Scalability Issues with the Hierarchical Feedback Aggregation for Large-Scale IPTV Systems

D.Komosny[1], K.Ganeshan[2], M.Jelinek[1] and R.Burget[1]

[1]Dept. of Telecommunications, Brno University of Technology,
Brno, Czech Republic
[2]School of Computing & Information Technology, Auckland, New Zealand
e-mail: {komosny, xjelin25, burgetrm}@feec.vutbr.cz, kganeshan@unitec.ac.nz

Abstract

Hierarchical feedback aggregation is an enhancement to the RTP/RTCP (Real Time Protocol/Real Time Control Protocol) when the SSM (Source-Specific Multicast) is employed. The main purpose of feedback transmission is to provide information about media delivery and information needed for RTP/RTCP session management. This paper deals with a tree structure for hierarchical feedback aggregation specifically designed for use in large-scale IPTV systems. This tree consists of end nodes and feedback targets. A feedback target collects feedback from a group of nodes, which are either end nodes or other feedback targets. The feedback gathering process continues up to the main feedback target, which is the top of the tree. The maximum number of nodes below a feedback target in the tree hierarchy should be known in order to avoid overloading the feedback target. For the purpose of estimating this maximum number, we used PlanetLab. The use of PlanetLab allowed us involving many more nodes spread around the globe in measurement. Finally, we propose the maximum recommended number of nodes below a feedback target in the tree hierarchy.

Keywords

Multicast, Feedback, Hierarchical Tree, IPTV

1. Introduction

Multimedia broadcasting on the Internet usually uses one-to-many multicast delivery from one source to many receivers. Aside form this forward media stream, feedback is usually transmitted in the opposite direction i.e. from receivers to one node. This node may or may not be the media source. The main purpose of feedback is to provide information about media delivery and reception. Feedback on reception usually involves information such as packet loss, delay and jitter. This feedback information is used by the upper layer protocols to control and monitor the session properties. Thus, the session control accuracy is strongly affected by the feedback reporting interval. Even minimally outdated feedback values can burden further data processing with an error and, consequently, the behaviour of the session can be faulty. Therefore, the feedback should be sent as often as possible. On the other hand, uncontrolled feedback transmission can overload the network to such an extent that not enough bandwidth is left for media transmission from source to receivers.

The feedback is usually sent by every session member. This is the case with Real Time Protocol/Real Time Control Protocol (RTP/RTCP) (Schulzrinne et al. 2003). In particular, the RTCP protocol is the one used for feedback distribution among session members. With RTCP, several packets for feedback transmission are used. The two basic packets are the sender report SR, which carries transmission and reception statistics, transmitted from the source to receivers, and the receiver report RR, which carries reception statistics, transmitted from receivers to the source. The feedback reporting interval varies a great deal depending on the session size. The reason for this is that in the case of large sessions, RTCP packets can interfere with RTP packets, which carry media, and cause RTP packet to be delayed and lost. Therefore, a mechanism to control the frequency of RTCP packets is used. The algorithm used keeps the frequency of RTCP packets at a value corresponding to 5% of the total allowed session bandwidth. Furthermore, 3.75% of the session bandwidth is used by RR packets and 1.25% of the session bandwidth is used by SR packets. Supposing that Source-Specific Multicast (SSM) is employed (Bhattacharyya, 2003),(Holbrook and Cain, 2004), the RR packet rate PR_{RR} is identified as,

$$PR_{RR} = \frac{0.75 \times BW_{RTCP}}{PS_{RR} \times n_{rec}} \tag{1}$$

and the SR packet rate PR_{SR} is identified as

$$PR_{SR} = \frac{0.25 \times BW_{RTCP}}{PS_{SR} \times n_{ser}} \tag{2}$$

where PS_{RR} is the average size of the RR packet, PS_{SR} is the average size of the SR packet, BW_{RTCP} is the allowed bandwidth used for RTCP packets, n_{rec} is the number of receivers, and n_{ser} is the number of senders (Schulzrinne et al. 2003). These formulas assume that the number of senders is less than or equal to 25% of session members. Furthermore, it is assumed that the session has four receivers or more. The algorithm used gives 25% of the RTCP bandwidth to the source and 75% of the RTCP bandwidth to receivers. Considering equation (1), large-scale sessions such as IPTV broadcasting experience large delays between sending RTCP packets. This drawback was identified as one of the major problems with RTP/RTCP (Rosenberg and Schulzrinne, 1998). However, feedback produced by individual session members could be aggregated to reduce the amount of transmitted data, and consequently, the feedback reporting interval (Chesterfield and Schooler, 2003; El-marakby and Hutchison, 1998).

