Building a network embedded FEC protocol by using game theory

Christian Esposito[a,*], Arcangelo Castiglione[a], Francesco Palmieri[a], Massimo Ficco[b]

[a] Department of Computer Science, University of Salerno, Via Giovanni Paolo II, 132 84084 Fisciano, SA, Italy
[b] Department of Industrial and Information Engineering, Second University of Naples, Via Roma 29, 81031 Aversa, CE, Italy

A B S T R A C T

Due to their decoupling and scalability properties, multicast technologies are widely adopted in all the software projects aiming at integrating off-the-shelf systems, in order to compose large-scale complex infrastructures. This also holds within the context of mission critical systems, where strong requirements for reliable and timely data sharing are imposed. In the current literature, reliable multicast is always achieved at the expenses of violations of the temporal constraints, since retransmissions are used to recover lost messages. When timeliness requirements assume the same importance as the reliability ones, the techniques based on spatial redundancy, such as forward error correction, are preferable. However, such coding techniques are scarcely adopted within multicast services due to the difficulties related to the optimal tuning of the introduced redundancy. In this paper, we present a solution applying a proper coding scheme that jointly achieves reliability and timeliness in all those scenarios involving tree-based multicasting over the Internet. Such a solution employs game theory in order to select the best locations within the multicast tree where coding operations have to be performed. We prove the quality of this solution by using a series of simulations running on OMNET++, that compare the achievable quality with respect to the optimal solution of the underlying optimization problem.

Keywords:
Application level multicast
Forward error correction
Message loss tolerance
Game theory

1. Introduction

Due to security and availability concerns, critical systems rely on traditional architectural solutions based on a monolithic, "close world", perspective, *i.e.*, several computing nodes are interconnected by a dedicated network with limited or no connectivity to the outside world. This causes large-scale critical infrastructures, *e.g.*, telecommunication, water supply, electric grid or transportation systems, to be fragmented in several sub-systems, each focused on controlling only a certain portion of the whole infrastructure, but completely blind on what happens into the other portions. However, this lack of cooperation implies a non-optimal control of these large-scale critical systems. In fact, the decisions taken by a certain portion also have a negative impact on other ones, since there are dependancies among the different portions of the infrastructure that should orchestrate their decisions in order to have an optimal control.

* Corresponding author.
E-mail addresses: esposito@unisa.it, christian.esposito@unina.it (C. Esposito), arcastiglione@unisa.it (A. Castiglione), fpalmieri@unisa.it (F. Palmieri),

https://doi.org/10.1016/j.ins.2017.04.010

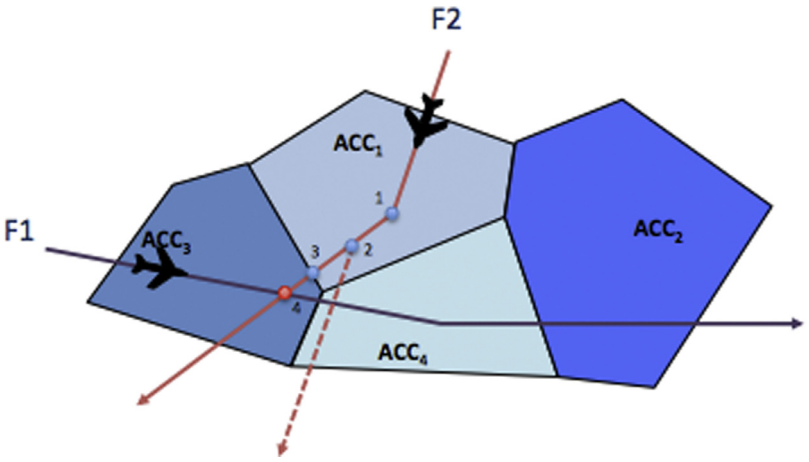

Fig. 1. Route tracking of two flights for collision avoidance.

