transcoding solution to the problem. The proposed solution has been implemented and evaluated through simulation using the VINT network simulator [7]. The simulation results can be found in section 4.

2 Background and Related Work

The current multimedia conferencing architecture over the Mbone / Internet has the following characteristics:

- Conferencing applications use the Real-time Transport Protocol (RTP) [8, 9] to transmit information over an unreliable best-effort multipoint network.
- Receivers express interest in receiving traffic by tuning into a multicast address and the network forwards traffic only along links with downstream recipients.
- No knowledge of group membership or routing topology is available at the source or receivers.

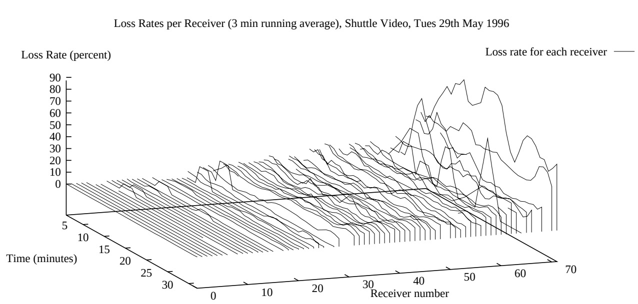

Figure 1: Loss rate against time for different receivers

The rest of this section discusses existing work and alternatives addressing the reliability issues for continuous media streams in a heterogeneous multicast environment.

2.1 Layered Encoding

McCanne et al [6] combine a layered compression scheme with a layered transmission scheme to address the network heterogeneity issue. The media stream is encoded into a number of layers that can be incrementally combined to provide refined versions of varying quality of the encoded signal. The individual layers are then transmitted on separate multicast addresses. Receivers adapt to network conditions by adjusting the number of levels they subscribe to and thus improving perceived quality by trading off average signal quality for packet loss. The application they propose for this scheme is multicast video. For this purpose they have developed a video codec that can compress a video frame providing very fine granularity layers.

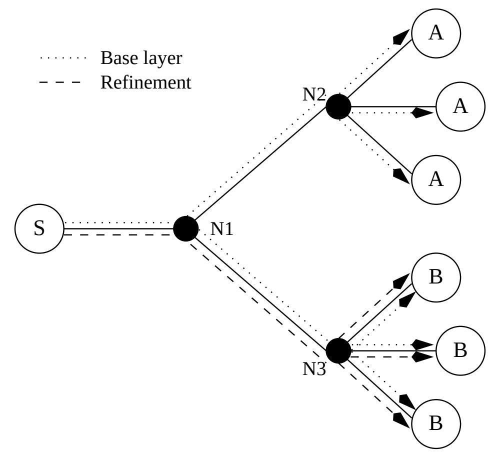

Figure 2: Example of layered encoding and transmission

Although layered encoding is possible with audio, the transmission bandwidth range is significantly more restricted in comparison to that available to video applications. This is definitely the case for sampled speech. There is none or very little improvement in intelligibility to be gained by sampling speech at full CD quality (44.1KHz stereo sampling) rather than a single channel sampled at 16KHz [10]. With music a wider range is available, from CD quality to a highly compressed low quality format.

Currently available speech codecs do not render themselves naturally to layering. It is possible to modify a scheme to split up the resulting compressed block into a number of sub-blocks that can be separately decoded to provide increasing levels of quality. However, to achieve the same quality that the original not split up version of the codec provides, a larger number of bits per codeword is required [11].

An additional requirement from a layered encoding scheme, in order for it to be suitable for use with a network adaptation algorithm, is that there must be an exponential relationship between the bandwidth of different layers. Such an arrangement maximises support for network bandwidth adaptation while keeping the routing overheads due to the number of multicast groups used low. This requirement in combination with the problems listed above makes layered transmission unsuitable for real-time audio steams.

Even with a video stream depending on the application there is a target frame rate from which you cannot deviate. In video conferencing the entire range can be used, from very slow scan video to the full potential of the camera, but when watching a film full frame rate is required.

2.2 Simulcasting

With simulcasting a group of receivers can adapt to network conditions by having the sender transmit a new parallel stream and customise it to match their requirements. The new stream can use a different compression scheme to reduce the bandwidth required and employ some form of FEC to counter packet loss. This approach is likely to create congestion on links that are close to the sender, as all simulcasted streams will have to traverse them.

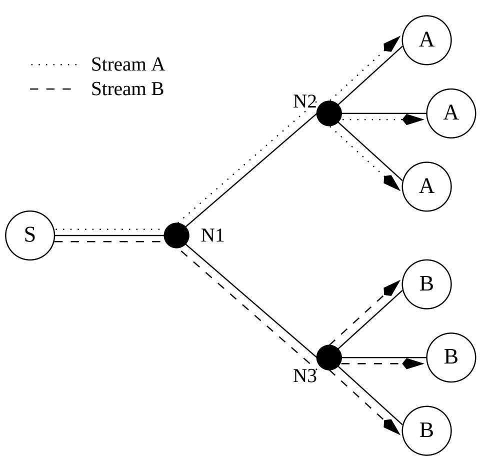

Figure 3: Example of simulcasting

The bandwidth utilisation advantage of layered encoding over simulcasting is illustrated in figures 2 and 3. Due to different bandwidth availability on links N1-N2 and N1-N3 groups of receivers A and B require different streams. With simulcasting the link between the source and N1 has to carry both the full bandwidth stream for receiver group B and a lower bandwidth stream for group A. Using layered encoding and transmission the link between the source and N1 does not carry duplicate redundant information. Ideally the sum in bandwidth of both layers will exactly cover the requirements of group B whereas the base layer will be customised for group A.

Simulcasting suffers from scalability problems because the sender is involved in the adaptation.

2.3 Retransmission Based Reliability

Proposals exist for integrating reliable multicast schemes into audio and video applications so that missing packets can be recovered from neighbours with better reception [12, 13, 14]. This is achieved by trading off quality for delay as any reliable multicast protocol has to request retransmission and wait for the repair. Although this may be acceptable in a real-time lecturing scenario it becomes less useful with interactive communication. An additional undesirable side effect is that the operation of the reliable protocol creates extra control traffic.

Maxemchuk et al [13] propose a hierarchy of retransmission servers positioned around expensive or over utilised links. The servers operate a NACK based reliable protocol between them and receivers use a similar scheme for requesting lost packets. Their proposal significantly improves reception quality but requires manual configuration of the retransmission servers.

In [14] Xu et al describe the STORM protocol that develops parent child relationships between participants of a multicast using an expanding ring search technique. Parents are chosen according to loss statistics so that they have a good chance of receiving packets their children are likely to request.

Streaming of stored data makes little sense unless browsing and selective playback is a requirement. For totally non real-time scenarios, a normal transport protocol and pre-fetch can be used to achieve perfect audio quality. TCP can be used in a single user scenario or a multicast congestion control protocol like RLC [15] for multiple recipients.