In the next section, we give an overview of a typical IPTV system. Then we discuss a new method called hierarchical feedback aggregation and, consequently, we introduce our modified feedback transmission tree to be used in IPTV systems. We demonstrate the use of the proposed tree using sample calculations for IPTV using our modified feedback transmission tree. In the rest of the paper we describe the measurement tool developed and present measured values and their application in the feedback tree establishment.

2. Hierarchical feedback aggregation in IPTV

Many service providers recognise the advantages of multicast and have started using it. An example of this is IPTV delivery. IPTV systems use both unicast and multicast for media distribution. Unicast is used for video on demand services and, on the other hand, multicast is used for broadcasting (see Figure 1). A dedicated multicast group is created for each TV channel being broadcast. However, in IPTV not all channels are sent to the subscriber at the same time. In a typical IPTV system, the subscriber is connected to the distribution network using DSL (Digital Subscriber Line). Multicast packets sent from DSLAM (Digital Subscriber Line Access Multiplexer) to the subscriber's DSL modem are only those belonging to what a TV channel subscriber wants to watch. This mechanism is implemented using multicast group membership and, to be more specific, IGMP (Internet Group Management Protocol) protocol (Cain et al., 2002) is used for joining specific multicast groups. DSLAM handles IGMP messages and, consequently, it allows traffic belonging to the chosen multicast group to be transmitted to the subscriber's DSL modem.

Besides media distribution, IPTV systems may also implement transmission from subscribers' devices for the purpose of status and performance monitoring. In order to assure this, the CPE WAN Management Protocol (Bernstein and Spets, 2004) can be used. This protocol offers a complex solution with implemented security based on SSL/TLS (Secure Sockets Layer/Transport Layer Security). Other features implemented in this protocol are auto-configuration, software and firmware management, and diagnostics. Hierarchical aggregation is another option for data collection from end devices in IPTV systems. The main difference is that hierarchical aggregation allows fast data collection and it does not implement such rich functionalities as CPE WAN Management Protocol. Hierarchical aggregation could be used in cases where the data collected needs to be ready for further processing in a short time. This data could be, for example, user polling results to influence an interactive programme. The following text deals with description of multicast types used on the Internet and how transmission from end nodes is addressed when specific-source multicast is used. Then we focus on hierarchical aggregation.

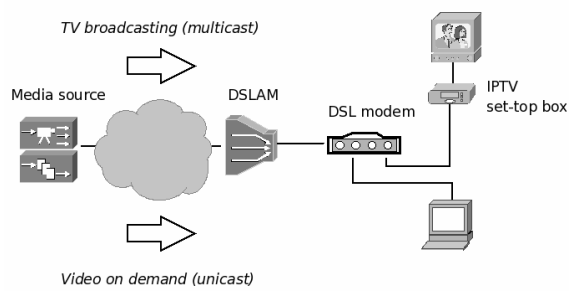

Figure 1: Typical IPTV system

There are two main types of multicast - any source multicast (ASM) (Chesterfield et al. 2007) and the above mentioned source-specific multicast. For IPTV, the SSM is the better productive technology. However, a drawback of SSM is that it lacks the

support for communication among session members since it offers only one-to-many communication. The feedback in the case of RTCP needs to be transmitted to all session members. The existing solutions use unicast connections for feedback transmission from receivers to the source. The two known methods are reflection and summarization (Chesterfield and Schooler, 2003). Both methods deal with the same transmission of RR packets from receivers, but they differ in processing the data received at the source. After processing, the feedback is sent back by the source using SSM to all session members. Using this method, every session member is capable of sending to and receiving feedback from other session members as with any source multicast.