A practical example is represented by the current *Air Traffic Management* (ATM) framework in the European Union (EU), which is organized as a composition of several systems called *Area Control Centers* (ACCs), spanning across the EU members. All the ACCs in a given country are managed by a civil aviation authority, which handles all the technical aspects for the underlying ACCs. ACCs in different countries differ in the implementation technologies or regulatory/legal frameworks. Due to this heterogeneity, each ACC is separated from the rest of the infrastructure and takes decisions without considering the other ACCs. In fact, each ACC is responsible for the en-route and landing/departing control of the aircrafts in a given volume of the airspace [14], called *Area of Responsibility* (AoR), and such a control is handed over from an ACC to another one only via radio communications among human operators. This mechanism to hand-over the control of a flight among different ACCs has been proved to be unsuitable to handle the growing avionic traffic in Europe and may cause severe negative effects. Let us consider Fig. 1 that shows the route of two flights, namely, *F*1 and *F*2, which pass through four ACCs. The aircraft *F*2 is currently flying in the airspace assigned to ACC_1, while *F*2 is flying in the airspace of ACC_3. When there is no cooperation, each ACC has no information on the flights that are located in airspace volumes different from its own. Specifically, ACC_3 is unaware that *F*2 represents a serious threat to the aircraft *F*1. In fact, if no one of the two aircrafts will change its route, a collision will occur at the point 4 of Fig. 1. Since ACC_1 does not know anything concerning *F*1, since it is completely out of its AoR, it cannot command any change of route to the pilot of *F*2. On the other hand, ACC_3 will get from its radars the position of the aircraft *F*2 only when it will enter its AoR by passing over the point 3, maybe too late for the pilot of *F*1 to take the needed countermeasures for avoiding the collision. When ACC_1 is approaching to handover the control of *F*2 to ACC_3, its operator contacts the other ACC to inform it of the route taken by the aircraft. However, such information cannot be timely delivered, since exchanged among two human operators using radio communications. In addition, for some unexpected reasons, *e.g.*, turbulence or technical problems, the actual position of the aircraft may diverge, referring to the dashed blue line in the figure, and the flight surveillance needs to be guaranteed without interruptions, maybe handing the control of the flight over another ACC, in the example ACC_4, different that the expected one, as ACC_3. The increase of the avionic traffic is progressively reducing the distances among the flights, making the airspace more crowded and requiring faster and faster control decisions among ACCs. Fast control handover is crucial to avoid dangers to human lives in the EU skies, so an almost unattended computer-based solution for the control handover has been envisioned as the only viable solution for this problem.

Recent developments in networking and the extraordinary success of the Internet as a global communication channel are making possible the gradual disposal of these traditional and ineffective strategies and architectures and encouraging an architectural revolution that is leading to a new generation of critical systems, the so-called *Large scale Complex Critical Infrastructures* (LCCIs) [9]. Specifically, LCCIs adopt a federated, “open world”, perspective, *i.e.*, they consist of a dynamic Internet-scale hierarchy/constellation of interacting heterogeneous legacy systems, which cooperate to perform critical functionalities [15]. LCCIs provide a solution to the unsuitability of traditional isolated architectures, and to the urgent need for more integrated control schemes. In fact, the federation that characterizes LCCIs goes beyond the trivial exchange of monitoring data between distinct control systems, by fostering the implementation of complex decentralized and distributed control algorithms, where decisions are taken through the cooperation of several geographically distributed controllers. LCCIs introduce a novel factor in the design of critical systems: Internet connections are adopted as means to convey critical information [16]. This introduces three main challenges that have to be properly considered by the data dissemination infrastructure used to architect an LCCI:

- *Timely data Dissemination*: although critical systems have been used in several different domains, spanning from military applications to the civil ones, all these use cases have one feature in common: the right answer delivered too late becomes the wrong answer, and may lead to wrong control decisions.

- *Reliable information distribution*: several failures, *e.g.*, link or node crashes and message losses, may occur and compromise the effectiveness of the data distribution process. So, a well-defined reliability strategy must be used to cope with such failures and ensure that messages reach the right destination.
- *Scalable message delivery*: the scale of LCCIs represents a serious challenge when designing such infrastructures and must be taken into high consideration. The time necessary to deliver a message does not have to be strongly affected by the scale of LCCIs, both in terms of the number of devices composing the system (vertical scale) and the traffic generated by the data sources (horizontal scale). Moreover, the adopted reliability strategy should cope with the large number of nodes and volume of traffic, without requiring any centralization of the message recovery duties.

Thanks to their peculiarity of offering decoupled and scalable multicasting, publish/subscribe services [12] represent a viable solution for realizing the federating middleware needed to implement LCCIs. However, they also need to provide means to guarantee timeliness and reliability in all their activities. Since its inception, the publish/subscribe community has been more focused on scalable architectures, efficient delivery, and expressive subscriptions rather than reliable event dissemination [10]. However, this situation significantly changed as more and more publish/subscribe services have started to be used in application domains that expose stringent reliability requirements. As a concrete example, in the context of the SESAR EU project, the EuroControl agency has chosen an OMG standard for publish/subscribe services as the reference technology for the novel European Air Traffic Management framework. In [10], we have noticed that only few publish/subscribe services handle message losses, while most of them only focus on how to manage node and link crashes. Until now, studying how to cope with network anomalies has not been the focus of the community, since it was deemed to be easily addressed by using a proper reliable event dissemination protocol, such as one of the reliable multicast protocols developed during the last decades within the networking community. However, in infrastructures such as LCCIs, nodes can be made as reliable as needed, while there is no control on the network dynamics, so that message losses represent the main source of inefficiency and need to be particularly taken in consideration. Moreover, in the current solutions for reliable data dissemination, the gain in reliability is always achieved at the expenses of the achievable delivery time, since retransmissions are used to recover lost messages. Therefore, the goal of this paper is to fill the above mentioned gap in reliable publish/subscribe services by proposing an innovative reliability strategy that guarantees message multicasting without compromising the notification performances. Specifically, we propose a method based on *Forward Error Correction* (FEC) [34], where the generation of redundant packets is not only performed at the root, as commonly done in the available literature, but also in some interior nodes within the multicast tree, called codecs. Our work aims at proposing a distributed algorithm to find the best nodes where performing FEC, in order to support a high degree of reliability without incurring in an excessive traffic overhead within the network. The novel contribution of this work is the following:

- Modeling the problem of codec placement in terms of an optimization problem;
- Resolving the optimization problem in a distributed manner by using game theory;
- Assessing the achievable quality of the proposed approaches with respect to the optimal solution.

