Achieving Reliable and Timely Event Dissemination over WAN

C. Esposito[1], S. Russo[1], R. Beraldi[2], M. Platania[2], R. Baldoni[2]

[1] Dipartimento di Informatica e Sistemistica (DIS)
Università di Napoli Federico II, Napoli, 80125 - Italy
[2] Dipartimento di Informatica e Sistemistica (DIS)
Universitá degli studi di Roma "La Sapienza", Roma, 00185 - Italy
{christian.esposito, sterusso}@unina.it - {beraldi, platania, baldoni}@dis.uniroma.it

Abstract. The design of large-scale critical infrastructures demands for innovative data dissemination services, able to jointly provide reliability and timeliness guarantees. Current middleware solutions do not address both these aspects. Indeed, fault tolerance is typically achieved at the cost of severe performance fluctuations, or timeliness is always obtained by softening the fault-tolerance requirements. In this paper we propose to fulfill this lack by combining two different approaches, namely coding and gossiping. We provide a theoretical model to evaluate the potential benefit of coding on the information delivery performance. These results are also confirmed by an experimental analysis conducted on a real air traffic control workload, which evidences how coding mitigates latency and overhead penalties to ensure reliable event notification.

Key words: Publish/Subscribe Middleware, Network Coding, Gossiping

1 Introduction

Publish/Subscribe services are facing a growing interest in designing innovative critical infrastructures, named *Large-scale Complex Critical Infrastructures* (LCCIs), which consist of federating geographically-distributed systems by means of wide-area networks [1]. The reason behind this interest is the intrinsic decoupling properties offered by such middleware, which are suitable for satisfying the scalability requirements exhibited by large-scale infrastructures. A concrete example is represented by the novel Air Traffic Management (ATM) framework under implementation in Europe by EUROControl, within the "*Single European Sky ATM Research*" (SESAR) European project, where the information backbone for enabling Internet-scale data sharing is constituted by a publish/subscribe service compliant with the recent OMG standard called *Data Distribution Service* (DDS). The SESAR architecture consists of legacy and geographically sparse ATM systems, each one implementing domain specific functionalities, federated as a *System of Systems* (SoS) for exchanging data by means of a middleware built on top of DDS, namely *System Wide Information Management* (SWIM).

Information sharing is conveyed by wide-area channels due to the geographical extension of the SoS. As such, the communication may be affected by the

unpredictable behavior of the network, where messages can be dropped or delayed by possible link failures or congestions. However, due to their critical nature, LCCIs require that all exchanged information has to be *reliably* and *timely* delivered by all intended destinations. In fact, for this kind of infrastructures, a message loss or a late delivery can compromise the mission of the overall system, leading to possible wrong decisions. As an example, consider the data about flight plans exchanged between several flight processors in the Air Traffic Control (ATC) scenario. This data contains information about the trajectory and the coordinates of an aircraft *en route*; if such a message is lost or delayed, a flight processor may not be able to infer the current position of an aircraft and to detect possible collisions. So, publish/subscribe services can be truly the winning solution for the novel generation of critical infrastructure only if they are able to provide both reliable and timely communication. Most of the research on publish/subscribe services only focuses on techniques to provide fault-tolerance by means of retransmissions, and any reliability improvement is gained at the expenses of severe performance fluctuations.

The work presented in this paper aims to fill this gap by proposing a strategy to achieve both reliability and timeliness [2] in event dissemination performed by publish/subscribe services over the Internet. To this end, we combine network coding [3] and gossiping [4], each known to be timely and reliable respectively, so to achieve the best from both. We consider a scenario where a source sends coded information to a set of destinations over Internet links that exhibit a non negligible probability to have bursty losses. Then, destinations apply a gossiping strategy to recover from possible lost messages.

The remainder of the paper is organized as follows: Section 2 discusses reliability and coding techniques present in literature; Section 3 provides a theoretical model to evaluate the potential benefit of coding on the information delivery performance, even when the sender introduces a redundancy to improve reliability. In Section 4, an experimental analysis conducted on a real workload, taken from the previously described ATC scenario, evidences how the use of coding is able to reduce the delay for a reliable delivery. Finally, Section 5 concludes the paper.

2 Related Work

The QoS in publish/subscribe systems has been poorly addressed, giving more attention to subscription expressiveness and scalable notification architectures. In particular, timeliness and reliability have been considered just as separate aspects, sometimes resulting in conflict between them, making current solutions not suitable for LCCIs. As a concrete example, reliability is often ensured by means of retransmissions, i.e., by means of the TCP links or adopting gossiping algorithms, such as [5] [6] [7]. Retransmissions implies that the time needed to deliver a notification becomes unpredictable and highly fluctuating, so to violate timeliness, since the number of retransmissions needed to deliver a notification depends on the network conditions. In the literature of reliable communications, there is a different approach than using retransmission, named Network Coding [3]. Specifically, with network coding, a node can send linear combinations of previously received information: a message is encoded into packets that generally convey information about several original packets. This technique provides

potential throughput improvement and a high degree of robustness. To this end, it has been widely applied to data dissemination in large scale systems; several works have shown improvements in the expected file download time in content distribution networks [8], robustness to network changes or link failures [9] and resiliency against random packet loss and delay [10].

