Dynamic Replier Active Reliable Multicast (DyRAM)

M. Maimour and C. D. Pham
RESAM, Lyon1
ENS, 46 allée d'Italie
69364 Lyon Cedex 07 - France
tel: (0033) 7272 8504, fax: (0033) 7272 8080.
email:{mmaimour,cpham}@ens-lyon.fr

Abstract

Desirable features of reliable multicast include low end-to-end delays, high throughput and scalability, and meeting these objectives is not an easy task. We propose a receiver-based (replier) local recovery multicast protocol with dynamic repliers elected on a per-packet basis. Designed to provide an efficient reliable multicast service without any cache facilities in the multicast tree, our approach, noted DyRAM, uses low-overhead active services in routers. After presenting the protocol and some designed and implementation issues the paper compares DyRAM to ARM, the nearest active reliable multicast protocol in term of functionalities proposed in the active multicast community. Simulation results show that DyRAM performs much better than an ARM-like protocol which needs a significant amount of cache to exhibit performances. Additionally, in case some cache is available for DyRAM and ARM, the study shows that DyRAM always performs better than ARM.

1. Introduction

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 (XTP [10] for instance). 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, 8, 13, 7, 9, 3].

Recently, the use of active network concepts [11] where routers themselves could contribute to enhance the network services by customized functionalities have been proposed in the multicast research community. Contributing mainly on feedback implosion problems, retransmission scoping and cache of data, active reliable multicast offer a general and flexible framework for customized functionalities in network protocols (although many problems regarding deployment and security remains). ARM (Active Reliable Multicast) [5] and AER (Active Error Recovery) [4] are two protocols that were recently proposed in the research community and that use active services within routers. They differ in the strategy adopted for solving the NACK implosion problem. In ARM, a receiver experiencing a packet loss sends immediately a NACK to the source. Active services in routers then consist in the aggregation of the multiple NACKs. In contrast, AER uses a strategy similar to the one used by SRM and based on local timers at the receivers: prior to send a NACK packet, a receiver initiates a timer and waits for a random amount of time. In addition, an active router in ARM would send the repair packet only to the set of receivers that have sent a NACK packet (subcast). In AER, the active router simply multicasts the repair packet to all its associated receivers. In both protocols, the cache of data packets is performed within the network (at the routers in ARM and in servers co-located with the routers in AER) to permit local recoveries.

Local recoveries can dramatically decrease the recovery latency. There are basically 2 possibilities for enabling local recoveries: (i) use some receivers or dedicated servers as repliers, or (ii) use network elements such as routers. For instance, the replier could be any receiver in the neighborhood (SRM [2]), a designated receiver (RMTP [8], TMTP [13], LMS [7], PGM [9]) or a logging server (LBRM [3]) in a hierarchical structure. Using a replier has the main advantage of requiring a memory usage as low as possible within network elements and is potentially more scalable. Now, the choice of the replier can be done in several ways. Protocols that use some kind of router assistance are LMS and PGM. LMS consists in adding more topology information with little help from routers. With some specific forwarding functions within routers, the protocol elects a designated replier (leader) for each subtree to respond to the requests made by the end hosts. Routers in this case help in the process of electing a replier for each subtree and perform specific routing in order to forward requests to the elected replier. PGM described in [9] is another protocol that uses some router assitance (not active router, as described previously, but rather PGM router) to discover and elect Designed Local Retransmitters (DLR must be on the path towards the source).

The approach we propose use local recoveries performed by repliers elected amongst a sub-set of receivers. Instead of an approximate solution based on timers as in SRM, or a complex DLR discovery as in PGM, the elected replier in our case is dynamically determined at each lost packet (without much overhead as it will be described later on), and not determined at the beginning of the multicast session, thus justifying the “dynamic replier” property of our active reliable multicast protocol (noted DyRAM). Since a specific replier can be chosen for each lost packet, it is therefore possible to have several logical subtrees at the same time for the recovery process. In addition, this very dynamic choice of the replier provide load balance features that decreases the end-to-end latency and reduce the receiver overhead. As opposed to LMS and PGM, DyRAM uses the concept of active networking with active services within routers for implementing several advanced mechanisms including the replier election. This choice of active networking is motivated by the fact that, although difficult, the deployment on-the-fly of customized services appears to be more general and flexible than a solution based on a dedicated, static router (PGM routers for example). However, this choice leads to some design constraints where the active services must incur as low as possible execution/memory overhead within routers since the service will mainly be executed in software (dedicated routers can take the ASIC solution for achieving performance).

