An Analysis of a Router-based Loss Detection Service for Active Reliable Multicast Protocols

Moufida Maimour, Cong-Duc Pham

To cite this version:
Moufida Maimour, Cong-Duc Pham. An Analysis of a Router-based Loss Detection Service for Active Reliable Multicast Protocols. 2002, pp.49-56. hal-00096689

HAL Id: hal-00096689
https://hal.science/hal-00096689
Submitted on 19 Sep 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
AN ANALYSIS OF A ROUTER-BASED LOSS DETECTION SERVICE FOR ACTIVE RELIABLE MULTICAST PROTOCOLS

Moufida Maimour and Cong-Duc Pham

RESAM, University Lyon 1
ENS, 46 alle d’Italie
69364 Lyon Cedex 07 - France
e-mail:{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 the reliability. In this context, routers are able to perform customized computations on the messages flowing through them. In this paper, we propose a new active service which consists in a 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 messages (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
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 generate immediately 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 $p_I$. Therefore the end-to-end loss probability perceived by a receiver is $p = 1 - (1 - p_I)^2$. 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 $\rho$ 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:

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

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. $M_r$ 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 $E[M] = cd\sum_{m=0}^{\infty} \frac{1 - (1 - p^m)^B}{(1 - p^m)^B}$. For $M_r$, we have $P[M_r \leq m] = 1 - p^m$ thus $E[M_r] = 1/(1 - p)$. Similarly $E[M_n] = 1/(1 - p_I)$. We will note $X$ (respectively $X_n$) and $Y$ (respectively $Y_r$) 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 $\lambda$ packets per unit of time.

### 2.1. Waiting times

In protocol A, the data packet arrival rate at the source is $\lambda_d^{A} = \lambda$ with a mean requirement service of $E[X]$. The NACK packet arrival rate at the source is $\lambda_r^{A} = \lambda \frac{E[M_r] - 1}{E[M_r]}$ with a mean requirement service of $E[X] + E[Y]$. Therefore the load at the source
receivers are identical to those of protocol for forwards it). On average the active router receives only one NACK because of the NACKs aggregation functionality).

At a randomly chosen receiver the data packet arrival rate is $\lambda_{\Delta}^A = \lambda E[M_{r}] - \lambda_{r}$. Therefore the data packet arrival rate at the receiver is $\lambda_{r}^A = \lambda E[M_{r}] - \lambda_{r}$ 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 $\lambda_{r}^A = \lambda E[M_{r}] - \lambda_{r}$ with a mean requirement service of $E[Y_{r}] + E[Y_{r}] / B$ (the router receives all the NACKs generated for a lost data packet but forwards upstream only one NACK because of the NACKs aggregation functionality). The load at the active router for protocol $A$ is as follows:

$$\rho_{A}^r = \lambda E[M_{r}] - \lambda_{r}$$

The mean waiting time at the active router is:

$$E[W_{A}^r] = \frac{4\lambda_{r}^A E[X_{a}] + \lambda_{r}^A E[Y_{r}](B + 1)^2 / B^2}{2(1 - \rho_{A}^r)} \tag{4}$$

At a randomly chosen receiver the data packet arrival rate is $\lambda_{r}^A = \lambda E[M_{r}] - \lambda_{r}$ with a required service of $E[X_{r}]$. 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 $\lambda_{r}^A = \lambda E[M_{r}] - \lambda_{r}$ with a mean requirement service of $E[Y_{r}]$. The load at the receiver side for protocol $A$ is:

$$\rho_{A}^r = \lambda E[X_{r}] + \lambda E[M_{r}] - 1) E[Y_{r}] \tag{5}$$

The mean waiting time at the active router is given by:

$$E[W_{A}^r] = \frac{\lambda E[X_{r}] + (E[M_{r}] - 1) E[Y_{r}]}{2(1 - \rho_{A}^r)} \tag{6}$$

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

$$E[W_{r}^D] = \frac{4\lambda_{r}^D E[X_{a}] + (\lambda_{r}^D + \lambda_{r}^D)(B + 1)^2 / B^2 E[Y_{r}]}{2(1 - \rho_{A}^D)} \tag{8}$$

### 2.2. Overall delay analysis

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

$$E[\Delta_{L}] = E[\Delta_{L}^{recov}] + E[\Delta_{L}^{cov}]$$

To compute these times, we need to introduce two random variables $L_{r}$ and $L_{d}$. 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 $i$th data packet. The expression for the mean of $L_{r}$ is given in [13]:

$$E[L_{r}] = \sum_{h=0}^{2k-1} C_{h-1}^k p^h (1 - p)^{(B-k-1)} [p^{k+1} / (1 - p^{k+1})]$$

Similarly $L_{d}$ 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 $E[L_{d}] = p_{i} / (1 - p_{i})$.