2.4 Statically Configured Transcoders

In many situations where a group of people with limited network resources want to participate in a high bandwidth multicast conference, the use of transcoders is employed. A transcoder is an application that is placed at the far end of the low bandwidth link to down-convert a high bit rate stream so that it can fit through the link. Transcoders for video can reduce the frame rate and image quality and audio transcoders can re-encode the audio signal using a higher compression scheme. A feature of audio transcoding is that it adds minimal delay to the signal when relaying it as it can be done on a per packet bases. Apart from changing the bandwidth requirements of a stream a transcoder can also introduce or remove forward error correction information to counter packet loss.

In a multicast scenario a transcoder can be positioned on the sender's end of a problematic link to re-encode the stream to use lower bandwidth and add FEC information. The resulting stream can be re-multicast to a new address. If all receivers beyond that link tune to receive the new customised stream then there will be no bandwidth wasted as the original stream will no longer traverse the problematic link. This solution is static and has problems with the dynamic nature of Mbone multicast routing.

In [16] Pasquale et al 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 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.

3 Self Organised Transcoding

By combining the simulcasting, local repair and transcoding schemes we have developed a solution that does not suffer from the above problems. What is needed is a control scheme that automatically configures transcoders within the multicast tree to support branches with bad reception.

When a group of receivers detect loss caused by a congested link, an upstream receiver with better reception at the far end of the bottleneck link affecting the group needs to act as a transcoder and provide a customised version of the stream. This new stream will be multicast to a different address to which receivers affected by the bottleneck need to switch. To achieve this in a scalable way, the suffering receivers need to elect a representative which will attempt to locate an upstream receiver willing to serve them and co-ordinate the transcoding process (figure 4).

Ideally there should be no links carrying more than

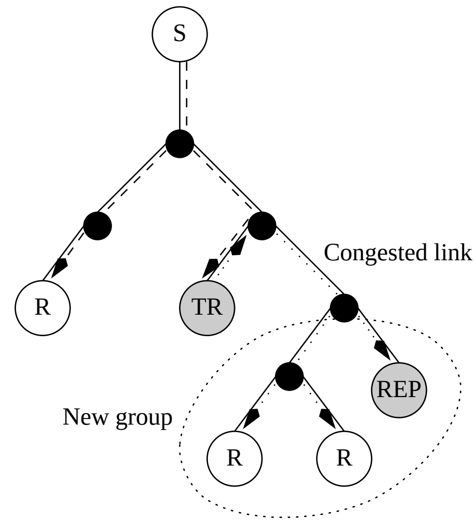

Figure 4: Transcoder requester and provider configuration around a congested link

one of the transcoded streams (including the original encoded stream). To achieve this the following conditions must hold:

- The transcoder servicing a sub-tree should be as closely located to the sub-tree as possible. The preferable location is at the end of the bottleneck link closer to the source.
- A group of receivers behind a bottleneck link has to be co-ordinated in its actions. All the receivers responding to congestion have to switch to a new transcoded stream at the same time.

To avoid the need of processing capabilities at nodes of the tree that do not have any receivers we co-locate the transcoders with active receivers and allow the users media tool to execute our protocol. Although this makes our proposal applicable on the currently available network infrastructure, it results in sub-optimal transcoder configurations. In the example of figure 4 the ideal placement for the transcoder would be on the network node above the requesting site. The current placement of the transcoder results in wasted resources for the transcoded stream out of the transcoding site. If the transcoding site was connected to the network through a shared medium link, like an ethernet, than both the incoming and outgoing stream would have to traverse the same link possibly causing congestion.

3.1 Transcoding Provider Selection

When a receiver detects packet loss, it schedules a transcoder request message. To avoid multiple receivers that see the same loss simultaneously sending a request and overloading the network, requests are multicast and sending them is delayed by a time proportional to the distance of the requester from the stream source plus an additional small random interval. If a receiver sees a request while it has one scheduled then the request is cancelled.

The request includes a description of the loss patterns observed. Receivers of the request that have better reception from the requester can offer to provide a transcoded stream. This is achieved in a similar manner to the request. The response message is scheduled to be multicast after an amount of time proportional to their distance from the requester plus a small random interval. The response message contains a description of the loss experienced by the offering receiver and its distance from the requester. On reception of the offer other receivers that have offers scheduled suppress their messages unless they can provide a better service.

The quality of service that a site offering a transcoded stream can provide is calculated as a function of the difference in loss rates observed by the requester and the offering site and the distance between the two. The quality is better for larger loss rate differences and smaller distances.

After a short timeout period, long enough for the offers from potential transcoders to arrive, the requesting receiver multicasts a transcoding initiation message. This message serves two purposes. It notifies the offering receivers of who is going to provide the transcoded stream and instructs all other members in the loss group to switch to the new stream.

As control messages are sent to the whole receiver group, while a transcoding negotiation is in progress, control messages from other receivers are suppressed.

3.1.1 Receiver Distance Calculation

The timer based-scheme described above is similar to that used in the SRM [17] reliable multicast protocol for retransmission requests and repairs. In SRM round-trip times (RTT) are used as distances between receivers and are calculated from timestamps in session messages. Reporting receivers include timestamps received from other receivers plus the amount of time elapsed between receiving the stamp and sending the report. Round-trip time estimates can thus be obtained. RTP uses the same scheme with timestamps in RTCP messages just for sources so that they can calculate the RTT to receivers. The SRM extension to obtain distance estimates between all the receivers does not scale for large sessions since every pair of receivers has to exchange at least three messages. Puneet et al [18] have developed a hierarchical self-organising scheme that elects top level receivers for different regions of the distribution tree that represent their region in distance calculation estimates. This scheme significantly improves the scalability of RTT calculation in SRM.

The requirement for background session messages to build distance information is removed if receivers use NTP [19] and have synchronised clocks. Although the level of deployment of NTP on current Mbone hosts is not very good, there is no reason why it should not be in use. With synchronised clocks a distance estimate can be obtained from a single timestamped message. A receiver of a request or an offer can calculate the distance from the sender by subtracting the timestamp in the message from the current time. This estimate may not be very accurate for the reverse distance from the receiver to the sender, as paths in the Internet are not always symmetric, but for our purposes it is good enough.

3.1.2 Multiple Offer Resolution

Depending on how the delay timers are set it is likely that the requester will receive more than one offer. Some of these may even not have originated from another receiver further up the delivery tree from the source but by a receiver on a side branch with a better link to the requester. A control protocol can be used in such cases to measure the performance of different links and decide on which one to use.

3.2 Receiver Group Control

In order to reduce the amount of traffic flowing through bottleneck links when a transcoded stream is initiated, all receivers behind that link should stop receiving the original stream. The transcoding initiation message provides the synchronisation needed to co-ordinate the switching action. Receivers of the initiation message decide individually whether they belong to the group and accordingly switch to receiving the new stream or continue without change.

The decision on whether a receiver belongs to the same loss group as the requester is based on correlated loss information between the two. The main principle behind this is that receivers behind the same congested link will miss the same packets and see similar loss patterns.