With the reflection method, the source forwards every RR packet received from each receiver to all receivers. However, the forwarding could be harmful to the network load, especially with the session size growing. In addition, the source does not need to forward all the received feedback data – some of them are valuable only for the source, not for the receivers. The second method, summarization, aggregates feedback data at the source. When the aggregation process is over, a summary is created and sent to all receivers. The feedback summary is conveyed using the RSI packet - Receiver Summary Information (Quinn and Almeroth, 2001), which contains several elementary fields for encapsulating the mandatory information, such as timestamps and number of session members.

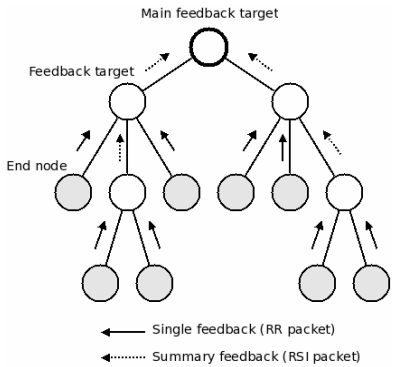

Figure 2: Tree structure for hierarchical feedback aggregation

Another step in the development was the hierarchical feedback aggregation, which in particular, is an enhancement to the summarization method (Chesterfield and Schooler, 2003). With this method, a node acts as a feedback target for a group of nodes, which could be end nodes or feedback targets as depicted in Figure 1. Feedback targets are organized hierarchically and they are supposed to be dedicated servers in order to have adequate resources available for feedback data processing. The feedback is transmitted from end nodes using RR packets to a feedback target. The feedback target aggregates the feedback data into one RSI packet and sends it to a feedback target on the tree level above. Then the process continues up to the main feedback target, which populates the top of the tree. Finally, the main feedback target sends the feedback to the media source for further redistribution to all nodes via SSM as required by RTP/RTCP. The separation of the media source and the main

feedback target is proposed to make the system more scalable in case of large-scale IPTV systems. This scenario is shown in Figure 2.

3. Definition of hierarchical tree for feedback transmission in IPTV systems

For use in IPTV systems, we propose a tree for feedback transmission as follows: Feedback targets are presented only on all tree levels except the lowest and the lowest level consists of end nodes only. We will refer to this tree concept through the rest of this paper. For the purpose of establishing a tree structure that meets the lowest session feedback transmission interval possible (i.e. sum of feedback transmission intervals through all tree levels plus feedback transmission interval from the media source to all session nodes), we can identify the number of tree levels I as

$$I = \log_{n_{\mathrm{gFT}}}\left(\frac{n_{\mathrm{end}}}{n_{\mathrm{gend}}}\right) + 1 \tag{3}$$

where n_{end} is the number of session end nodes, n_{gend} is the number of end nodes in a group (i.e. end nodes below the same feedback target on the level above) and ng_{FT} is the number of feedback targets in a group (i.e. feedback targets below the same feedback target on the level above). Furthermore, we use symbol i for identification of a tree level, where $i=I$ is the lowest tree level and $i=0$ is the level where main feedback target is presented. The equation is true provided that the maximum number of feedback targets in groups on every tree level i is of the same value ($n_{gFT}=n_{gFT(i)}$).

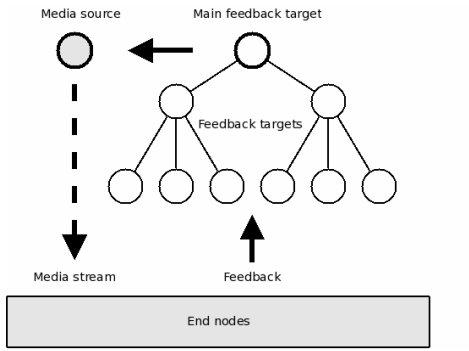

Figure 3: IPTV system with hierarchical feedback transmission

When establishing a tree, we know the required number of session members n_{end}. Furthermore, the number of end nodes in a group n_{gend} and the number of feedback targets in a group n_{gFT} need to be identified. We can identify n_{gend} using equation (1) and substituting RI_{RR} with RI_{min}. RI_{min} is a limit value defined in RTP/RTCP to be 5sec (Schulzrinne et al., 2003). The purpose of the limitation is to avoid packet floods when the session behaves unexpectedly. For instance, when a session is

