Self-Adapting Epidemic Broadcast Algorithms*

L. Rodrigues
U. Lisboa
ler@di.fc.ul.pt

J. Pereira
U. Minho
jop@di.uminho.pt

July 19, 2004

Abstract

Epidemic broadcast algorithms have a number of characteristics, such as strong resilience to node failures, that make them an appealing technology to build survivable systems. However, the performance of existing protocols is highly dependent of run-time parameters, such as the network load or network topology. In the current extended abstract we advocate, using concrete illustrations, the use of self-adapting epidemic broadcast algorithms.

1 Introduction

Gossip-based broadcast algorithms [1, 2, 4, 9, 8, 6], also called "epidemic" or "probabilistic" broadcast algorithms, do have inherent scalability properties that make them suitable tools for disseminating information among a large number of nodes. Furthermore, they are extremely resilient to failures of participants and to changes in the underlying network topology. Therefore, they are quite appealing for building survivable systems.

However, the good performance of an epidemic broadcast protocol is not independent of the underlying infrastructure. In fact, issues such as available resources and topology may have a strong impact in properties such as reliability and latency.

Indeed, in order to operate in a reliable manner, the nodes participating in the broadcast must be equipped with enough resources to ensure that messages are gossiped a sufficient number of times. If a node does not have enough resources, it may drop a large number of messages that are being forwarded. If several nodes do not have enough resources, reliability might end up being drastically impacted.

On the other hand, to most recipients, delivery only happens after a message has been relayed a number of times by participant nodes. In consequence, although gossiping is highly resilient, it makes end-to-end latency be several times larger than the latency between any pair of nodes. It is possible to tune the gossiping strategy to minimize the average latency of a gossip protocol.

*This position paper was presented at the FuDiCo II: S.O.S. Survivability: Obstacles and Solutions 2nd Bertinoro Workshop on Future Directions in Distributed Computing, June 2004, Bertinoro (Forlì), Italy. The work was partially supported by LASIGE and FCT project RUMOR (POSI/40088/CHS/2001).

In a static and stable environment, it is possible to tune the protocol behavior *a priori*. For instance, one may limit the actual load imposed on the system by analyzing existing resources, or bias the probabilistic broadcast to avoid worst case paths. In a survivable system, such static configuration is not feasible.

In the current extended abstract we advocate the use of self-adapting epidemic broadcasts and provide an overview of some of our current research in this area. Namely, we concentrate on two complementary aspects:

- How to automatically control the load as a function of the available resources [12].
- How to automatically bias the gossip procedure to favor low latency [10].

The next two sections briefly describe these two adaptation mechanisms.

2 Adapting the Accepted Load

In order to operate in a reliable manner, the nodes participating in the epidemic broadcast must be equipped with enough resources to ensure that messages are gossiped a sufficient number of times. If a node does not have enough resources, it may drop a large number of messages that are being forwarded. If several nodes do not have enough resources, reliability might end up being drastically impacted.

A survivable system must cope with changes in the available resources, as the number of available nodes and links may change significantly in run-time.

We have recently proposed a novel adaptive mechanism for gossip-based broadcast algorithms [12]. The idea is to disseminate and gather information about the resources available in a broadcast group such that every sender can adjust its emission accordingly. The challenge consists in ensuring that senders are able to perceive the quality of the algorithm operation, in terms of reliability with the current system configuration, without interacting explicitly with other nodes of the system; such interaction would hamper the distinctive scalability of gossip-based broadcast algorithms.

The intuitive idea underlying our mechanism is the following. We periodically evaluate the available resources in every broadcast group. In each period, nodes gossip the minimum buffer size in the group for that period. They do so by maintaining and gossiping the minimum of their own buffer size with the value received in gossip messages for that period. The value computed at the end of a period is used as the estimation for that period and maintained for a predefined number of periods Δ. Then, during the normal operation of the gossiping protocol, each sender computes the average *age* of messages stored locally that would be discarded if the local buffer was the smallest in the group. The age of a message is the number of times it has been forwarded from one node to another and is directly related with the level of dissemination among nodes. If the average age of messages that would be discarded by members with low buffers is lower than the required age to ensure reliability, the sender decreases its transmission rate. If the average age of discarded messages is higher than needed, then the sender is allowed to increase its transmission rate.

