AN ANALYSIS OF A ROUTER-BASED LOSS DETECTION SERVICE FOR ACTIVE RELIABLE MULTICAST PROTOCOLS

Moufida Maimour and Cong-Duc Pham

RESAM, University Lyon 1-LIP-INRIA RESO
46 alle d'Italie
69364 Lyon Cedex 07 - France
email:{mmaimour,cpham}@ens-lyon.fr

ABSTRACT

Group communications (multicast) are foreseen to be one of the most critical yet challenging technologies to meet the exponentially growing demands for data distribution in a large variety of applications of the Internet (grid computing, web applications, distributed simulations...). When reliability is required, there is no straightforward solutions and meeting the objectives of reliable multicast is not an easy task. Active networks open a new perspective in providing more efficient solutions for the problem of reliability. In this context, routers are able to perform customized computations on the packets flowing through them. In this paper, we propose a new service consisting in an early loss detection service to be deployed into routers. We also show how the loss detection service can improve the performances of the DyRAM active reliable multicast protocol in term of the recovery delays making DyRAM suitable for applications requiring low latencies.

1. INTRODUCTION

Group communications (multicast) are foreseen to be one of the most critical yet challenging technologies to meet the exponentially growing demands for data distribution in a large variety of applications of the Internet (grid computing, web applications, distributed simulations...). At the network level IP multicast provides an efficient one-to-many IP packets delivery but without any reliability guarantees. However data dissemination applications such as distributed computing or interactive simulations usually require a reliable transfer and meeting the objectives of reliable multicast is not an easy task.

The problem of reliability in multicast protocols has been quite widely covered during the last 10 years. Early reliable multicast protocols use an end-to-end solution to perform the loss recovery. Most of them fall into one of the following classes: sender-initiated, receiver-initiated and receiver-initiated with local recovery protocols. In sender-initiated protocols, the sender is responsible for both the loss detection and the recovery. These protocols usually do not scale well to a large number of receivers due to the ACK implosion problem. Receiver-initiated protocols move the loss detection responsibility to the receivers. They use NACKs instead of ACKs. However they still suffer from the NACK implosion problem when a large number of receivers have subscribed to the multicast session. In receiver-initiated protocols with local recovery, the retransmission of a lost packet can be performed by some other nodes in the multicast tree [2, 10, 14, 9, 11, 3].

Recently, the use of active network concepts [12] where routers themselves could contribute to enhance the network services by customized functionalities has been proposed in the multicast research community. Contributing mainly on feedback implosion problems, retransmission scoping and cache of data, active reliable multicast offers a general and flexible framework for customized functionalities in network protocols (although many problems regarding deployment and security remains). ARM (Active Reliable Multicast) [6] and AER (Active Error Recovery) [4] are two protocols that were recently proposed in the research community and that use active services within routers. DyRAM (Dynamic Replier Active Reliable Multicast) [7] is another active protocol that can dynamically elect a receiver as a replier on a per-packet basis.

In this paper we investigate an other possible functionality which consists in an early loss detection service by the routers themselves. In this case, routers are capable to detect packet losses and consequently generate corresponding NACKs to be sent to the source. Although simple, moving the loss detection into routers arises many questions such as "where to place such detection-capable routers?" and "what is the overhead of such an additional service?". The results presented in the paper show that one must be careful when doing so in order to get any real benefit. This paper presents an analytical evaluation of this new functionality. The study is based on the processing overhead at both the end hosts and the active routers to derive the overall delay required by any receiver to correctly receive a data packet. The rest of the paper is organized as follows. Section 2 presents the delay analysis of an early loss detection service and Section 3 presents the numerical results. In Section 4, we show how such a service can be added to the DyRAM protocol and, using simulations, how it can reduce the recovery delay. Section 5 concludes.

2. DELAY ANALYSIS

A commonly used model for evaluating multicast protocols is to have a multicast tree rooted at the source with receivers as leaves. Intermediate nodes are the routers. In the context of active networking, we will consider that a subset or all of the routers can be active. Consequently these routers are able to perform customized processing (services) on the packets (data packets and NACKs) flowing through them. The first supported service is the NACK suppression service which consists in ignoring subsequent NACKs for the same data packet during a given amount of time. For our analysis, we will assume that this "*duplicate discard*" period is well chosen. Therefore only one NACK is forwarded toward the

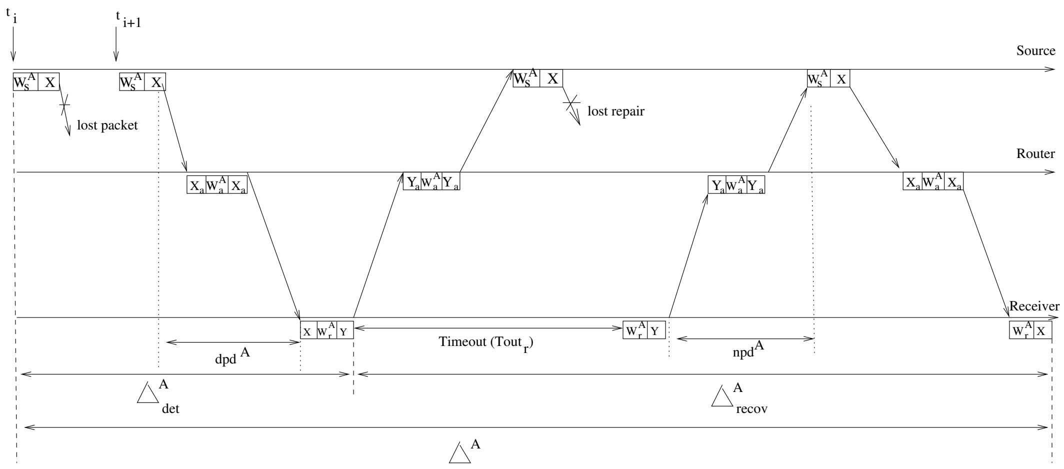


Figure 2: Overall delay diagram for protocol A.