experiencing a network part failure, the number of session members could become extremely small during a short period of time. This would be harmful to the network load as feedback interval calculation is based on a temporary low number of session members. We use RI_{min} for the value of RI_{RR} to keep the session feedback interval RI_{SS} as low as possible. Thus, after substituting RI_{RR} with RI_{min}, we obtain,

$$n_{\text{gend}} = \frac{RI_{min} \times 0.75 \times BW_{RTCP}}{PS_{RR}} \tag{4}$$

and similarly,

$$n_{\text{gFT}} = \frac{RI_{min} \times 0.75 \times BW_{RTCP}}{PS_{RSI}} \tag{5}$$

where PS_{RSI} is the size of the RSI packet Finally, we can easily identify the session feedback reporting interval RI_{SS} as,

$$RI_{SS} = RI_{SR} + RI_{min} \times (I) \tag{6}$$

Note that with IPTV sessions there is only one source, therefore the reporting interval for SR packet is also limited to RI_{min}. Thus we can also identify RI_{SS} as,

$$RI_{SS} = RI_{min} \times (I + 1) \tag{7}$$

4. IPTV broadcasting with hierarchical feedback aggregation

In this section, we discuss an IPTV broadcasting application example working with the hierarchical feedback aggregation. In particular, we describe the tree establishment process. We assume the following initial values: (1) The maximum required video bandwidth is slightly less than 10Mbps (MP@ML 720x576, 25fps). Some small bandwidth is used for audio. For simplicity, we suggest the session bandwidth for media transmission to be 10Mbps. Moreover in conformity with RTP/RTCP, 5% of the used bandwidth should be dedicated for the feedback transmission. In our case it is 500Kbps. Therefore, we estimate the final session bandwidth to be 10,5Mpbs. (2) We suggest that the number of session receivers n_{end} is 250000. According to some studies, the number of IPTV receivers per international TV channel is likely to be in around this figure in the near future. (3) Finally, we need to know the RR packet size PS_{RR} for identifying the next three values: size of RSI packet PS_{RSI}, the number of end nodes in a group ngend and the number of feedback targets in a group n_{gFT}. The size of RR packet can be identified from (Schulzrinne et al., 2003; Quinn and Almeroth, 2001) (IP header – 20B; UDP header – 8B, RR header – 8B, report block (only one SSRC) – 24B). The result is 480 bits. Note that we omit the SDES packet with a CNAME. CNAME combined with SSRC is used for SSRC collision detection for media sources within a multicast session (Schulzrinne et al. 2003). However, in sessions where session members cannot act as sources the SSRC collision detection is not needed. First of all, we can simply calculate the number of end nodes in a group as

$$n_{\text{gend}} = \frac{5 \times 0.75 \times 500000}{480} = 3907$$

For identification of n_{gFT}, we need to know the size of the RSI packet. We can calculate the RSI packet size as,

$$PS_{RSI} = \text{RSI}_{\text{fix}} + \Sigma\left(RBL_{fix} + RBL_{data}(k)\right) \tag{8}$$

where RSI_{fix} is the fixed part size of RSI packet, K is the number of report blocks, RBL_{fix} is the fixed part size of report block, and $RBL_{data}(k)$ is the variable part size of report block k. We suppose that the RSI packet carries two report blocks – jitter and loss distributions. In each report block, the number of distribution buckets DBN is 256. Note that we do not consider the mandatory blocks "RTCP group and average packet size" and "RTCP bandwidth indication" as defined in (Quinn and Almeroth, 2001). Instead, we try to provide this information in a dedicated packet together with the identified tree structure. The fixed part size of the RSI packet RSI_{fix} is 352 bits (IP header – 20B; UDP header – 8B, RSI header 16B) and the fixed part size of report block RBL_{fix} is 96 bits (Quinn and Almeroth, 2001). We can identify the variable part size of the report block RBL_{data} as