For the overall delay analysis, we note $T_{a}$ ($T_{d}$) as the propagation delay from the source (a receiver) to the active router. The propagation delay from the source to a receiver is noted $T_{r} = T_{s} + T_{a}$. The required delay to receive a data packet from the source by a receiver in protocol $\omega$, $d_{p}^{\text{delay}} = T_{s} + 2X_{r} + W_{c}$. The required delay to receive a NACK packet from any receiver by the source in protocol $\omega$ is noted $d_{p}^{\text{delay}} = T_{s} + 2X_{r} + W_{c}$.

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

$$E[\Delta_{L}^{recov}] = E[L_{r}] / \lambda + E[W_{A}^{r}] + E[X_{r}] + E[d_{p}^{\text{delay}}] + E[Y_{r}] + E[W_{r}^{d}] + E[X_{r}] \tag{9}$$

To estimate a mean for $\Delta_{L}^{recov}$, we must take into consideration that a repair may be lost. Consequently $\Delta_{L}^{recov}$ 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 repair 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 ($d_{p}^{\text{delay}}$), the processing time at the source ($W_{s} + X_{r}$), the delay required for the repair to be received by the receiver ($d_{p}^{d}$) and the processing time at the receiver ($W_{r} + X_{r}$). Therefore we can write:

$$E[\Delta_{L}^{recov}] = E[d_{p}^{\text{delay}}] + E[W_{A}^{r}] + E[X_{r}] + E[d_{p}^{d}] + (E[W_{r}^{d}] + E[X_{r}]) (T_{out} + E[W_{r}^{d}] + E[Y_{r}]) - \sum_{j=0}^{j-1} (1 - p)^{j+1} (1 - p)$$

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. \textit{Tout}, is the value of the timeout set at the receivers. Finally, we have:

\[
E[\Delta_{\text{recov}}^t] = E[\text{rpd}^t] + E[\text{dpt}^t] + E[W_s^t] + E[W_r^t] + 2E[X] + (\text{Tout} + E[W_s^t] + E[Y]) p/(1-p) \quad (10)
\]

For protocol D, a loss may be detected by the router with a probability \( p_1 \) (the probability that the data packet is lost at the source) or by any of the affected receivers otherwise. Therefore \( \Delta_{\text{det}, \text{router}}^D \) is a function of \( \Delta_{\text{det}, \text{receiver}}^D \) and \( \Delta_{\text{det}, \text{router}}^D \), which are respectively the time required to detect a loss by a receiver and by an active router. Therefore we have:

\[
E[\Delta_{\text{det}, \text{router}}^D] = (1 - p_1)E[\Delta_{\text{det}, \text{receiver}}^D] + p_1 E[\Delta_{\text{det}, \text{router}}^D]
\]

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

\[
\]

and

\[
\]

giving:

\[
E[\Delta_{\text{det}, \text{router}}^D] = p_1(1 - p_1)/\lambda + E[W_u^D] + E[X] + T_{\text{d}} + E[X_u] + E[Y_u] + E[W_u^D] \quad (12)
\]

Similarly, the recovery delay can be given by:

\[
E[\Delta_{\text{recov}}^D] = (1 - p_1)E[\Delta_{\text{recov}, \text{receiver}}^D] + p_1 E[\Delta_{\text{recov}, \text{router}}^D]
\]

where \( \Delta_{\text{recov}, \text{receiver}}^D \) is the required time to recover if the loss is detected by the router. Similarly \( \Delta_{\text{recov}, \text{router}}^D \) is the required time to recover if the loss is detected by a receiver. \( \Delta_{\text{recov}, \text{receiver}}^D \) is similar to \( \Delta_{\text{recov}}^A \) so:

\[
\]

Since a router sets a timeout and retransmits the NACK if the required data packet has not been received yet, \( \Delta_{\text{recov}, \text{router}}^D \) can be expressed by:

\[
E[\Delta_{\text{recov}, \text{router}}^D] = E[\Delta_{\text{recov}, \text{receiver}}^D] + E[\Delta_{\text{recov}}^D]
\]

where \( E[\Delta_{\text{recov}}^D] \) is the required delay until the active router has received the data packet so it will never generate a NACK for it. \( E[\Delta_{\text{recov}}^D] \) 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 \( \Delta_{\text{recov}}^A \), we find:

\[
E[\Delta_{\text{recov}, \text{router}}^D] = 2T_{\text{t}} + E[W_s^D] + E[X] + E[W_r^D] + 2E[X_u] + (\text{Tout} + E[W_r^D] + E[Y_u]) p/(1-p) \quad (13)
\]

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 \( \Delta_{\text{recov}}^A \), we can derive:

\[
E[\Delta_{\text{recov}}^D] = E[\text{rpd}^D] + E[\text{dpt}^D] + E[W_s^D] + E[W_r^D] + 2E[X] + \text{Tout} + T_{\text{t}} + (\text{Tout} + E[W_r^D] + E[Y]) p/(1-p) \quad (14)
\]

### 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 \( \mu \text{sec} \). We can take \( E[Y] = 0.2 \) accordingly which corresponds to 100 \( \mu \text{sec} \). 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. \( E[X_u] = E[Y_u] = 2E[X] = 2 \). In order to consider the detection-capable active router position, we introduce \( \alpha \in [0, 1] \) which is the ratio of \( T_{\text{d}} \) to \( T_{\text{t}} \). These two parameters are expressed as a function of the number of links crossed by a packet. We assume that we need 1 unit of time (this corresponds to 500 \( \mu \text{sec} \)) to go across one link. Therefore we can use \( T_{\text{d}} \) (respectively \( T_{\text{t}} \)) to represent the number of links between the router (respectively the source) and any receiver. We will also consider
two particular values of \( \alpha \): \( \alpha_c = (T_{sr} - 1)/T_{sr} \) 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.