Starting from the considerations made in our previous work [11], where we have introduced how network coding works and theoretically shown its benefit on the information delivery performance, in this paper we investigate how network coding and gossiping can ensure both timeliness and reliability in publish/subscribe systems. Specifically, we have improved a best-effort topic-based publish/subscribe by (i) introducing the possibility of applying coding at the publisher, (ii) using a gossiping strategy to recover any lost data at each destination, and (iii) integrating coding within the retransmissions used by gossiping.

The most similar work to ours is [12], where coding is combined with gossiping at receiver side: coding is used only when gossipers have to retransmit in push mode their received data. We considerably differ from this work in the following ways: (i) we apply coding also at the multicaster and study the relation of such redundancy with the one applied by the gossipers, (ii) we investigate more than one gossiping approach and the improvement that coding can bring to them, and (iii) we evaluate the effects of coding not only on the delivery performance but also for the imposed overhead and in different network conditions. Another similar work is presented in [13], which differs from ours due to its theoretical nature (i.e., dissemination approaches are only studied by mean of analytical models), and its evaluation metrics (i.e., gossiping with and without coding are studied only with respect to performance). In addition, it uses only gossiping with coding applied, without the use of a prior tree-based dissemination within which we also apply coding.

3 Protocol analysis under ideal conditions

We consider a publish/subscribe service implemented as an Application Level Multicast (ALM) tree, i.e., a tree-based overlay network upon the IP-level connections. For simplicity sake, in the theoretical analysis we refer to the scenario depicted in Figure 1, where a publisher is directly connected to several subscribers. However, in Section 4 we relax this assumption by considering a more scalable topology for WAN. In both cases we assume the presence of unreliable links. The protocol we use to reliably and timely deliver events to subscribers is composed by two phases:

1. *Dissemination:* the publisher fills n packets, named as plain, with the context of an event and sends them to all destinations. In addition, a additional packets are also sent. We call these the *redundancy* of the protocol. We separately consider two different cases: (i) *no-coding:* the a packets are randomly selected among the n plain packets; (ii) *coding:* the a packets are linear combinations of the n packets.
2. *Recovery:* when a subscriber detects the loss of a packet (we assume the presence of a fixed timeout), then it starts a gossip-based recovery procedure to fully reconstruct that event.

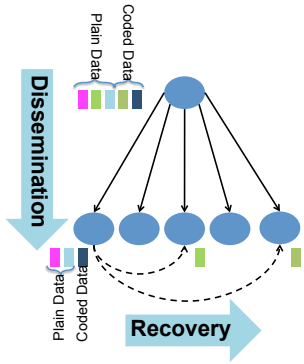

Fig. 1: Schema of the proposed approach

In the remainder of this section we analyze the effect of coding on a gossip protocol. In order to keep the analysis simple while capturing the main characteristics of our protocol, we consider a pull-based gossip procedure. A complete comparison among gossip strategies will be provided in Section 4

The goal of the model is to capture the number N_R of gossip rounds required for a tagged destination node to retrieve an event composed by n packets. During each round, the destination performs F elementary operations, each called *contact*, consisting of contacting another node and pulling useful data from it. The parameter F is also called fanout. We assume a discrete-time model, in which the contacts are numbered progressively. In the following we calculate $P_R(k) = Pr\{N_R \leq k\}$ for $F = 1$ and $F > 1$, i.e., the probability of retrieving the event by k rounds. We call this probability the *success rate*.

3.1 Success rate, $F = 1$

To numerically compute $P_R(k)$ we will exploit a Markov chain with $n+1$ states and time variable transition probabilities. The state of the chain is the number of *useful* packets stored at the destination node at the end of the gossip rounds. For the no coding case, the useful packets correspond to packets with different IDs, whereas for the coding case they are linearly independent combinations.

The state probability is denoted by $\pi^k(i)$, which represents the probability that after k gossip rounds the tagged destination stores i useful packets. We assume that the node contacted at k-th round did not execute yet the gossip operation at that round, i.e., the contacting destination node sees the contacted node as it was at the end of round $k-1$. The evolution of the Markov chain is described through the following equation: $\pi^{(k)}(j) = \sum_{i=0}^{n} \pi^{(k-1)}(i) P_{ij}(k)$, $k > 0$, where $\pi^{(0)}(i)$ is the probability of storing i packets at the end of the dissemination phase of the algorithm, and $P_{ij}(k)$ the state transition probability from state i to state j at gossip round k. This probability is $P_{ij}(k) = 0$ for $j < i$ (the number of stored packets cannot decrease) whereas $P_{nn}(k) = 1$. With these assumptions, the state n is an absorbing state and $\pi^k(n)$ represents the probability that at the end of round k the useful packets stored at the destination node is n, i.e., the event is fully received. Hence, $P_R(k) = \pi^k(n)$.

The transition probabilities as well as the initial state are computed in terms of the following three probabilities:
• Holding probability $P_H(k,r)$: probability of contacting a node holding r packets at the beginning of round k.
• Increasing probability $P_I(j|i,r)$: probability that after merging two random groups of i and r different packets, the resulting group has j useful packets [3].
• Loss probability $P_L(n,d)$: probability of loosing d out of n transmitted packets over a Gilbert-Elliott channel [14], starting from a random instant of time.
As far as the transition probabilities are concerned, after performing a round the destination node increases its state from i to j if and only if the following events occur:
• The contacted node holds r packets, where $r \geq j-i+d$ and $d \geq 0$. This event occurs with probability $P_H(k,r)$.
• After adding these r packets the number of useful packets for the destination potentially becomes $j+d$, i.e., the number of additional useful packets is $j+d-i$. This happens with probability $P_I(j+d|i,r)$.
• The contacted node sends to the destination the $j+d-i$ useful packets and, during the transmissions, d packets are lost. This happens with probability $P_L(j+d-i,d)$.