The current Mbone media tools implementing the RTP protocol provide periodic loss measurements in RTCP receiver reports. These reports are multicast so that all receivers see reports from other receivers. By correlating the variations in loss between what is reported and what is observed locally, some grouping information can be derived. The problem with RTCP reports is that they become very infrequent as the size of the receiver group grows to maintain the amount of bandwidth used by control traffic small. Furthermore the period of time over which the loss is reported is not obvious. A guess at the reporting interval can be made as the last correctly received packet is given in the report. By monitoring the frequency of reports from each receiver we can figure out the period over which the report is referring to. The estimation process is complicated by lost report messages. To perform the correlation the loss observed locally over the same period has to be calculated. To achieve this, a history of arrival timestamps and sequence numbers of correctly received packets has to be maintained. The history has to be long enough to cover the maximum possible reporting interval. Apart from the excessive amount of storage this method requires, the results produced cannot be very accurate.

The need for background control messages to exchange locality information can be removed by including a description of the loss pattern observed by the requester in the transcoding initiation message. The loss description can be in the form of a bitmask representing which of the last transmitted packets were received and which not. The sequence number of the last packet in the bitmask can also be included. Other receivers can use the information in the bitmap to correlate the loss and decide if they belong to the group or not. This alleviates the problem of infrequent RTCP reports in large sessions, as even receivers that have not had a chance to send a report will know if they belong to the group.

To perform the correlation each receiver must maintain a history of received packets. The size of the state can be as little as one bit per packet as all we are interested in is whether a packet was received or not. The length of the history does not have to be much longer than the length of the loss bitmaps as the loss data in a received initiation message will always be recent.

A received bitmap can be compared with the local log to provide a loss proximity measure. Packets that are lost in both sites increase the likelihood that the receivers are located close to each other. Packets lost at one receiver but not at the other reduce the likelihood. The accuracy of the result can be improved by increasing the size of the bitmap at the expense of larger control messages. An optimisation would be to use a Huffman encoding for the bitmap. This way information about more packets can be packed in the same space in the message. The encoding method used can vary and be optimised for different loss rates.

The bitmap control scheme requires less state and processing and provides much more accurate results to the RTCP loss variation correlation.

3.2.1 Loss Bitmap Comparison

As the result of the loss bitmap comparison determines the stability of Self Organised Transcoding (SOT) it is crucial to minimise decision errors. Errors can have two outcomes:

- A receiver can decide to join a group it doesn't belong in, thus pulling the transcoded stream to some remote network location.
- A receiver that should join the transcoded group does not do so and the original data stream continues to flow down the bottleneck link.

Thus the decision process has to be as precise as possible and cannot be either over optimistic or under optimistic.

SOT uses three summary measures when comparing the loss bitmaps from two different receivers. These measures are the number of packets *lost in common*, the number of packets *lost at one receiver* but not the other and the overall *loss rate*. The overall loss rate is calculated as the average between the rates at the two sites as it is assumed that in order for two bitmaps to match the two rates have to be similar.

In order for the comparison to be useful, some information has to exist in the bitmaps. A pattern full of losses is not useful, as it will be the result of a broken link. Two such patterns although identical could be the results of two different broken links. The same holds for a pattern full of received packets. In contrast we can be almost certain that two identical bitmaps with a number of transitions between lost and received packets are the result of the same problematic link.

In some cases it may be desirable to accept a small number of not common losses in order to form groups of receivers that are affected by a common major bottleneck but also have smaller problems of their own. An example can be seen in figure 5. Receiver X can be grouped together with Y and have a single stream provided by Z. The stream can be customised to cover for the additional low loss on the link to X at the expense of having the link to Y slightly under-utilised. The alternative configuration would be to have Z transcode a stream for Y and then Y transcode a stream for X. There is a trade-off between the amount of separate losses that we should allow in the comparison to achieve such configurations and the probability of error in the decision.

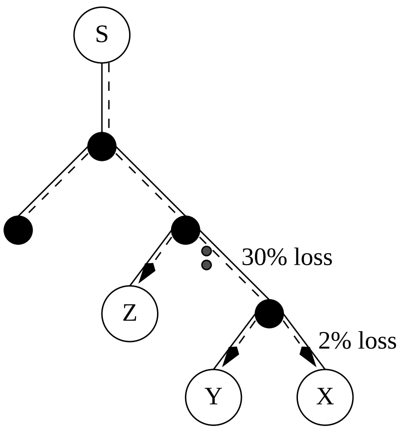

Figure 5: Receivers with slightly different loss affected by the same main bottleneck

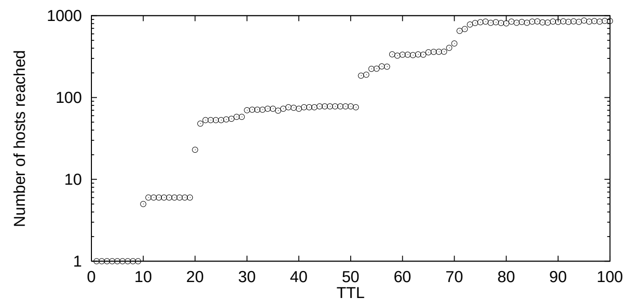

Figure 6: Number of hosts running sdr reachable for different TTL

3.3 Congestion Control

The transcoder initiation protocol results in a natural pairing between requester and transcoding provider on either side of a bottleneck link (figure 4). This provides a solution to the scaling problem of multicast congestion control. The original requester can represent the receiver group and provide the feedback needed to the transcoder provider to adapt to the available bandwidth of the bottleneck. This can be achieved on similar time-scales to TCP congestion control thus resulting in fair sharing of the network with non-multicast traffic. A possible design of a congestion control algorithm for real-time streams and results from simulation are presented in sections 4.2 and 4.5 respectively.

3.4 Membership Changes

The above discussion does not address start and end time issues. Receivers may join and leave at different times during a conference. When new receivers join they have to find out if their branch of the network is being serviced by a transcoder or the real source, and which address they should join to receive the traffic.

To achieve this a new receiver has to be able to probe existing receivers in its network neighbourhood in a scalable manner. With currently deployed multicast distribution protocols this can be achieved through the use of a time to live (TTL) based expanding ring search (ERS) algorithm. The idea is that the TTL field of the IP header can be used to limit the lifetime of multicast packets and restrict their distribution to a local part of the network. Using larger TTL values allows packets to live longer and reach further. A new receiver can send query messages to a control group starting with a low TTL and increasing until it receives a response.

Unfortunately increasing the TTL over the threshold necessary for the query packets to live beyond some router means that a whole new part of the network is reached and not a single potential responder. Figure 6 shows the number of hosts listening to the SDR [20] session announcement multicast address reachable from UCL for increasing TTL values. The measurements were collected in September 1997 using the multicast ping mechanism. Because the use of ping on multicast addresses causes an implosion of responses resulting in lost messages, the values in the graph are lower than the real ones especially for larger TTL values. The graph clearly shows that as the TTL exceeds various thresholds, large groups of receivers become reachable.

