Probabilistic Semantically Reliable Multicast*

José PEREIRA
Universidade do Minho
jop@di.uminho.pt

Luís RODRIGUES
Universidade de Lisboa
ler@di.fc.ul.pt

Rui OLIVEIRA
Universidade do Minho
rco@di.uminho.pt

Anne-Marie KERMARREC
Microsoft Research,Cambridge, UK
annemk@microsoft.com

Abstract

Traditional broadcast protocols fail to scale to large settings. Several recent attempts have proven efficient to ensure reliable information dissemination in groups composed of a large number of participants. This paper proposes a reliable multicast protocol that integrates two complementary approaches to deal with the large-scale dimension in group communication protocols: gossip-based probabilistic and semantic-based protocols.

Although it seems intuitive that the combination of probabilistic and semantic reliable protocols should provide good results, we show that a naive composition of both protocols offers disapointing results. We then propose a specialized probabilistic semantically reliable layer, evaluate different implementation policies and study the impact of relevant system parameters in the performance of the protocol.

Our approach allows us to identify which protocol configuration provides the best results, combining the advantages of both models in a single primitive that: *(i)* is scalable to

* An extended abstract of this report was published as a short paper in the *Proceedings of the IEEE International Symposium on Network Computing and Applications*, October, 8-10, 2001 Royal Sonesta Hotel Cambridge, MA, USA.

large number of participants and highly resilient to network and process failures and; *(ii)* delivers a high quality data flow even when the load exceeds the available bandwidth.

1 Introduction

Reliable group communication protocols are a fundamental building block to build complex distributed applications [2]. Many of the earlier group communication systems were targeted at small and medium scale fault-tolerant systems, with emphasis on strong primitives such as atomic broadcast [9].

However, providing stable high throughput to large groups while, at the same time, enforcing strong semantics is a very hard task [3]. For instance, protocols that provide sender reliability, such as RMTP [14], generate a large number of acknowledgments which must be received and processed by the sender. For large groups or high throughput this rapidly overwhelms the sender, a situation often referred as *ack implosion*.

Even with protocols that avoid such problems by using more sophisticated mechanisms to track message stability, non probabilistic protocols have to retransmit the messages until all recipients acknowledge their reception or are declared failed. Therefore, when the network is congested or a receiver is perturbed, messages accumulate in buffers eventually preventing further messages to be sent for some time. This means that a single slow receiver or network link slows down the entire group [18].

Two approaches have been proposed recently to address the scalability issues of reliable multicast protocols. One is the use of probabilistic epidemic protocols that support the efficient dissemination of data among a large number of nodes while providing probabilistic guarantee of delivery [4]. The other is the use of semantic knowledge to support higher sustained throughput in groups with heterogeneous performance behavior [16].

Gossip-based probabilistic multicast protocols [8, 10, 13, 15], support throughput stability even for very large groups, as the load required to ensure reliability is evenly spread across all members of the group. The usage of epidemic dissemination results also in high resilience to process and network failures. However, the probabilistic assumptions on which the delivery guarantee is built cannot be enforced if the input load exceeds the capacity of the network. For instance, perturbed processes might have to be excluded from the group due to message losses and exceeding network bandwidth might lead to generalized failure to meet delivery guarantees.

Semantically reliable multicast protocols exploit the application's semantics to improve the tolerance of slow group members or network links. This approach stems from the observation that in distributed applications messages often overwrite the content and purpose of other messages sent shortly before, therefore making them obsolete while still in transit. By not delivering obsolete messages to perturbed members, semantically reliable multicast accommodates members with different throughputs in the same group without endangering the application's correctness. The advantages of semantically reliable multicast have been demonstrated in the context of both information delivery [16] and strongly consistent replication [17].

This paper proposes a protocol that combines these two approaches in a communication primitive that is highly resilient to environmental adversity and that delivers a high quality data flow even when some messages are lost. Although it seems intuitive that such combination should provide good results, we show that a naive compositing of both protocols offers disapointing results. We then propose the use of a specialized probabilistic semantically reliable protocol. We show how configuration parameters, such as buffer size and purging policies affect the final performance of the protocol. Finally, the paper identifies the configuration that offers better performance, in particular, it provides a better quality of delivery in overloaded networks without negatively impacting performance in lightly loaded networks.

The paper is organized as follows. Section 2 provides a brief introduction to the probabilistic and semantically reliable multicast approaches when used in isolation and underlines the interest of combining the two approaches. Different alternatives to obtain a combined protocol are described in Section 3. Section 4 evaluates these alternatives and identifies the one that offers better results. Finally, Section 5 concludes the paper.

