Classification of Receivers in Large Multicast Groups using Distributed Clustering

Kavé Salamatian, Thierry Turletti

Classification of Receivers in Large Multicast Groups using Distributed Clustering

Kavé Salamatian
LIP6
University of Paris VI

Thierry Turletti
INRIA
Sophia Antipolis

{Kave.Salamatian,Thierry.Turletti}@sophia.inria.fr

Abstract

In this paper, we describe and evaluate a mechanism for filtering RTCP receiver reports sent from receivers to the whole session. In large multicast groups, such feedbacks could induce network congestion, receivers overload and feedback delay. The proposed filtering mechanism provides a classification of receivers according to a predefined similarity measure (e.g. packet loss rate observed or *TCP-friendly* bandwidth share). This mechanism could be used for example by layered transmission schemes in which coders adapt dynamically the number and the rate of layers according to clusters of receivers.

1 Introduction

Transmission of multimedia flows over multicast channels is confronted with the receivers heterogeneity problem. A multicast transmission, in which all receivers in the multicast group are similar and receive the same packets, can be assimilated to a single unicast transmission, and the feedback of only one receiver is sufficient to estimate the state of all. Unfortunately, it is a matter of fact that receivers in a multicast group are heterogeneous in term of CPU power (or CPU load) and in term of connection characteristics.

Layered transmission has been proposed to cope with receivers heterogeneity [1, 2, 3]. In this approach, the source is represented using a base layer, and several successive enhancement layers refining the quality of the source reconstruction. Each layer is transmitted over a separate multicast group and receivers decide the number of groups to join (or leave) according to the quality of their reception. At the other side, the sender can decide the optimal number of layers and the encoding rate of each layer according to the feedback sent by all receivers.

In fact, the sender does not need reports of each receiver in the multicast group. It rather needs a partition of the receivers into some homogeneous classes. Each layer of the layered coding can be adapted to the characteristics of one (or a group) of these classes. Each class represents a group of homogeneous receivers, for which the sender should take adaptive actions. Classification should be done using feedback of discriminative variables that are related to the subjective quality. These feedbacks could be transmitted using RTCP [4] Receiver Reports (RRs) possibly modified to fit specific application requirements.

Packet loss is an important curse for delay-constrained multimedia applications, such as video conferencing in multicasting environments. These applications experience quality degradation with increased packet losses and transmission delays in the network. Recent results suggest that error control schemes using FEC (*Forward Error Correction*) strongly reduce the impact of packet losses [5, 6, 7]. In these schemes, redundant information are sent along with the original information so that the lost data (or at least part of it) can be recovered from the redundant information.

Clearly, sending redundancy increases the probability of recovering packets lost, but it also increases the bandwidth requirements and thus, the loss rate of the multimedia stream. FEC schemes should be coupled to adaptive control schemes that determine the transmission parameters (redundancy level, source coding rate, type of FEC scheme, etc.) as a function of the state of the multicast channel, to achieve the best subjective quality at receivers. For such adaptive mechanisms, it is important to have simple channel models that can be estimated in an on-line manner. Feedback about quality degradations sustained by receivers are necessary to make adaptive decisions at the sender side. But multicast channels are highly asymmetric : they are point-to-multipoint in one direction and multipoint-to-point in the other direction. Only one multicast flow gets out of the sender and several feedback flows get into it. To avoid possible feedback implosion in a large group of receivers, the RTP standard specifies that all the RTCP reports sent must not consume more than 5% of the bandwidth assigned for whole the session. So, when the group size increases, the RTCP interval increases resulting in RTCP infrequent feedback reports which may decrease the interest of the feedback.

Recent investigations have shown that in unicast [8, 9] and in multicast [10, 11], the loss process exhibits strong temporal dependences and important memories. These temporal dependences show themselves as frequent consecutive packet losses. Analytic models [12], confirmed by empirical studies [8], suggest that low order Markov chains are suitable for modelling temporal dependences.

To cope with empirical fluctuations of the packet loss rate, priority encoding techniques are sometimes used in conjunction with layered (or hierarchical) representation of the source [7]. With this approach, base layers are protected by more powerful FEC schemes than enhancement layers in order to guarantee the reception in priority of base layers. The level of protection to provide for each layer of the hierarchical representation should be chosen by taking into account the past history of temporal variation of loss rate in each layer. The main argument here is based on the stationary assumption : if a receiver has experienced a particular network state, it is more likely to experience it one more time in the near future. In fact, fluctuation of the network quality can be assimilated to receivers heterogeneity and the same set of solutions could be used to reduce the negative impact of it. Theoretic relationship between priority encoding techniques and multicast in heterogeneous environment has been shown in [13].

