Performance of Active Multicast Congestion Control

Marifeli Sedano[1] marifeli@aut.alcala.es

Arturo Azcorra[2] azcorra@it.uc3m.es

María Calderón[2] mcalderon@it.uc3m.es

[1]*Área de Ingeniería Telemática, Universidad de Alcalá de Henares*
28871 Alcalá de Henares (Madrid), Spain

[2]*Área de Ingeniería Telemática, Universidad Carlos III de Madrid*
28911 Leganés (Madrid), Spain

April 30, 2000

Abstract

This paper aims to provide insight into the behavior of congestion control mechanisms for reliable multicast protocols. A multicast congestion control based on active networks has been proposed and simulated using ns-2 over a network topology obtained using the Tiers tool. The congestion control mechanism has been simulated under different network conditions and with different settings of its configuration parameters. The objective is to analyze its performance and the impact of the different configuration parameters on its behavior. The simulation results show that the performance of the protocol is good in terms of delay and bandwidth utilization. The compatibility of the protocol with TCP flows has not been demonstrated, but the simulations performed show that by altering the parameter settings, the proportion of total bandwidth taken up by the two types of flow, multicast and TCP, may be modified.

Keywords: Congestion Control, Active Networks, Reliable Multicast, Network Protocols, TCP

1 Introduction

Congestion control for heterogeneous traffic mixes is one of the most challenging problems of the Internet. End-to-end congestion control mechanisms that combine fairness, resilience, high network utilization, low transit delay and that support of a mix of traffic are being investigated. The problem is particularly difficult for multicast applications, although in the reliable multicast case the elastic nature of the traffic leads to fewer restrictions than in the real-time multicast application case.

Congestion control for reliable multicast protocols is an active area of research since the IETF [1] stated that any such protocol must incorporate a congestion control mechanism compatible with current Internet approaches (i.e. TCP). However, currently established proposals for reliable multicast protocols lack congestion control because initial work on the field was mainly focused on solving the scalability problem. Most recently-published proposals that incorporate congestion control are thus in a rather immature state.

Multicast applications come in all shapes and sizes. Their different functional and performance requirements usually imply differences in congestion control needs, in particular as regards internal group fairness, traffic elasticity and minimum acceptable throughput. A first classification of multicast congestion control algorithms divides them into two groups: receiver-driven or sender-driven. Current proposals for receiver-driven algorithms are mainly based on using layered communication. Receivers are responsible for controlling congestion by measuring the lost rate and disconnecting from group(s) if it goes up a certain threshold. This approach is simple and effective, but can only be applied to applications that do not require full reliability (e.g. quality adaptive) or to bulk transfer combined with FEC to achieve reliability. A further disadvantage of this approach is that it is not TCP-friendly, which can be particularly problematic when the delay to join/drop from a group is high.

Proposals for sender-driven protocols are mainly oriented towards bulk-data transfer or fully-reliable interactive applications. In the latter type of applications, the average group rate will be that of the slowest receiver admitted to the group. Congestion control is the joint responsibility of sender and receivers. Each receiver estimates the "proper" rate and communicates it to the sender, which adjusts its rate to the lowest one indicated by the receivers. An advantage of this approach is that it appears to make a TCP-friendly mechanism more tractable. An example of this line of research is the usage of a TCP rate-estimation function [2] at the receiver end. To apply this function, measurements of the loss rate and estimations of the RTT are required. Two of the still unsolved challenges in this area are to provide an appropriate response time, while using measurements averaged over time, and the design of a feedback control mechanism that avoids implosion caused by rate notifications from receivers.

The approach studied in this article is the congestion control mechanism designed for the Reliable Multicast Active Network Protocol (RMANP) [3]. This protocol is a sender-based fully-reliable multicast protocol in which the reliability and congestion control can benefit from processing performed at some of the routers. These routers use active network technology [4], therefore allowing the implementation of an active multicast congestion control mechanism. Active routers participate in congestion detection, congestion recovery and congestion notification. The advantages are obvious: within the network it is possible to know where, when, and how the transmission rate needs to be adapted to the particular congestion state of each network area.

