Designing a Caching-Based Reliable Multicast Protocol

Carolos Livadas Idit Keidar Nancy A. Lynch
Laboratory for Computer Science, Massachusetts Institute of Technology
$\{$*clivadas,idish,lynch*$\}$*@theory.lcs.mit.edu*

1. Introduction

With the increasing use of the Internet, multi-party communication and collaboration applications are becoming mainstream. This trend calls for high-performance multicast services that scale to large groups and higher bandwidth requirements. Although packet loss characteristics have a large impact on the performance of multicast services, few protocols attempt to adapt and actively exploit such characteristics.

In this paper, we describe a reliable multicast protocol that exploits packet loss locality through caching. Several studies [1, 3, 5] have observed that packet losses in multicast communication are bursty, *i.e.*, links drop numerous multicast packets while temporarily congested. Thus, consecutive losses as witnessed by individual hosts are likely to occur on the same lossy link. By caching pertinent information regarding the error recovery of prior losses and optimistically presuming that future losses occur on the link responsible for prior losses, our protocol streamlines the recovery of future losses. This scheme demonstrates how packet loss locality can be actively used to reduce the recovery latency and the bandwidth overhead of multicast error control. Moreover, in view of increasing our confidence in the correctness and performance of our protocol, we use a rigorous design approach.

2. Background

Multicast communication refers the transmission of data in the one-to-many and many-to-many settings; that is, where one or more hosts within a group transmit data that is destined for all the members of the group. Reliable multicast refers to the reliable transmission of data in such settings; that is, when the data transmitted is guaranteed to reach all the members of the group. Moreover, a (reliable) multicast *session* refers to a particular instance in which a set of hosts, which may be dynamic, engage in (reliable) multicast communication. Among the slew of reliable multicast protocols proposed to date, Scalable Reliable Multicast (SRM) [2] is a simple and robust retransmission-based protocol. SRM uses IP multicast to multicast messages to all the members of the reliable multicast group. In turn, IP multicast uses underlying spanning trees to disseminate these messages to all group members in a best-effort manner, *i.e.*, with no delivery or performance guarantees.

Packet recovery in SRM is initiated when a receiver detects a loss and schedules the transmission of a *request*; an error control message requesting the retransmission of the missing packet. If a request for the same packet is received prior to the transmission of this local request, then the local request is rescheduled by performing an exponential backoff. When a group member receives a request for a packet that it has already received, the group member schedules a *reply*; a retransmission of the requested packet. If a reply for the same packet is received prior to the transmission of this local reply, then the local reply is canceled. Using this scheme, all session members participate in the packet recovery process and share the associated overhead.

SRM minimizes duplicate error control and retransmission traffic through *deterministic* and *probabilistic* suppression. These suppression techniques prescribe how requests and replies should be scheduled so that only few requests and replies are transmitted for each loss. Deterministic suppression prescribes that request and reply scheduling timers be set proportionately to the distance from the source and the requestor, respectively. Thus, the requests of ancestors suppress those of their descendants. Probabilistic suppression prescribes that members that are equidistant from the source and the requestor probabilistically vary the scheduling times of their requests and replies, respectively. Thus, sibling requestor and replier hosts are afforded the opportunity to suppress each other. Unfortunately, suppression introduces a tradeoff between the number of duplicate requests and replies and the recovery latency — the scheduling of requests and replies must be delayed sufficiently so as to minimize the number of duplicate requests and replies.

SRM, as do other reliable multicast protocols, presupposes that packet losses are independent. However, in several studies of multicast communication, such as Bolot *et al*. [1], Yajnik *et al*. [5], and Handley [3], packet losses in multicast sessions were found to be nonindependent and to exhibit *spatial* and *temporal correlation* — spatial correlation refers to the correlation of packet losses across receivers, *i.e.*, the degree to which the losses are shared among receivers, and temporal correlation refers to the correlation of packet losses at each receiver, *i.e.*, the burstiness of packet losses.

3. Caching-Based Multicast Error Control

Our reliable multicast protocol is inspired by SRM. The distinction between SRM and our novel protocol lies in the scheduling of requests and replies. In particular, we adopt SRM's deterministic suppression scheme to achieve the suppression of descendant receivers by their ancestors within the underlying IP multicast spanning tree. In contrast, we replace the probabilistic suppression scheme for reducing the number of requests and replies generated by sibling hosts with a novel caching-based scheme. We illustrate this scheme by describing the scheduling of requests — the scheme for scheduling replies is analogous.