Such a paper is an extended version of a previous publication [8], where the authors have approached the resolution of the above-mentioned optimization problem by using a non-cooperative game. Such a solution was proven to obtain good levels of reliability and timeliness, with a simple implementation of the approach within each node of a publish/subscribe service. However, such a solution was not able to find an optimal solution of the optimization problem; moreover, the erroneous monitoring of the loss pattern severely influences the outcome of the game. For this reason, we have improved our previous work by adding the resolution of the codec placement problem in terms of cooperative and Bayesian games, by using:

- *Cooperative game theory* whose scope is to reach a solution over the Pareto front of the codec placement optimization problem, so as to obtain optimality in the problem resolution;
- *Bayesian game theory* by adopting a probabilistic formulation of our games in order to reduce the effects that the perturbation in the observed loss patterns may have on the obtained solution.

Last, we have qualitatively and analytically assessed the superiority of these approaches with respect of the non-cooperative formulation in [8].

The reminder of the paper is organized as follows. Section 2 provides an overview of the current techniques to provide loss-tolerance in multicast services deployable over Internet, and points out their main limitations to jointly provide timeliness and reliability. In Section 3, we describe in details our approach; while, in Section 4 we report experimental results of our simulation-based study to assess the quality of our approach. Finally, we conclude in Section 5 with some remarks on future work.

2. Background and open issues

Multicast communication has been an active research topic in the last decade, and several protocols have been developed to support reliable dissemination of messages among the members of a certain group. Such protocols are traditionally grouped in two main classes, based on the layer in the ISO/OSI stack where the protocol has been implemented:

- *Transport-Level Multicast* (TLM), where multicast is realized by means of IP Multicast, and
- *Application-Level Multicast* (ALM), where applications autonomously organize themselves into logical overlay networks, and take care of performing the appropriate operations for effective data delivery.

Although TLM solutions are more efficient, since they exhibit lower delivery latencies and better usage of network resources, their adoption for Internet-scale systems has been dogged by deployment issues and scaling limitations [4,5]. For these reasons, ALM services represent a more viable solution, despite their intrinsic inefficiency in the usage of network resources. Within the context of these TLM and ALM solutions, several strategies have been proposed in order to achieve reliable dissemination of messages [24,26]. A possible taxonomy of such strategies traditionally consists in two main categories: the ones based on the *Temporal redundancy*, which use retransmissions to recover the lost messages, and the ones based on the *Spatial redundancy*, which forward additional information along to the application data, so as to recover dropped messages without using retransmission. We remark that the latter one encompasses the first solutions, investigated within the context of Computer Networks to tolerate message losses. In fact, such a solution, known as *Automatic Repeat reQuest* (ARQ) [28], has met a considerable success for unicast communication, *e.g.*, TCP uses a variant of ARQ to ensure reliable transmission of data over IP. For this reason, it has been applied also in the most mature publish/subscribe services available in the market. For a concrete example, the products compliant to the OMG specification called Data Distribution Service (DDS) [20] adopt ARQ as the method for providing reliability. However, ARQ suffers from serious scalability and reliability limitations over large-scale networks, due to the centralization of the recovery duties at the publisher side. To overcome such a drawback, retransmission-based approaches have evolved by performing recovery duties in a more distributed manner. Some practical examples of such techniques are the following ones:

1. *Lateral Error Recovery* (LER) [39]: all the members of a group are randomly clustered in ω distinct planes, where multicast trees are independently formed. Each process selects as its neighbors one process per each plane which is different than its own one. When a process experiences a message loss, it requests a retransmission to one of its neighbors that exhibits the lowest recovery latency. If this process does not have the requested message, another neighbor is contacted. Last, if none of the neighbors contains the given message, then the root of the multicast tree and the multicasted are contacted;
2. *Cooperative Error Recovery* (CER) [35]: processes are clustered in the so-called *minimum-loss correlation* (MLC) groups, whose members are characterized by a negligible loss correlation (*i.e.*, it is unlikely that, if a process experiences a message loss, all the members of its MLC group have lost the same message). Therefore, if a process experiences a message loss, it asks to the other members of its MLC group to retransmit the dropped message.
3. *Gossiping* [13]: applications receive messages through a multicast tree, and periodically start a gossip round. During this round, a given destination sends to other randomly-chosen destinations a summary of the received messages. When a destination detects an inconsistency by comparing the received summary and its own history of received messages, it sends a NACK to the sender of the summary to solicit a retransmission and to converge to the same history of received messages.