The complexity of multicast protocols, and their associated congestion control, and the interactions between different traffic flows in the network make it very difficult to predict the system behavior in an analytical way. The cost of actual implementation and experimentation makes simulation a valuable intermediate solution. We have therefore simulated the functionality and performance of the RMANP congestion control in order to obtain data for its evaluation and design refinement. In the following sections we first give an overview of the congestion control of RMANP and then describe the simulation topology obtained with the Tiers tool, together with the simulation scenario and methodology used. This description will allow the reader to evaluate the validity both of the simulation results obtained, and of the conclusions that have been drawn from them.

2 Active Multicast Congestion Control Overview

We briefly describe the multicast congestion control used in RMANP. A more detailed description of the congestion control may be found in [5]. The active congestion control requires the participation of the source, receivers and active routers (ARs). The source performs rate control, beginning the session with the minimum acceptable rate set by the application (parameter *Rm*). It adaptively adjusts the rate, decreasing it in case of congestion or increasing it when acknowledgements are received. A relevant difference with TCP is that in our case the control is rate-based instead of window-based.

The multicast session will involve many receivers with different bandwidth and delay characteristics in the paths to them from the source so that the group must evolve at the throughput of the slowest receiver. However, as the application declares a minimum acceptable rate, receivers that cannot reach *Rm* are excluded from the session. This section describes how congestion is detected and notified to other systems, how the systems react to congestion, and how they recover when the congestion is relieved.

2.1 Congestion Detection

The ARs and the receivers perform sequence number control. If the loss of a packet is detected, the system concludes that there is congestion in the upstream subnetwork. In the example shown in Figure 1, congestion at subnetwork 3 will be first detected at AR4. As packets are immediately forwarded downstream, the packet that is used to detect congestion is marked to avoid congestion being inferred again at the next downstream AR. In addition to this mechanism, an AR detects congestion when the buffer it uses to store excess packets (*new_queue*) fills up and overflows. Lost retransmitted packets cannot be detected by intermediate sequence control, and for this reason the receiver controls the number of successive retransmission requests, signalling congestion if they exceed a certain threshold.

Other multicast congestion control proposals only perform detection at end-systems. The source detects congestion by time-out of a positive feedback mechanism (usually ACKs) and receivers detect congestion by looking for lost packets in the data stream. The advantage of having the ARs perform detection is that the system can react faster to congestion, leading to lower packet loss which, in turn, implies lower average delay and higher network utilization.

Because congestion is detected in RMANP by intermediate sequence number control, the downstream AR will detect congestion occurring at the upstream subnetwork. Other approaches, such as the one proposed by Faber for unicast communications in [6] can only detect losses that occur at ARs. This has the disadvantage that congestion in non-active subnetworks cannot be detected.