Multicast transmission are also known to exhibit strong spatial correlations [10, 14]. These spatial correlations are due to the tree structure of multicast routing: losses occurred upstream of a routing node will be observed by all receivers in the attached subtree. Receivers that obtain a multicast flow from the same routing node are more likely to expect similar losses. This has fuelled the idea that feedbacks from these receivers can be aggregated into a single compact feedback report.

We show in this paper, that a simple distributed aggregation mechanism can classify receivers and solve by this ways the feedback implosion problem and a the same time provide the sender with a highly compressed and efficient representation of the receivers.

In the remainder of this paper, we describe a mechanism to solve the scalability problem of RTCP receiver reports in large sessions. The mechanism is based on the classification of receiver reports done by aggregation agents organised into a hierarchy of local regions as described in Section 2. The clustering mechanism is described in Section 3 and simulations of the mechanism using real MBone traces are shown in Section 4. In Section 5, we conclude and present some related work.

2 Aggregators organisation within the network

In this section, we discuss how to set up aggregation agents (AAs) at strategic positions within the network in order to minimise the bandwidth

overhead of RTCP receiver reports (RRs).

We have chosen a multi-level hierarchical approach such as described in the RMTP [15] protocol in which receivers are grouped into a hierarchy of local regions, see Fig. 1. However, in our approach, there are no designated receivers: all receivers send their feedback to their associated aggregator.

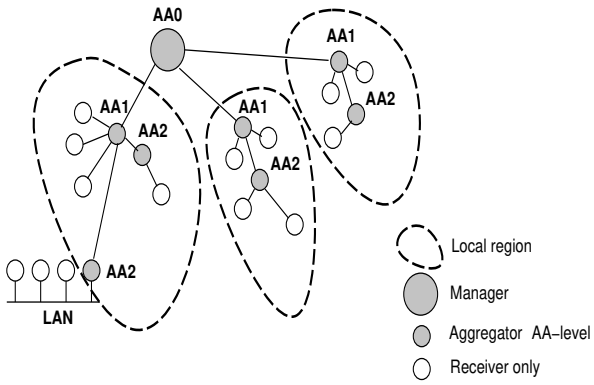

Figure 1: Multi-level hierarchy of aggregators

Each region contains an aggregator that receives feedback, performs some aggregation statistics and send them in point-to-point to the higher-level aggregator (*merger*). The root of the aggregator tree hierarchy (called the *Manager*) is based at the sender and receives the overall summary reports. In our approach, the overall summary report is a classification containing the number of receivers in each class, the mean behaviour of the class and the variance around this mean. The mechanism of aggregation is described in section 3.

In our experiments, aggregators are manually set up within the network. However, other approaches could be used to automatically launch aggregators within the network. For example, we could implement the *aggregator* function using a *custom concast* [16]. *Concast* addresses are the opposite of multicast addresses, i.e., they represent groups of senders instead of groups of receivers. So, a *concast* datagram contains a multicast group source address and a unicast destination address. With such a scheme, all receivers send their RRs feedback packets using the RTCP source group address and to the sender's unicast address, and only one aggregated packet is delivered to the sender. The *custom concast* signalling interface allows the application to provide the network the description of the merging algorithm function (e.g. using Java).

Another possibility could be to implement the aggregator function as an *active service* in [17]. Within this architecture built on top of the existing Internet service model, receivers behave as *clients* that launch agents (such as aggregators) at strategic locations within the network. Agents are application-level services called *servents* and within the network reside one or more pools of active service nodes each called a *cluster*. The algorithm is as follows. Before a client can instantiate a servent, it has first to obtain bootstrap configuration information that enables it to rendezvous with an AS1 cluster. This can be done either using DHCP[18] or by listening to a well-known multicast address over which the required information is periodically transmitted. Once clients rendezvous with the active service, they can create instances of servents within the service cluster. AS1 implements a distributed *Active Service Control Protocol* (ASCP) that uses an *announce-listen* communication model. Basically, clients announce their requests for service instances and host managers (HM) respond to these announcements by instantiating a single servent. Then, servents announce their existence to both the client and other potential servent sites to avoid duplicate servents. If a servent crashes, its state will time out and so the next client announcement will be used as a service request to the cluster. If a client crashes, its state in the system will expire and will trigger the termination of its servent. Note that using this approach, aggregators servents only subscribe to RTCP flow and not to both data and control flows.