2 Motivations

For self-containment, and to motivate our approach, in the following two sections we provide a brief introduction to previous work in the area of probabilistic multicast protocols and semantically reliable multicast. We then discuss the challenges of combining the two approaches.

2.1 Probabilistic broadcast

In probabilistic reliable protocols [10, 13] messages are not disseminated in a deterministic manner. Instead, *(i)* each participant relays each message received to a random subset of processes and; *(ii)* each message is relayed at most a bounded number of times. It can be shown that these protocols can be configured such that each message is delivered to none or all processes with a high probability. The probability of a message being delivered to some but not all processes can be made as small as required by adjusting the protocol's parameters, therefore providing as much reliability as any deterministic approach. The decentralized nature of epidemic dissemination results in a protocol that is scalable to a large number of participants without overloading any particular member of the group (as happens with deterministic protocols and ack implosion).

In addition to their performance in the large scale, gossip-based protocols are highly resilient both to process failures and to independent packet losses in the network. However, these protocols do not tolerate correlated message losses, such as resulting from network congestion. Unless there is a fixed set of bounded rate senders, the maximum input rate has to be set conservatively thus preventing full network usage. This is a potential weakness of the approach since with probabilistic protocols the network usage is notably high in comparison with deterministic reliable broadcast protocols.

Several optimizations of the original protocol have been proposed. Recognizing that the probability of the message being delivered to all processes grows with the size of the initial set of processes receiving the message, it has been proposed that an initial optimistic broadcast phase is used [4]. The epidemic phase is then done by negative acknowledgment thus saving network bandwidth. It has also been shown that a global deterministic view of the participant set is not required [8]. Instead, it is sufficient that each participant keeps a random subset of the entire participant set. Finally, given knowledge about network topology it is possible to better spread network traffic across physical links, thus better coping with wide area networks [15].

2.2 Semantic reliability

Deterministic protocols have to retransmit the messages until all recipients acknowledge their reception or are declared failed. When the network is congested or a receiver is perturbed, messages accumulate in buffers until fully acknowledged. Since buffers are bounded, when buffers fill up the sender is not allowed to send further messages. This means that a single slow receiver or network link may slow down the entire group.

Semantic reliability stems from the observation that in distributed applications many messages are made obsolete by messages sent shortly after. As such, when messages accumulate in buffers it is likely that already obsolete messages are in the same buffer as the messages that make them obsolete. If obsolete messages are purged, the resulting buffer space can be used ensuring that the sender is never blocked and thus fast receivers are not affected. This allows receivers with different throughputs to be accommodated within the same group: Fast receivers get all the messages with low latency and slower receivers get less messages with higher latency.

The performance of semantic reliability depends both on the obsolescence profile of the traffic and on the size of buffers available for purging. A simple analytical model that enables reasoning about the efficiency of the protocol and the configuration of system parameters according to the obsolescence function of the target application has been proposed in [16]. This model was validated through simulation. It was shown that when applied to a traffic profile of an on-line trading system, the protocol can be configured to allow a receiver to exhibit processing delays 40% higher than those required to process all messages in due time without disturbing the sender. An equivalent model was also proposed to reason about the advantages of the approach to support strongly consistent replication [17].

2.3 Challenges in combining both approaches

Intuitively the combination of the two previous approaches is appealing. However, to fulfill its potential, the combination needs to be implemented by a concrete protocol. The straighforward manner of obtaining a probabilistic semantic protocol would be to apply semantic purging to the buffers maintained internally by the probabilistic protocol. Unfortunately, obtaining an efficient protocol from such architecture is harder than it looks at first sight. To start with, some probabilitic protocols, such as pbcast [10], do not rely on buffering and that prevents semantic purging from being applied. Thereby the use of purging would be restricted to protocols that use buffering such as lpbcast [8]. Furthermore, even in this case it is hard to balance the buffering requirements of the semantic reliability with the latency requirements of the protocol. In fact, as it will be shown in the evaluation section a static configuration of protocol parameters does not yield satisfactory results.

The intuitive reason for the poor performance of such a naive combination of both protocols is the following: Consider that the gossip protocol is statically configured to use small buffers. In such case it is hard to apply obsolescence relations since it unlikely that related messages can be found simultaneously in the buffer. On the other hand if, to increase the chances of applying semantic purging, the protocol is statically configured to use large buffers then the following disadvantages may be observed: excessive purging occurs even when there is enough network bandwidth to disseminate all the messages; the latency of message dissemination increases since messages are buffered for longer periods at each hop and; the probability of overloading the network when the buffers are flushed is higher, which may cause correlated losses at the network level. Integrating straightforwardly semantic buffering management into gossip-based protocols does not enable the system to react efficiently to changes in the network load. Therefore, it is necessary to design a solution that allows purging to be applyed as a function of the network usage.