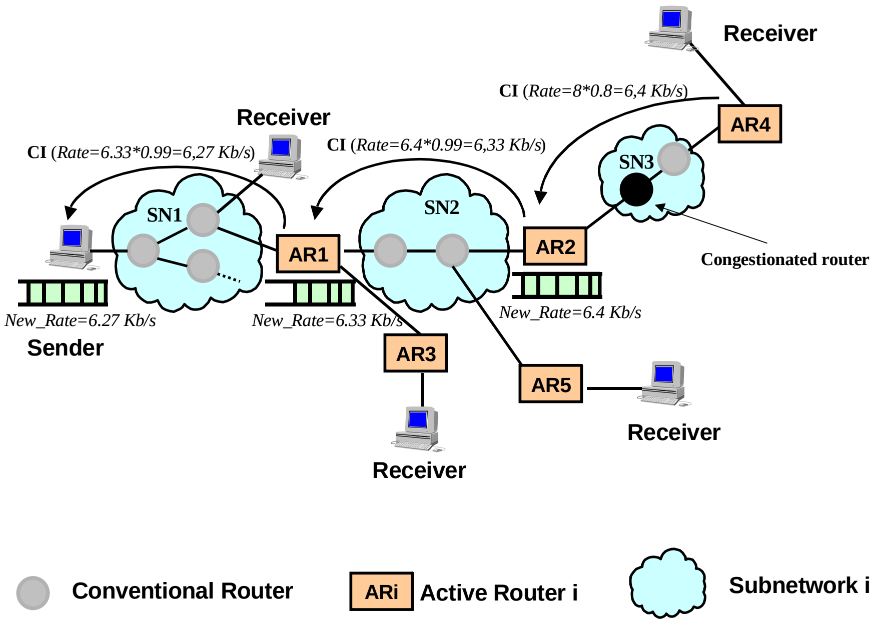

Figure1. Active multicast congestion control example

2.2 Congestion Notification

An AR or receiver that has detected congestion sends an explicit congestion indication (CI packet) towards the source. It first computes the proper rate (adjusted rate) of the session and places it in the CI sent. To compute the adjusted rate it multiplies the rate at which the packet was sent (included in control information in data packets) by an absolute multiplicative decrement (parameter *Da*).

The CI packet will be processed by all upstream ARs . In this way not only the source but also the ARs react to the congestion . To be able to handle severe congestion, involving loss of congestion indications , the source implements a fall-back congestion detection mechanism based on the use of a time-out for the reception of acknowledgements.

It is important to remark that using explicit notifications instead of implicit ones, such as the expiration of retransmission timers or the reception of duplicated ACKs, avoids the systems (ARs and source) reacting to non-existent congestion situations.

2.3 Reaction to Congestion

The AR that has detected congestion and sent the notification towards the source, does not react further to this congestion because it is upstream from this AR and the systems responsible for reducing the rate of transmission into the congested area are therefore the upstream ARs. The congestion control mechanism is hop-by-hop, as corrective actions will be taken at every AR upstream from the congestion detection point, and finally also at the source. The basic mechanism is a gradual rate control distributed between the congestion point and the source. The basic idea is that each AR controls the rate at its output interfaces setting it to the value contained in the CI with the lowest value received. Since after applying rate control, packets are received at a faster rate than that of the output , excess packets are buffered in a queue in the AR called "*new_queue*". The adjusted output rate will not be the same at all upstream ARs, but will be gradually reduced at each one. The rate will be decreased at each AR by a relative decrement factor (parameter *Dr*): each AR multiplies the rate received in the incoming CI by *Dr* in order to calculate the one placed in the CI that it will, in turn, send upstream. The objective of this mechanism is to distribute the effort of buffering the overflow of traffic that is in transit, until the moment at which the source finally receives the congestion notification and reduces its rate. With this solution the network can react rapidly to transitory congestion episodes, reducing the number of lost packets and consequently improving delay and network utilization. The use of a gradual decrement allows the temporary traffic overflow to be distributed among all the ARs involved, rather than only being buffered by the one immediately upstream from the congested point. Its use also allows each AR to gradually empty its *new_queue* by ensuring its output rate is higher than that of its input .

Figure 1 shows an example in which the source is initially transmitting at 8 Kb/s. Subnetwork 3 becomes congested and this fact is detected by AR4, that notifies it to AR2. The CI sent includes the adjusted rate calculated by AR4. Assuming that parameter *Da* is set to 0.8, the rate in the first CI would be 6.4 Kb/s. Consequently, AR2 would adjust its output rate to 6.4 Kb/s, and excess incoming packets (those whose transmission would case the output rate ceiling to be breached) would be buffered in its *new_queue*. AR2 would also send upstream a CI with a rate indication of 6.33 Kb/s (assuming that *Dr* is set to 0.99). This behavior would be repeated at every AR until the last CI reaches the source.

Other proposals [6,7] install filters that perform rate control and discard incoming packets that overflow the output rate. Our proposal is to absorb the overflow in the ARs, avoiding the retransmission of overflow packets from the source at the cost of some memory consumption at the ARs. Since in RMANP the ARs perform retransmissions of packets (local loss recovery), the rate control must also be applied to retransmitted packets, although these are sent with higher priority than new packets buffered at *new_queue*.