3 Clustering mechanisms

3.1 Preliminaries

Multicast transmission has been reported to exhibit strong spatial correlations [10]. As in the case of temporal dependences in which compression methods can greatly reduce the size of a description of temporally correlated data, spatial correlation can be used to compress the amount of data required for representing feedback data sent by receiver.

Moreover, for sender-based multicast regulation, only a classification of receivers is sufficient to apply adaptation decisions. In this case, the compression of feedback reports can be viewed as a vector quantisation [19] that use spatial dependences of receivers reports to achieve a compact representation of the receivers. Vector quantisation achieves a classification by clustering receivers issuing similar (by a predefined similarity measure) receiver reports.

The classification of receivers should not only be based on actual receiver reports, but it should also integrate the recent history of receivers. As we explained in the introduction, different reception states experienced by receivers during past periods, should be treated as reports of different and heterogeneous receivers. In this way a receiver can change its class during consecutive reporting periods. However, since over large time scales, the stationary hypothesis cannot be always validated, a procedure should be added to ensure that we forget about far past time reports. Temporal fluctuations of receiver state can be directly introduced into the clustering or be treated independently at sender side by tracking variation in the classification.

The objective of the classification algorithm is to cluster similar reception behaviours into homogeneous classes. Before describing the classification algorithm, several concepts should be introduced. First, we should choose the discriminative characteristic we need to cluster for the classification. One obvious characteristic that can be used is the loss rate observed by a receiver over a fixed size interval of time. However this value does not completely describes the state of reception in the receiver because it does not take into account the memory of the loss process. Moreover, empiric loss rate can have very large fluctuations, specially in the case of Markovian loss models. Another choice is to use more sophisticated model-based characteristic as Markov chain parameter [6, 20]. Indeed, recent results using model order estimation with entropy and minimum description code approaches [21] show that a Gilbert model, i.e., a 2-state Markov process, is a good model for the loss process observed in traces [1] (e.g. [10]). This is consistent with others, more general, results on end-to-end Internet characteristics (e.g. [9]). In this approach, the loss process is supposed to be a 2-state Markovian model (the Elliot-Gilbert Model) (Fig. 2) in which one state (which we refer to as state B BAD) represents a packet loss, and the other state (which we refer to as state G $GOOD$) represents a packet reaching the destination. Let p denote the probability of going from state G to state B, and let q denote the probability of going from state B to state G. Then the residence times for states B and G are both geometrically distributed with means $1/p$ and $1/q$, respectively. The mean observed loss for this model is $\bar{p_l} = \frac{p}{p+q}$. These properties are used to estimate the parameters of the Elliot-Gilbert model.

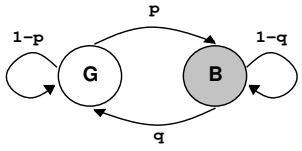

Figure 2: The Gilbert model

The model above can be refined to an hidden Markov chain model. In this model, the probability of loss in the bad state (p_B) is not equal to one ($p_B < 1$) and in the good state (p_G) is not equal to zero ($p_G > 0$). In this case, the mean observed loss will be $\bar{p_l} = \frac{p}{p+q}p_B + \frac{q}{p+q}p_G$. As the states

[1] Note that this does not indicate that the Gilbert model is the model that best fits the traces, but rather that the Gilbert model best strikes a balance between better fit and increased model complexity (or equivalently increased number of states).

of the model are unobservable, the parameter of this model should be obtained using Expectation Maximisation (EM) methods [22].

Other kinds of characteristics could be interesting to cluster. For example, we could use the available bandwidth estimated by each receiver using an analytical model of TCP for estimating the TCP-Friendly bandwidth share [23]. Sometimes, some combinations of characteristics can be suitable. The clustering mechanism can be applied to such a combination of characteristics or independently to each characteristic (or subgroup of characteristics).

Note that the current version of RTCP receiver reports (RRs) [4] only includes information about the mean loss rate and does not allow to transfer more complete parameter, unless using RTCP report extension mechanism.