We then have: $P_{ij}(k) = \sum_{r=j-i}^{n} \sum_{d=0}^{r} P_H(k,r) P_I(j+d|i,r) P_L(i+d-j,d)$. We now compute the key probabilities defined above as well as the initial distribution for the coding and no coding cases.
Holding Probability - Our model assumes that the state of a node contacted during round k is equal to the state of that node at the end of round $k-1$. We assume that the contacted node is itself experiencing the same sort of evolution as the contacting node, i.e., the *number* of packets stored by the contacted node follows the same statistic of the number of packets stored in the observed node, but delayed of one unit of "time"; hence $P_H(k,i) = \pi^{(k-1)}(i)$.
Increasing probability, baseline protocol - The probability $P_I(j|i,r)$ for the baseline protocol can be found through a combinatorial argument. Let divide the r packets into two subsets U of $a = j-i$ elements and $\overline{U}$ of $r-a$ elements, $0 \leq a \leq r$. This can be done into $\binom{r}{a}$ different ways. Now, let examine the packets according to some order (for example their IDs). The first packet in U is useful with probability $\frac{n-i}{n}$, the second one with probability $\frac{n-i-1}{n-1}$ and so on. Hence, the element in subset U are all useful with probability

$$p_u = \frac{n-i}{n} \times \frac{n-i-1}{n-1} \times \ldots \times \frac{n-i-d+1}{n-d+1} = \frac{(n-a)!}{n!} \times \frac{(n-i)!}{(n-i-a)!}$$

Similarly, the first packet in the complementary subset, $\overline{U}$, is not useful with probability $\frac{i}{n-d}$, the second one is not useful with probability $\frac{i-1}{n-d-1}$, and so on. Hence, all the $r-a$ elements of $\overline{U}$ are not useful with probability:

$$p_{\overline{u}} = \frac{i}{n-a} \times \frac{i-1}{n-a-1} \times \ldots \times \frac{i-(r-a-1)}{n-d-(r-a-1)} = \frac{i!}{(i-r+a)!} \times \frac{(n-r)!}{(n-a)!}$$

[3] In the coding case *different* means linearly independent from each other.

$P_I(j|i,r)$ is then the product of the above two probabilities times the number of possible subset of a elements: $P_I(j|i,r) = \binom{r}{a} p_u p_{\overline{u}}$.
Increasing probability, coding - In [11], we have shown that the probability of a random linear combination being useful is bounded by $1-\frac{1}{q}$, with q the size of the Galois Field where coefficients used to compute the linear combination are taken uniformly at random. In practice, the value q is sufficiently high to make this probability almost one [11]. Thus, we get: $P_I(j|i,r) = \begin{cases} 1 \; j = i+r, i+r \leq n \\ 0 \text{ otherwise} \end{cases}$
$P_L(n,d)$ - This probability has been computed in several papers, *e.g.*, [15]; due to lack of spaces, we do not report further details.
Initial distribution, baseline protocol - Let assume that the source node sends the n original packets plus a additional duplications, $0 \leq a \leq n$, in a random order, starting from a random instant of time. As the order of packet transmissions is random, the loss events are in turn randomized over the whole set of sent packets. Hence, a loss hits any of the $n+a$ packets with the same probability. Now, the are $\binom{n}{d_1} \times \binom{a}{d_2}$ ways to form a pair of groups, in which the first (second) group is a subset of d_1 (d_2) elements, taken from a set of n (a) elements. Therefore, the probability that n' out of the n original packet are lost and a' out of the a duplications are also lost, given that $d_1 + d_2 = n' + a'$ packets are lost, is $p(n',a') = \frac{\binom{n}{n'} \times \binom{a}{a'}}{\sum_{d_1=0..n}^{n} \sum_{d_2=n'+a'-d_1}^{a} \binom{n}{d_1} \times \binom{a}{d_2}}$. The initial distribution can be found by summing up the probabilities associated to the following three events: (i) the total number of packet lost is $n'+a'$; (ii) among them n' are from the original n packets and a' from the duplication; (iii) the number of different packets after merging the received $n-n'$ original packets and $a-a'$ duplications is i. Hence:

$$\pi^{(0)}(i) = \sum_{n'=0}^{n} \sum_{a'=0}^{a} P_L(n+a, n'+a') \times p(n',a') \times P_I(i|n-n', a-a')$$

Initial distribution, coding - A random linear combination is very likely to be independent from any other group of random linear combinations [11]. Hence, the initial distribution of the Markov chain is well approximated exploiting the received packet's distribution:

$$\pi^{(0)}(i) = \begin{cases} P_L(n+a, n+a-i) & i < n \\ \sum_{k=0}^{a} P_L(n+a, n+a-k) & i = n \end{cases}$$

3.2 Success probability, $F > 1$

In a gossip protocol with fanout F, during a round the destination node contacts F nodes and pull data from them. Although in the real protocol these contacts occur in parallel, we treat them as elementary gossip operations occurring in sequence and changing the state of a node. For convenience, the initial state gets index $k = -1$, rounds are numbered starting from 0 and the first elementary operation of round 0 occurs at time index $k = 0$. Hence, for example for $F = 2$, the contacts with number $k = 0, 1$ belong to round 0, contacts $2, 3$ to round 1, etc. In general, the $k-$th belongs to the round $\lfloor k/F \rfloor$. The node performing