Note that the purpose of applying semantic purging as described in this paper differs significantly from its previous application to deterministic protocols [16]. In those protocols, purging is applied to isolate fast receivers from performance degradation caused by slow receivers. With probabilistic protocols, fast receivers are already unaffected by perturbed receivers. The goal is to be able to exceed available network capacity without endangering application correctness and reliability guarantees.

Note also that the resulting architecture resembles a system where messages have different priorities and lower priority messages may be dropped to allow the dissemination of high priority messages. There is however a significant difference: In a priority based system, the importance of each message must be known a priory.[1] With semantic reliability the importance of each message is defined dynamically in function of the transmission (or not) of other messages that make it obsolete.

Naturally, benefits from the combined approach can only be achieved if the traffic pattern exhibits some amount of obsolescence. In the next section we discuss a number of applications that can benefit from the new protocol.

2.4 Targeted applications

In publish-subscribe systems [1, 5, 6], subscribers register their interest in an event or a pattern of events in order to be subsequently notified of any event published which matches their interest regardless of how and by whom that information is produced. Many variants exist and differ mainly by their ability to filter events [7].

Message dissemination may take the subscription pattern into account. For instance, in content-based publish-subscribe systems the structure of the message can be inspected by the protocol in order to route messages from publishers to subscribers in situations where interest is sparse, thus avoiding flooding the network. However, in situations where interest in events is dense flooding cannot be avoided. In this situation knowledge of message contents by the protocol can be used to improve flooding itself by using a semantically reliable protocol. This makes the protocol described in this paper ideal for publish subscribe applications for very large numbers of publishers and subscribers with uniformly distributed interest. This fulfills the first requirement for usage of a probabilistic semantically reliable protocol: Message semantics is

[1] In real-time systems, the priority may be derived from the deadline, which must be known when the message is sent.

already available at the protocol level.

In addition, applications supported by such publish-subscribe systems are likely to also satisfy the second requirement: Exhibiting message obsolescence. For example, a stock-based publish-subscriber stock-market application would benefit from message obsolescence. The semantic obsolescence may be carried by the absolute time a message had been sent. Any message carrying a more recent value (either absolute, either from the same publisher) would obsolete any older message.

Likewise, a large-scale auction system hosting by a publish-subscribe system could implement a very natural obsolescence semantic. Message carrying a bid higher than that of any other message would obsolete the latter one. In this case, the semantic information is not computed locally but is carried by the message itself, the bid absolute value as the parameter to define obsolescence relations. Some other causal criteria might be useful in this context: if p publishes m, m may obsolete all messages received by p before firing m. Again, the semantic knowledge would be carried by the messages themselves.

3 Probabilistic semantically reliable multicast

The architecture proposed in this section stems from the observation that probabilistic broadcast protocols do not tolerate network congestion as it induces correlated loss. This can be avoided by some sort of flow control mechanism that temporarily buffers messages, allowing the system to cope with short load peaks. If the system is subject to a sustained high load, it is likely that messages have to be buffered long enough so that obsolete messages can be recognized and subsequently purged. Resources saved by purging obsolete messages can then be applied to ensure that the delivery of non-obsolete messages is done. Therefore, despite not delivering all messages, the protocol selectively tries to deliver those messages that are required for consistent operation. On the other hand, when the network is not overloaded messages do not need to be buffered and the dissemination latency is not affected by the introduction of semantic purging.

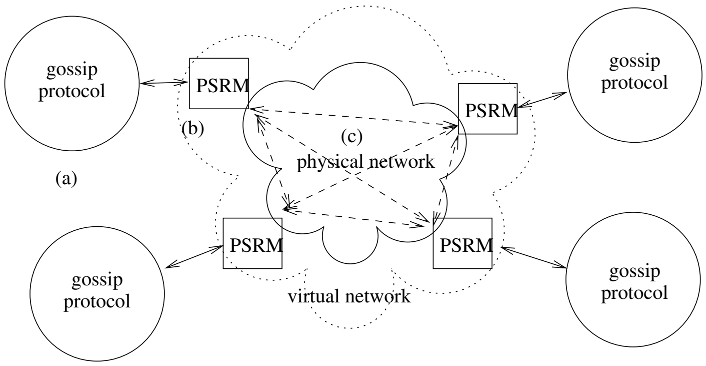


Figure 1: Three-tiered architecture: (a) Gossip protocol; (b) PSRM; (c) network.