It is important to remark that due to the explicit inclusion of the adjusted rate in the CIs, multiple congestion indications, which may arise in several different ways, can be filtered. In the example shown in Figure 1, if congestion arose in subnetwork 2 it would be detected by AR2 and AR5, which would then send a CI to their upstream AR. AR1 would thus receive both CIs, but the one arriving second would be filtered. Hence, the AR only reacts once in the event of a single congestion incident being detected by several systems (e.g. a LAN), or being detected several times (e.g. different packets lost at different downstream ARs). It also means that the AR only reacts to the worst case in the event of several congestion incident occurring simultaneously in the distribution tree. An important consequence is that RMANP is loss-tolerant and does not react twice for losses occurring simultaneously at different places in the distribution tree. This requirement of multicast congestion control has already been addressed by other researchers [8].

2.4 Leaving the Congested State

In order to readjust and stabilize in function of the newly-available network capacity, the transmission rate needs to be increased. The source and the ARs increase the rate each time an ACK is received by adding a fraction of *Rm*, multiplied by the number of packets acknowledged, to the current rate. The increased fraction of *Rm* used by the source is controlled by the parameter *Is*. The ARs use the parameter *In* in the same way.

An AR considers that the congestion situation is over when it has emptied its *new_queue*. It will then desist from controlling the output rate and will just forward incoming packets as soon as they are received. The source will continue increasing the rate as long as the application has data to send and no CIs are received.

3. Simulation Environment

All simulations were made using ns-2 [9]. The simulator was extended to implement the RMANP and its congestion control mechanism. The network topology, shown in Figure 2, was generated using the Tiers Topology Generator. One WAN, two MANs and four LANs compose the internetwork. The receivers are connected to different LANs and the source is located in LAN1. There are seven ARs placed at different points of the internetwork.

The WAN links were set to 180 Kbps, 100 Mbps for MAN links and 10 Mbps in the case of LANs. The propagation delay of LAN links was considered null and the propagation delays of MAN and WAN links are shown in the figure 2. With these figures, the propagation delay from the source to the farthest receiver is 587 ms.

- Some of the RMANP parameters were fixed throughout all the simulation experiments performed. The settings used were:
- Source packet size was set to 1057 octets (data plus headers), parameter *Rm* to 16 Kb/s and parameter to 0.005.

Packet cache used for local loss recovery at Active Routers: 100 packets. *New_queue* cache: 10 packets. Conventional router interface queue: 3 packets with FIFO droptail. The Active Routers *In* parameter was set to 0.05. The value of the *Dr* parameter used was 0.99.

These settings are considered representative of the type of applications that would use the RMANP service. Moreover, the impact of these settings is reasonably predictable, and our goal was to simulate the impact of other parameters considered more critical. Several sets of simulations of each type were carried out and since the dispersion in the average results was minimal, the protocol is considered stable at least under the simulated conditions.

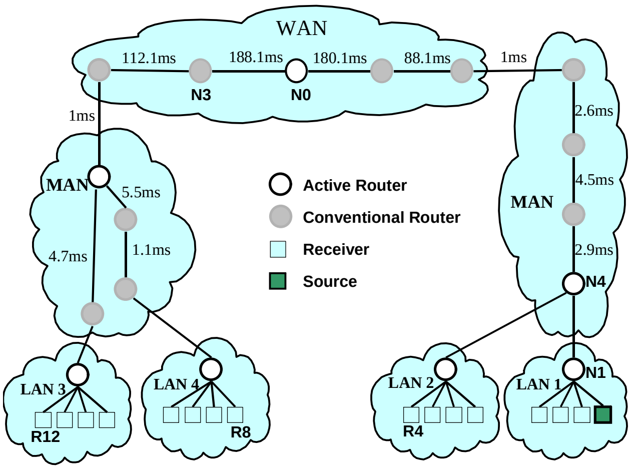

Figure 2. Simulated Topology

4 Simulation Results

To evaluate the RMANP performance, we simulated a session, between a source and 15 receivers, that coexists with a constant background traffic that suddenly increases and later decreases again. In this set of simulations, parameter *Da* was set to 0.8. Figure 3 shows the evolution of the instantaneous transmit rate of the source (in Bytes/s) in one of the simulations. The simulated time is 950 seconds. The background traffic is injected between routers N0 and N3. Its rate is 14.5 KByte/s between 0 and 350 seconds, it is then increased to 18.5 KByte/s until t = 650 seconds, when it is reduced to 14.5 KB/s. The average bandwidth obtained by RMANP was 3.787 KB/s between 350 and 650 seconds, and 6.467 KB/s over the rest of the simulation. Notice that RMANP makes use of 94.7% of available bandwidth between 350 and 650 seconds and only 80.8% the rest of the time. The reason for the lower figure in this last interval is the slow start at the beginning of the session.