$$RBL_{data} = DBN \times DBL \tag{9}$$

where DBN is the number of distribution buckets, and DBL is the size of distribution buckets. For the purpose of identifying the bucket size, we need to consider the worst-case scenario i.e. all end nodes report feedback values belonging into one bucket. In other words, we need to have enough bits to express the number of all end nodes in a session. Therefore, the worst-case bucket size DBL is,

$$DBL(i) = \log_2\left(n_{\text{gend}} \times n_{\text{gFT}}^{(I-i-1)}\right) \tag{10}$$

where DBL(i) is the bucket size on tree level i. It can be seen from the equation above that on the higher tree levels, the distribution bucket size is large (ie. more end nodes are involved in aggregation). However, as stated before, we need to keep the RSI packet size constant to keep notation $n_{\text{gFT}}=n_{gFT(i)}$ to be true through all tree levels. To assure this, we use the multiplicative factor MF defined in (Quinn and Almeroth, 2001). Utilizing the following equation,

$$DBL = DBL(i) = \log_2\left(\frac{n_{\text{gend}} \times n_{\text{gFT}}^{(I-i-1)}}{D(i)}\right) \tag{11}$$

we are able to calculate the divisor D for the specific tree level i as,

$$D(i) = n_{gFT}^{(I-i-1)} \tag{12}$$

and with the definition of $D=2^{MF}$ in (Quinn and Almeroth, 2001), we are able to identify the multiplicative factor *MF* for each tree level as,

$$MF(i) = \log_2 n_{\mathrm{gFT}}^{(I-i-1)} \tag{13}$$

Thus, when a constant size of RSI packet is achieved by applying *MF*, we can identify the RSI packet size as, PS_{RSI}=352 + 2 x (96 + 256 x $\log_2$(3907)) = 6654bit and, consequently, n_{gFT}, *I* and RI_{SS} as

$$n_{\mathrm{gFT}} = \frac{5 \times 0.75 \times 500000}{6654} = 282 \ , \ I = \log_{282}\left(\frac{250000}{3907}\right) + 1 = 2 \ , \ RI_{SS} = 5 \times 3 = 15\mathrm{sec}$$

If we compare the obtained result with the standard RTP/RTCP, session feedback reporting interval is, in this case, equal to the RR packet reporting interval plus SR packet interval. We can utilize equation (2) to calculate the RR packet reporting interval as 320sec. Adding 5sec (RI_{min}) for the SR packet reporting interval, we then obtain the value of 325sec. It can be seen, that the session feedback interval with hierarchical feedback aggregation is significantly smaller. An overview of the calculated tree is shown in Table 1. The multiplicative factor *MF* is at the top level, and set to 9 (15 is the maximum possible value since four bits are used to carry *MF* value in the RSI packet header) to keep the size of DBL at a constant value of 12 bits through all tree levels where RSI packet is used. The calculated tree is able to cover as many as over one million end nodes with the same session feedback reporting interval equal to 15sec.

5. Scalability issue of large-scale sessions

As described above, an end node produces a single summary for a feedback target up to the main feedback target. In large sessions, both feedback targets and main feedback targets should have adequate resources available for feedback data processing. The question is how many nodes a single feedback target could serve before it starts to drop packets due to overloading. The following section deals with the measurements done to answer this question.

i	n_{FT}	n_{end}	*DBL* without *MF* (bit)	MF	*DBL* with *MF* (bit)
0	1 *	-	21	9	12
1	282	-	12	0	12
2	-	1 101 744	-	-	-

* This feedback target is the main feedback target

Table 1: Overview of the specified tree structure

5.1. The test-bed

A pair of software tools was developed to test how many end nodes or feedback targets respectively can send their feedback to a single feedback target/main feedback target. The first tool is the sender acting like an end node/feedback target and the second is the receiver acting like a feedback target. To describe briefly their

function, the sender starts to send data to a receiver after receiving the initiation packet from receiver with constant length of 12 bytes (Figure 3). The packet carries the specification of the feedback packets to be sent: packet length *L*, number of packets *N* and time period *T* between sending packets. When the sender gets the initiation packet, it starts sending feedback packets as required by the receiver. When all the feedback packets are sent, the receiver processes measured results and it prints the statistical values to the standard program output. Both receiver and sender were implemented in the C++ programming language and the code was compiled and tested under the Linux operating system, kernel 2.6.18, compiler GCC v4.1. This application should be compatible and fully functional under all UNIX-like operating systems.