The main criticism to all the retransmission-based schemes is that an high reliability degree can be only obtained at the expenses of the achievable performances. In fact, these approaches exhibit a reactive nature, since recovery actions are taken only after a loss has been detected, and based on the quite simplistic consideration that retransmitting a message once is enough in order to correctly deliver it. However, the Internet, provides no guarantees about the successful reception of a retransmitted message, so it may be necessary to perform retransmission several times before being sure that the message has been correctly received by all the interested destinations. In fact, message deliveries over wide-area networks exhibit not-negligible bursty loss patterns [37], *i.e.*, a message has a considerable probability P to be lost during the delivery and the succession of consecutive dropped messages has an average length, namely ABL, greater than two. In addition, [29] has shown that loss statistics are not constant during the day, or even over the week/month, due to the higher amount of traffic, especially HTTP requests, corresponding to intense web surfing activities during specific hours of the day, in the weekends or vacations. So, the number of consecutive retransmissions that are needed to correctly deliver a message cannot be predicted beforehand, and the notification delay results unpredictable and characterized by severe performance fluctuations. To have an empirical evidence of these issues, we have conducted an experiment by considering the latency associated to event notification between a publisher and three subscribers. The communication is realized by means of a publish/subscribe service complaint to the OMG DDS specification, which, as mentioned before, adopts retransmissions on top of UDP for loss tolerance. Again, we have plugged the four computers running the publisher and the subscribers by means of a network emulator called Shunra Virtual Enterprise (VE)[1], which allows users to recreate a specific network behavior. We have considered three different scenarios with different network configurations, starting from a network exhibiting scarce isolated losses to one with a high number of losses occurring in burst (more details on these scenarios are given in [7]). As it can be noticed in Fig. 2(a), as soon as the network conditions become worse, we have an increase in the average delivery time and the occurrence of remarkable spikes in the measured latencies. In fact, in the figure we can observe a progressive increase of the interquartile rage in the delivery time, which is an evidence of severe performance fluctuations. The performance penalty can be experienced not only when ARQ is used, but also for more advanced retransmission-based schemes, such as gossiping, as seen in our previous work [11].

[1] www.cnrood.nl/PHP/files/telecom_pdf/Shunra_Virtual-Enterprise.pdf.

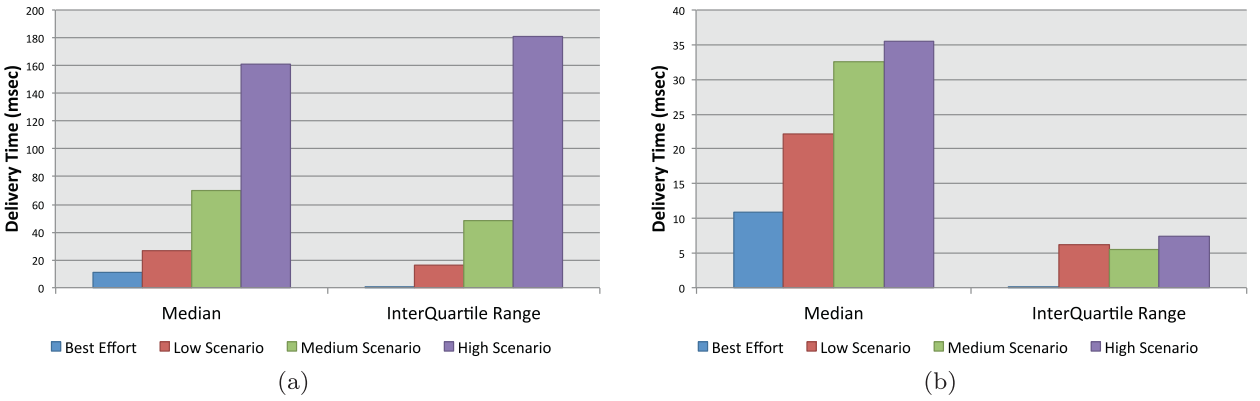

Fig. 2. Latency experienced in three different scenarios with (*a*) ARQ and (*b*) FEC used on top of a UDP-based publish/subscribe service.

A more effective solution to provide joint reliability and timeliness, is to proactively take reliability actions even if losses have not occurred, by adopting spatial redundancy rather than temporal one. One of the most common examples of spatial redundancy is the *Forward Error Correction* (FEC), where the multicaster generates redundant data by encoding the messages to be distributed, so as that the applied redundancy can be used to reconstruct the original application data when some losses occur. Unlike ARQ, the overhead imposed by FEC does not depend on the network conditions, making the delivery time more predictable. To have an empirical evidence of this statement, we have re-executed the previous experiments without using any fault-tolerant capability of the adopted publish/subscribe service and by inserting a FEC approach in the publishing applications before passing the message to the service, and in the subscribing application after having obtained the messaged from the service. Fig. 2(b) shows that the performance worsening of the FEC-enabled loss recovery is moderate if compared with the ones illustrated in Fig. 2(a), and we do not find the same high values for the interquartile range, meaning that the performance fluctuations are more moderate. In fact, we can see that the interquartile range is slightly larger than the one with only the best-effort transport protocol, and the increasing of losses causes small changes in this range. Also, the increase in the median value is lower than the one in Fig. 2(a). The main criticism against FEC schemes is that the achievable reliability strongly depends on the proper tuning of the applied redundancy. In case such redundancy is lower than the loss rate applied by the network, then dropped packets cannot be reconstructed and a given message is irremediably lost. On the contrary, thanks to the closed control loop applied in retransmission-based approaches, we are always able to recover dropped packets. Therefore, the reliability degree achievable by retransmission-based approaches outperforms the one offered by proactive approaches as FEC.