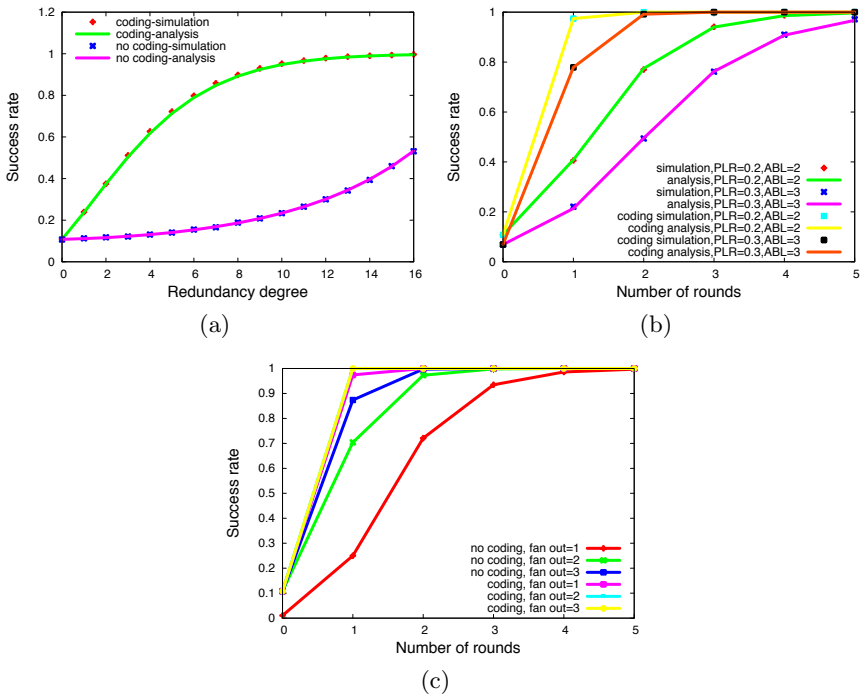

Fig. 2: (a) Reliability improvement in coding and no coding cases by varying the redundancy degree; (b) Impact of the number of gossiping rounds on success rate; (c) Effect of fanout on success rate in coding and no coding cases.

round F sees the contacted node being in the state $F\lfloor k/F \rfloor - 1$. Under these considerations, the previous model can still be applied; the only formal changes are the definition of the holding probability $P_H(k,i) = \pi^{(F\lfloor k/F \rfloor - 1)}(i)$ and the fact that the meaningful probability values are those for $k = jF-1$, $j = 0, 1, 2, ...$

3.3 Results and discussion

Figure 2(a) compares the success rate in coding and no coding cases, by varying the number of redundancy packets and the network conditions in terms of: (i) *Packet Loss Rate (PLR)*, i.e., the rate at which the network drops packets, and (ii) *Average Burst Length (ABL)*, i.e., the number of consecutive dropped packets. The success rate is defined as the fraction of system nodes that fully recover an event. In both coding and no coding protocols, PLR is set to 0.2 and ABL to 2. The results obtained through simulations of 100 nodes are also reported. Simulations results have been obtained with a custom time-driven simulator. We can see how coding exponentially improves the reliability by augmenting the number of redundant packets. This is direct consequence of the high probability for a linear combination to be independent from all the others. It is anticipated that a similar behavior is also found when a tree-based ALM protocol is used.

Figure 2(b), instead, shows how reliability augments in presence of gossip. In case of coding, 2 rounds are enough to fully recover the whole event. The figure thus confirms that coding is expected to improve the delay performance as a consequence of a less number of rounds required to get the missed packets in case of loss. Finally, Figure 2(c) shows the effect of fanout for $PLR = 0.2$ and $ABL = 2$. By Increasing the fanout, a node pulls a larger amount of data per round, thus making a round more effective.

4 Experimental Evaluation

The goal of this section is to present experimental results that point out the improvements achievable when combining coding and gossiping. We implemented our solution by using the OMNET++ (www.omnetpp.org) simulator: for the networking components we have used the INET framework, while, as ALM protocol, we used the Scribe [16] implementation provided by the OVERSIM library. We avoided to use real wide-area networks, such as PlanetLab, due to the uncontrollable loss patterns that make obtained results non reproducible [17].

The workload used in our experiments has been taken from the requirements of the SESAR project, since it is representative of a real LCCI case. Specifically, exchanged messages have a size of 23KB, the publication rate is one message per second and the number of nodes is 40 (this is the estimated number of ATM entities involved in the first phase of the SESAR project, deployed in Italy, Switzerland and France). The fault load consists in omissions only, that have been applied at the overlay level. This is motivated by the consideration that it is possible to have a simple estimation of losses due to the Internet by means of application-level measurements, as done in [17], while network-level measurements are difficult, or even impossible, to obtain. The values of the parameters that characterize the network dynamics are: 50ms as link delay, PLR=0.05 and ABL=2 [17]. We modeled the time to obtain a redundant block by encoding two information blocks equal to 5ms, while the time for the dual operation is equal to 10ms. We also assumed the block size equal to the payload of MTU in Ethernet (i.e., 1472 bytes) so that an event is fragmented in 16 blocks.

Finally, without loss of generality, we considered a system with a publisher and 39 subscribers, all subscribed to the same topic. We simulated a period of 1000 publications and reported the average of three different experiments on the same scenario (we did not observe standard deviations above 5% of reported values, thus they are not plotted on the curves).