3.1 Architecture based on a specialized PSRM

The purpose of our architecture is to separate the mechanism exploiting message semantics both from the probabilistic protocol and from the physical network. It consists of inserting a specialized probabilitic semantically reliable multicast layer in between the gossip protocol and the network. The layer takes into account the available bandwith at each link and peforms buffering only to avoid overloading the network. Figure 1 presents a graphical overview of the proposed architecture: The lower tier (c) is the physical network. The middle tier (b) is a buffering layer, where any purging mechanism may be applied. The top tier (a) is an unmodified gossip protocol.

Lower tier: Network

The lower tier is the physical network which, by controlling admission of messages, provides bounded independently distributed message loss among each pair of participating processes. This ensures that, when the system is overloaded, message losses do not happen inside the network but at the edges under control of a purging policy that minimizes their consequences. The concrete congestion control mechanism used depends on the characteristics of the physical network and is out of the scope of this paper. Possible options are a window-based congestion control mechanism compatible with TCP/IP [11] and rate-based congestion control in ATM networks [12].

Since probabilistic broadcast protocols naturally spread the load evenly across node pairs, a static partition of available bandwidth achieves this goal satisfactorily (i.e. fully uses the network) when using a shared network with predictable bandwidth. We used this approach to obtain the experimental results presented in this paper.

Middle tier: the specialized PSRM layer

The middle tier is our specialized probabilitic semantically reliable multicast layer: it buffers messages when the capacity of the network is exceeded and, in this case, it selectively drops messages in order to improve the quality of service. This makes it the key component in the proposed architecture.

For short bursts of incoming messages, buffering alone is enough to spread the load in time and thus to avoid message losses. However, for continued loads, buffers are eventually exhausted. Therefore, there are different ways of configuring the protocol by using different purging policies, as enumerated below:

1. Semantic purging discards messages that have been made obsolete to accommodate each newly arrived message. This optimizes system usage by forcing correlated loss of obsolete messages and avoiding loss of non-obsolete messages. This technique exploits the semantic of the application to increase the fairness of the purging mechanism.

2. Random selection of messages in the buffer to make up room for newly arrived messages is the next possible mechanism. This ensures proper operation of the probabilistic protocol within the bounds of system capacity due to the intrinsic redundancy of gossip protocols [8]. Note that it does not represent the fairest criterion. However, it favors most recent (and less likely to be obsolete) messages.

3. New messages are simply discarded while buffer space is not available. This approach makes little use of the intrinsic redundancy of the probabilistic protocols and is acceptable only during the first transmission as it otherwise leads to correlated message loss (i.e. lossage of entire rounds) and failure of the reliability guarantees of the protocol.

In this paper we make use of the first two purging methods, naturally favoring semantic purging. The resulting performance depends on how exactly semantic purging is performed and on buffer configuration. Regarding the use of semantic purging, we consider two policies:

- Eager semantic purging. In this approach, the reception of every message triggers semantic purging and every obsolete message, if any, is immediately removed from the buffers.
- Lazy semantic purging. In this approach, purging occurs only when the buffer is full and a new message arrives.

The size of buffers affects the performance of the system because if buffers are too short, message losses will occur event for light traffic loads and the reduced number of messages available diminishes the probability of detecting obsolescence relations. On the other hand, if buffers are appropriately configured, messages losses only occur at high loads and, in that case, it is possible to find older and newer messages in the same node and thus detect and purge messages that are already obsolete.

Top tier: Gossip protocol

The top tier is a standard gossip-based broadcast protocol. Each message broadcast or locally received is forwarded to a subset of processes randomly chosen. Important parameters are the cardinality of this subset (also called the fan-out) and the maximum number of times each message is relayed (called the number of rounds). The configuration of such parameters is out of the scope of this paper and has been described in [10, 13].

4 Configuring PSRM

This section evaluates the different architectures and configurations parameters. We start by introducing the evaluation criteria and the simulation model. Later, we present simulation results from the different configurations and finally we identify the one that provides the best results.

4.1 Evaluation criteria

The evaluation of the proposed system in the face of different application traffic profiles and purging configuration parameters requires the definition of a suitable performance metric. As noted before, the combination of semantic reliability with probabilistic protocols does not aim at reducing the number of message losses when the network is congested. If the load exceeds the network capacity either the source is controlled or a number of message drops will inevitably occur. The purpose of our protocol is to create the conditions to drop more obsolete information than up-to-date information, thus improving the quality of the information provided to the users according to the semantics of the application.