Although FEC is strongly adopted in TLM solutions, in ALM ones it has not found the same enthusiastic use due to severe tuning and efficiency issues. Generally speaking, there are two available approaches to embody FEC techniques in ALM solutions [18], depending on where the FEC operations are performed. On one hand, *End-to-End FEC* consists in having the multicaster to encode the messages to be delivered, while all the destinations decode the received packets in order to obtain the original message even if the network was affected by some losses. Such an approach can be abstracted as follows: a FEC layer is placed between the ALM protocol and the application, so that the messages generated by the application are encoded before being passed to the ALM, which will carry on the dissemination, while the packets received by the ALM will be decoded in order to deliver the obtained message to the application. This method is easy to implement, due to such layer separation, while it is strongly affected by an efficiency issue referred to as “the boat unbalanced by the heaviest” (BUH) problem. Specifically, FEC redundancy degree is usually decided by the encoder with respect to the worst loss pattern experienced along the path towards the *i*th destination, namely λ_i:

$$\rho \geq \max_{i=0,\ldots,N} \lambda_i, \tag{1}$$

where N is the total number oh nodes within the tree. So, if only few paths exhibit heavy losses, the multicaster has to generate a large amount of redundant data, even if the other destinations do not need it. A concrete example is shown in Fig. 3(a), where coding is applied at the root of a multicast tree, with a degree computed by Eq. (1). To evaluate the efficiency of such a solution in terms of traffic load, we have to measure the link stress, *i.e.*, the number of the redundant packets passing through a given link per each packet containing the application data to be delivered. In this example, such a metric assumes a value equal to 20.5. This overwhelming redundancy degree, and the consequent traffic load, may overload the nodes that need less redundancy and/or cause serious congestion in certain portions of the network. Such a situation of highly-heterogeneous loss pattern is not rare in the Internet, and can be caused by two main reasons:

1. The loss pattern of a node depends on the ones placed along its path to the root of the multicast tree. Therefore, the nodes closer to the root typically require a redundancy degree that is much lower than the ones that are further away.

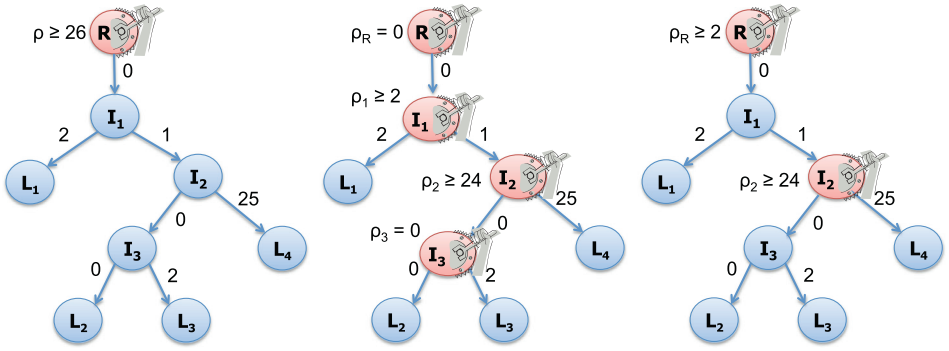

Fig. 3. The redundancy degree applied by the different FEC approaches: (a) End-to-End FEC, (b) Link-by-Link FEC, and (c) Network-Embedded FEC. The average loss rate is indicated per each link in the multicast tree.

2. Wide-area networks are made of the interconnection of different routing domains, each characterized by a certain network behavior that varies over time; therefore, Internet-size ALMs can have nodes in different domains, which cn potentially experience heterogeneous loss patterns.

On the other hand, *Link-by-Link FEC* allows every node to perform encoding and decoding, in order to protect the delivery on each link from message losses, and the redundancy degree is chosen considering the worst loss pattern on the link towards one of the children, namely λ_j, and the incoming redundancy ρ_{in} (*i.e.*, the redundancy applied by the parent of the ith node minus the losses occurred over the link connecting such a node to its parent: $\rho_{in} = \rho_j - \lambda_i$):

$$\rho_i \geq \max_{j=0,\ldots,C} \lambda_j - \rho_{in}, \tag{2}$$

where C is the total number of children that the ith node has. It can be implemented by putting the FEC layer between the ALM and the UDP protocol, so that every delivery operation invoked by the ALM will trigger an encoding, and every receive operation causes a decoding.

This method is more flexible since the redundancy degree is chosen only with respect to the quality of an overlay link between two nodes. A concrete example is shown in Fig. 3(b), where coding is applied by each interior node with a degree computed by using Eq. (2). In this case, the link stress is equal to the value of 10.57, which is almost an half of the one in Fig. 3(a). Therefore, this resolves the BUH problem, but not without any drawback. In fact, such a solution causes strong degradations in performance due to the continuous execution of the two coding operations at every overlay node.