We evaluate three different gossip strategies:
• *Push*: messages are forwarded to the other nodes as soon as they are received;
• *Pull*: nodes periodically send a list of recently-received messages; if a lost message is detected by comparing the received lists with the local history, then a transmission is requested;
• *Push/Pull*: a node forwards to the other nodes only the identifier of the last received message; if one of the receivers does not have such message, then it makes an explicit pull request.

The metrics we evaluate in our study are:

• *Success rate:* the ratio between the number of received events and the number of the published ones, and it is referred to as reliability, which is the ability of the publish/subscribe service to deliver all the published messages to all the interested subscribers. If success rate is 1, then all the published events have been correctly delivered by all subscribers.
• *Overhead:* the ratio between the total number of datagrams exchanged during an experiment and the number of datagrams generated by the publisher (that is the product of the published notifications and the number of datagrams in which a notification is segmented to be conveyed by the network). It is a measure of the traffic load that the dissemination strategy imposes on the network, and should be kept as lower as possible, in order to avoid congestions.
• *Performance:* mean latency is a measure of how fast the given dissemination algorithm is able to deliver notifications, while the standard deviation indicates possible performance fluctuations due to the applied fault-tolerance mechanisms, highlighting timing penalties that can compromise the timeliness requirement.
• *Dependance on Network Dynamics:* difference in percentage of the dissemination latency between two distinct experimental scenarios where network parameters are different. It indicates if the performance depends on network dynamics and the degree of unpredictability for dissemination latency, which makes timeliness not achievable.

The parameters we vary in our analysis are:
• *Fanout*: number of nodes contacted by the gossiper during a single gossip round.
• *Fanin*: each node has an history of the ids of received notifications that is accessed when a pull or push/pull round starts. The fanin indicates the number of rounds that an id is sent to gossip partners, after which it is deleted from the history. When not explicitly declared, we assume that the fanin is set to 1.
• *Redundancy degree*: it indicates both the number of redundant datagrams sent by the publisher in addition to the original packets and the fanout when gossiping is used. This is motivated by the fact that fanout implicitly defines a redundancy.

In our evaluation, we report the results obtained to reach a success rate equal to 1 (thus, some curves may be truncated).

4.1 Success rate

In Figure 3(a) we compare the coding and no coding cases by varying the redundancy degree. The publisher publishes an event on Scribe, by sending several redundant packets. In the coding case, these packets are linear combinations of the original datagrams that compose the event. We can notice that without coding the number of redundant packets to deal with the adopted fault load is largerly higher. Coding, on the contrary, exhibits a stronger recover capacity: in fact, in the former case, a full success rate (complete delivery) is achieved only with a redundancy equal to 29 (i.e., the event is sent almost three times), while in the latter one just with a redundancy of 8. This improvement is particularly meaningful for the considered ATC context, where real ATC systems currently ensure a complete delivery by sending a plain event three times (private dis-

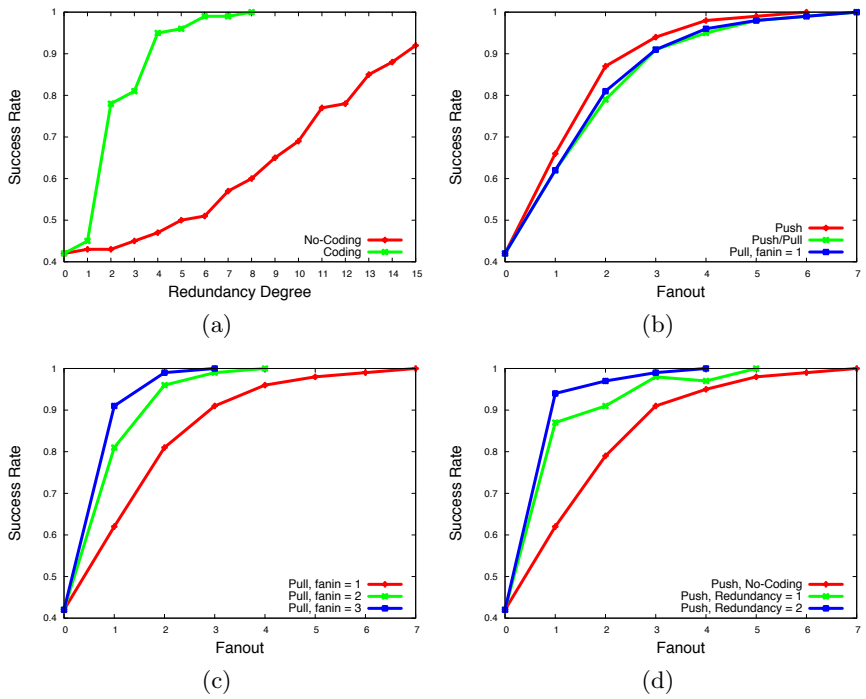

Fig. 3: (a) Success rate when retransmitting plain datagrams and coded datagrams; (b) success rate of the three gossiping styles; (c) success rate for pull gossiping when fanin is varied; (d) success rate for coded push gossiping when redundancy is varied.

cussion with dr. Angelo Corsaro, CTO at PrismTech and co-chair of the Data Distribution Service Special Interest Group).