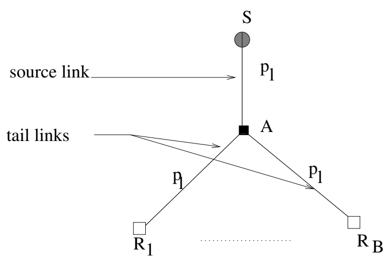


Figure 1: Simple network model.

source for each loss. The second active service consists in the subcast functionality where repair packets are sent only to the affected receivers. The subcast is performed thanks to the soft state information maintained at the routers so that a repair is only retransmitted to the affected receivers. Now, in order to evaluate the impact of adding a loss detection service, we consider two protocols noted A and D. Both of them benefit from the NACK suppression and the subcast services but D, in addition, benefits from a loss detection service at the active routers which are able to detect a gap in a data packet sequence. In this case, the router would immediately generate a NACK packet toward the source and would initialize a timer. On expiration of this timer, without having received the data packet, the router will send another NACK. All NACK packets received for this data packet from the downstream links are ignored until the expiration of the corresponding timer.

For the delay analysis, we consider a simple network model with a two level multicast tree. One source multicasts data packets to one group composed of B receivers $R_1 \ldots R_B$ connected to the source via an active router A. We will call the *source link* the set of point-to-point links that connects the source to the active router. Similarly, a *tail link* is composed of the point-to-point links connecting the active router to each of the receivers (see figure 1). We consider that the source link and the tail links have a loss probability of p_l. Therefore the end-to-end loss probability perceived by a receiver is $p = 1 - (1 - p_l)^2$. The losses are assumed to be temporally independent and those at the tail links are assumed to be mutually independent. We will also assume that the NACKs will never be lost.

The computational framework adopted in our analysis is similar to the one provided in [13]. Each node is modeled by a M/G/1 queue (Poisson arrivals and arbitrary service time distribution). The delay analysis is largely based on the mean waiting time of the system. In order to estimate this mean waiting time for each node under the evaluated protocols, we proceed as follows. First the different flow rates $\lambda_1, \lambda_2, \ldots, \lambda_n$ of the node with their respective service requirement $X_1, X_2, \ldots, X_n$ are determined. Provided that each of these random variables have means and second moments, then the load ρ at this node can be computed using :

$$\rho = \sum_{i=1}^{n} \lambda_i E[X_i]$$

Finally, the mean waiting time $E[W]$ can be computed using the Pollaczek-Khinchine mean value formula as follows:

$$E[W] = \frac{\sum_i \lambda_i E[X_i^2]}{2(1-\rho)}$$

In what follows, let M (respectively M_r) be the number of transmissions of a data packet from the source until all the receivers (one receiver) have (has) correctly received the data packet. M_a is the number of transmissions of a data packet from the source until the active router has correctly received the data packet. We have $P[M \leq m] = (1 - p^m)^B$ thus $E[M] = cd \sum_{m=0}^{\infty} 1 - (1 - p^m)^B$. For M_r we have $P[M_r \leq m] = 1 - p^m$ thus $E[M_r] = 1/(1 - p)$. Similarly $E[M_a] = 1/(1 - p_l)$. We will note X (respectively X_a) and Y (respectively Y_a) the service time required for processing a data packet and a NACK at the source and the receivers (the active router). We assume that the source is multicasting at a rate of λ packets per unit of time.

2.1. Waiting times

In protocol A, the data packet arrival rate at the source is $\lambda_d^{s,A} = \lambda$ with a mean requirement service of $E[X]$. The NACK packet arrival rate at the source is $\lambda_n^{s,A} = \lambda \frac{E[M]-1}{B}$ with a mean requirement service of $E[X] + E[Y]$. Therefore the load at the source is

$$\rho_s^A = \lambda E[X] + \lambda \frac{E[M]-1}{B}(E[X] + E[Y])$$

giving

$$\rho_s^A = \lambda E[X](1+\frac{E[M]-1}{B}) + \lambda E[Y]\frac{E[M]-1}{B} \quad (1)$$

The mean waiting time at the source is :

$$E[W_s^A] = \frac{\lambda E[X^2] + \lambda \frac{E[M]-1}{B} E[(X+Y)^2]}{2(1-\rho_s^A)}$$

with $E[(X+Y)^2] = E[X^2] + 2E[X]E[Y] + E[Y^2]$, we have:

$$E[W_s^A] = \lambda \frac{E[X^2](1+\frac{E[M]-1}{B}) + E[Y^2] + 2E[X]E[Y]}{2(1-\rho_s^A)} \quad (2)$$

The active router receives on average $E[M]$ times a packet with a probability of $1-p_l$. Therefore the data packet arrival rate at the router is $\lambda_d^{a,A} = \lambda E[M](1-p_l)$ with a mean requirement service of $2E[X_a]$ (the router receives the packet and then forwards it). On average the active router receives $(M_r-1)B$ NACK packets from the B receivers downstream. Therefore the NACK packet arrival rate is $\lambda_n^{a,A} = \lambda(E[M_r]-1)B$ with a mean requirement service of $E[Y_a] + E[Y_a]/B$ (the router receives all the NACKs generated for a lost data packet but forwards only one NACK upstream because of the NACKs aggregation functionality). The load at the active router for protocol A is:

$$\rho_a^A = 2\lambda\, E[M](1-p_l)E[X_a] + \lambda(E[M_r]-1)(1+B)E[Y_a] \quad (3)$$