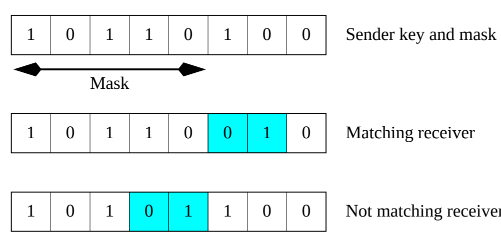

Figure 7: Using a sliding key to probe receiver group

To avoid an implosion of responses to a query message that suddenly reaches a large number of receivers, an additional mechanism is required to restrict the number of potential responders. A sliding key probing mechanism (introduced in [21]) can be employed. This operates by having the query sender and receivers each choose a random key. The sender's key is included in the request and only receivers with a matching key are allowed to respond. The number of matching keys can be controlled through the use of a mask that specifies the number of significant bits in the key that have to match (figure 7). Thus the number of responses to a request can be controlled by selecting an initial mask length and then re-sending the request with a reduced mask size until a response is received.

An additional problem with the locality achieved through TTL scoped ring searches is that they cannot be constrained to work along the distribution tree from a given source. That would be desirable behaviour as we would be able to constrain configuration of transcoding groups along the original source distribution tree and have more predictable behaviour from our protocol. The use of subcasting, which has recently been proposed as a modification to multicast to support reliable multicast applications, can provide this functionality [22].

Receivers quitting the session are not a problem except in the cases of the requester or the last member of the group leaving. By having the requester periodically multicast alive messages after group formation to the formed group other members can detect when the requester has left. The messages are sent on a separate control address to prevent distribution to other session participants. On detection of departure, the remaining group members schedule a message to take over the role of the requester. The transmission is delayed proportionally to the distance from the transcoder so as to achieve election of the member closest to the bottleneck link. A transcoder can detect the departure of the last member of the group and stop transmission by the cease of congestion feedback information.

3.5 Topology Changes

Link and router outages although not very frequent are quite common in the Internet / Mbone [23]. As a result of an outage the multicast routing to some members of a session being serviced by a transcoder may change resulting in a non-optimal or even problematic configuration (figure 8). To recover from such situations SOT needs to periodically repeat the initiation protocol. The task of doing this can be left up to the original requester.

The resolution of congestion problems that initially caused group formation or the appearance of a new bottleneck splitting an existing group in half also affect group dynamics. The first effect can be addressed by having the transcoder dissolve a group that has been following the transmission rate of the original sender for some period of time. The introduction of a new bottleneck will cause some members of the loss group to observe different loss patterns to those of the requester. By introducing a loss bitmap in the alive messages sent by the requester, this can be detected and the affected members can rerun the initiation protocol.

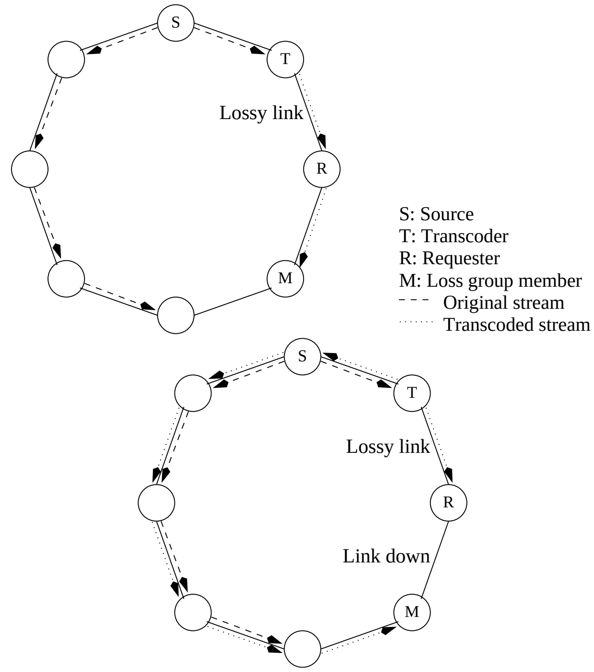

Figure 8: Possible effect of a topology change on a transcoder setup (before and after)

3.6 Multicast Address Allocation

Multicast distribution trees vary for different sources in a session so SOT adaptation has to be per source. With currently deployed multicast protocols there is no way a receiver can express interest in a particular source. Instead subscription is per group and every sender sending to this group is received. For SOT to work in this environment each sending participant has to use a separate multicast address to transmit data. In addition each new transcoder instantiation has to transmit on a new unused address. In sessions with a large number of participants this can be a problem. Typically very large sessions are lecture based where only a few participants transmit data and the majority are only spectators which somewhat alleviates the problem.

The real solution to this problem is the deployment of the Internet Group Management Protocol version 3 (IGMPv3) [24] which is currently under development by the IETF IDMR working group. IGMPv3 supports expression of interest in particular sources for each multicast address joined. This reduces the number of multicast addresses required by SOT to the maximum number of transcoded streams forwarded by any node.

4 Simulation

As part of the development of some of the ideas in SOT and in order to evaluate its performance, we implemented the protocol in version 2.1 of the VINT network simulator ns [7]. Ns is an event driven packet-level simulator. Within ns there are several multicast protocol implementations. We chose to use the dense mode (DM) version as it behaves similarly to what is currently available on the Mbone. We extended the implementation by adding source-specific group membership control, which is expected to be available on the Mbone with the deployment of IGMPv3 [24]. The Self Organised Transcoding protocol was implemented as an extension to ns using C++ and otcl. The protocol implementation and simulation scripts are available upon request from the authors.

4.1 SOT Implementation

This section describes the design of the SOT transcoder initiation implementation in ns. In the initiation stage SOT uses the following three messages:

request: Sent by the loss group representative to locate possible transcoders. This message identifies the requester and includes the observed loss rate so that sites willing to offer a transcoded stream can compare their reception and respond only if it is better.

response: Sent by receivers that have received a request, have better reception to the requester and are willing to provide a stream. The locally observed loss rate is included so that the requester can select the best transcoder if multiple offers are received.

command: After responses have been collected by a requester, a transcoder is selected and this message sent to instruct the transcoder to initiate the new stream and other loss group members to switch streams. The message includes a loss bitmap for other receivers to compare against and decide if they belong to the group or not.

In addition to receiving the above messages there are two more events that can occur during protocol operation. The first is a *loss report* that is triggered by the reception of a data packet. The second is a *timeout* from the internal protocol timer.

These events cause each member of a SOT session to be in one of the following states:

Idle: This is the initial state.

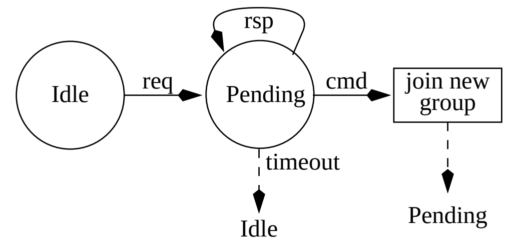

Figure 11: Transcoding initiation state transition diagram (loss group member)

Schedule request: In order to locate a transcoder after loss has been reported a request message has to be sent. To avoid message implosion, these messages are randomly delayed as described earlier in section 3. A timeout event is scheduled and the receiver waits in this state before sending the request.