Consider for instance the case of an on-line auction system where messages carry different bids on a given item. If a user does not receive a bid that is made obsolete by another (higher) bid, but receives this higher bid she can still participate in the auction. On the other hand, if she does not receive the (last) highest bid, she may be not aware that the item is being taken by another bidder.

A good metric of the quality of the information delivered to the users in a system where a degree of message obsolescence exists is to count the loss of messages that never become obsolete. This is an interesting metric in the sense that messages that don't ever become obsolete are the absolute minimum messages that have to be reliably delivered in order to ensure consistency.

In short, the primary evaluation criterion is that non-obsolete messages are atomically delivered according to the probabilistic protocols metric (i.e. either to almost all or to almost none recipients) and that those messages are given higher priority than obsolete messages (i.e. if the total amount of non obsolete messages is less than system capacity then all non-obsolete messages are delivered).

Secondary criteria are reducing latency when the system is not overloaded (i.e. messages are not arbitrarily delayed in order to allow purging) and that system configuration is robust (i.e. performance is stable despite variation of traffic and system parameters).

Notice that the fulfillment of the latency criterion depends on buffer occupancy reflecting system usage. When the system is not overloaded, optimal latency results are obtained if buffer occupancy is residual. When the system becomes overloaded, optimal purging is obtained if buffers rapidly fill up.

4.2 Simulation model

Due to the complexity of the behavior of the target algorithms, it is difficult to derive which of the previous approaches performs better. Therefore we have resorted to simulation to get insight on the performance of the different alternatives.

In order to evaluate our protocol architecture we have used a simple event-based simulation model. The system is composed of a set of processes fully connected by a point-to-point network. For the results described in this paper, we have used the following parameters:

- 50 nodes attached to the network, processing messages with no overhead;
- a total of 10Mbps network bandwidth equally divided among all point-to-point connections;
- all messages are assumed to be 100 byte long;
- there is a fixed $100\mu s$ latency for each message, attributable to the protocol stack;
- it is assumed that the congestion control mechanism used within the physical network introduces 5% uniformly distributed messages losses.

For the probabilistic broadcast protocol, the following parameters have been used: a fanout of 5 and maximum of 4 rounds per message. All measurements have been done by running the simulation for 40s of simulated time, discarding data related to messages sent in the initial and final 10s, in order to obtain a stationary state as verified by observing individual simulations.

The traffic is characterized by an obsolescence relation. A pair of messages is in the relation if one message is made obsolete by the other. The relation that is generated is composed of a fixed number of concurrent chains with fixed length and is described by three parameters:

- Ratio r of related messages, which is the share of messages that participate in the relation. The rest of the traffic is never obsolete and never makes other messages obsolete.

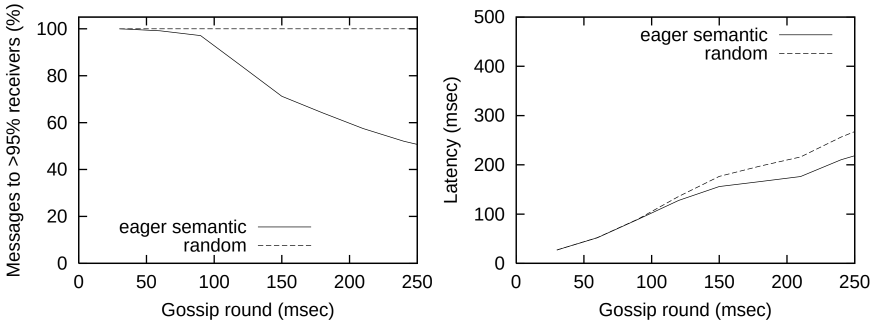
- Traffic diversity d. This is the number of non-related chains of obsolete messages that are generated concurrently.
- Chain length l. This is the length of each obsolescence chain generated.

Although the primary purpose of such traffic model is to explore how the protocol reacts to different applications characteristics, real applications can be mapped into it. As an example, the distributed auction application can be mapped as follows: The number of concurrent items being auctioned at any given time gives the traffic diversity d and the number of bids for each item gives parameter l. As not all items are traded with equal frequency and thus messages containing bids regarding infrequent items are unlikely to ever become obsolete, some amount of traffic can be reserved as never becoming obsolete using parameter r.