The mean waiting time at the active router is :

$$E[W_a^A] = \frac{4\lambda_d^{a,A}\, E[X_a^2] + \lambda_n^{a,A}\, E[Y_a^2](B+1)^2/B^2}{2(1-\rho_a^A)} \quad (4)$$

At a randomly chosen receiver the data packet arrival rate is $\lambda_d^{r,A} = \lambda$ with a required service of $E[X]$. This is due to the fact that the receiver receives only once each data packet thanks to the subcast functionality. The NACK packet arrival rate is $\lambda_n^{r,A} = \lambda(E[M_r]-1)$ with a mean requirement service of $E[Y]$. The load at the receiver side for protocol A is:

$$\rho_r^A = \lambda E[X] + \lambda(E[M_r]-1)E[Y] \quad (5)$$

The mean waiting time for protocol A is given by :

$$E[W_r^A] = \lambda \frac{E[X^2] + (E[M_r]-1)E[Y^2]}{2(1-\rho_r^A)} \quad (6)$$

The waiting times in protocol D for both the sender and the receivers are identical to those of protocol A. At the active router the data packet arrival rate is $\lambda_d^{a,D} = \lambda E[M](1-p_l)$ with a mean requirement service of $2E[X_a]$. On average the active router receives $(M_r-1)B$ NACK packets from the receivers. $M_a - 1$ NACK packets are ignored by the router because it has recently sent a similar NACK. Therefore only $(M_r-1)B - M_a + 1$ will be forwarded toward the source. Hence, the arrival rate of received but ignored NACKs is $\lambda_g^{a,D} = \lambda(E[M_a]-1)$ with a mean requirement service of $E[Y_a]$. For the received and forwarded NACKs, we have $\lambda_n^{a,D} = \lambda((E[M_r]-1)B - E[M_a]+1)$ with a mean requirement service of $(E[Y_a] + E[Y_a]/B)$.

The load at the active router for protocol D is:

$$\begin{aligned}\rho_a^D = \; & 2\lambda E[M](1-p_l)E[X_a] + \\ & \lambda(E[M_a]-1+(B+1)(E[M_r]-1) - \\ & (M_a-1)/B)E[Y_a] \end{aligned} \quad (7)$$

The mean waiting time at the active router for protocol D is:

$$E[W_a^D] = \frac{4\lambda_d^{a,D} E[X_a^2] + (\lambda_g^{a,D} + \lambda_n^{a,D}(B+1)^2/B^2)E[Y_a^2]}{2(1-\rho_a^D)} \quad (8)$$

2.2. Overall delay analysis

For protocol $\omega \in \{A, D\}$, the overall delay Δ^ω for a data packet to be received by a randomly chosen receiver includes the time required to detect the loss Δ_{det}^ω and the time required to perform the recovery Δ_{recov}^ω, therefore we have:

$$E[\Delta^\omega] = E[\Delta_{det}^\omega] + E[\Delta_{recov}^\omega]$$

To compute these times, we need to introduce two random variables L_r and L_a. Assuming that the lost packet has a sequence number of i, L_r is the number of subsequent packets with a sequence number greater than i which are lost by both the randomly chosen receiver and all the other receivers that have also lost the ith data packet. The expression for the mean of L_r is given in [13]:

$$E[L_r] = \sum_{k=0}^{B-1} C_{B-1}^k\, p^k (1-p)^{(B-k-1)} \frac{p^{k+1}}{1-p^{k+1}}$$

Similarly L_a is the number of subsequent packets with a sequence number greater than i that are lost by the active router. Since we have only one active router in our model then $E[L_a] = p_l/(1-p_l)$. For the overall delay analysis, we note T_{sa} (T_{ar}) as the propagation delay from the source (a receiver) to the active router. The propagation delay from the source to a receiver is noted $T_{sr} = T_{sa} + T_{ar}$. The required delay to receive a data packet from the source by a receiver in protocol ω is $dpd^\omega = T_{sr} + 2X_a + W_a^\omega$. The required delay to receive a NACK packet from any receiver by the source in protocol ω is noted $npd^\omega = T_{sr} + 2Y_a + W_a^\omega$.

Figures 2 and 3 show the overall delay diagrams for protocol A and D. We have accordingly $\lambda = t_{i+1} - t_i\ \forall i$. For protocol A, the loss detection time is given by :

$$\begin{aligned} E[\Delta_{det}^A] = \; & E[L_r]/\lambda + E[W_s^A] + E[X] + \\ & E[dpd^A] + E[Y] + E[W_r^A] + E[X] \end{aligned} \quad (9)$$

To estimate a mean for Δ_{recov}^A, we must take into consideration that a repair may be lost. Consequently Δ_{recov}^A includes the delay incurred by a given number $(j-1)$ of timeout expirations (if the data packet needs to be transmitted j times), the time required to receive the last NACK by the source and the time required to receive the repair by the receiver. From figure 2, we can see that in addition to this time, there is the delay required for the NACK to be received by the source (npd^A), the processing time at the source ($W_s^A + X$), the delay required for the repair to be received by the receiver (dpd^A) and the processing time at the receiver ($W_r^A + X$). Therefore we can write:

$$\begin{aligned} E[\Delta_{recov}^A] = \; & E[npd^A] + (E[W_s^A] + E[X]) + E[dpd^A] + \\ & (E[W_r^A] + E[X]) + (Tout_r + E[W_r^A] + E[Y]) \\ & \textstyle\sum_{j=1}^{\infty} (j-1)p^{j-1}(1-p) \end{aligned}$$

where $p^{j-1}(1-p)$ is the probability that there are $(j-1)$ retransmissions of the data packet until its correct reception by the randomly chosen receiver. $Tout_r$ is the value of the timeout set at the receivers. Finally, we have :

$$\begin{aligned} E[\Delta_{recov}^A] = \; & E[npd^A] + E[dpd^A] + E[W_s^A] + E[W_r^A] + \\ & 2\, E[X] + (Tout_r + E[W_r^A] + E[Y])\, p/(1-p) \end{aligned} \quad (10)$$