Wait for response: After a request has been sent, a requester waits in this state for offers from transcoders. A timer is set to retransmit the request if no responses are received for a predefined period of time. If at the expiration of the timer an offer has been received a command message is sent out to initiate the transcoded stream and instruct other receivers in the loss group to receive it.

Schedule response: A site willing to offer its services as a transcoder waits in this state for a timeout before sending a response. Reception of a better offer from another transcoder before the timeout cancels the scheduled response.

Pending: As SOT messages are multicast to the entire receiver group, in order to reduce the amount of bandwidth consumed at any instant the messages have to be spread out in time. To this end an attempt to have only one request in progress is made. This is achieved by having all other traffic cancelled and the senders back off when a setup with a worse loss problem is seen to be in progress. Backing off is implemented by waiting in this state for a timeout that returns the receiver to the idle state.

The transition diagrams for the states and events listed above are shown in figures 9, 10 and 11 for the requester, the site offering to transcode and a loss group member respectively.

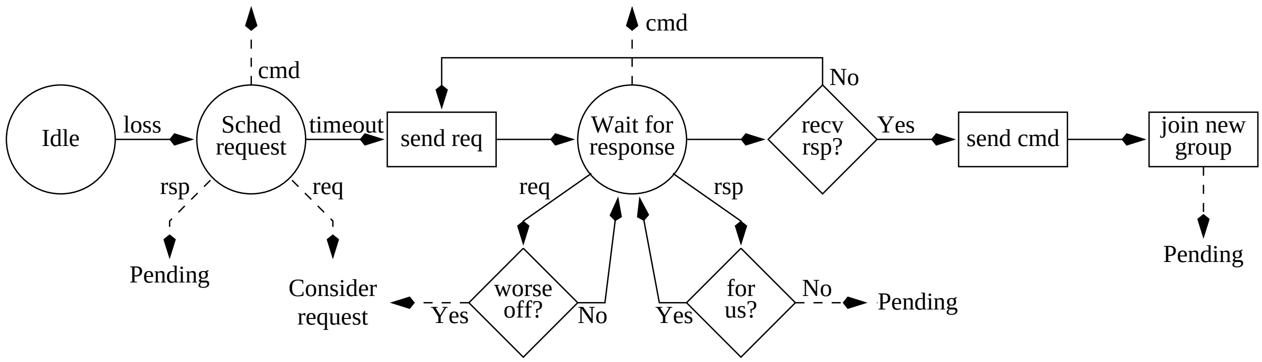

Figure 9: Transcoding initiation state transition diagram (requester)

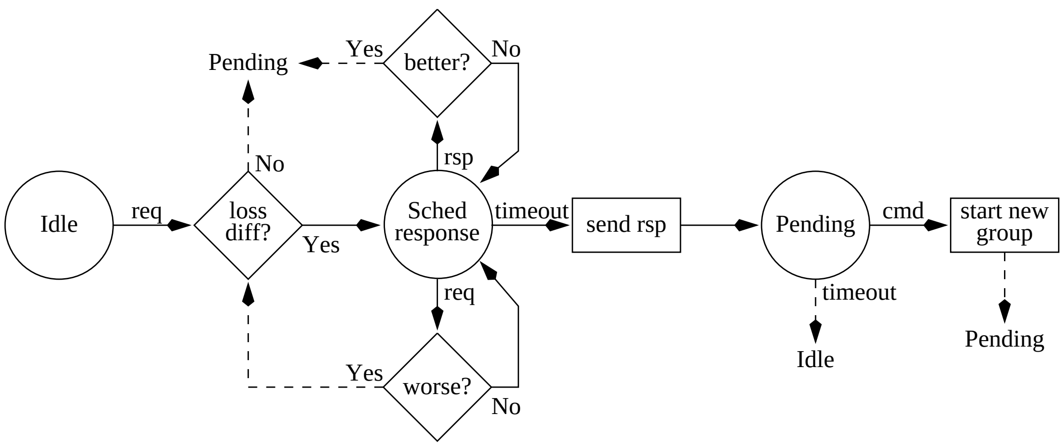

Figure 10: Transcoding initiation state transition diagram (transcoding offer)

4.2 Congestion Control Implementation

After the initiation stage is complete, the requester provides feedback to the transcoder concerning the bottleneck behaviour. The information consists of loss / no loss signals. The transcoder uses this information to adapt the bandwidth of the transcoded stream. Although the adaptation algorithm is independent from the operation of SOT, we implemented a simple version for the purposes of our simulations. The implemented algorithm tries to behave in a manner similar to the congestion control algorithm in TCP [25] by halving the stream bandwidth when loss is detected and linearly increasing the bandwidth when no loss is signalled. When transcoders start the initial bandwidth of the transcoded stream is set to a very low rate thus performing the equivalent of a slow start.

Bandwidth selection is achieved by varying the transcoded packet size. This is the behaviour that would be expected by an audio transcoder when selecting a different encoding scheme for the outgoing stream. The number and frequency of outgoing packets is the same as that of incoming ones. This is true of most transcoding techniques as each packet corresponds to a specific time interval. In video transcoding there are two different options to control the transmission rate. The simplest solution is to reduce the frame rate which will result in a smaller number of packets. A better approach is to reduce the quality of the encoded image resulting in the same number of smaller packets. Codecs available in current Mbone tools support image quality selection.

When stream bandwidth is increased in response to a period with no loss we are performing an experiment to see if the bottleneck can accommodate some additional traffic. If the link is full this will result in congestion and some packets will be dropped. To avoid degradation in perceived quality due to the loss, the additional bandwidth can be used to carry FEC redundant information [4] for a short period after the bandwidth increase. The switch to a higher quality encoding without redundancy can be postponed until we feel that the link can take the new traffic.

4.3 Simulation Metrics and Parameters

SOT was simulated on a number of simple network topologies, which were designed to include specific problematic configurations (figure 12), and on larger random topologies that were created with the assistance of topology generators (figure 13).

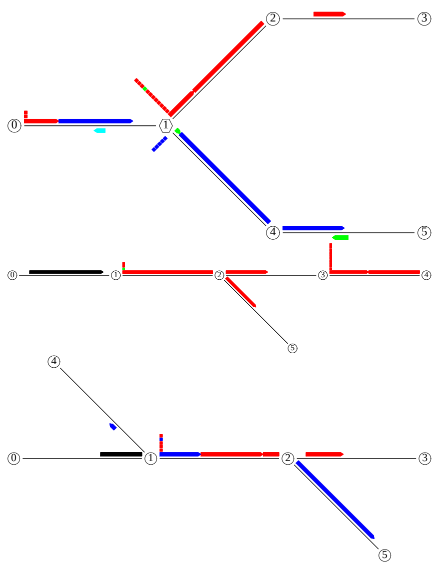

Figure 12: SOT simulation on specific problem topologies

Tests included sparse and dense topologies, bottlenecks in series and introduction of additional non-real-time TCP flows during SOT sessions. In all situations the simulations performed as expected forming groups and setting up transcoders around the problem links.