In figures 3(b) and 3(c) we evaluate the three gossiping strategies, push, pull and push/pull, by varying the fanout. The publisher periodically publishes an event on Scribe; in the push and push/pull cases, when a node completely receives a message (all datagrams), it starts a gossiping procedure by contacting a subset of the other nodes according to the fanout value. On the contrary, the pull procedure is on a periodical basis, with the period set to 1.5 seconds. Figure 3(b) evidences that push is better than the other gossiping styles, since the loss of the gossip message or the retransmission request compromise the recovery of a given notification. However, Figure 3(c) shows an interesting trade-off between fanin and fanout in the pull approach: augmenting the fanin, augments the number of time that a message identifier is sent to other nodes, that, in turn, reduces the number of partners per round to succesfully spread this information.

In figures 3(d) and 4(a) we combine coding and gossiping: the publisher sends a plain message, as before, and introduces a redundancy by sending one or two linear combinations of that message. During the gossip phase, nodes retransmit just the redundancy. Results show that when coding is teamed up with gossiping, the provided reliability increases; incrementing the redundancy degree (from one

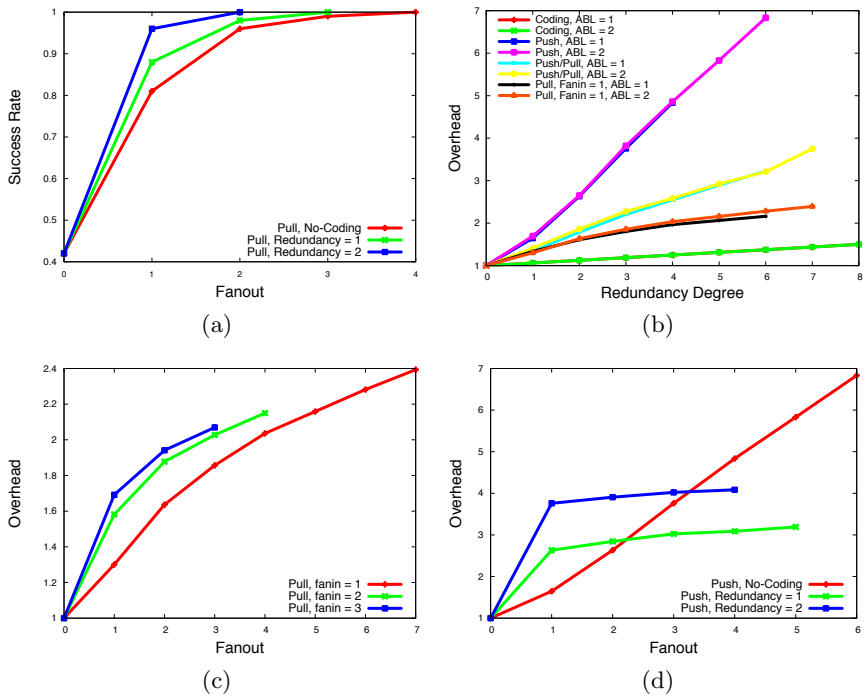

Fig. 4: (a) Success rate for combined pull gossiping and coding when redundancy degree is varied; (b) overhead for the different strategies; (c) overhead for pull gossiping when fanin is varied; (d) overhead for combined push gossiping and coding when redundancy degree is varied.

to two linear combinations) implies the achievement of complete delivery with smaller fanout with respect to the case of no coding. These results confirm what we mentioned in the previous section, i.e., coding is able to improve the reliability provided by gossiping without requiring a high redundancy degree.

4.2 Overhead

Figure 4(b) shows the overhead produced by coding, and the three gossiping strategies without applying any coding. The overhead caused by using coding only depends on the applied redundancy degree, and not on the experienced network conditions (in fact, it does not change even varying ABL). Such characteristic is also present for push and push/pull gossiping, while pull exhibits a slight variation caused by the different ABL. This difference is due to the proactive nature of push and push/pull gossiping (i.e., a gossip message is always sent without considering current network conditions). Moreover, the improvements introduced in push/pull by sending only the identifier of the received notifications, allows to reduce in a remarkable manner the experienced overhead. It is worth noticing that overhead is low for pull gossiping, even better than push/pull. This is due to its reactive nature: it sends notifications only

when needed (i.e., after the detection of a loss), so to reduce the overall traffic load. In addition, the gossiping period (always set equal to 1,5 seconds) is able to reduce the number of gossiping messages. Moreover, in Figure 4(c) we separately analyze the impact of fanin on the overhead for the pull-based strategy. Due to the slightly difference in the overhead measure by varying the network conditions, for clarity of presentation we plot only the case with ABL = 2. As expected, augmenting the fanin augments the probability that a message will be retransmitted several times, so as to increase the measured overhead.

Figures 4(d) and 5(a) show the impact of coding on push and pull gossip styles respectively. We notice that the redundancy initially introduces a higher overhead, but it is mitigated by the lower fanout required to reach a success rate equal to 1 with respect to the case without coding. In particular, Figure 5(a) shows an interesting property: augmenting the redundancy in the pull strategy decreases the overhead to reach a reliable delivery. This is motivated by the fact that coding is able to reduce the number of needed retransmissions and the fanout to successfully deliver a message, as depicted in Figure 5(b).

4.3 Performance

Figure 5(c) illustrates the mean latency of the gossiping and coding approaches to reach a full success rate by varying the redundancy degree. As expected, coding has an almost linear trend when incrementing the applied redundancy degree, while push and push/pull gossiping have better performance, further improved when fanout is incremented. On the contrary, pull has lower performance due to its reactive nature. Figure 5(d) clearly shows that in pull gossiping the fanin directly compromises the performance: the higher is the fanin, the greater is the delivery latency. This is motivated by the trade-off between fanin and fanout: a higher fanin decreases the fanout, but the complete reconstruction of a message is spread over more (periodic) gossip executions.