For protocol D, a loss may be detected by the router with a probability p_l (the probability that the data packet is lost at the source) or by any of the affected receivers otherwise. Therefore

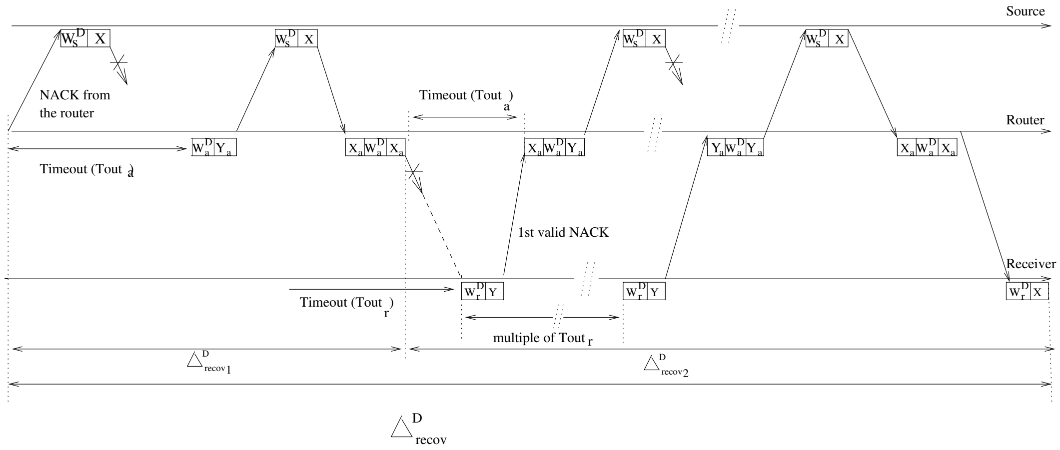


Figure 3: Overall delay diagram for protocol D.

Δ^D_{det} is a function of $\Delta^D_{det,receiver}$ and $\Delta^D_{det,router}$ which are respectively the time required to detect a loss by a receiver and by an active router. Therefore we have :

$$E[\Delta^D_{det}] = (1-p_l)E[\Delta^D_{det,receiver}] + p_l E[\Delta^D_{det,router}]$$

Using the same method to obtain the loss detection delay in protocol A, we find for protocol D that:

$$E[\Delta^D_{det,receiver}] = E[L_r]/\lambda + E[W^D_s] + E[dpd^D] + E[Y] + E[W^D_r] + 2E[X] \quad (11)$$

and finally,

$$E[\Delta^D_{det,router}] = E[L_a]/\lambda + E[W^D_s] + E[X] + T_{sa} + E[X_a] + E[Y_a] + E[W^D_a]$$

which gives,

$$E[\Delta^D_{det,router}] = p_l(1-p_l)/\lambda + E[W^D_s] + E[X] + T_{sa} + E[X_a] + E[Y_a] + E[W^D_a] \quad (12)$$

Similarly, the recovery delay can be given by:

$$E[\Delta^D_{recov}] = (1-p_l)E[\Delta^D_{recov,receiver}] + p_l E[\Delta^D_{recov,router}]$$

where $\Delta^D_{recov,receiver}$ is the required time to recover if the loss is detected by the router. Similarly $\Delta^D_{recov,router}$ is the required time to recover if the loss is detected by a receiver. $\Delta^D_{recov,receiver}$ is similar to Δ^A_{recov} so:

$$E[\Delta^D_{recov,receiver}] = E[npd^D] + E[dpd^D] + E[W^D_s] + E[W^D_r] + 2\,E[X] + (Tout_r + E[W^D_r] + E[Y])\,p/(1-p)$$

Since a router sets a timeout and retransmits the NACK if the required data packet has not been received yet, $\Delta^D_{recov,router}$ can be expressed by:

$$E[\Delta^D_{recov,router}] = E[\Delta^D_{recov1}] + E[\Delta^D_{recov2}]$$

where $E[\Delta^D_{recov1}]$ is the required delay until the active router has received the data packet so it will never generate a NACK for it. $E[\Delta^D_{recov2}]$ is the required delay to receive the data packet by the randomly chosen receiver after the active router has already received it. Referring to figure 3 and following the same method used for Δ^A_{recov}, we find :

$$E[\Delta^D_{recov1}] = 2T_{sa} + E[W^D_s] + E[X] + E[W^D_a] + 2E[X_a] + (Tout_a + E[W^D_a] + E[Y_a])\,p_l/(1-p_l) \quad (13)$$

Under the assumption that the first valid NACK is received just at the expiration of the timeout at the router, and using the same method applied for Δ^A_{recov}, we can derive:

$$E[\Delta^D_{recov2}] = E[npd^D] + E[dpd^D] + E[W^D_s] + E[W^D_r] + 2E[X] + Tout_a - T_{ar} + (Tout_r + E[W^D_r] + E[Y])\,p/(1-p) \quad (14)$$

3. NUMERICAL RESULTS

For the numerical study, we set the values of the different parameters to those measured in [5] and normalized to the time needed to transmit a data packet ($E[X] = 1$) which is of about 500 μsecs. We can take $E[Y] = 0.2$ accordingly which corresponds to 100 μsecs. The processing overhead at the routers is considered to be the double of the time required to process a data packet at the end hosts, $E[X_a] = E[Y_a] = 2E[X] = 2$. In order to consider the detection-capable active router's position, we introduce $\alpha \in]0,1[$ which is the ratio of T_{ar} to T_{sr}. These two parameters are expressed as a function of the number of links crossed by a packet. The link propagation time is set to 1 time unit. Therefore we can use T_{ar} (respectively T_{sr}) to represent the number of links between the router (respectively the source) and any receiver. We will also consider two particular values of α: $\alpha_c = (T_{sr}-1)/T_{sr}$ where the router is one link far from the source ($T_{sr}-1$ links between the router and a receiver) and $\alpha_f = 1/T_{sr}$ where the router is one link far from a receiver.