However, in many cases, only the mean loss rate information is aggregated as most adaptation mechanisms only use this parameter to adapt [6, 7].

Another pre-requisite of the clustering algorithm is the similarity (or dissimilarity measure) to use. Two kinds of measure could be chosen. The similarity measure between two observed report x and y ($d(x, y)$) and between an observed report x and a cluster C ($d(x, C)$). The former similarity measure can stand for the simple L^p distance ($d(x, y) = \sqrt[p]{\sum_i (x_i - y_i)^p}$) or any other more sophisticated distance suitable to a particular application.

The latter similarity measure is more difficult to apprehend. The simplest way is to choose in each cluster a representative $\hat{x}_C$ and to assign the distance $d(x, \hat{x}_C)$ to the distance between the point and the cluster ($d(x, C) = d(x, \hat{x}_C)$). We can also define the distance to cluster as the distance to the nearest or the furthest point of the cluster ($d(x, C) = \min_{y \in C} d(x, y)$ or $d(x, C) = \max_{y \in C} d(x, y)$). The distance can also be a likelihood derived over a model mixture approach. The type of measure used will impact over the shape of the cluster and over the classification.

3.2 Algorithm

In this paper, we use a very simple Nearest Neighbour (NN) clustering algorithm to introduce the concepts, see pseudo-code shown in Fig. 3. Each cluster is represented by a representative point and a weight. When a new point joins a cluster, it changes slightly the representative point and updates the weight of the cluster; afterwards the point is dropped to achieve compression. The representative point is derived using a weighted average formula over the points in the cluster.

Aggregation agents regularly receive RTCP reports from receivers and/or other aggregation agents in their coverage area as described in section 2. A new report joins the cluster that has the lowest euclidian (L^2) distance to it, and updates the cluster representative by a weighted average. If this minimal distance is more than a predefined threshold, a new cluster is created. This bounds the size of the cluster. We also use a maximal number of clusters (or classes) which is fixed to 5, as it is not realistic to have more layers in such a layered multicast scheme. So, the number of clusters is not predefined (but bounded by 5) and is derived by the clustering mechanism itself. At the end of each reporting round, the resulting classification is sent back to the higher level aggregation agent (i.e., the manager) in form of a vector of cluster representative and their associated weights.

Classification derived at lower level aggregation agents are integrated to existing clustering by corresponding each class representative point as a new receiver report with a weight corresponding to the representative weight.

Receiver reports during past reporting rounds are integrated into the mechanism by using the cluster obtained during past reporting round as a base for the new clusters. Another approach consists of renewing the clustering in each reporting round and forget about past reports into the clustering. The temporal fluctuation of receivers can be integrated to the final classification at the sender. However, in each case a mechanism should be introduced to forget about far past to not bias the cluster representative by out of date

reports. This mechanism is implemented by an exponential weighting heuristic: at each reporting round the weight of a cluster is reduced by a constant factor, see Fig. 4. If the weight of a cluster falls down below a cluster suppression threshold level, the cluster is removed.

Finally, at the higher level of the aggregators hierarchy, the clustering generated by aggregating lower level aggregator reports is renewed at the beginning of each reporting round.

d_{th} =*predefined threshold*
$N_{\max}$ =*maximal number of clusters (5)*
$\mathbf{r}$ = *received receiver report*
search for the nearest cluster $d(\mathbf{r}, \hat{C}) = \min_C d(\mathbf{r}, C)$
if $(d(\mathbf{r}, \hat{C}) \geq d_{th})$
  if (Number of existing cluster $< N_{\max}$)
    Add a new cluster C_{new} and set $\hat{C} = C_{\text{new}}$
Recalculate representative of cluster $\hat{C}$,
$\hat{x}_{\hat{C}} = \frac{weight(\hat{C})\hat{x}_{\hat{C}} + \mathbf{r}}{weight(\hat{C})+1}$
Increment the weight of cluster $\hat{C}$

Figure 3: NN clustering algorithm

$w_{\min}$ =*predefined cluster suppression threshold*
γ =*memory weight*
At the beginning each reporting round
  for all clusters C
    %*Weight the current cluster by* γ
    $weight(C) = weight(C) * \gamma$
    if $weight(C) < w_{\min}$
      Remove cluster C
  Aggregate new reports
  Send aggregate reports to the higher level aggregator

Figure 4: Aggregation mechanism with memory weighting

4 Simulations

In this section, we evaluate the mechanism described in the previous section on real MBONE traces [2]. These traces are gathered by recording multicast packets received on the MBONE at 12 different sites located both in the US and in Europe. The sources of the packets are MBONE audio sessions sending at regular intervals of 40 *ms* from University of California at Berkeley. The multicast distribution tree of the used scenario is depicted in Fig. 5. Traces used as a basis for our scenario contain around 93 700 packets representing approximatively one hour of audio transmission.