The severe drawbacks that affect these two approaches have limited so far the efficient usage of FEC techniques in the context of ALM solutions. Accordingly, we propose a novel FEC technique that can be effectively applied to ALM solutions in order to achieve both timeliness and reliability in a multicast communication.

3. An optimized distributed FEC scheme

An appealing approach to mitigate the issues affecting the above mentioned ways to embody FEC-based spatial redundancy within an ALM solution is represented by the so-called **Network-embedded FEC** [38], where only a subset of interior nodes, called *codecs*, is allowed to perform encoding in order to generate redundant data. In this way, the redundancy degree is chosen based on the worst loss pattern on the path towards one of the overlying codec in the tree, and C in Eq. (2) becomes the total number of nodes belonging to the subtree rooted at the ith codec. This adapts the applied redundancy to better face the loss patterns exhibited by subtrees delimited between two codecs, by optimizing the need of a flexible setting of the redundancy and avoiding excessive over-provisioning without worsening the achievable reliability degree in case of message losses. Such an approach does not provide a coarse-grained control over the applied redundancy degree as seen for End-to-End FEC. On the other hand, since the number of codecs is kept lower than the total number of interior nodes, it does not introduce the performance overhead experienced by the Link-by-Link FEC. Fig. 3(c) shows a practical example where coding is applied only by an interior node, and link stress is equal to 10.85, close to the one of Link-by-Link FEC, but with less coding operations.

The problem of finding where to place codecs within a multicast tree can be formulated in the following way. Consider a set L of m possible locations in the multicast tree where a codec can be placed, a set U of n destinations of the multicast communication conveyed by the tree, and a $n \times m$ matrix D with the transportation costs $d_{i,j}$ for delivering a piece of information from the location j to the destination i, for all $j \in L$ and $i \in U$. In addition, let us define two sets of decision variables: (*i*) $y_j = 1$, if a codec is placed in $j \in L$, and 0, otherwise; and (*ii*) $x_{ij} = 1$, if the destination i receives redundancy from a codec located in $j \in L$, and 0, otherwise. The objective is to find the number of needed codecs and their locations by

minimizing the sum of these costs:

$$min \sum_{i \in U} \sum_{j \in L} d_{i,j} x_{i,j} + \sum_{j \in L} \alpha_j, \tag{3}$$

subject to:

$$\sum_{j \in L} x_{i,j} = 1, \quad \forall i, \tag{4}$$

$$x_{i,j} \leq y_j, \quad \forall i, j, \tag{5}$$

$$\sum_{j \in L} y_j = p \leq n = |L|, \tag{6}$$

$$x_{i,j}, y_j \in \{0, 1\}. \tag{7}$$

where the transportation cost $d_{i,j}$ is the redundancy that the codec in location j has to generate according to Eq. (2), and α_j represents the costs of placing a codec in the jth location. Such codec costs are needed to make the trivial solution, placing a codec in all the interior nodes within the multicast tree, unfeasible. The constraint in Eq. (4) indicates that each destination has to receive redundancy from only one codec; the one in Eq. (5) prevents each destination to receive redundancy from unactivated codec; and the total number of codecs in the multicast tree is set equal to p, lower than the total number of interior nodes within the multicast tree, namely n, by the constraint in Eq. (6). Such formulation belongs to a class of optimization problems known as the *P-Median* problems, with the slight exception representing the number of medians that have not been fixed a-priori. Such problems can be solved directly by using Mixed-Integer Program (MIP), or it is possible to relax the integral constraints and solve the corresponding pure Linear Program (LP) [25,33]. The resolution of such an optimization problem can be centralized at a node of the tree, or distributed among all the nodes.

Although a centralized solution allows us to simplify the problem resolution and take optimal decisions, it is unfeasible in a large-scale distributed system, since collecting global information is extremely difficult, if not impossible. In addition, even if assuming that is possible to acquire any global knowledge at a single node of the system, it still exhibits serious scalability limits that prejudice its usage for systems composed by a large number of nodes. In fact, the time needed to collect loss statistics in a centralized point linearly increases with the number of nodes and the links among them. Moreover, the memory required to store all the collected data handle the resolution of the optimization problem with a large number of nodes may overwhelm the resources of the central decision node and cause the impossibility to reach a proper solution. A distributed approach has the strength of overcoming these issues, since codec placement is performed by using local computations at each node of the multicast tree, thus avoiding the presence of a central decision maker with global knowledge of the system. However, this advantage is paid at the expenses of obtaining a placement that is slightly far away from the optimal one.