We begin by examining the performances of D as a function of the router position. Figure 4 plots the ratio of the achieved delay by D when we change the router position (parameter α) to the case where $\alpha = \alpha_f$. We can see that the position of the router with respect to the source is deciding. The gain is increased when the

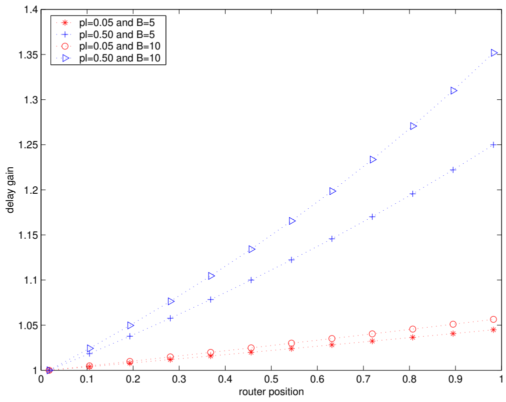
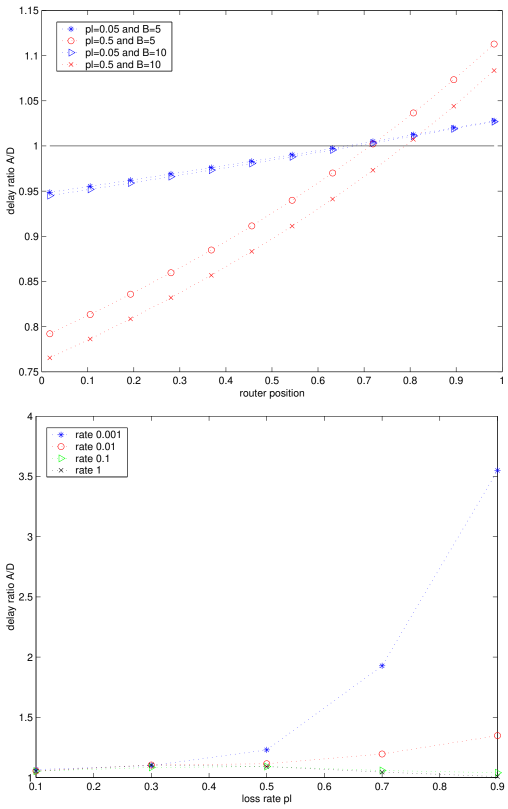


Figure 4: The delay gain in D as a function of the router position.

router is the closest to the source. The gain in delay observed is more important for high loss rates. In fact, for a loss rate of 50% the delay is improved by 25% up to 35% if we move the router near the source instead of putting it near the receivers.

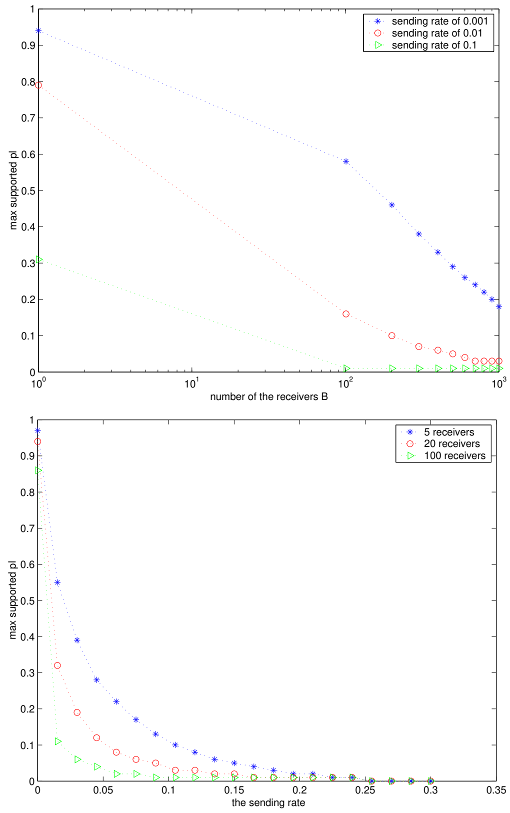


(a) (b)

Figure 5: The delay ratio A/D as a function of (a) the router position (b) loss rate p_l.

In order to compare D to A, figure 5(a) plots the delay ratio of A to D as a function of the router position. We can see that D performs better than A only when the router is sufficiently close to the source. Otherwise, the overhead introduced by generating NACKs in D becomes unjustified. This is due to the longer distance that must be crossed by the NACK packet before reaching the source.

Henceforth, we will consider only the case where the active router is sufficiently close to the source ($\alpha = \alpha_c$). Figure 5(b) plots the gain achieved by D with respect to A as a function of p_l for different sending rates. B is set to 5. The benefit of D over A increases as the loss rate increases. For instance, for a sending rate of 0.001, protocol D reduced the delay compared to protocol A by a factor of 3.5 for a loss rate of 90%. In the other cases, even if the gain is not significant D still performs better than A.



(a) (b)

Figure 6: Maximum supported p_l in D as a function of (a) B, the number of the receivers and (b) λ, the sending rate.