Not surprisingly, applying coding has good effects on the mean latency. This improvement is particularly evident in the pull approach, as depicted in Figure 6(a): a redundancy degree equal to 2 decreases the mean latency from 1.25 to 0.99 seconds. The reason of this behaviour is the ability of coding to reduce the number of retransmissions, as previously shown in Figure 5(b). Finally, Figure 6(b) illustrates the standard deviation for the investigated strategies: as expected, coding has the lowest values (proving its high timeliness guarantees, as mentioned), while pull exhibits the worst value (proving the low timeliness guarantees of retransmission-based strategies). The interesting result is that coding is able to decrease the standard deviation. In fact, applying a coding degree of 2, push and pull experience respectively a reduction of 14% and 19%.

4.4 Dependance on Network Dynamics

To investigate any effect of network dynamics on the obtainable performance, we have compared the mean latency when PLR is varied, and results are depicted in Figure 6(c). Specifically, coding exhibits always the same performance, while push and pull show a variation (respectively 8% for push and 31% for pull). In addition, coding is able to lower the measured variation in latency for pull, due to the increase of PLR (i.e., 13% for pull).

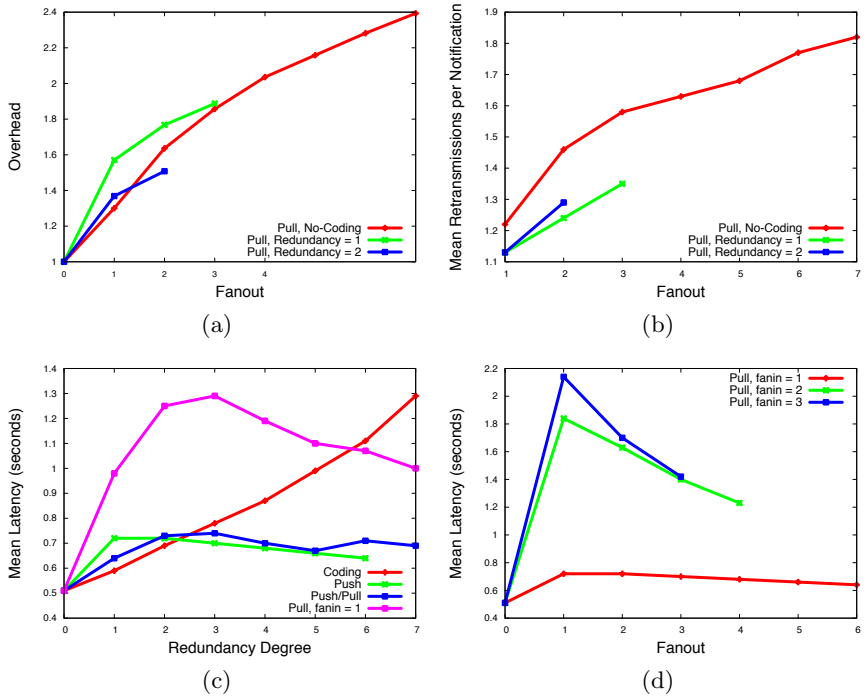

Fig. 5: (a) Overhead for combined pull gossiping and coding when redundancy degree is varied; (b) number of needed retransmissions per notification when redundancy degree is varied; (c) mean latency for the different strategies; (d) mean latency for pull gossiping when fanin is varied.

5 Final Remarks

The inherent scalability provided by the publish/subscribe paradigm makes it appealing for the design of the new generation of LCCIs over WAN. However, the critical nature of these systems requires that all information has to reach the interested destinations within strict temporal constraints. Recently, researchers started to investigate how to improve reliability in publish/subscribe systems, but this improvement, often obtained through retransmission techniques, is tipically gained at the expenses of performance penalties. In this paper, we proposed a strategy that combines coding and gossiping for reliable and timely data distribution over WAN. We conducted a theoretical analysis to evaluate the potential benefit of coding on delivery performance, even when the sender introduces redundant information to improve reliability. In addition, we evaluated the impact of coding through an experimental study conducted on a real workload taken from the air traffic control scenario. The obtained results state that coding improves the reliability of gossip techniques by also decreasing the overhead in the data dissemination. In addition, coding has a positive impact even on the mean notification latency, due to a decrease in the number of retransmissions required to fully reconstruct a message. Finally, coding helps to stabilize the performance