We assume that some aggregation agents are positioned (not necessarily on the multicast tree). Each aggregation agent (noted by AA 1 to AA 6) manages several receivers and/or other AAs. The region assigned to each AA is shown in Fig. 5. During each reporting round that lasts 4 seconds (or equivalently each 100 packets), each receiver sends a RTCP RR to its corresponding AA. We also assume that RRs follow the RTCP recommendations and contain packet loss rate information derived over windows of 5 seconds. We use the clustering algorithm described above to achieve a distributed classification of receivers as a function of the observed empiric loss rate.

Fig. 6 shows the evolution of the empiric loss rate reported in RRs for six out of the twelve receivers. We can notice a strong temporal variability of loss rates observed for each receiver. For example, receiver *alps* has most of the time a fluctuating loss rate of around 10%; however, sometimes long bursts of losses are observed that drive loss rate to 1. This figure also shows the strong existing spatial correlation. For example, receivers *alps* and *bagpipe* have almost the same behaviour. However, this high spatial correlation does not exempt us for doing classification. Indeed, Fig. 6 shows that even if almost all receivers suffer the same loss rush on round 70, receiver *anhur* does not experiment it. So, at least for that period of time, there are clearly two distinct classes of receivers in the multicast tree.

Fig. 7 shows the cluster representative points obtained in AA 2 using the clustering algorithm and without any weighting to reduce the effect of far past reports. The cluster representative points

[2] We have used traces gathered by Maya Yajnick available on URL *http://www.cs.umass.edu/ yajnik/datasets.html*.

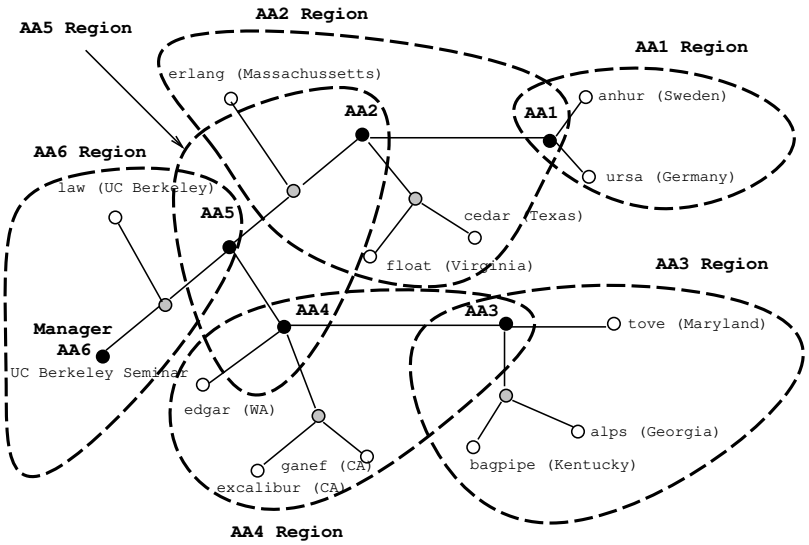

Figure 5: The simulated scenario

obtained in the same AA using a weighting factor of 0.9 and a cluster suppression threshold of 0.1 are shown in Fig. 8.

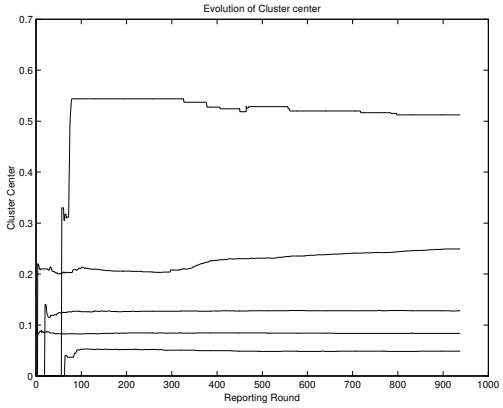

Figure 7: Evolution of cluster representative points without weighting factor

Comparison of Fig. 8 and Fig. 7 shows the main advantage of using the weighting factor: a faster tracking of non stationarity in the network. In Fig. 7, after a fast transient time, cluster representative points stabilise and do not change anymore. Fig. 8 shows that cluster representative points clearly track the increase in loss rate that is obvious in Fig. 6 for the receiver named *ursa*.