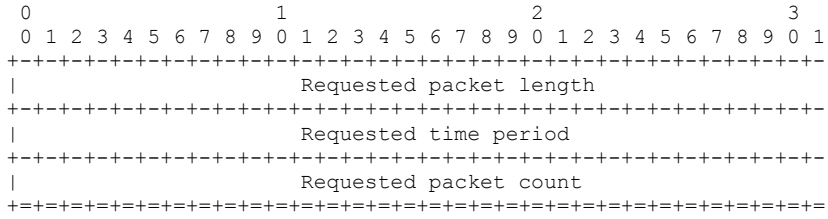

Figure 4: Initiation packet structure

The feedback packet structure is depicted in Figure 4. The first four bytes consist of so-called synchronization source (SSRC) number, which is the identification number of sender. Next 4 bytes carry the sequence number, which counts the transmitted feedback packets. The last sequence of bytes called payload data is filled with random data to achieve the required feedback packet length. In order to assure that feedback packets are not sent in the same intervals from each sender, the packet transmission period is randomized in the interval <0.5; 1.5> of the original value. Before the sender and receivers applications were deployed in an actual network, the correct function was validated in our laboratory network. Several tests were performed and all confirmed reliability of these applications. Then applications were moved to the planetary scale network called PlanetLab. PlanetLab is a global research network that supports the development of new network services. PlanetLab currently consists of 828 nodes at 409 sites.

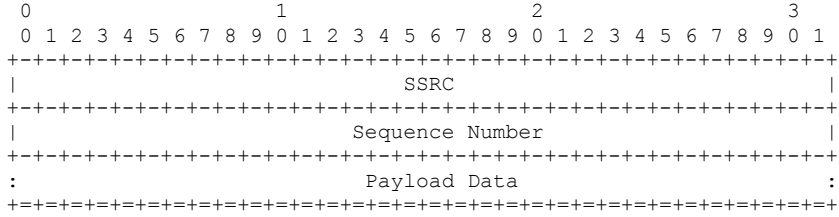

Fig. 5: Feedback packet structure

5.2. Measured results

The important results from measurement taken in PlanetLab are shown in Figures 5 and 6. During our measurement, 383 nodes were active in PlanetLab. Thus only 383

senders were used for generating feedback traffic to the dedicated server in our laboratory. Two feedback packet sizes were used. The size values were taken from the calculations above. The first size was set to 480 bits, which is the value of RR packet used for feedback transmission from end nodes to a feedback target. The second size was set to 6654 bits, which is the value of the RSI packet used for feedback transmission from feedback targets to a feedback target on the next tree level. The first graph shows CPU load vs. feedback transmission interval varying from the minimum value given by the OS to 20 seconds. The investigated value is 5 second, which is equal to the minimum feedback reporting interval RI_{min} defined in RTP/RTCP. The measured CPU load is quite low indicating that the feedback target should be able to deal with this traffic from 383 nodes without any problems. As shown in the figure, both feedback packet sizes give approximately the same result.

A limitation of our measurement was that only 383 nodes could be involved in the measurement since we were particularly interested in a number of end nodes equal to 3907, as previously identified in the calculated tree. We can get results for higher number of nodes by means of increased feedback packet transmission interval at each receiver. For the 3907 end nodes, the feedback packet transmission interval should 5/(3907/383) = 0,49 sec. In this case, we were interested at which interval the receiver will start dropping the packets, since the CPU load significantly increases with small interval values (Figure 5). Figure 6 shows that intervals below 0,5 sec. are critical as higher packet loss occurs there. Again, the result is similar for both feedback packet sizes. Observing this result, we can conclude that the number of nodes sending feedback to one feedback target should be no more than 3907.