The default parameters used for simulations shown below are a buffer size of 10 messages per connection, a traffic diversity of $d = 1$ and a chain length of $l = 5$. These provide a large default high purging rate which is used as a baseline to evaluate the impact of each parameter. In addition, the resulting default high purging rates are consistent with what is observed for applications generating high message throughput, even in applications with high traffic diversity [16].

Atomicity graphics were obtained by running the simulation for each input rate and observing to how many processes each non-obsolete message is delivered. Variability is measured by the standard deviation of the number of processes to which each non-obsolete is delivered (normalized by the mean in order for different measurements to be comparable). Latency was measured at a single process and only for delivered messages.

Maximum throughput graphics are calculated by binary searching (running a simulation for each combination of parameters of interest) of the input rate that causes the system to break the atomicity criterion. An error is introduced by the size of the interval used to stop the algorithm which is of 10 msg/sec.



(a) Atomicity. (b) Latency.

Figure 2: Increasing the gossip period, as required for purging, negatively impacts system performace when the system is lightly loaded. With a fixed load of 10 msg/sec which could be handled without purging, (a) messages delivered to more than 95% of recipients decrease and (d) latency increases.

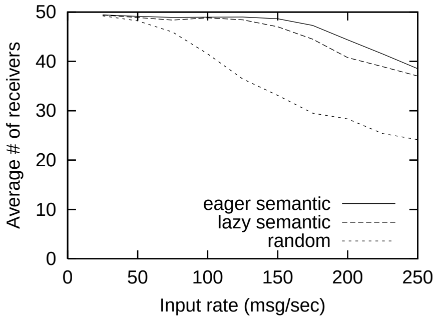
4.3 Simulation results of the naive protocol

With this protocol, it has been observed that only an eager semantic purging strategy is effective when the network is congested and only when the parameters of the gossip protocol are adjusted. Specifically, the lenght of the gossip round was increased in order to have enough buffer occupancy for semantic purging to be possible.

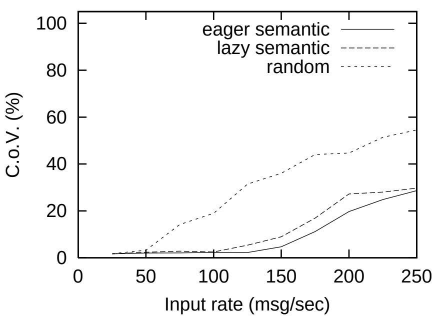
However, this results in degradation of performance when the system is lightly loaded as depicted in Figure 2, both in terms of messages discarded when there would be enough network capacity to transmit them (Figure 2a) as well as in the latency introduced (Figure 2b). This negative impact in the performance of lightly loaded systems rules out the usage of the simple combination of semantic purging with a gossip protocol.

4.4 Simulation results of PSRM

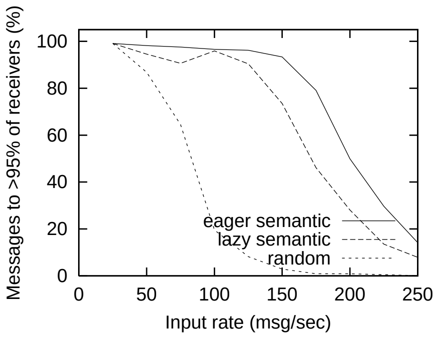
The evaluation of the proposed architecture focuses on measuring observed performance improvements as well as on determining appropriate configuration.



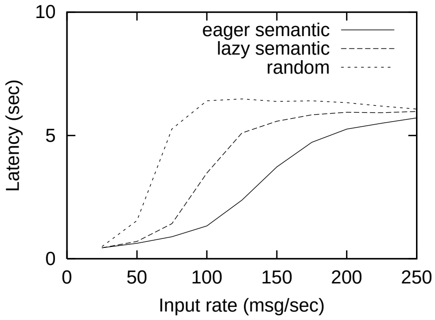
(a) Average receivers.



(b) Variability.



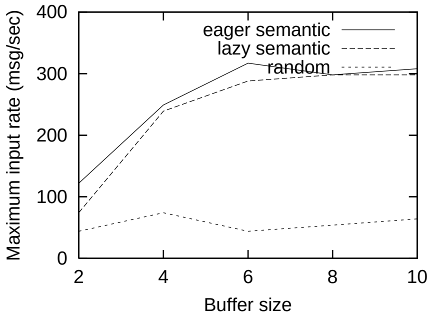
(c) Atomicity.