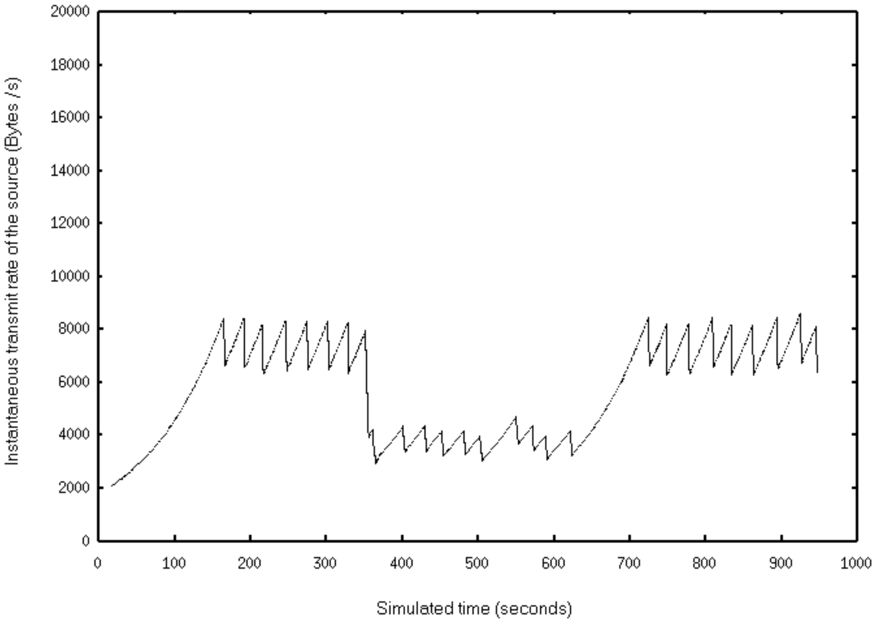

Figure 3. Transmission rate of the source

The packet transit delays experienced at different receivers are shown in Table 1. Notice that the average delay for Receiver R4 is 1% above the minimum one, in spite of no congestion being experienced on the path to it from the source. This slight increase is caused by the rate reduction which is self-imposed by RMANP at active routers N1 and N4. The average delay at receivers R8 and R12 undergoes an increase, of 8.5 % and 8.6%, with respect to the corresponding minimum values. This larger increase is caused by the bottleneck trunk line between routers N0 and N3 that affects both R8 and R12. The effect of the bottleneck may be seen in Figure 4, where the delay experienced by each packet between its emission at the source and its reception at R12 (in seconds) is shown. Comparing Figures 3 and 4 it can be seen that the peaks in packet delay correspond to the instants at which the session rate exceeds the available one. Congestion causes an increase of delay in two ways. First, rate reduction at active routers implies a queuing delay (at *new_queue*) until the upstream active router also reduces its rate. Second, congestion implies loss of packets, thus delaying delivery to receivers.

Receiver	Minimum Calculated (ms)	Simulation Results		
		Average (ms)	Lowest (ms)	Highest (ms)
R4	3.36	3.40	3.36	3.42
R8	777	843	777	3339
R12	778	845	778	3352