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.

The purpose of the paper is to present the DyRAM approach and comparisons with ARM, the nearest truely active reliable multicast protocol in term of functionalities (global NACK suppression and subcasting) proposed in the active multicast community. We show that the dynamic replier election successes in providing high-bandwidth, low-latency and low-overhead for the local recovery procedure. The rest of the paper is organized as follows. Section 2 describes the protocol. Section 3 presents the simulation results and section 4 concludes with some future directions.

2. Protocol description

DyRAM is a reliable multicast protocol with a recovery strategy based on a tree structure constructed on a per-packet basis with the assistance of routers. It is an incremental step from the existing propositions but it has been designed to provide low-overhead active functionalities with a dynamic replier election on a per-packet basis. 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:

- the NACK suppression of duplicate NACKs in order to limit the NACK implosion problem.
- 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.
- the replier election which consists in choosing a link as a replier one to perform local recoveries from the receivers instead of caching data packets at the routers.

2.1. DyRAM general features

Packet structure. In order for DyRAM to work correctly, we will assume that data packets are labelled uniquely by a sequence number and also that it is possible to distinguish between an original transmission from a repair (dedicated field in the packet at the IP-multicast level for example). Additionally, a NACK packet contains in its header the source address, the affected receiver address, the multicast group address and the sequence number of the requested data packet.

The NACK-based strategy. In DyRAM, the source sends data packets to the multicast address subscribed to by all the receivers. The core functionalities of IP multicast are assumed to deliver the packets to the receivers, as best as possible. The receivers are responsible for detecting, requesting and in most cases, retransmitting a lost data packet.

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. Practically, the timeout must be set to at least the estimated round trip time (RTT) to the source. In order to detect the lost of the last packets (no gap in sequence number in this case), a receiver will request them if no data has been received within a pre-defined receiving time window. Upon reception of a NACK packet, the source sends the repair packet to the multicast address.

In order to limit the processing overheads of duplicate NACKs and to avoid the corresponding retransmissions, the source, the active routers and the receivers (remember that DyRAM uses repliers among receivers) ignore similar NACKs for a certain period of time. According to this rule, a NACK will be said "valid" if it is received for the first time or no similar NACKs are received during the discard time window.

As the receivers contribute in the recovery process, a receiver that receives a valid NACK will check if the requested packet has already been received. If so it will send a repair for this packet toward the source otherwise it will send a similar NACK instead. This NACK will help the upstream active router to correctly perform another replier election.

2.2. Routers' contributions

At the active routers side, the NACK suppression, the subcast and the replier election services can be implemented simply by maintaining information about the received NACKs. This set of information is uniquely identified by the multicast address. For each received NACK, the router creates or simply updates an existing NACK state (NS) structure which has a given life time. Such a structure contains during its life time, the following information:

- seq: the sequence number of the requested packet,
- $rank$: initially set to 0, is increased for each NACK packet received. Is reset to 0 after the election.
- $subList$: a subcast list that contains the list of links (downstream or upstream) on which NACKs for this packet have arrived.

NACK suppression. 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 NS structure (rank and subcast list) and are dropped afterward. We use the rank information in some cases to be sure that all downstream receivers have sent a NACK packet.

Subcast functionality. The subcast list in the NS structure is an important component for the subcast functionality. This list contains the set of links (downstream or upstream) from which a NACK has been received. 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.

An NS is created as soon as a valid NACK arrived for which no previous NS was created, and can be replaced only on receipt of the corresponding repair. An NS structure can be freed also when all the downstream links have lost the corresponding data packet. This is the case when the active router receives similar NACK packets from all the downstream receivers or routers. As we expect to only free an NS when the corresponding repair is received, when a repair arrives at the router and finds the corresponding NS structure, the subList is used to perform the subcast functionality. Nevertheless, if such NS structure is not found, this means that it was not possible to perform the subcast at all and the repair is multicast on all the links downstream.