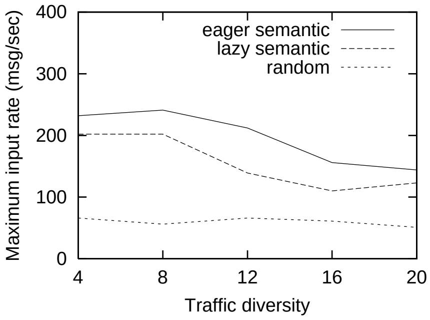
(d) Latency.

Figure 3: Measurements of throughput and latency: (a) Average receivers of non-obsolete messages; (b) standard deviation of (a) normalized by the mean (coeficient of variation); (c) number (%) of non-obsolete messages delivered to more than 95% of recipients; (d) latency.

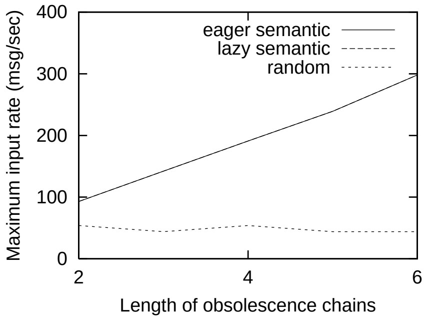
Reliability guarantees Figure 3a presents the average receivers of each non-obsolete message for different message input rates and purging strategies. Note that for low message rates any purging strategy is suitable, as messages are in average delivered to all processes, with low variance as shown in Figure 3b. However this graphic is somewhat misleading. For an input rate of 250 msg/sec the random purging method is not “half as good” as semantic purging as the graphic shows. This reason for this is that “half of the atomicity” is not useful. A better metric is presented in Figure 3c, which counts the number of messages which were delivered to more than 95% of processes and shows how atomicity rapidly degrades when system capacity is exceeded and how much semantic purging can improve the situation over random purging alone. For instance, when the input rate is 100 msg/sec, an eager purging strategy provides intact atomicity



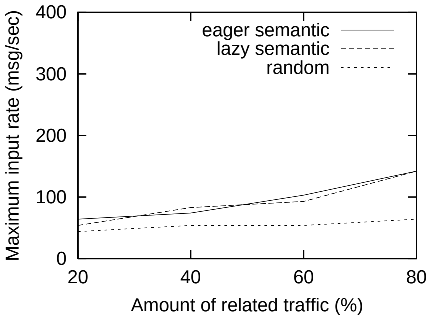
(a) Buffer size.



(b) Traffic diversity



(c) Obsolescence chain length.



(d) Amount of related traffic.

Figure 4: Impact of buffer size and traffic profile in maximum throughput.

of non-obsolete messages, while a random purging criterion results in only 20% of the same messages being correctly delivered.

Latency A secondary evaluation criterion is the latency introduced by buffering required for purging. As shown in Figure 3d, this latency is introduced only when the system is congested. It can also be observed that an eager purging strategy results in better latency by eliminating transmission of as much obsolete messages as possible.

Impact of buffer size Figure 4a illustrates the impact of varying buffer size available at each node. We observe that there is a minimal buffer size that enables optimum throughput. Further increasing the buffer size presents no advantage for purging and would negatively impact latency.

Impact of traffic parameters Finally, it is important to determine how evaluation criteria are affected by the characteristics of the traffic. Figure 4b illustrates how the purging rate is affected by traffic diversity d when using a reduced buffer size of 3 messages per connection. Notice how increasing traffic diversity for a fixed buffer configuration decreases the purging performance. However, performance degradation is not sudden, ensuring that a small change in traffic characteristics does not result in a drastic change in the system output.

Figure 4c presents the impact of increasing sequence lengths which cause increasing maximum achievable throughput. As this reduces the number of messages that never become obsolete it proportionally increases the amount of messages that can be purged (therefore, the system performance improves with the length of the obsolescence chains). Finally, Figure 4d presents the impact of sharing the channel with traffic that does not ever become obsolete, which naturally reduces the amount of traffic that can be purged (therefore, the system performs better with a larger amount of related traffic).

4.5 Selecting the best configuration

Our simulation results confirm that the naive combination of both protocols does not provide satisfactory results. Specifically, purging can only be achieved at the expense of degrading latency even when the system is not loaded.

On the other hand, our three tier architecture only increases latency when the input load exceeds the available network capacity. We have compared different purging strategies, namely eager and lazy purging with a pure gossip protocol without semantic purging, and concluded that eager purging represents a significative advantage in all performance metrics considered.