Tabla 1. Packet Transit Delays

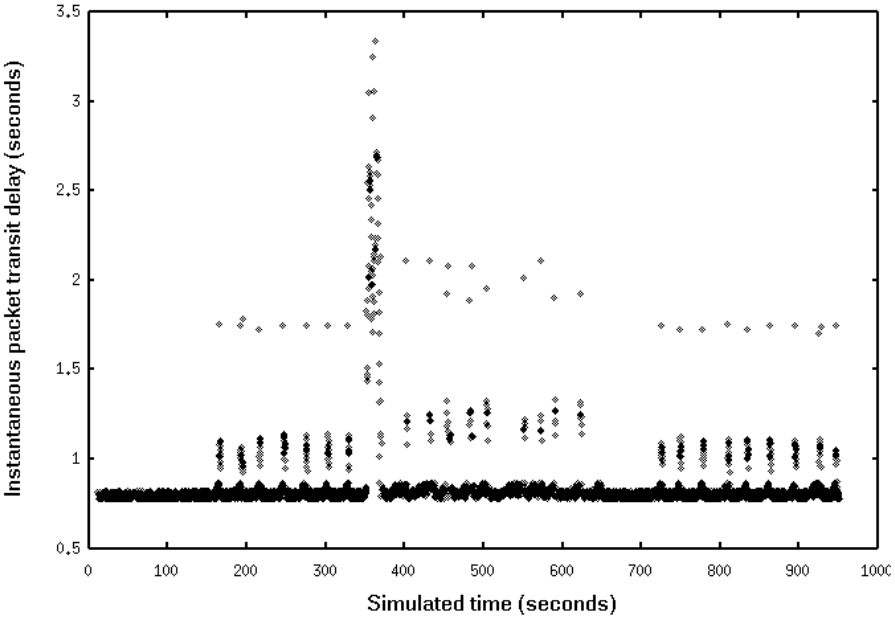

Figure 4 Packet Transit Delay in receiver R12

The average packet delay is reduced by the effect of intermediate buffering, which implies a queuing delay for a packet instead of a much higher retransmission delay. The price to pay is the buffering requirements at active routers, which have also been measured in the simulations. The average number of packets stored at the different ARs ranges from 4.97 to 9.79, with peaks ranging from 10 to 28 packets. Therefore, the absolute peak of memory required is 28 KByte for this session. Assuming that the memory requirement is linear with the number of flows, a router handling a T3 line that could handle 1002 such flows would imply a peak memory requirement of 27.4 MByte.

From the different sets of simulations carried out, the parameter *Da* can be seen to affect three relevant performance figures: packet delay, end-to-end average throughput, and active router memory requirements. A second round of simulations has been carried out specifically to determine the effects of different values of *Da*, using values of 0.9, 0.7 and 0.6, in addition to the previous value of 0.8. The performance results of the four cases are compared in Table 2. Notice how a lower *Da* leads to an improvement on end-to-end delay and buffer requirements, but also to a decrease in throughput. A lower *Da* implies a greater rate reduction when congestion is detected, causing an increase in queuing delay at *new_queue*. The queuing delay introduced for the worst-case packet is: T*(Rout/Rin - 1), where T is the time during which the input rate is higher than the output rate (reaction time of previous active router), and the quotient between the output rate (Rout) and the input rate (Rin) is precisely 1/*Da*. However, a higher *Da* causes more packet loss because the instantaneous rate exceeds the available one more frequently. Notice that both effects have an inverse influence on delay, but the loss effect is more significant.

Da	Average Delay in R12 (ms)	Average end-to-end Throughput (KB/s)	Average Queue size (packets)
0.6	823	4978	8.76
0.7	830	5276	9.14
0.8	845	5612	9.79
0.9	941	5872	12.17

Tabla 2. Performance results with different settings of *Da*

A lower *Da* decreases throughput since a higher rate cut means that the time required to increase the rate up to the available one is also higher, and therefore the average rate is lower. The impact of *Da* on buffering is caused by a combined effect of throughput and loss. Higher throughput obviously requires higher buffering because acknowledgment time is constant. Higher losses also imply higher average buffering because the period a lost packet is buffered is much higher for a packet which is lost than for one which is successfully transmitted on the first attempt.