In the rest of this section simulation results evaluating the performance of SOT and the congestion control algorithm are presented. The experiments performed were designed to measure the time-scales over which SOT reacts to a congestion problem, the amount of extra bandwidth used by SOT control messages and the level of network friendliness achieved by the congestion control algorithm.

In all the simulations packets are sent with a frequency of 50 pps (20 ms duration) simulating an audio source[1]. Transcoded packet sizes start from 32 bytes and are increased in steps of 32 bytes up to the incoming stream bandwidth. Although the number of available coding algorithms is limited and hence the number of possible packet sizes, with the combination of redundant information all the sizes used in the simulation should be possible in a real audio tool. For video transcoding things are simpler as image quality selection provides a fuller transmission range.

[1] The simulation in this paper is tailored for multicast audio however the proposed scheme can be applied equally well to other types of real-time multicast streams including video.

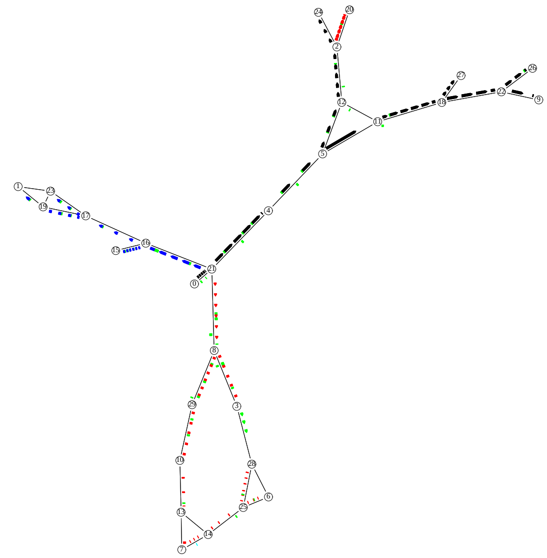

Figure 13: SOT simulation on randomly created topologies

4.4 Transcoder Initiation Evaluation

The main goal of the transcoding initiation algorithm is to quickly respond to a congestion problem by setting up a transcoder. The experiments performed aim at measuring the amount of control traffic introduced to the network during initiation and the delay between detecting a problem and completing initiation.

As transcoder request messages are also used to suppress further requests from other members of a loss group, only one request can be in progress at any one time. In the simulations we resolve conflicts by giving priority to the request reporting the highest loss. Back-off of competing request groups is achieved by introducing a random delay in the message that instructs other receivers how long they should wait before retrying. This resolves a request synchronisation problem and reduces convergence time.

To measure the back-off algorithm performance we used sessions with varying number of bottlenecks. The designed network topology is shown in figure 14. The session source is positioned in the centre of a star topol-

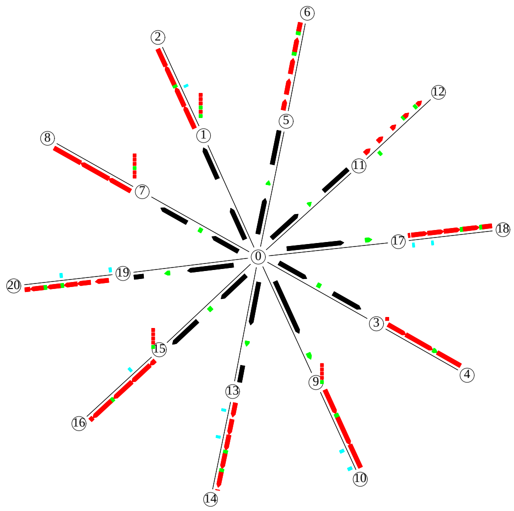

Figure 14: SOT simulation on session with 10 bottlenecks

ogy. Each branch contains two receivers connected in series. The connection from the sender to the first receiver is a high bandwidth link (0.5 Mb/s) with delay varying between 20ms and 40ms. The connection between the first and second receiver is a lower bandwidth link (150Kb/s to 200Kb/s) with longer delay varying between 50ms and 100ms. The session bandwidth is set at 256Kb/s and as a result transcoders need to be set up on all branches between the first and second receivers.

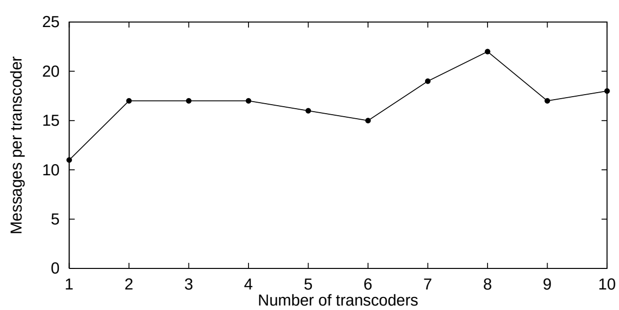

Figure 15: Number of SOT control messages sent during transcoder initiation for different scale simulations

The experiment was repeated with scale varying from one to ten branches. The number of messages that were sent during initiation over the number of transcoders that were set up is shown in figure 15 for different session sizes. The messages per transcoder is roughly constant showing that the back-off mechanism does not get stuck in loops as a result of request conflicts.

In all the simulations the source starts sending RTP

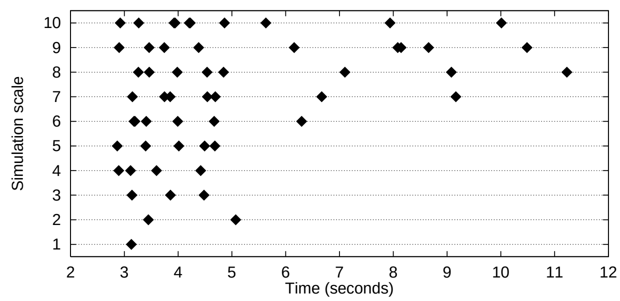

Figure 16: Transcoder initiation times for different scale simulations

packets at time 1.5 seconds and the first packets are dropped when the queues fill up somewhere around time 2 seconds. From then on the transcoder initiation process begins. Figure 16 shows the completion times for the individual transcoder initiations for each simulation. It can be seen that transcoders are set up more densely at the beginning of the simulation and then initiations reach a steadier rate. The initial concentration can be attributed to control packet loss. The links are very congested before the transcoders start and as a result request messages may not reach other requesters behind different bottleneck links thus allowing more than one initiation to progress in parallel. As the simulation progresses and congestion relaxes, the back-off algorithm works better and reduces conflicts by spreading out initiation times.