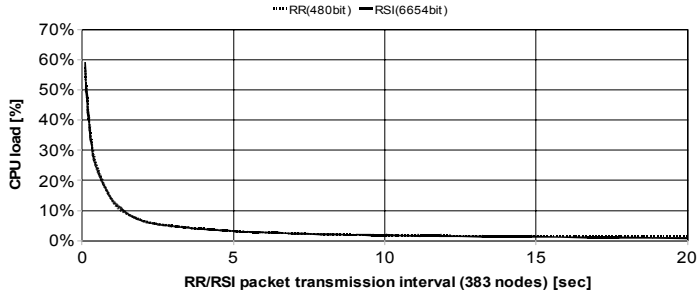

Figure 6: CPU load vs. feedback packet transmission interval

6. Conclusion

In this paper, we focussed on hierarchical feedback aggregation, which is one of the methods for data collection in IPTV systems. We addressed the problem of data collection from a large number of receivers in a short time. Furthermore, we dealt with finding the maximal number of nodes that can send feedback to a single feedback target in hierarchical feedback aggregation. This motivated us to carry out a measurement using two developed applications - sender and receiver. The measurements were taken in PlanetLab, which offered us the opportunity to involve many more nodes than those available in our laboratory. The results show that there is no great difference between the two feedback packet sizes investigated. Finally,

we identified the maximum number of nodes, which should send their feedback to a single feedback target without having significant number of packet being lost.

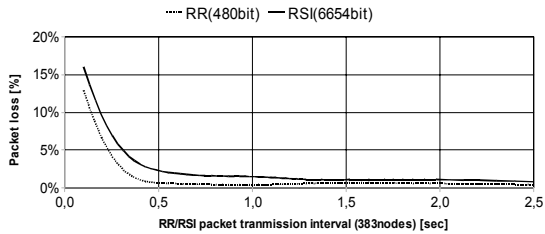

Figure 7: Packet loss vs. feedback packet transmission interval

Acknowledgement: This work was supported by the Academy of Sciences of the Czech Republic project 1ET301710510.

7. References

Bernstein, J. and Spets, T., (2004), "CPE WAN Management Protocol". *Technical Report 069, Digital Subscriber Line Forum*.

Bhattacharyya, S., (2003), "An Overview of Source-Specific Multicast (SSM)", *Request for Comments 3569*, Internet Engineering Task Force.

Cain, B., Deering, S., Kouvelas, I., Fenner, B. and Thyagarajan, A., (2002), "Internet Group Management Protocol", Version 3. *Request for Comments 3376*, Internet Engineering Task Force.

Chesterfield, J. and Schooler, E., (2003), "An Extensible RTCP Control Framework for Large Multimedia Distributions", *Proceedings of the Second IEEE International Symposium on Network Computing and Applications*, IEEE Computer Society.

Chesterfield, J., Schooler, E. and Ott, J., (2007), "RTCP Extensions for Single-Source Multicast Sessions with Unicast Feedback", *Internet Draft, work in progress*, Internet Engineering Task Force.

El-Marakby, R. and Hutchison, D., (1998), "Scalability Improvement of the Real-time Control Protocol (RTCP) Leading to Management Facilities in the Internet", *Proceedings of the Third IEEE Symposium on Computers & Communications*, IEEE Computer Society.

Holbrook, H. and Cain, B., (2004), "Source-Specific Multicast for IP". *Internet Draft, work in progress*, Internet Engineering Task Force.

Quinn, B. and Almeroth, K., (2001), "IP Multicast Applications: Challenges and Solutions", *Request for Comments 3170*, Internet Engineering Task Force.

Rosenberg, J. and Schulzrinne, H., (1998), "Timer reconsideration for enhanced RTP scalability", *Proceedings of Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies*, IEEE.

Schulzrinne, H., Casner, S. and Frederick, R., (2003), "RTP Profile for Audio and Video Conferences with Minimal Control", *Request for Comments 3551*, Internet Engineering Task Force.

Schulzrinne, H., Casner, S., Frederick, R. and Jacobson, V., (2003), "RTP: A Transport Protocol for Real-Time Applications", *Request for Comments 3550*, Internet Engineering Task Force.