This adaptation mechanism is highly scalable as it does not require nodes to maintain information about every other node in the system. It also does not require additional messages to be exchanged: it relies on a small amount of control information that is included in the

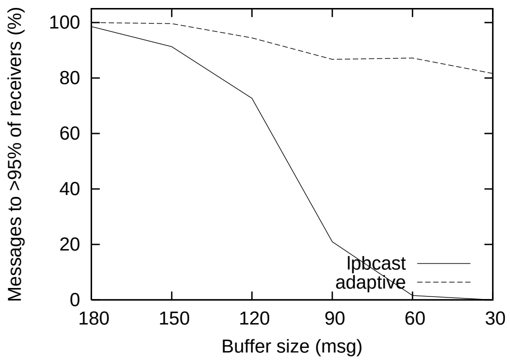

Figure 1: Improving reliability by adaptation.

header of normal data messages. The mechanism takes into account the dynamic nature of the system and continuously adapts to changing operational conditions. As conveyed by our performance measures, without such a mechanism, the reliability of message dissemination in a large scale gossip-based setting can hardly be sufficient in a practical setting.

Figure 1 shows the impact of the adaptation procedure on atomicity using simulation. The impact of adaptive mechanisms is evaluated against a non-adaptive epidemic protocols, namely the *lpbcast* protocols described in [2]. In detail, simulations with increasingly smaller buffer sizes are run while offering the same load. It can easily be observed that the number of messages delivered to more at least 95% of processes drops sharply with the original protocol. In contrast, the adaptive protocol controls the offered load and thus ensures that most messages are delivered to almost all processes. Other simulations show that the admitted load converges to the maximum load that results in high atomicity.

Unfortunately, the protocol described above is not fully self-configurable. There a number of parameters that need to be preset by the protocol designer, such as the increments/decrements in load in response to a change in the system. We are currently working on making these parameters self-adjusting.

3 Adapting the Gossip Procedure

In an epidemic broadcast protocol delivery only happens after a message has been relayed a number of times by participant nodes. If the message path consists of slow links and congested nodes, the latency of message delivery can be significantly higher than the latency of a non-gossip based broadcast.

The high latency of probabilistic broadcast is exacerbated in wide area area settings, where participant are connected by a variety of links. In such cases, the gossip path between the sender and a given recipient may include several non-optimal hops. Furthermore, in such settings, the combined effect of network omissions (which may require the protocol to be configured with additional gossip rounds) and congested links and nodes (resulting in queuing delays) introduce additional latency.

Based on the previous observations, there have been several different proposals to take advantage to network topology to improve the performance of gossip based protocols, including HiScamp [3], Directional Gossip [9], and others [5, 7, 13].

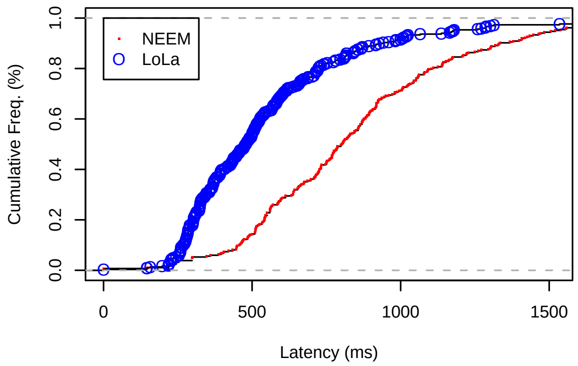

Figure 2: Reducing latency by adaptation.

Following these works, we advocate the design a self-configuring adaptive mechanisms to reduce the average end-to-end latency [10]. We propose to use a novel strategy that consists in adjusting the fanout and preferred targets for different gossip rounds as a function of the properties of each node. Node classification is light-weight and may be integrated in the protocol membership management. Therefore, each node is not required to have full knowledge on the group membership or on the network topology.

We have designed a new protocol based on the following idea: Average latency can be reduced by biasing gossiping to promote the usage of those paths with fewer delays. Our protocol, that we have named LoLa (Low-Latency Probabilistic Broadcast), reaches this goal by using an unique strategy that consists in adjusting both the fanout and the preferred targets for each gossip according to the properties of each node. Basically the protocol: *i)* favors gossip to nodes that result in lower latency (i.e. biasing gossiping), and; *ii)* gossips messages that have accumulated less latency towards a larger number of destinations (i.e. adjusting fanout).