Replier election. The set of information contained in a NS structure for each received NACK can be used to perform an other key functionality of our approach in addition to the NACK suppression and the subcast features. This key functionality is the "replier election" and is specific to DyRAM.

On reception of a valid NACK, the router initializes a timer noted DTD (Delay To Decide) in order to collect during this time window 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. This link may end up to be the upstream one if all the downstream links appear to be in the subcast list. In some cases, an active router could decide to send the NACK on the upstream link even before the expiration of the DTD timer. This mainly happens when similar NACK packets are received from all the downstream links. 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 (in addition to provide robustness to replier and link failure). The replier would then send back the repair packet to the DyRAM router which would in turn subcast the repair packet on all the links in the subcast list.

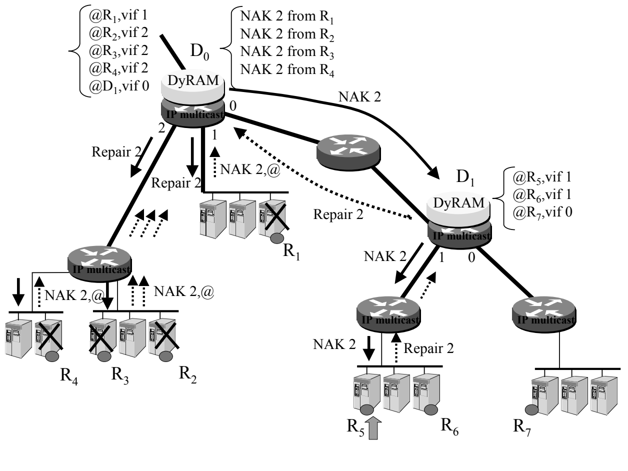


Figure 1. Replier election process.

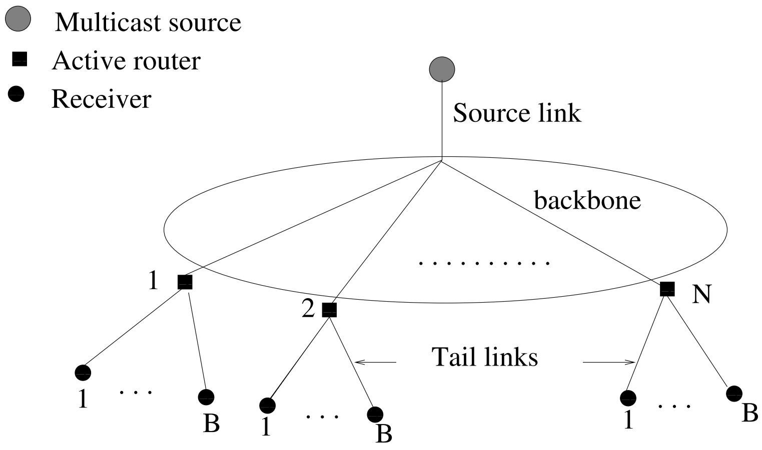


Figure 2. Network Model.

Figure 1 shows the replier election procedure where $R_1 ... R_4$ have lost the data packet number 2. The NACK will be issued on link 0, will go through some regular router, arrives at the local DyRAM router (D_1) which will select the link 1 as a replier. For more details about the election procedure refer to [6].

In some cases, a non up-to-date subcast list may impact on the decision of the router in the replier election process. Such a situation may happen due to the fact that receivers detect losses primarily by sequence gaps in the data packets. This is why receivers with a shorter latency to the source are likely to detect losses before receivers farther away. One consequence may be the election of a replier which has itself lost the requested data packet (and has not issued the NACK before the expiration of the DTD timer). In this case, this receiver will simply return a NACK back to the source and the router will update its subcast list and choose another replier link.