A third set of simulations were performed using a mix of an RMANP session with several TCP flows. Different parameters (line bandwidth, *Da*, ...) of RMANP were modified in the various sets of simulations. A bandwidth share between the TCP flows and the RMANP session was observed. The quotient between the throughput achieved by RMANP and each TCP flow ranged from 1.55 to 0.41. The difficulty in evaluating these results arises when trying to define what should be the “proper” share in the different cases. TCP throughput depends on round trip time and number of flows, and varies significantly among different flows that share the same end-points. RMANP throughput depends largely on the *Rm* (minimum rate) parameter, the worst case combination of leaf round trip time and branch link bandwidth, the value of *Da*, and the increase rates *Is* and *In*. Therefore, for a given topology and collection of TCP and RMANP flows, what should be each of them’s “proper” share? Some authors suggest that in order to be TCP-friendly, the bandwidth achieved by a multicast session should be equal to the lowest of the bandwidths that a TCP session would achieve between the source and any of receivers, but this approach is still not generally accepted. A possible direction for future work is to study the definition of TCP compatibility and how to achieve it.

5 Conclusions and Future work

The congestion control mechanism of RMANP has been shown to adapt the group throughput to that available. It has also been shown to achieve a high utilization of the available bandwidth, subject to a trade-off between transit delay and utilization, depending on the value of *Da*. Average delays obtained are better than those achievable by end-to-end protocols because of the local retransmissions from active routers along the distribution tree. This makes RMANP suitable for applications sensitive to delay (e.g. interactive traffic).

Active network technology has several advantages in the implementation of multicast congestion control mechanisms. Local retransmission, intermediate sequence number control, local rate-control and buffering of overrate packets are all feasible with active networks. In particular, these advantages can be achieved with only a partial deployment of active routers. One possibility would be to place them at the endpoints of expensive transmission trunks, such as intercontinental lines, where the cost of the required processing power would be worth the improvements in delay and line utilization.

The TCP compatibility of the mechanism has not yet been established, but a promising result has been obtained: in the simulations performed the bandwidth was shared, in varying proportions, between RMANP and TCP. Further work will be aimed at providing a quantitative definition of TCP compatibility and trying to achieve it by appropriate setting of the protocol parameters. It is clear, however, that TCP compatibility must be demonstrated under any topology and combination of RMANP and TCP flows. Due to the complexity involved, any such demonstration will inevitably require a very considerable simulation effort, even under the assumption that the proper parameter combination is known in advance.

Among the weak aspects of the congestion control mechanism proposed we highlight its rate-increase mechanism. Simulation results have shown that a better adaptation to changing available bandwidth could be obtained by using a low *Da* combined with an increase mechanism that takes into account the history of the rate in use. This could be implemented at the source without any need to increase the processing complexity at active routers.

Acknowledgments

This work has been supported by CICYT (the Government Commission of Science and Technology) under projects TEL99-0988-C02-01 and TEL99-0988-C02-02.

References

[1] A. Mankin, A. Romanow, S. Bradner and V. Paxson. IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols. RFC 2357, June 1998.

[2] J. Padhye, V. Firoiu, D. Towsley, J. Kurose. Modeling TCP Throghput: A Simple Model and its Empirical Validation. *Proc. ACM Sigcomm*, 1998, Vancouver, Canada.

[3] M. Calderón, M. Sedano, A Azcorra and C. Alonso. Active Network Support for Multicast Applications. *IEEE Network Magazine*, pp. 46-52, May/June 1998.

[4] D.L. Tennenhouse, J.M. Smith, W.D. Sincoskie, D.J. Wetherall and G.J. Minden. A Survey of Active Network Research. *IEEE Comunications Magazine, pp.* 80-86, January 1997.

[5] A. Azcorra, M. Calderón, M. Sedano, José I. Moreno. Multicast Congestion Control for Active Network Services. European Transactions on Telecommunications, Vol. 10, No. 3, May/June 1999

[6] T. Faber. ACC: Using Active Networking to Enhance Feedback Congestion Control Mechanisms. *IEEE Network Magazine*, pp. 61-65, May/June 1998.

[7] M. Luby and L. Vicisano. Heterogeneous Multicast Congestion Control based on Router Packet Filtering. Document sent to the IRTF RM mailing list, June 1999.

[8] C T. Montgomery. A Loss Tolerant Rate Controller for Reliable Multicast. Technical Report: NASA-IVV-97-011, August 1997.

[9] S. McCanne, S. Floyd, and K. Fall, UCB/LBL Network Simulator, 1996, available from http://www-mash.cs.berkeley.edu/ns/ns.html