The strategy above requires nodes and links to be classified according to their throughput and latency. To preserve scalability, such classification must not require global knowledge of the system. In our approach, node classification is light-weight (it relies only on local information and feedback from directly connected peers) and integrated in the protocol membership management. Local information is gathered mostly from TCP/IP connections used to connect to peers and, as a side effect, diagnose the availability of resources.

A detailed network simulation is used to evaluate the adaptation strategy by comparing it with the NEEM protcol [11]. The network is configured with a large number of hosts connected by V.90 modems (70%), some connected using ADSL (20%) and some directly attached to the backbone (10%), Figure 2 shows the distribution of latency of all message deliveries in the system. Notice that with the original NEEM protocol, only 20% of message deliveries have latency lower than 500 ms. In contrast, LoLa delivers almost 60% of messages in less that 500 ms and almost all messages in less than 750 ms. Simulation results show also that there is an increase of bandwidth usage of nodes where it is available.

4 Discussion

Our research has shown that self adapting mechanisms may strongly improve the performance of epidemic broadcast protocols. We have illustrated these benefits using two complementary aspects of adaptation: adaptation to changes in the system load and adaptation to changes in the network topology. This work has lead to improvements in reliability and latency of epidemic protocols in realistic settings. However, further research needs to be performed to assess the full implications of self-adapting strategies in survivable systems. A particular aspect of concern is that potential weaknesses in the adaption algorithm may be exploited by malicious attackers to cause instability in the system.

References

[1] K. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Transactions on Computer Systems*, 17(2):41–88, May 1999.

[2] P. Eugster, R. Guerraoui, S. Handrukande, A.-M. Kermarrec, and P. Kouznetsov. Lighweight probabilistic broadcast. In *Proceedings of IEEE Intl. Conf. on Dependable Systems and Networks (DSN'2001)*, 2001.

[3] A. Ganesh, A. Kermarrec, and L. Massoulie. Hiscamp: self-organising hierarchical membership protocol. In *10th European ACM SIGOPS workshop*, 2002.

[4] A.J. Ganesh, A.-M. Kermarrec, and L. Massoulie. Peer-to-peer membership management for gossip-based protocols. *IEEE Transactions on Computers*, Feb, 2003.

[5] I. Gupta, A.-M. Kermarrec, and A. Ganesh. Efficient epidemic-style protocols for reliable and scalable multicast. In *IEEE International Symposium on Reliable Distributed Systems (SRDS)*, 2002.

[6] I. Gupta, R. van Renesse, and K. Birman. Scalable fault-tolerant aggregation in large process groups. In *Intl. Conf. Networked Computing and Applications (NCA'2001)*, 2001.

[7] A. Kermarrec, L. Massoulie, and A. Ganesh. Probabilistic reliable dissemination in large-scale systems, 2001.

[8] M. Lin, K. Marzullo, and S. Masini. Gossip versus deterministic flooding: low-message overhead and high-reliability for broadcasting on small networks. In *Proceedings of Intl. Symposium on Distributed Computing (DISC 2000)*, Toledo, Spain, October 2000.

[9] M.-J. Lin and K. Marzullo. Directional gossip: Gossip in a wide area network. Technical Report CS1999-0622, University of California, San Diego, Computer Science and Engineering, June 16, 1999.

[10] J. Pereira, L. Rodrigues, and R. Oliveira. Low-latency probabilistic broadcast in wide area networks. Technical report, U. Uminho/U. Lisboa, 2004. (In preparation).

[11] J. Pereira, L. Rodrigues, R. Oliveira, and A.-M. Kermarrec. Neem: Network-friendly epidemic multicast. In *IEEE Symp. Reliable Distributed Systems (SRDS)*, 2003.

[12] L. Rodrigues, S.Handurukande, J. Pereira, R. Guerraoui, and A.-M. Kermarrec. Adaptive gossip-based broadcast. In *IEEE Intl. Conf. on Distributed Systems and Networks (DSN)*, 2003.

[13] R. van Renesse, Y. Minsky, and M. Hayden. A gossip-style failure detection service. Technical Report TR98-1687, 28, 1998.