Reducing the "Delay To Decide" timer overhead. In order to decrease the overhead (longer recovery latency) introduced by the DTD, we propose to keep track within a router of the received data packets in order to quickly detect packet losses occurring from upstream and affecting all its subtree. This can be done simply by maintaining a track list structure (TL) for each multicast session handled by the router. 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 sequence number of the last received data packet.

- $lostList$ contains the list of data packets not received by the router with sequence number greater than $lastOrdered$ and less than $lastReceived$. This list is empty when ($lastReceived < lastOrdered + 1$) and contains at least one element otherwise.

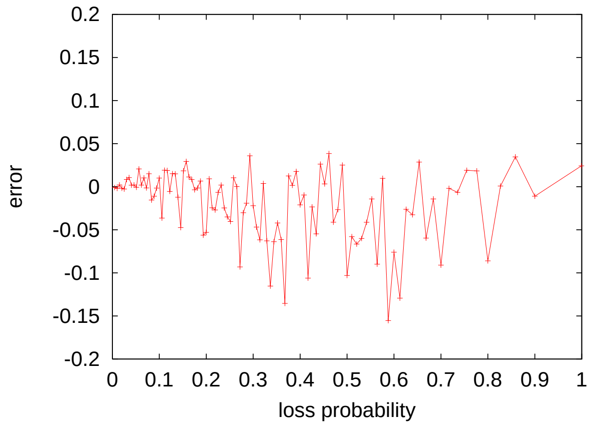
A router maintaining such a track list (TL) structure is able to decide to forward the NACK immediately toward the source instead of waiting for the expiration of the DTD timer when the requested data packet sequence number is greater than $lastOrdered$ and contained in the $lostList$.

3. Simulations

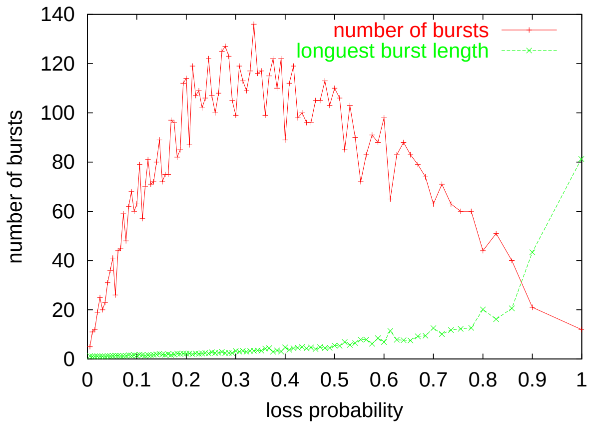
3.1. Network model

We implemented a simulation model of DyRAM (in the PARSEC language developed at UCLA [1]) and evaluated its performances on a network model. The network model considers one source that multicasts data packets to R receivers through a packet network composed of a fast core network and several slower edge access networks. We will call *source link* the set of point-to-point links and traditional routers that connects the source to the core network. Similarly, a *tail link* is composed of point-to-point links and routers connecting a receiver to the core network (see Fig. 2). We only consider active routers at the edge of the core network. This is due to the fact that the core network is reliable and a very high-speed network. Adding complex processing functions inside the core network will certainly slow down the packet forwarding functions.

Each active router A_i is responsible of B receivers noted $R_{i1}, \cdots, R_{iB}$ forming a local group. We assume that there is l_b backbone links between the source and every active router A_i.



(a)



(b)

Figure 3. The temporal correlation.

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. The core network is considered reliable as mentioned previously. For the other links (the source link or the tail links), the loss probability is noted p_l. Therefore, the end-to-end probability of a packet loss perceived by a receiver is $p = 1 - (1 - p_l)^2$. The losses at the tail links are assumed to be mutually independent. To take into consideration the temporal correlation of losses, we use the Markov chain model proposed in [12]. Let M be the transition matrix $M(i, j) = P[j/i]$ where $P[j/i]$ is the probability to be in state j knowing that the current state is i. Our simulations showed that using the M matrix proposed in [12], p_l is of about 0.9154. In order to be able to perform simulations with different values of p_l, we introduced the M_λ matrix defined by:

$$M_\lambda = \frac{1}{\lambda} \cdot M$$

Having p_l as an input, λ can be computed using:

$$\lambda = \frac{b}{p_l} + a$$

where $a = 0.8068$ and $b = 0.1768$. Fig. 3(a) shows that with this loss model, we achieve in most cases the targeted loss rate. Moreover the losses occur in bursts whose length increases with the loss probability as shown in fig. 3(b).