We have also shown that this configuration of the PSRM layer is robust in face of changing traffic diversity, which makes deployment easier due to less strict configuration requirements.

5 Conclusions

Recently, gossip-based and semantically reliable protocols have independently emerged as appropriate solutions to cope with large number of participants in group communications. The scalability of gossip-based protocols rely on a peer-to-peer interaction model and they provide probabilistic guarantee of delivery. In semantic reliability, semantics of the application is used to implement an efficient purging mechanism as well as a network congestion control mechanism.

In this paper we have proposed to combine them. Using semantic knowledge in gossip-based broadcast protocols enables to increase the quality of the information delivered to the users in the cases the load exceeds the available bandwith for applications exhibiting obsolescence patterns. The paper presented and evaluated different strategies to combine the two protocols. We have shown that a three tier architecture, that uses a specialized PSRM layer that intercepts the interactions between the gossip protocol and the network can provide good results if configured appropriately. The best configuration uses a congestion control mechanism to prevent correlated loss inside the network and eager semantic purging on each of the resulting buffers. We have shown that the quality of the delivery, measured as the percentage on non-obsoleted information that is delivered to more than 95% of the intended recipients, is better for such configuration than for other approaches, and much better than relying on random drop upon network congestion.

Obviously, the efficiency of the protocol depends of the obsolescence patterns of applications, but even with short chains improvements can be obtained. However, when appropriate, the combined protocol performs better that a standard gossip-based protocol both in congested networks and under normal circumstances. Especially, it enables to extend the applicability of gossip-based protocols to congested networks whereas such configurations are hardly addressed in the design of gossip-based protocols.

References

[1] G. Banavar, T. Chandra, B. Mukherjee, R. Strom J. Nagarajarao, and D. Sturman. An efficient multicast protocol for content-based publish-subscribe systems. In *The 19th IEEE International Conference on Distributed Computing Systems (ICDCS '99)*, 1999.

[2] K. Birman. The process group approach to reliable distributed computing. *Communications of the ACM*, 1993.

[3] K. Birman. A review of experiences with reliable multicast. *Software Practice and Experience*, 29(9):741–774, July 1999.

[4] 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.

[5] A. Carzaniga, D. Rosenblum, and A. Wolf. Content-based addressing and routing: A general model and its application. Technical Report CU-CS-902-00, Department of Computer Science, University of Colorado, January 2000.

[6] Talarian Corporation. Topic-based publish/subscribe though the well-known socket abstraction, see ”Everything you need to know anout Middleware: Mission-Critical Interprocess Communication (White paper). White Paper, 1999. http://www.talarian.com.

[7] P. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec. The many faces of publish/subscribe. Technical Report DSC ID:2000104, EPF Lausanne, 2001.

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

[9] V. Hadzilacos and S. Toueg. Fault-tolerant broadcasts and related problems. In Sape Mullender, editor, *Distributed Systems*, chapter 5, pages 97–145. Addison Wesley, 1993.

[10] M. Hayden and K. Birman. Probabilistic broadcast. Technical Report TR96-1606, Cornell University, Computer Science, 1996.

[11] V. Jacobson. Congestion avoidance and control. *ACM Computer Communication Review; Proceedings of the Sigcomm '88 Symposium in Stanford, CA, August, 1988*, 18(4):314–329, 1988.

[12] R. Jain. Congestion control and traffic management in ATM networks: Recent advances and A survey. *Computer Networks and ISDN Systems*, 28(13):1723–1738, October 1996.

[13] A.-M. Kermarrec, L. Masssoulié, and A. Ganesh. Reliable probabilistic communication in large-scale information dissemination systems. Technical Report 2000-105, Microsoft Research, 2000.

[14] J. Lin and S. Paul. RMTP: A reliable multicast transport protocol. In *Proceedings of IEEE INFOCOM '96*, pages 1414–1424, 1996.

[15] 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.

[16] J. Pereira, L. Rodrigues, and R. Oliveira. Semantically reliable multicast protocols. In *Proceedings of the Nineteenth IEEE Symposium on Reliable Distributed Systems*, pages 60–69, October 2000.

[17] J. Pereira, L. Rodrigues, and R. Oliveira. Enforcing strong consistency with semantically reliable multicast. Technical report, FCUL and UM, 2001. (submitted for publication).

[18] R. Piantoni and C. Stancescu. Implementing the Swiss Exchange Trading System. In *Proceedings of The Twenty-Seventh Annual International Symposium on Fault-Tolerant Computing (FTCS'97)*, pages 309–313. IEEE, June 1997.