An other important aspect to examine is the maximum loss rate supported by the entire system before one of the nodes is overloaded. Figure 6(a) shows the maximum supported loss rate (p_l) as a function of the number of receivers. We can see that for a sending rate of $\lambda = 0.001$ with one receiver, the system will be able to support more than 90% of losses. However when the number of receivers increases, the supported loss rate decreases. To see the impact of the sending rate, figure 6(b) plots the maximum supported loss rate as a function of the sending rate for several group sizes. As expected, we can see that the maximum supported loss rate decreases as the sending rate increases.

Till now we have considered only the case where the processing overhead of the active router is twice the processing time required at the end hosts. Figure 7 plots the maximum supported p_l as a function of the processing power. Figure 7(a) shows that the maximum supported loss rate increases as the processing power increases from 0.1 (corresponding to a reduction by a factor of 10) to 1 (the router has the same processing power as the end hosts). Figure 7(b) shows that one does not need to increase the processing power infinitely. In fact, for 5 receivers the maximum supported loss rate does not increase if the processing power is increased beyond 9 times. Even if the number of receivers is multiplied by 20, increasing the processing power beyond 16 will not increase the supported loss rate. This is due to the fact that the routers are not the bottleneck.

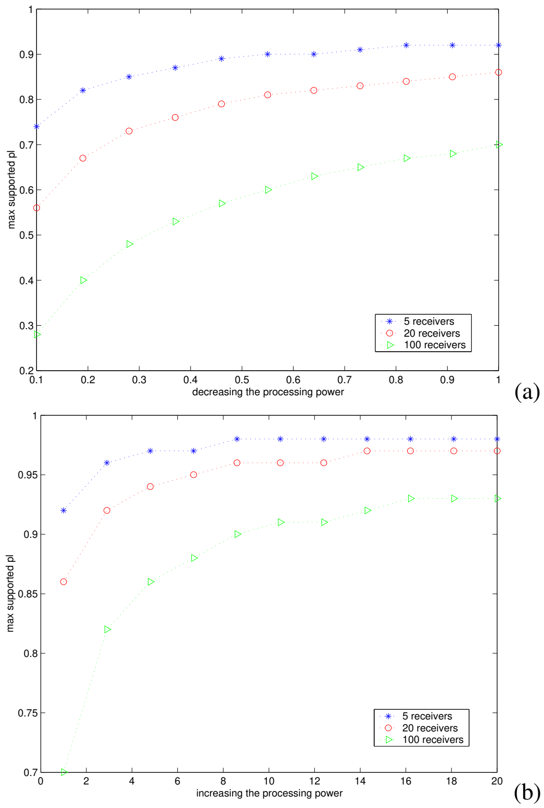


Figure 7: Maximum p_l supported by D as a function of the routers processing power.

Figure 8 plots the minimal processing power required at the routers so that they are never the bottleneck. This minimum processing power increases with the loss rate and the number of receivers. For instance, a loss rate of 50% and 5 receivers require the router to be approximately 20 times faster than the end hosts.

To accurately examine the behavior of the different nodes and to know which node is overloaded before the others, Figure 9 shows the load at the different nodes in A and D as a function of the loss rate. The processing overhead at the routers is considered twice the required processing time at the end hosts. We can see that the load at the source and the receivers is the same in both A and D. The load at the router in D is only slightly greater than A so the loss detection service does not introduce a significant processing overhead at the routers. The routers are the most overloaded nodes because of the processing overhead introduced by the active services. The source is more loaded than the receivers since it is responsible of the retransmissions. We can see from figure 9(b) where we increased the sending rate that the load increases in all nodes of the system.

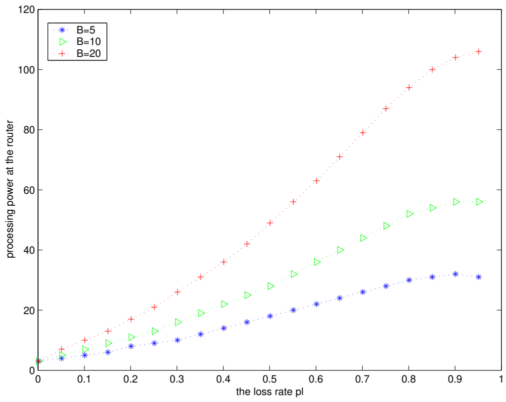


Figure 8: The required processing power at the router.

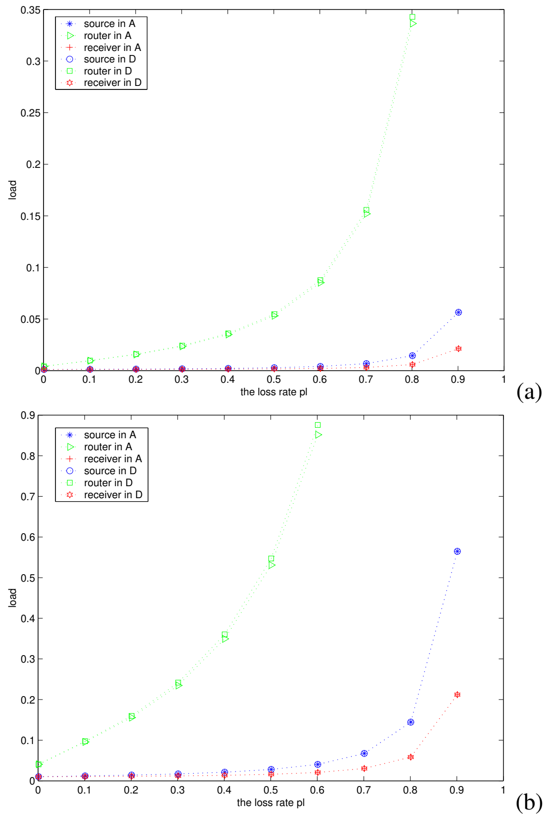


Figure 9: Load at the different nodes with 5 receivers (a) $\lambda = 0.001$ (b) $\lambda = 0.01$.