3.2. Metrics

To evaluate our protocol we have defined a set of metrics. Firstly, M_S is defined as the number of retransmissions from the source per packet and gives an idea on the load at the source.

$$M_S = \frac{Number\ of\ retransmissions}{Number\ of\ sent\ packets}$$

To quantify the load at the network level, we use BW which is the average bandwidth consumed per link during the multicast session:

$$BW = \frac{Bw_{NACK} + BW_{Data}}{N.B + l_b.(N+1)}$$

where BW_{NACK} and BW_{Data} are respectively the bandwidth consumed by the NACK packets and the data packets during the multicast session:

$$BW_{NACK} = \sum_{i=1}^{NackNb} l_{NACK_i}$$

$$BW_{Data} = \sum_{i=1}^{DataNb} l_{DP_i}$$

where l_{NACK_i} and l_{DP_i} are respectively the number of links crossed by the ith NACK packet and the ith data packet.

The third metric is the completion time per packet which is the required time to successfully receive the packet by all the receivers (can be seen as the latency).

3.3. Simulation results

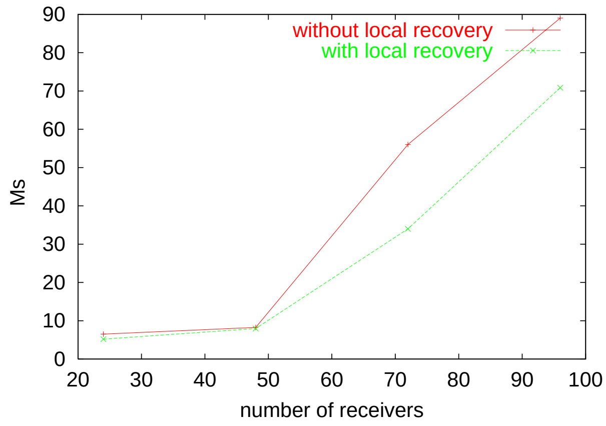
For all the simulations, we set $l_b = 55$. A NACK and a data packet are considered to be of 32 and 1024 bytes respectively. All simulation model values are normalized to the NACK transmission time (e.g. the time required to send or receive a NACK is set to 1, for a data packet this time is set to 32). For the processing overheads at the routers, we assume that both NACKs and data packets are processed in 32 time units. In our simulation models, we have taken into consideration the fact that the repairs may also be lost.

We verified that the three metrics defined above behave in the same manner and are dependent on each other. First we will show the benefit of performing the local recoveries from the receivers. Then a comparison between our approach (no cache in routers) and ARM (with cache in routers, no repliers) will be presented. Finally, we will consider the case when DyRAM also benefits from some cache amount in routers in the multicast tree.

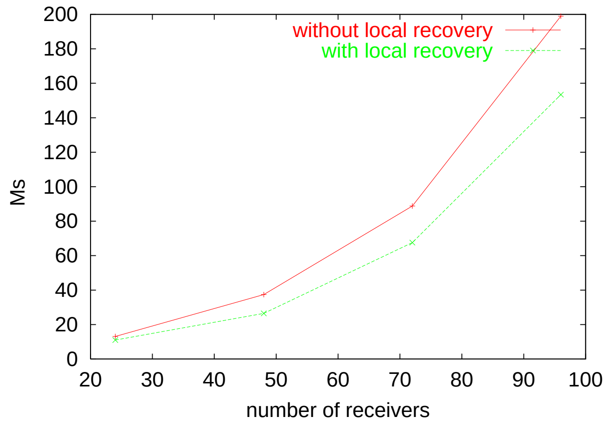
Local recovery from the receivers. Figure 4 plots for DyRAM the number of retransmissions (M_S) from the source as a function of number the receivers, with and without local recovery, for 5% and 25% loss rates. These results have been obtained from simulations of a multicast session with 48 receivers distributed among 12 local groups. The curves show that the local recovery introduces less load at the source as the number of receivers increases, especially for high loss rates. Even not shown, the consumed bandwidth and the completion time are reduced in about the same proportions.