3.1. Facility location games

The distributed resolution of p-median problems, and more generally the facility location optimization problems, has been an hot research topic within the context of game theory [32], leading to the so-called facility location games. Such a rich branch of the literature is rooted in the seminal works of Hotelling [23] and Downs [6], where the problems is modeled by a set of p players that simultaneously select a location, with the impossibility for two, or more, players to pick up the same location, to attract the maximum number of consumers and minimize the relative costs. More formally, let us consider an undirected graph (X, E), consisting of a set of nodes, namely X, and a set of edges connecting two nodes, namely E. Within the set of the available nodes, we define a subset of all the nodes, namely $Y \subseteq X$, containing the nodes where a player can be located. We consider a set of players $P := c_1, c_2, \ldots, c_p$ of finite size $p \geq 2$, for which the strategy set $c \in P$ is defined as $S^c = Y$, such that a strategy of a player is the selection of a node s^c in Y. Combining the strategy sets of all the players, namely $S = S^{c_1} \times S^{c_2} \times \ldots \times S^{c_p}$, a strategy profile $s \in S$ implies a certain payoff to each player c, namely $\Phi^c(s)$, which are aggregated in the so-called profile of payoffs, denoted as π. The payoff is the gain achievable by a player to serve a set of consumers, considering both the cost to open a facility and the transportation costs, as above mentioned. The scope of the game is to determine the best strategy profile that implies the maximum payoff for all the players. Despite the several possible formulations that came out over the years within the literature, we can roughly describe facility location games in terms of a non-cooperative game [1] and a cooperative one [19].

In the first case, a game is defined non-cooperative, since players are selfish, *i.e.*, there is no direct communication among the players, and each one only cares to maximize its own profit or to minimize its own costs, without considering the state of the other players (with the eventuality of damaging them, even if it is not intentional). Then, the normal form for the non-cooperative location game is given by $\Gamma = (P, S, \pi)$, with the objective of maximizing the payoff for all the players. One of the most studied aspects of such class of games is the existence of Nash equilibria, *i.e.*, given a certain strategy $s \in S$, it is not profitable for a player to select a different node than the one in the current strategy profile, since moving to a neighbor

node will not change or even reduce the achievable payoff, so a player has no incentive to change strategy. More formally,

$$\exists s \in S : \forall c \in P, \forall x \in Y, \pi^c(s^c, s^{-c}) \geq \pi^c(x, s^{-c}) \\ \rightarrow s \text{ is a Nash equilibrium.} \tag{8}$$

The demonstration of the existence of such equilibria is a known NP-hard problem and is resolved by means of theorems, by making proper assumptions on the characteristics of certain elements of the game, such as the profit function or how customers prefers a player instead of another. For a concrete example, the authors in [36] demonstrate the existence of at least one Nash equilibrium for such games and the conditions to induce such equilibrium are presented. Moreover, it is possible that a game admits more than one Nash equilibrium, and the ratio between the objective function value of the best Nash equilibrium and the one of the worst one is defined as the Price of Stability (PoS), which has been investigated in several papers, such as [1,21].

3.2. Modeling codec placement

We can model our codec placement problem as a non-cooperative game with n players (where n is the number of interior nodes within the multicast tree), whose strategy is represented by a binary decision associated to playing the role of codec or not. Rather than formalizing the payoff of each player, we consider their costs, according to Eq. (3). The global cost function specified in such an equation is decomposed in a set of local cost functions to be assigned to each player [27]. Specifically, let us indicate with S_i the binary value representing the strategy chosen by the ith player (which is 1 if the ith player decides to be a codec; 0 otherwise), and with U_i the set of nodes belonging to the subtree rooted at the node where the ith player is located. The cost paid by the ith player to follow its strategy can be formalized as follows:

$$C_i(S_i) = \alpha_i \cdot S_i + (\rho_i + \max_{j \in U_i} \lambda_j) \cdot (1 - S_i), \tag{9}$$

where ρ_i is the redundancy received by the ith node, λ_j is the loss pattern experienced by the jth node having the ith node as parent within the multicast tree, and α_i is the cost to pay if the player has chosen to act as a codec. The game can start with a random strategy profile and evolve over the time where each player changes its strategy in order to minimize its costs formulated in Eq. (9). Such an evolution will bring to a stable solution represented by the Nash Equilibrium, where no player has an incentive to change its strategy. A non-cooperative game with pure strategies, as in the our case, can admit zero, one or multiple Nash equilibrium points in its strategy profile space. Since our game belongs to the class of location games, it admits more than one Nash equilibrium. Based on the definition in Eq. (8), it is possible to see that a strategy profile s represents a Nash Equilibrium if and only if the two following conditions are guaranteed:

$$\exists i \in Y \text{ s.t. } \rho_i + \max_{j \in U_i} \lambda_j \leq \alpha_i \tag{10}$$

$$\nexists i \in Y \text{ s.t. } \rho_i < \alpha_i - \max_{j \in U_i} \lambda_j \tag{11}$$

The first condition indicates that the redundancy generated by a codec placed at the i-th node is never greater than the α factor of its children; so, none of its children has an incentive to act as a codec. While, the second condition states that, when the ith node is the codec, it is not convenient to stop being a codec, since the paid cost is already minimized. The condition in Eq. (10) defines the control behavior of the agents to act as a codec or not, which can be formalized through Algorithm 1.