One advantage of the weighting factor can be appreciated around reporting round 80. Fig. 7 shows that a bad reception state of the multicast channel around the reporting round 80 triggers the creation of a new cluster with a mean loss rate around 55%. Note that in this case, no mechanism for reducing the future effect of this bad network state is applied, and this cluster remains even if it contains only a small number of representative points. However, Fig. 8 shows that using the weighting mechanism this cluster disappears after about 100 rounds and a new cluster is constructed representing loss rate of about 5%. A suitable choice of weighting factor makes possible a faster freeing of the cluster, but at the same time, it reduces the inherent smoothing effect of the cluster representative points calculation mechanism which can be annoying for adaptation.

Another important point to evaluate is the error made when using a distributed mechanism instead of a centralised algorithm to aggregate all receiver reports. To compare the two approaches, we have measured the error done by replacing the value of each receiver report by the correspond-

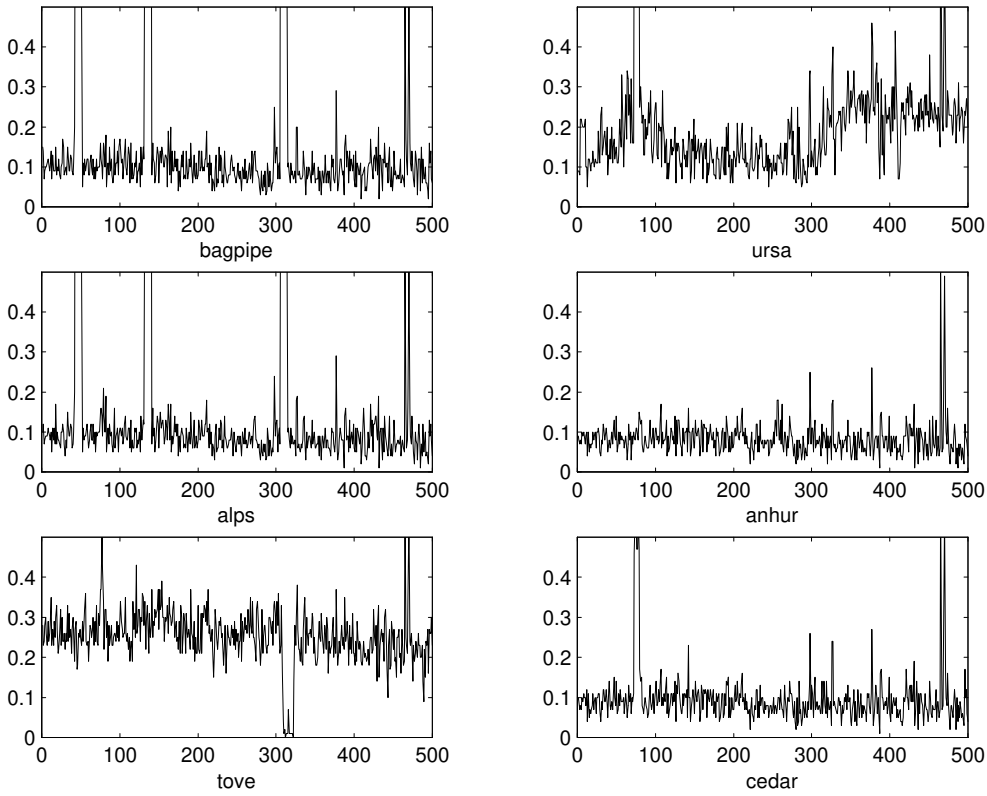

Figure 6: Empiric packet loss rates calculated over windows of $4s$ (100 packets) for six receivers

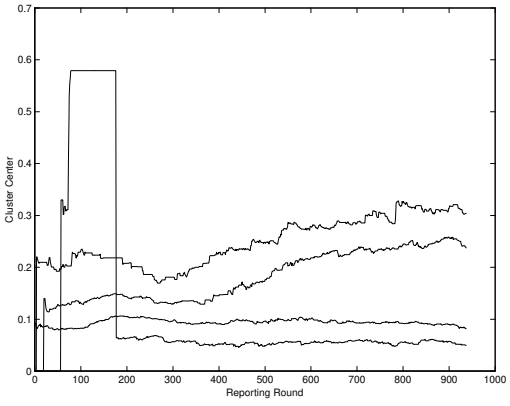

Figure 8: Evolution of cluster representative points using a weighting factor of 0.9

ing cluster centre, in two ways: with a cluster obtained at the sender using all receiver reports in each reporting round, and with a cluster obtained using the distributed algorithm described in section 3.