Putting the recovery process in the receivers requires at least 2 receivers per active router otherwise local recoveries can not be realized. Therefore the local group size (B) is an important parameter. In order to study the impact of B simulations are performed with 48 receivers distributed among groups of different sizes. Figure 5 compares the per-link consumed bandwidth ratio of DyRAM to an other approach without any local recovery facilities as the group size is varied. As the receivers number per group increases, the consumed bandwidth is reduced with larger ratios for large loss rates. In fact when the group size is larger, there are more chances that the members of the group can recover from each other. For instance, figure 5 shows that with only 6 receivers per group and local recovery we achieve a gain of 80%.

DyRAM vs. ARM. In this section, we compare our approach to ARM. Figure 6 shows for different loss rates, the ratio of the consumed bandwidth (BW), the load at the source in term of number of retransmissions and the completion time for ARM and DyRAM. We took into consideration the percentage of the available cache at the ARM routers. Without any cache at the routers, DyRAM always performs better than ARM. When ARM benefits from caching at the routers, it performs better only for very low loss rates or large cache size at the routers. However for large loss rates (e.g. 25%), ARM requires more than 50% of cache at the routers to perform better than DyRAM. Our approach is more promising because even without any caching overhead at the routers DyRAM still has better performances than ARM when the latter can use more than 50% of cache.



(a)



(b)

Figure 4. Load at the source in DRARM with and without local recovery (a) $p = 0.05$, (b) $p = 0.25$.

DyRAM combined with cache at the routers. Designed to provide an efficient reliable multicast service without any cache within active routers, the results presented so far show that DyRAM does not provide better performances than an ARM-like protocol when the loss rate is small. However DyRAM performs much better for large loss rates without requiring higher overheads.

The results we present now assume that some cache memory are available in routers in addition to the DyRAM native local recovery strategy. One interesting question is: of how much DyRAM's performances can be enhanced? To answer this question, figure 7 plots for different loss rates the gain in consumed bandwidth achieved by ARM and DyRAM when the cache size is varied over a simple ARM without caching in routers. The main results are that with cache facilities at the routers for both protocols, ARM never performs better than DyRAM even for low loss rates. When

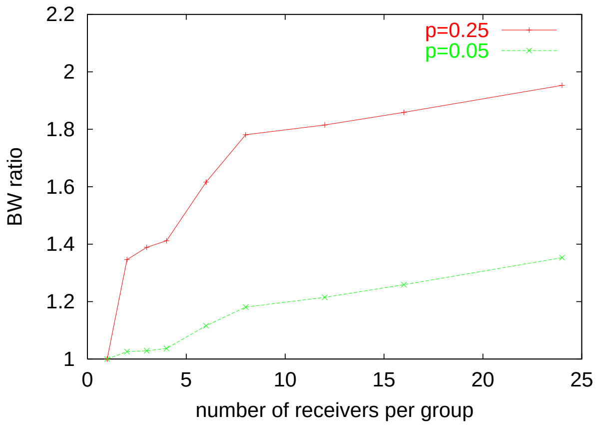


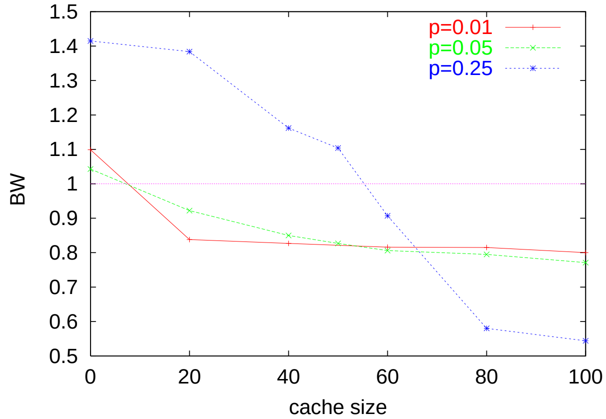
Figure 5. Consumed bandwidth ratio in DyRAM with and without local recovery as a function of the group size.