An other deciding factor is the number of receivers associated to the active router. Figure 10 shows that D has lower end-to-end delay than A. For low loss rates (i.e. 5% to 10%) protocol D introduces a gain of only 5% that does not vary even if we increase the number of receivers. The benefit of D over A can be better seen when the loss rate is increased. In fact, for a loss rate of 25%, the gain of D over A increases from 7% for 1 receiver to 27% for 50 receivers. Protocol D shows better performances than A especially for high loss rates and a large number of receivers.

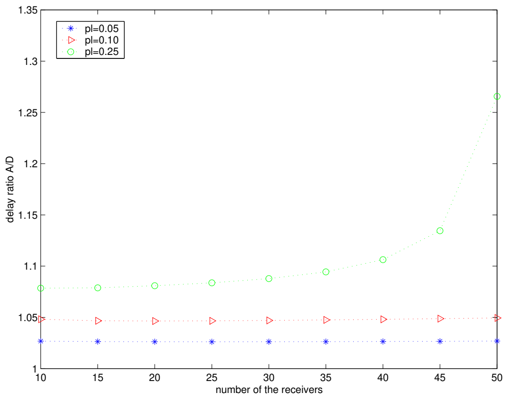


Figure 10: The gain for $\lambda = 0.01$ with $B = 5$.

4. ADDING THE LOSS DETECTION SERVICE TO A RELIABLE MULTICAST PROTOCOL

4.1. DyRAM: an Overview

DyRAM is a reliable multicast protocol suite with a recovery strategy based on a tree structure constructed on a per-packet basis with the assistance of routers [7]. The protocol uses a NACK-based scheme with receiver-based local recoveries where receivers are responsible for both the loss detection and the retransmission of repair packets. Routers play an active role in DyRAM which consists in the following active services:

1. the NACK suppression of duplicate NACKs in order to limit the NACK implosion problem.
2. the subcast of the repair packets only to the relevant set of receivers that have experienced a loss. This helps to limit the scope of the repairs to the affected subtree.
3. the replier election which consists in choosing a link as a replier one to perform local recoveries from the receivers.

Therefore we can summarize the motivations behind DyRAM with the following design goals: (i) to minimize active routers load in order to make them supporting more sessions (mainly in unloading them from the function of data caching), (ii) to perform a more efficient replier link election procedure with active services, (iii) to dynamically distribute the recovery task among the downstream receivers to not overload one replier and (iv) to incur as low as possible execution/memory overheads.

4.2. Simulation Results

A set of simulations are performed to show how a loss detection service could decrease the delay of recovery of the DyRAM framework. To do so, four protocols noted A, D, $DyRAM$ and $DyRAM^+$ are simulated on a network model derived from the proposed architecture (fig. 11). In addition to the source active router A_S, we consider N active routers A_i, $i \in \{1 \ldots N\}$. Each active router A_i is responsible of B receivers forming a local group. All of the four protocols benefit from the NACK suppression and the subcast services. Whereas A only benefits from these two services, D benefits from the loss detection service at the source router. $DyRAM$ is similar to DyRAM where local recoveries from the receivers are possible. $DyRAM^+$ behaves like $DyRAM$ except that additionally the source router performs the loss detection service. In our loss model, we consider both the spatial and the temporal correlation of data packet losses. The spatial correlation is introduced by considering a per-link loss rate and the core network is considered reliable. The temporal correlation of losses is achieved by using the same model as in [7]. We also consider that there is l_b backbone links between the source router A_S and every active router A_i. The simulations are implemented using the PARSEC language developed at UCLA [1].

For all the simulations, we set $l_b = 55$. A NACK and a data packet are considered to be of 32 and 1024 bytes respectively. All simulation model values are normalized to the NACK transmission time. For the processing overheads at the routers, we assume that both NACKs and data packets are processed in 32 time units. These values are derived from measures in [6].

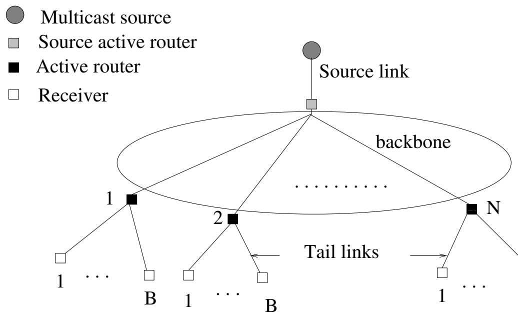


Figure 11: Network model.

Figure 12 plots the recovery delay (normalized to the RTT) for the four protocols as a function of the number of the receivers for different loss rates. First of all, it is noticeable that protocol $DyRAM$ and $DyRAM^+$ (both with local recovery from the receivers) always perform better. For instance, we can see in figure 12(a) that $DyRAM$ is 10 times faster than A for a loss rate of 5%. Now, when the loss detection service is applied to A (giving protocol D) the recovery delay can be reduced. In fact as we can see for the different loss rates, D always performs better than A thanks to the loss detection service. When applying the loss detection service to $DyRAM$, the delay of recovery decreased mainly for high loss rates and a large number of receivers. For instance, the loss detection service allows DyRAM to go 4 times faster for 96 receivers and a loss rate of 25%. We can also notice in figures 12(a)(c) that $DyRAM$ performs slightly better than $DyRAM^+$ when the number of receivers is small. Therefore it is unjustified to perform the loss detection service for a small number of receivers since the local recovery is sufficient to reduce the recovery delay. This does not appear to be a limitation of the loss detection service since a multicast session has generally to support a large number of receivers.