We have depicted the evolution of these two errors in Fig. 9. This figure shows two important features. First, the overall error for the two clusters stabilise around an asymptotic value which shows that the algorithm is stable. Second, as it can be predicted, the error is slightly lower in the centralised clustering case than in the distributed one. The first point is very important. The stability of the mechanism is a function of the weighting factor, and the figure validates our choice of the weighting factor. The second point validates our approach of distributing the clustering. In fact, escaping from the feedback implosion at receiver has a cost, measured as the highest error observed for the distributed clustering compared to the centralised one.

Figure 10 shows the variation of classes experienced by receiver *ursa* during the first 150 reporting rounds. We can note that in a relatively large

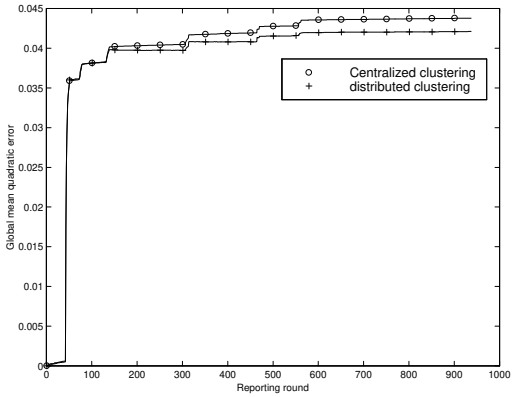

Figure 9: Comparison of clustering algorithms errors

range of rounds, the receiver oscillates between two different classes. For example, from reporting rounds 22 to 55, receiver *ursa* oscillates between classes 2 and 3, and from reporting rounds 115 up to 150, between classes 1 and 3. This observation confirms the two-states temporal behaviour (Elliot-Gilbert model) for a network channel as introduced in Section 3.

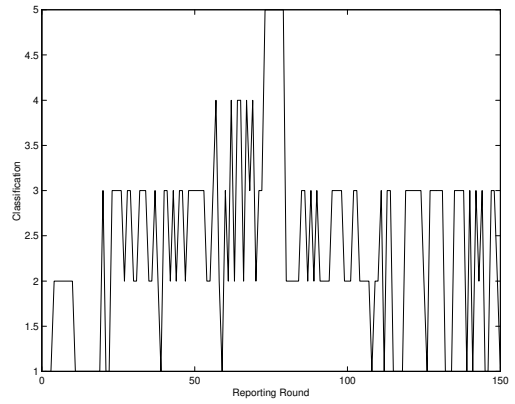

Figure 10: Class variation of receiver *ursa*

5 Conclusion

We have described a mechanism to solve the scalability problem of RTCP receiver reports in large sessions. The mechanism is based on the classification of receiver reports done by aggregation agents organised into a hierarchy of local regions. The idea is to cluster similar reception reports into homogeneous classes and report them back to the source in order to adapt both the transmission sending rate and the level of protection of each layer. We have shown in this paper, that a simple distributed aggregation mechanism can classify receivers and solve the feedback implosion problem. The algorithm provides the sender with a highly compressed and efficient representation of the receivers reports in the multicast session suitable for sender-based adaptation.

Another way to solve the scalability problem of RTCP receiver reports is to elect some *representative receivers* to send back their receiver reports to the source[24]. This mechanism is efficient to estimate the number of receivers in a large session without adding any agents in the session. However, we believe that such a mechanism cannot be used to track variations of the receivers states in order to implement an efficient congestion control scheme. For a different area, clustering mechanisms have recently been proposed to handle the problem of preference heterogeneity for large-scale virtual applications in which receivers change their preferences towards application data [25].

Future works include the use of this mechanism in a joint source and channel rate control algorithm for a layered video transmission application.