no cache is available at the routers, DyRAM outperforms ARM and this for any loss rate. As p increases, DyRAM performs better and better. In fact for 25% loss rate, ARM needs more than 50% of cache to achieve the same level of performances than DyRAM without any caching at the routers. To obtain a gain of 20%, ARM must benefit from 40% of cache. Instead, DyRAM without any cache already performs 40% better and with 40% of cache, it performs 60% better. One different way to look at the figures is that if DyRAM has only 40% of cache available then it already performs better than ARM with more than 60% of cache (for 1% and 5% loss rate). With 80% of cache, DyRAM has the same level of performances than ARM with 100% of cache available. Stated differently, about 80% of cache at the routers gives DyRAM the same features than a protocol with an infinite storage capacity.

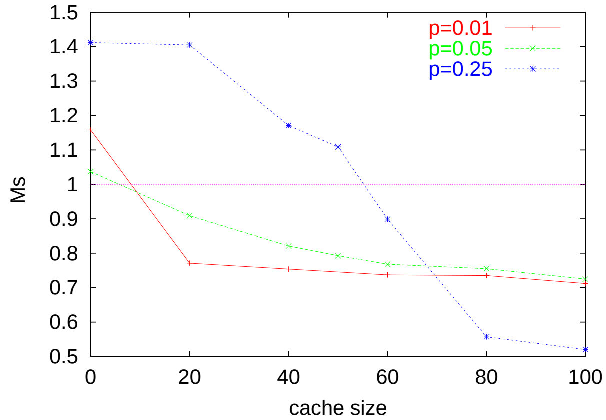
4. Conclusions

DyRAM is a reliable multicast protocol that uses on a per-packet basis a replier chosen among the receivers to perform the recoveries. DyRAM uses active services within routers to perform NACK suppression, subcasting facilities and the dynamic election of replier. When comparing DyRAM with ARM, a very similar protocol in term of functionalities (NACK suppression and subcasting), we found that DyRAM performs better in a large range of loss probabilities.

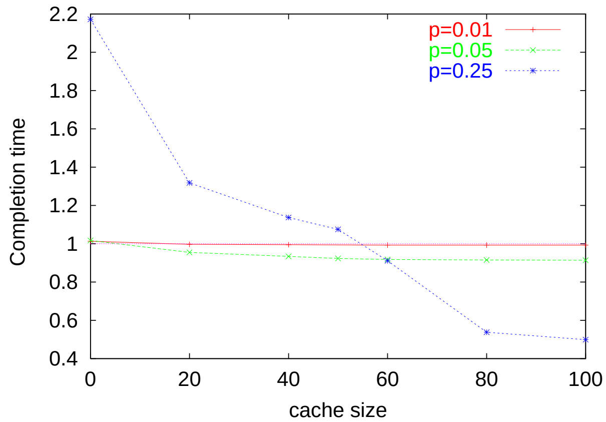
One of our main concerns while designing DyRAM was to propose low-overhead (in terms of processing and memory requirements) and easy to implement solutions for the active services to be put in routers. Based on small data structures we believe that NACK suppression, subcast-



(a)



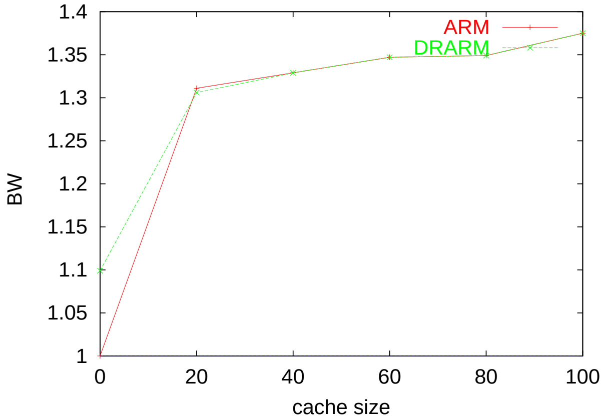
(b)