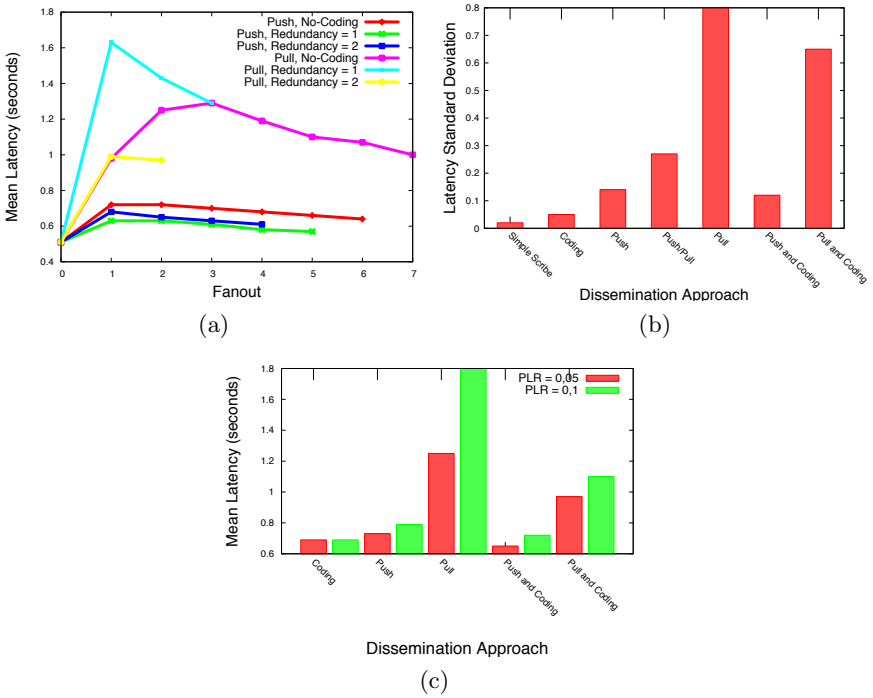

Fig. 6: (a) Mean latency for combined approaches when redundancy degree is varied; (b) standar deviation of dissemination latency; (c) network dependancy.

of the data dissemination strategy by decreasing the mean notification latency (from 30% to 10% in the pull strategy) and possible latency fluctuations in presence of changes in network conditions. As such, a decrease of these fluctuations makes the notification latency more predictable and, in turn, helps to satisfy timeliness constraints imposed by the applications.

Acknowledgment

This work has been partially supported by the Italian Ministry for Education, University, and Research (MIUR) in the framework of the Project of National Research Interest (PRIN) "DOTS-LCCI: Dependable Off-The-Shelf based middleware systems for Large-scale Complex Critical Infrastructures", and by the BLEND Eurostar European Project.

References

1. C. Esposito, D. Cotroneo, A. Gokhale and D.C. Schmidt, "Architectural Evolution of Monitor and Control Systems - Issues and Challenges," *International Journal of Network Protocols and Algorithms*, vol. 2, no. 3, pp. 1–17, 2010.

2. R. Baldoni, M. Contenti, S. Piergiovanni, and A. Virgillito, "Modelling publish/-subscribe communication systems: towards a formal approach," 2003.
3. C. Fragouli, J. Le Boudec, and J. Widmer, "Network coding: an instant primer," *Computer Communication Review*, vol. 36, no. 1, p. 63, 2006.
4. A.-M. Kermarrec, L-Massoulié, and A. J. Ganesh, "Probabilistic Reliable Dissemination in Large-Scale Systems," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 14, no. 2, pp. 1–11, February 2003.
5. P. Costa, M. Migliavacca, G. Picco, and G. Cugola, "Epidemic Algorithms for Reliable Content-Based Publish-Subscribe: An Evaluation," *Proceeding of the 24th IEEE International Conference on Distributed Computing Systems*, pp. 552–561, March 2000.
6. R. Baldoni, R. Beraldi, V. Quema, L. Querzoni, and S. Tucci-Piergiovanni, "Tera: topic-based event routing for peer-to-peer architectures," in *Proceedings of the 2007 inaugural international conference on Distributed event-based systems*. ACM, 2007, pp. 2–13.
7. R. Baldoni, R. Beraldi, L. Querzoni, G. Cugola, and M. Migliavacca, "Content-based routing in highly dynamic mobile ad hoc networks," *International Journal of Pervasive Computing and Communications*, vol. 1, no. 4, pp. 277–288, 2005.
8. C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution," in *Proceedings of the IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, 2005, pp. 2235–2245.
9. T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.
10. P. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, vol. 41, no. 1. The University; 1998, 2003, pp. 40–49.
11. C. Esposito, S. Russo, R. Beraldi, and M. Platania, "On the benefit of network coding for timely and reliable event dissemination in WAN," *Proceedings of the 1th International Workshop on Network Resilience, held jointly to the 30th IEEE SRDS, available at dots-lcci.prin.dis.unina.it*, October 2011.
12. M. Balakrishnan, K. Birman, A. Phanishayee, and S. Pleisch, "Ricochet: Lateral Error Correction for Time-Critical Multicast," *Proceedings of the 4th USENIX Symposium on Networked System Design & Implementation*, pp. 73–86, April 2007.
13. S. Deb, M. Medard, and C. Choute, "Algebraic gossip: a network coding approach to optimal multiple rumor mongering," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2486–2507, June 2006.
14. X. Yu, J. W. Modestino, and X. Tian, "The Accuracy of Gilbert Models in Predicting Packet-Loss Statistics for a Single-Multiplexer Network Model," *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 4, pp. 2602–2612, March 2005.
15. L. Wilhelmsson and L. Milstein, "On the effect of imperfect interleaving for the Gilbert-Elliott channel," *IEEE Transactions on Communications, Volume - 47 Issue:5, May 1999*, May 1999.
16. M. Castro, P. Drushel, A. Kermarec, and A. Rowstrom, "Scribe: A Large-scale and Decentralized Application-level Multicast Infrastructure," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1489–1499, January 2004.
17. C. Esposito, "Data Distribution Service (DDS) Limitations for Data Dissemination w.r.t. Large-scale Complex Critical Infrastructures (LCCI)," *Mobilab Technical Report (www.mobilab.unina.it)*, March 2011.