A reduction of initiation times can be achieved by removing the back-off algorithm and allowing multiple initiations to take place simultaneously. To ensure that other members of the same loss group are still suppressed when one member sends a request, a loss bitmap has to be included in the message. Receivers of the request can then decide if it refers to their loss problem or not. The problem with allowing simultaneous initiations is use of excessive bandwidth with control messages at any time. This is alleviated by the following factors:

- Very few messages need to be exchanged during initiation
- It is not important if control packets get lost and do not reach the rest of the net, as initiations should be localised close to the problem
- The number of initiations is proportional to the number of bottlenecks and not session size, although they may have a close relation depending on geographical coverage

Persistent Responses: Transcoder response messages serve two purposes. They suppress additional offers and double as a response to the requester. In order to reach the requester the response has to pass through a congested link. To improve the chances of reaching the requester the offering site can follow the multicast response with a small number of unicast copies of the message addressed to the requester. These should be spaced in time by a small random interval.

Initiation Congestion Reduction: When a new transcoder starts two new congestion problems can arise. Prunes from the receivers of the new transcoded group for the original data take time to be forwarded to the transcoding site and hence the original group will still be forwarded for some time causing more congestion on the problematic link. This can be partially avoided by starting the transcoder at very low bandwidth (like a slow-start). The second problem is that receivers of the original group that are not interested in the new multicast traffic need time to prune it. In the meantime the additional traffic may cause problems in previously non-congested links. The slow-start will help here as well. An additional measure can be to have transcoding offers followed by a single packet in the intended new group. In this way receivers that are not interested in the new traffic can have a head start with pruning.

4.5 Congestion Control Algorithm Performance

In order to measure the performance of the congestion control algorithm two different experiments were performed. In the first experiment ten SOT sessions are created that share the same bottleneck link. This arrangement is shown in the topology of figure 17. Each session contains two members, the sender and one receiver. All senders are positioned on nodes on one side of the bottleneck link and all receivers on the other. The bandwidth of all sessions is set at 256kb/s. The available bandwidth on the bottleneck link is set at 1Mb. The aim of the experiment is to show that transcoders are set up and that they fairly share the bandwidth of the bottleneck link.

Figure 18 shows the packet size variation for each of the transcoders during the simulation. There is considerable variation due to the adaptation process of slowly increasing and then halving the bandwidth but all the sessions oscillate around roughly the same packet size. This is better illustrated in figure 19, which shows the average packet size and the standard variation during adaptation for each of the transcoders.

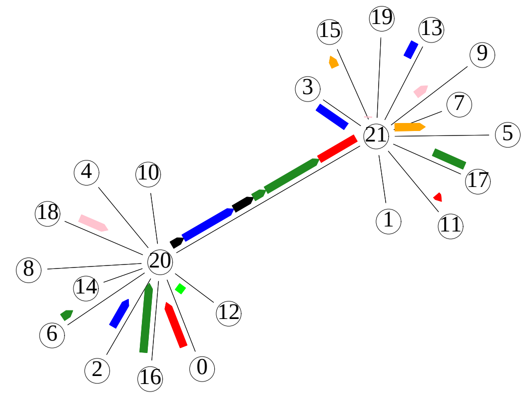

Figure 17: Multiple SOT sessions sharing a bottleneck

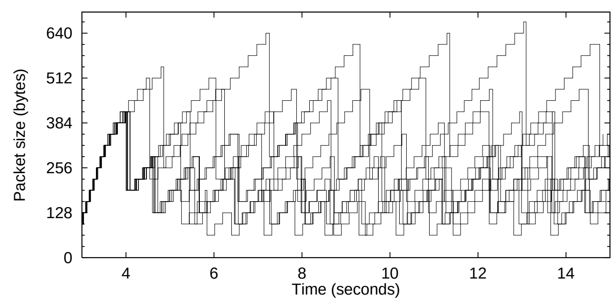

Figure 18: Packet size variation for each transcoder during adaptation

The percentage of simulation time that the transcoders spend sending each packet size can be seen in figure 20. This graph is cumulative for all ten sessions. Although during the adaptation some extreme small and large packet sizes are reached, for most of the simulation a small number of packet sizes is used that is close to the optimum.

The cycle of variation between packet sizes will require switching between different codecs and levels of redundancy. Care needs to be taken in order to limit the impact of this variation on the user. With audio the changes will be very hard to perceive except for the cases where really low bandwidth codecs (like LPC) are used. However, the amount of time spent by the adaptation algorithm in packet sizes requiring such codecs will be very small (a few packets). Perception experiments carried out at UCL have evaluated the impact of mixing small intervals of LPC synthetic speech with toll-quality speech for the purposes of audio redundancy [26]. Results show that for small intervals (around 40ms) intelligibility of speech does not deteriorate whereas for intervals larger than 80ms there is a slight deterioration.

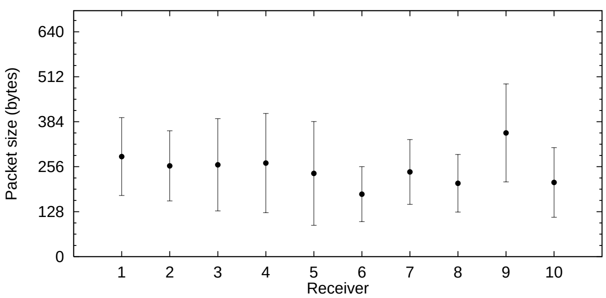

Figure 19: Average and standard deviation of packet size for each transcoder

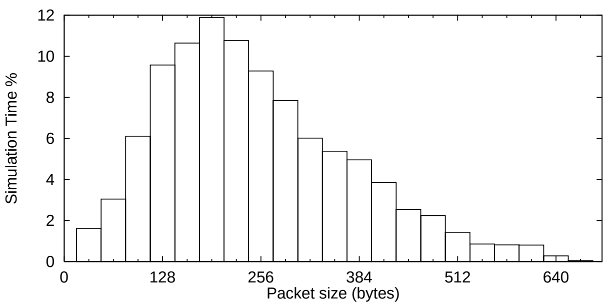

Figure 20: Percentage of simulation time that each packet size was used by all the transcoders

The second experiment shows fair sharing of the bottleneck bandwidth with TCP. The same topology was used and the bottleneck link shared between four SOT and four TCP sessions. The default ns drop-tail queueing strategy was used on the bottleneck link. Using this strategy, the queue capacity is measured in number of packets and packet size makes no difference. Figure 21 shows the total bottleneck bandwidth used by SOT and TCP for each second of the simulation after the SOT transcoders have started. The two curves are very evenly matched. Jain's index [27] with each individual flow as a user gives 99.5% fairness.

By modifying the simulation to use drop-tail queues that take into account packet size and have limited buffer space, the fairness drops to 96.8% The reason for the change is that SOT packets are smaller than TCP packets and hence have a better chance of fitting in a full queue.