For simplicity, consider a simple reliable multicast session comprised of a single source, multiple receivers, and an underlying IP multicast spanning tree containing a single faulty link. Following the detection of the first packet loss, receivers schedule retransmission requests. Although, deterministic suppression achieves the suppression of all descendant receivers, several *orphan* sibling receivers (receivers for which no ancestors share the particular packet loss) may still compete for sending requests — this is particularly plausible when the underlying multicast tree is sparse. During the recovery of the first loss on the faulty link, all orphan receivers multicast their requests, which include a field containing the particular orphan receiver's RTT estimate to the source. Upon receiving these requests, each receiver can determine which of the orphan receivers was the most appropriate requestor in terms of the orphan receivers' RTTs to the source. Thus, in view of streamlining the recovery of future losses, the orphan receiver that is closest to the source self-appoints itself the *leader* and all other receivers that shared the loss self-appoint themselves *non-leaders*. Presuming that the next loss occurs on the same link, non-leaders schedule their requests for future losses at a point in time that follows the time they expect to receive the leader's request. A cache hit occurs when the next loss occurs on the same link. In this case, the leader's request suppresses all non-leaders' requests and, thus, a single request is sent. A cache miss occurs when the leader either receives the packet, or gets suppressed by one of its ancestors. In the former case, our protocol elects a new leader at a lower level of the underlying multicast tree. In the latter case, our protocol elects a new leader at a higher level of the underlying multicast tree.

More complex loss patterns, involving losses that occur on distinct links, are handled by using the above leader appointment scheme to build a hierarchy of leaders; each such leader being responsible for sending requests (and analogously replies) on behalf of all descendants of a particular faulty link. Once this leader hierarchy is in place, active leaders alternate to match the packet loss characteristics.

Assuming high packet loss locality, our scheme produces only a single request and a single reply for every loss except the first. The costs associated with our reliable multicast protocol include: i) the recovery latency incurred due to deterministic suppression, and ii) the overhead in terms of duplicate requests and replies in building and managing the leader hierarchy. The benefits of our caching-based scheme are that: i) only single requests and replies are transmitted following cache hits, and ii) recovery latency is reduced with respect to SRM due to the elimination of the additional recovery latency incurred due to probabilistic suppression.

4. Design and Analysis Approach

In contrast to traditional protocol design techniques, we use a rigorous design approach that is based on the *timed I/O automaton* specification model [4] and the associated correctness and performance reasoning techniques. The first step in this approach is to precisely specify the high-level reliable multicast service. These abstract specifications constitute the metric for showing that a reliable multicast protocol is correct. The next step involves specifying our caching-based reliable multicast protocol. The final steps in our approach are the correctness and performance analyses of the proposed protocols. Protocol correctness is ascertained by showing that the protocol implements the high-level reliable multicast service specifications. Protocol performance is analyzed by providing conditional guarantees as to the protocol's overhead and recovery latency and comparative performance claims with respect to existing reliable multicast protocols, such as SRM.

References

[1] J.-C. Bolot, H. Crépin, and A. Vega Garcia. Analysis of Audio Packet Loss in the Internet. In *Proc. NOSSDAV'95*, volume 1018 of *LNCS*, pages 154–165, Apr. 1995.

[2] S. Floyd, V. Jacobson, S. McCanne, C.-G. Liu, and L. Zhang. A Reliable Multicast Framework For Light-Weight Sessions And Application Level Framing. *IEEE/ACM Transactions on Networking*, 5(6):784–803, Dec. 1997.

[3] M. Handley. An Examination of MBone Performance. Technical Report RR-97-450, USC/ISI, Jan. 1997.

[4] N. A. Lynch and F. Vaandrager. Forward and Backward Simulations — Part II: Timing-Based Systems. *Information and Computation*, 128(1):1–25, July 1996.

[5] M. Yajnik, J. Kurose, and D. Towsley. Packet Loss Correlation in the MBone Multicast Network. In *Proc. IEEE/GLOBECOM'96*, pages 94–99, Nov. 1996.