Figure 4: The delay gain in \( D \) as a function of the router position

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 \( \alpha \)) 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 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.

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 long 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 \( A \).

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 the 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 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 precisely 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.

An other deciding factor is the number of receivers associated to the active router. Figure 10 shows that $D$ performs better than $A$ in term of the end-to-end delay. It is important to note that for low loss rates (say 5% and 10%) protocol $D$ introduces a gain of only 5% that does not change even if we increase the number of the 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.

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.

A receiver detects a loss by sequence gaps and upon the detection of a loss, a receiver immediately sends a NACK toward the source and sets a timer. Since NACKs and repairs may also be lost, a receiver will re-send a similar NACK when the requested repair has not been received within the timeout interval. In order to limit the processing overheads of duplicate NACKs and to avoid the corresponding retransmissions, the source, the active routers and the receivers ignore similar NACKs for a certain period of time. Routers maintain information about NACKs flowing through them. For each received NACK, the router creates or simply updates an existing NACK state (NS) structure. Such a structure contains during its life time the following information:

- $seq$: the sequence number of the requested packet,
- $\text{subList}$: a subcast list that contains the list of links (downstream or upstream) on which NACKs for this packet have arrived.

4.2. NACK Suppression and Subcast

On receipt of the first NACK packet for a data packet, a router would create a corresponding NS structure, initialize a timer noted
DTD (Delay To Decide) that will trigger the election of a replier to whom this first NACK will be sent. All subsequent NACK packets received during the timeout interval are used to properly update the corresponding subcast list and are dropped afterward. When a data packet arrives at an active router it will simply be forwarded on all the downstream links if it is an original transmission. If the data packet is a repair packet the router searches for a corresponding NS structure and will send the repair on all the links that appear in the subcast list.

### 4.3. Replier Election

During the DTD time window, a router collects as much information as possible about the links affected by a loss (updates of the subcast list). On expiration of the DTD timer, the router is able to choose a replier link among those that are not in the subcast list. In an attempt to avoid for the overloading of a particular downstream link, the router always try to choose a different link from the previously elected one (if available) by respecting a ring order among them, thus realizing when possible a load balance.

When a router receives NACK packets from all of the downstream links before the expiration of the DTD timer, it will immediately forward the last NACK received toward the source and will cancel the replier election process. An active router keeps track of the received data packets by maintaining a track list structure (TL) for each multicast session. A TL has three components:

- **lastOrdered** is the sequence number of the last data packet received in order. All packets with a sequence number less or equal to lastOrdered have definitely been received by the router.
- **lastReceived** is the last received data packet sequence number. All packets with a sequence number greater than lastReceived have not been received by the router.
- **lostList** is the list of sequence numbers greater than lastOrdered and less than lastReceived of data packets not received yet by the router. This list is empty when (lastReceived < lastOrdered + 1) and contains at least one element otherwise.

The track list (TL) structure allows an active router to forward the first valid NACK immediately toward the source without waiting for the DTD timer expiration or the reception of similar NACKs from all the downstream links. This concerns the case when the requested data packet sequence number is greater than lastOrdered and contained in the lostList. The subcast list is updated so that the repair packet (from the source) would be forwarded on all downstream links.

### 4.4. 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. 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...
for different loss rates. First of all, it is noticeable that protocols
These values are derived from measures in [6].
5%. Now, when the loss detection service is applied to
for the four protocols as a function of the number of the receivers
that both NACKs and data packets are processed in 32 time units.
ery active router
simulation model values are normalized to the NACK transmis-
sion time. For the processing overheads at the routers, we assume
there is 6 backbone links between the source router A0 and ev-
ery active router Ai. The simulations are implemented using the
PARSEC language developed at UCLA [1].
For all the simulations, we set 6 = 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
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 12 plots the recovery delay (normalized to the RTT)
for four protocols as a function of the number of the receivers
different loss rates. First of all, it is noticeable that protocols
DyRAM and DyRAM+ with local recovery from the receivers
always perform better. For instance, we can see in figure 12(a)
that DyRAM goes up to 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
to the loss detection service. When applying the loss detec-
tion 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 slightly performs better than DyRAM+
when the number of receivers is small. Therefore it is unjustified
to perform the loss detection service for a few 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 have gained popularity with active
services contribution where routers implement additional function-
ali...
experienced load at the different nodes.

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.

6. REFERENCES