5 Conclusions

This paper has presented a solution to the problem of multicast congestion control for real-time streams. The proposed scheme uses self-organisation to form groups

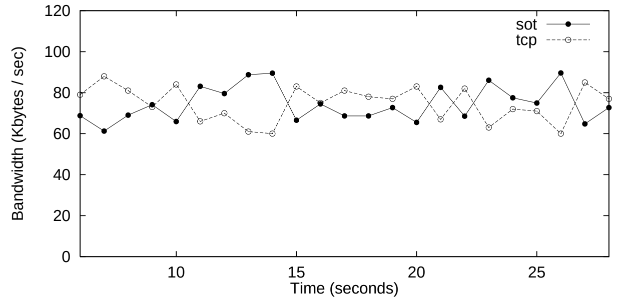

Figure 21: Bandwidth use of 4 SOT and 4 TCP sessions sharing a bottleneck link

out of co-located receivers with bad reception and provides local repair through the use of transcoders. The receiver driven nature of the protocol ensures high scalability and applicability to large Mbone sessions. An evaluation of the proposed scheme has been conducted through simulation. The results are encouraging as they show that the protocol is viable.

The simple congestion control implementation indicates that fair sharing of bottleneck links between real-time multicast traffic and traditional unicast traffic is possible. The simulation shows that adaptation is possible even with the limited bandwidth of audio communication. The significantly broader range available with video can provide additional freedom to the adaptation algorithm.

6 Acknowledgements

We would like to thank Mark Handley, Orion Hodson and Lorenzo Vicisano for their contribution through many useful discussions at UCL and Luigi Rizzo for his comments on drafts of the paper.

References

[1] Ramachandran Ramjee, Jim Kurose, Don Towsley, and Henning Schulzrinne. Adaptive playout mechanisms for packetized audio applications in wide-area networks. In *Conference on Computer Communications (IEEE Infocom)*, Toronto, Canada, June 1994.

[2] Isidor Kouvelas, Vicky Hardman, and Anna Watson. Lip synchronisation for use over the internet: Analysis and implementation. In *IEEE Globecom '96*, London, UK, November 1996.

[3] Colin Perkins, Vicky Hardman, Isidor Kouvelas, and Angela Sasse. Multicast audio: The next generation. In *International Networking Conference (INET)*, Kuala Lumpur, Malaysia, June 1997.

[4] Jean-Chrysostome Bolot, Sacha Fosse-Parisis, Mark Handley, Vicky Hardman, Orion Hodson, Isidor Kouvelas, Colin Perkins, and Andres Vega-Garcia. RTP payload for redundant audio data. Request for comments (Proposed Standard) RFC 2198, Internet Engineering Task Force, Audio / Video Transport Working Group, September 1997.

[5] Mark Handley. An examination of Mbone performance. Research Report ISI-RR-97-450, USC/ISI, April 1997.

[6] Steven McCanne, Van Jacobson, and Martin Vetterli. Receiver-driven layered multicast. In *SIGCOMM*, pages 117–130, Stanfort, CA, August 1996. ACM.

[7] Steve McCanne et al. UCB/LBNL/VINT network simulator - ns (version 2). Software and documentation available from http://www-mash.cs.berkeley.edu/ns/, November 1997.

[8] Henning Schulzrinne, Stephen Casner, Ron Frederick, and Van Jacobson. RTP: a transport protocol for real-time applications. Request for comments (Proposed Standard) RFC 1889, Internet Engineering Task Force, January 1996.

[9] Henning Schulzrinne. RTP profile for audio and video conferences with minimal control. Request for comments (Proposed Standard) RFC 1890, Internet Engineering Task Force, January 1996.

[10] L. R. Rabiner and R. W. Schafer. *Digital Processing of Speech Signals*. Prentice Hall, 1978.

[11] Mostafa Hashem Sherif, Duane O. Bowker, Guido Bertocci, Bruce A. Orford, and Gonzalo A. Mariano. Overview and performance of CCITT/ANSI embedded ADPCM algorithms. *IEEE Transactions on Communications*, 41(2):391–398, February 1993.

[12] S. Pejhan, M. Schwartz, and D. Anastassiou. Error control using retransmission schemes in multicast transport protocols for real-time media. *IEEE/ACM Transcations on Networking*, 4(3):413–327, June 1996.

[13] N.F. Maxemchuk, K. Padmanabhan, and S. Lo. A cooperative packet recovery protocol for multicast video. In *International Conference on Network Protocols*, Atlanta, Georgia, October 1997.

[14] X. Rex Xu, Andrew C. Myers, Hui Zhang, and Raj Yavatkar. Resilient multicast support for continuous-media applications. In *International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, St. Louis, Missouri, May 1997.

[15] Lorenzo Vicisano, Luigi Rizzo, and Jon Crowcroft. TCP-like congestion control for layered multicast data transfer. In *Conference on Computer Communications (IEEE Infocom)*, San Francisco, California, March 1998.

[16] Joseph C. Pasquale, George C. Polyzos, Eric W. Anderson, and Vachaspathi P. Kompella. Filter propagation in dissemination trees: Trading off bandwidth and processing in continuous media networks. In *International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, pages 259–268, Lancaster, U.K, November 1993.

[17] S. Floyd, V. Jacobson, S. Liu, S. McCanne, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transcations on Networking*, 5(6):784–803, December 1997. An earlier version of this paper appeared in *ACM SIGCOMM 95*, August 1995, pp. 342-356.

[18] Puneet Sharma, Deborah Estrin, Sally Floyd, and Lixia Zhang. Scalable session messages in SRM using self-configuration. submitted to Sigcomm '98, February 1998.

[19] D. Mills. Simple Network Time Protocol (SNTP) version 4 for IPv4, IPv6 and OSI. Request for comments (Proposed Standard) RFC 2030, Internet Engineering Task Force, October 1996. Obsoletes RFC 1769.

[20] Mark Handley. SAP: Session Announcement Protocol. Internet Draft, Internet Engineering Task Force, November 1996. Work in progress.

[21] Jean-Chrysostome Bolot, Thierry Turletti, and Ian Wakeman. Scalable feedback control for multicast video distribution in the internet. In *SIGCOMM*, London, UK, August 1994. ACM.

[22] Christos Papadopoulos, Guru Parulkar, and George Varghese. An error control scheme for large-scale multicast applications. In *Conference on Computer Communications (IEEE Infocom)*, San Francisco, California, March 1998.

[23] Vern Paxson. End-to-end routing behavior in the internet. *IEEE/ACM Transcations on Networking*, 5(5):601–615, October 1997.

[24] Brad Cain, Steve Deering, and Ajit Thyagarajan. Internet Group Management Protocol, version 3. Technical report, Internet Engineering Task Force, November 1997. Internet Draft (work in progress).

[25] W. Stevens. TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms. Request for comments (Proposed Standard) RFC 2001, Internet Engineering Task Force, January 1997.

[26] Vicky Hardman, Martina Angela Sasse, Mark Handley, and Anna Watson. Reliable audio for use over the Internet. In *International Networking Conference (INET)*, September 1995.

[27] R. Jain, D. Chiu, and W. Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. Technical Report DEC Research Report TR-301, DEC, September 1984.