5. CONCLUSIONS

Reliable multicast protocols can benefit from active networking technologies where routers can execute additional functionalities. In this paper, we proposed a new service which consists in the loss detection by the routers themselves. In order to evaluate the potential of this new service, we proposed a delay analysis of two active reliable protocols noted A and D.

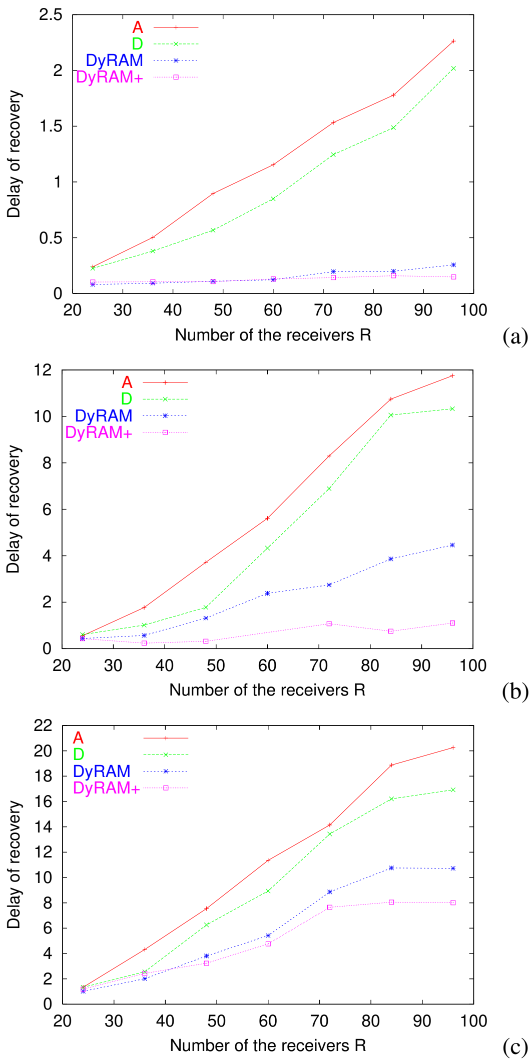


Figure 12: The recovery delay with (a) $p = 0.05$, (b) $p = 0.25$ and (c) $p = 0.50$.

The numerical results showed that one must be careful about where to place a loss detection capable-router. This latter must neither be too far from nor too close to the source. When the router is closer to the receivers, the load introduced by the loss detection service is unjustified because of the long distance to be crossed by the generated NACKs by the router. When the router is put sufficiently far from the source, we maximize the number of losses that can be detected. When the position of the active router is well chosen, we showed that D performs better than A especially for high loss rates. This result can be used to propose an active multicast architecture with specialized routers [8]. For instance the closest router to the source should perform the loss detection while the others will only perform the other active services such as the NACK suppression and the subcasting. The load at the different nodes was also examined and we observed that the routers are the bottleneck when their processing overhead is set to twice the processing requirement at the end hosts. Nevertheless we showed that we do not need to increase the processing power infinitely for the routers to never be the bottleneck.

Based on the analytical study, we added the loss detection service to the DyRAM protocol. Simulation results showed that adding such a service to the source router helps to reduce the delay of recovery without overwhelming the other active routers that perform the replier election service. In fact DyRAM protocol performs better with the loss detection service especially for high loss rates when increasing the number of the receivers.

Acknowledgements

This work is supported in part by the RNRT VTHD++ project and by the ACI GRID program.

6. REFERENCES

[1] R. Bagrodia et al. Parsec: A parallel simulation environment for complex systems. *Computer Magazine*, 1998.

[2] Sally Floyd and al. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking*, 5(6), 1997.

[3] Hugh W. Holbrook, Sandeep K. Singhal, and David R. Cheriton. Log-based receiver-reliable multicast for distributed interactive simulation. In *SIGCOMM*, Oct. 1995.

[4] S. Kasera and S. Bhattacharya. Scalabe fair reliable multicast using active services. *IEEE Network Magazine's Special Issue on Multicast*, 2000.

[5] S. Kasera, J. Kurose, and D. Towsley. Scalable reliable multicast using multiple multicast groups. In *Measurement and Modeling of Computer Systems*, 1997.

[6] L. Lehman, S. Garland, and D. Tennehouse. Active reliable multicast. In *Proc. of the IEEE INFOCOM, San Francisco, CA*, March 1998.

[7] M. Maimour and C. Pham. Dynamic replier active reliable multicast (dyram). In *Proc. of the 7th IEEE Symp. on Comp. and Comm. (ISCC 2002)*, July 2002.

[8] M. Maimour and C. Pham. A loss detection service for active reliable multicast protocols. In *Proc. of the International Network Conference (INC 2002)*, July 2002.

[9] Christos Papadopoulos, Guru M. Parulkar, and George Varghese. An error control scheme for large-scale multicast applications. In *Proc. of the IEEE INFOCOM*, March 1998.

[10] S. Paul and K. Sabnani. Reliable multicast transport protocol (RMTP). *IEEE JSAC, Spec. Issue on Network Support for Multipoint Communications*, 15(3), April 1997.

[11] T. Speakman et al. Pgm reliable transport protocol specification. internet draft, 1998.

[12] D. L. Tennehouse, J. M. Smith, W. D. Sincoskie, D. J. Wetherall, and G. J. Winden. A survey of active network research. *IEEE Communication Magazine*, January 1997.

[13] M. Yamamoto, J. Kurose, D. Towsley, and H. Ikeda. A delay analysis of sender-initiated and receiver-initiated reliable multicast protocols. In *Proc. of IEEE INFOCOM'97, Los Alamitos*, April 1997.

[14] Rajendra Yavatkar, James Griffoen, and Madhu Sudan. A reliable dissemination protocol for interactive collaborative applications. In *ACM Multimedia*, 1995.