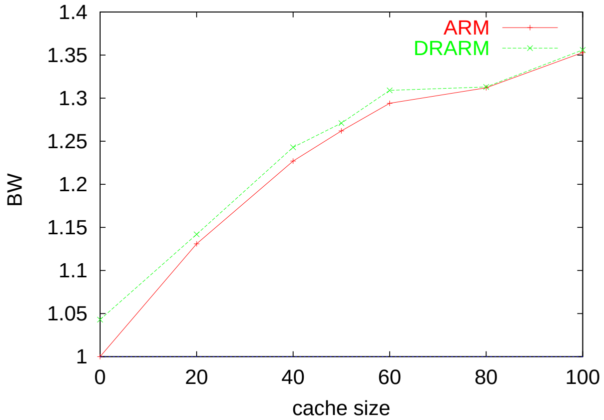
(c)

Figure 6. DyRAM vs. ARM (a) Consumed bandwidth, (b) Load at the source, (c) Completion time.

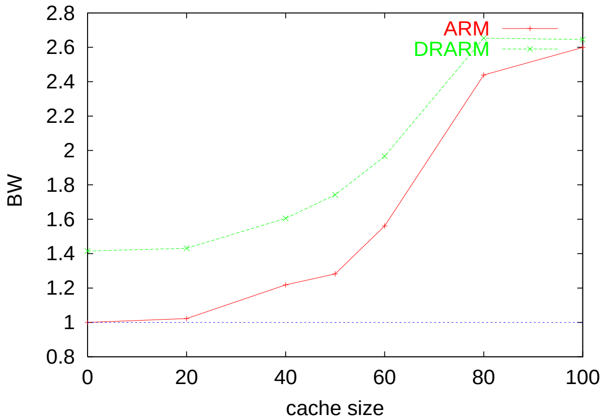
ing and replier link election introduce only low overheads in routers and that implementations without much performance degradation on the routing and forwarding functions are possible. We are currently in the process of prototyping DyRAM on an active network test-bed in order to verify the simplicity of our proposed solution.



(a)



(b)



(c)

Figure 7. Gain obtained by DyRAM and ARM when both benefit from cache in the routers (a) $p = 0.01$ **(b)** $p = 0.05$**, (c)**$p = 0.25$**.**

References

[1] R. B. et al. Parsec: A parallel simulation environment for complex systems. *Comp. Mag.*

[2] S. Floyd, V. Jacobson, C. Liu, S. McCanne, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM ToN*, 5(6):784–803, 1997.

[3] H. Holbrook, S. Singhal, and D. Cheriton. Log-based receiver-reliable multicast for distributed interactive simulation. In *SIGCOMM*, volume 25, pages 328–341, October 1995.

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

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

[6] M. Maimour and C. D. Pham. Dynamic replier active reliable multicast (drarm). Technical report, RESAM, http://www.ens-lyon.fr/˜mmaimour/Paper/TR/TR02-2001.ps.gz, July 2001.

[7] C. Papadopoulos, G. Parulkar, and G. Varghese. An error control scheme for large-scale multicast applications. In *Proc. of the IEEE INFOCOM*, pages 1188–1996, March 1998.

[8] S. Paul and K. Sabnani. Reliable multicast transport protocol (RMTP). *IEEE journal of Selected Areas in Communication, Special Issue on Network Support for Multipoint Communications*, 15(3):407–421, April 1997.

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

[10] T. Strayer, B. Dempsey, and A. Weaver. XTP – THE XPRESS TRANSFER PROTOCOL. *Addison-Wesley Publishing Company*, 1992.

[11] D. L. Tennehouse, J. M. Smith, W. D. Sincoskie, D. J. Wetherall, and G. J. Winden. A survey of active network research. *IEEE Com. Mag.*, pages 80–86, January 1997.

[12] M. Yajnick, J. Kurose, and D. Tosley. Packet loss correlation in the mbone multicast network. In *Proc. of Global Internet conference*, Nov 1996.

[13] R. Yavatkar, J. Griffoen, and M. Sudan. A reliable dissemination protocol for interactive collaborative applications. In *ACM Multimedia*, pages 333–344, 1995.