References

[1] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," in *Proc. ACM SIGCOMM'96*, Stanford, CA, August 1996.

[2] T. Turletti, S. Fosse-Parisis, and J.C. Bolot, "Experiments with a layered transmission scheme over the internet," Tech. Rep. RR-3296, INRIA Sophia-Antipolis, 1998.

[3] C. Albuquerque, B.J. Vickers, and T. Suda, "An end-to-end source-adaptive multilayered multicast (samm) algorithm," in

Proc. Packet Video Workshop, PVW'99, April 1999.

[4] H. Schultzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," Request for Comments 1889, IETF Network Working Group, Jan. 1996.

[5] Y. Wang and Q.F. Zhu, "Error control and concealment for video communication: A review," *Proceedings of the IEEE*, vol. 86, no. 5, pp. 974–997, May 1998.

[6] J.C. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive fec-based error control for internet telephony," in *Proceedings of IEEE INFOCOM*, NY, March 1999, pp. 1453–1460.

[7] K. Salamatian, "Joint source-channel coding applied to multimedia transmission over lossy packet network," in *Proceedings Video Packet 99*, New York, Columbia University, April 1999.

[8] J.C. Bolot and A. Vega-Garcia, "Control mechanisms for packet audio in the Internet," in *Proc. IEEE Infocom'96*, San Francisco, CA, April 1996, vol. 1, pp. 232–239.

[9] V. Paxson, *Measurements and Analysis of End-to-End Internet Traffic*, Ph.D. thesis, UC Berkeley, February 1997.

[10] M. Yajnik, J. Kurose, and D. Towsley, "Packet loss correlation in the mbone multicast network," in *IEEE Global Internet Conf.,London,UK*, 1996.

[11] M. Handley, "An examination of mbone performance," Tech. Rep. ISI/RR-97-450, USC/ISI Research Report, August 1997.

[12] H. Schulzrinne, J. Kurose, and D. Towsley, "Loss correlation for queues with bursty input streams," in *Proc. IEEE ICC '92*, 1992, pp. 219–224.

[13] S. Boucheron and K. Salamatian, "About priority encoding transmission," *IEEE Transaction on Information Theory*, March 2000.

[14] R. Caceres, N.G. duffield, J. Horowitz, and D. Towsley, "Multicast-based inference of network internal loss characteristics," *IEEE Transactions on Information Theory*, vol. 45, no. 7, pp. pp. 2462–2480, November 1999.

[15] S. Paul, K.K. Sabnani, J.C. Lin, and S. Bhattacharyya, "Reliable multicast transport protocol (rmtp)," *IEEE Journal On Selected Areas in Communications*, vol. 15, no. 3, pp. 407–421, April 1997.

[16] K.L. Calvert, J. Griffioen, A. Sehgal, and S. Wen, "Concast: Design and implementation of a new network service," in *Proceedings of International Conference on Network Protocols*, Toronto, Ontario, 1999.

[17] E. Amir, S. McCanne, and H. Zhang, "An application level video gateway," in *Proc. of ACM Multimedia*, San Francisco, California, November 1995.

[18] R. Droms, "Dynamic host configuration protocol," *RFC-1541*, October 1993.

[19] Y. Linde, A. Buzo, and R.M. Gray, "An algorithm for vector quantiser design," *IEEE Transactions on Communications*, vol. COM-28, pp. 84–95, January 1980.

[20] K. Salamatian, *Transmission Multimedia fiable sur Internet*, Ph.D. thesis, Université Paris Sud, Décembre 1999.

[21] N. Merhav, M. Gutman, and J. Ziv, "On the estimation of the order of a markov chain," *IEEE Trans. Info. Theory*, vol. 35, no. 5, September 1989.

[22] L. R. Rabiner, *Fundamentals of speech recognition*, Prentice Hall, 1993.

[23] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling tcp throughput: A simple model and its empirical validation," in *Proc. ACM SIGCOMM'98*, 1998.

[24] J. Bolot, T. Turletti, and I. Wakeman, "Scalable feedback control for multicast video

distribution in the internet," in *Proc. ACM SIGCOMM'94*, London, UK, September 1994.

[25] T. Wong, R. Katz, and S. McCanne, "An evaluation of preference clustering in large-scale multicast applications," in *Proc. of Networked Group Communication*, Pisa, Italy, 17-20 November 1999.