The solution achieved by modeling our problem as a non-cooperative game is not optimal, due to the lack of coordination among the players within the game. This is a well-known problem in game theory. Indeed, the solution that can be achieved by a non-cooperative game is typically worse than the optimum for the global cost function of a distributed optimization, and such a difference is measured with the so called Price of Anarchy (PoA), which assesses how the efficiency of a system degrades due to the selfish behavior of its agents. Specifically, within a game the sum of the costs paid by the players is known as social cost, and in our case, it is the sum of the costs for all the players:

$$C(S) = \sum_{j=0}^{n-1} C_i(S_i). \tag{12}$$

Naming as $C(S_O)$ the cost of the optimal strategy S_O, known also as social optimum, with the best possible placement, *i.e.*, the one with the lowest social cost:

$$C(S_O) = \min_S C(S), \tag{13}$$

Based on these equations, we formulate PoA as follows:

$$PoA = \frac{C(S_W)}{C(S_O)}, \tag{14}$$

with $C(S_W)$ indicating the cost of the worst case Nash equilibrium. PoA for location games have been investigated in several papers, such as [1,3]. In our case, it depends on the value assumed by the α_i factors. To prove this, let us assume all the

Algorithm 1 Agent behaviour within a non-cooperative game.

Require: τ, α

```
Function run()
find_Noncoop_Solution(∞);
EndFunction
Function find_Noncoop_Solution(TermCond)
isCodec ← false
current_redundancy ← 0
t ← 0;
while t < TermCond do
  wait(τ);
  demands ← collectDemands();
  if isCodec == true then
    if ρ + demands.max() > α then
      ρ ← ρ + demands.max();
    else
      isCodec ← false
      sendMessage(Parent, ρ + demands.max());
    end if
  else
    ρ ← getRedundancy(Parent);
    if ρ + demands.max() > α then
      isCodec ← true;
      ρ ← demands.max();
    else
      sendMessage(Parent, ρ + demands.max());
    end if
  end if
  t ← t +1;
end while
return isCodec;
EndFunction
```

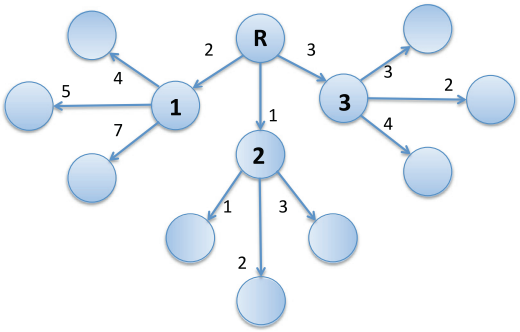

Fig. 4. Example multicast tree.

nodes sharing the same value for the factor α, and consider the example depicted in Fig. 4, whose admissible solutions for the codec placement are the ones illustrated in Table 1 with their relative social costs. We can state the following considerations:

1. if $\alpha \leq \lambda_{\min}$, where $\lambda_{\min}$ is the minimum loss pattern experienced along a path to the root of the multicast tree, then all the nodes will decide to act as codecs, since such solution is more convenient. This applies also for the social optimum (*e.g.*, the solution 4 has a social cost of 55 when $\alpha = 1$, having the lowest value respect to the other solutions), so the PoA assumes a value of 1;
2. if $\alpha > \lambda_{\max}$, where $\lambda_{\max}$ is the maximum loss pattern experienced along a path to the root of the multicast tree, then no interior node will act as a codec and only the root would generate the needed redundancy. This does not apply in the social optimum where multiple nodes may act as codecs, causing a PoA > 1. If we set α equal to 10 in our example,

Table 1
Admissible codec placements for the example in Fig. 4.

Placement	Codec Id	Applied redundancy	Social cost
1	Root	9	$\alpha + 108$
2	Root	7	$2\alpha + 84$
	Interior Node 1	2	
3	Root	4	$3\alpha + 57$
	Interior Node 1	5	
	Interior Node 3	3	
4	Root	3	$4\alpha + 51$
	Interior Node 1	6	
	Interior Node 2	1	
	Interior Node 3	4	

then the solution 3 is the one with the lowest social cost, not the solution 2, which is a Nash Equilibrium. In this case we have a PoA of 1.19.

3. if $\lambda_{\min} > \alpha > \lambda_{\max}$, it is possible that a certain subset of interior nodes decide to be codecs, while others do not. Knowing that the sum of the minima for a function series is greater than or equal to the minimum of the sum of a function series, we have the following:

$$PoA = \frac{C(S_W)}{C(S_O)} = \frac{\sum_{i \in N} \min_{S \in \Sigma} C_i(S_i)}{\min_{S \in \Sigma} \sum_{i \in N} C_i(S_i)} \geq 1. \tag{15}$$

As a practical example, consider the aforementioned one, with α equal to 5 and 8, respectively. In the first case, the social optimum and the Nash Equilibrium coincide and are equal to placement 4 in Table 1 (with a consequent PoA of 1). In the second case, the social optimum is placement 3; whereas, the Nash Equilibrium is placement 2 (with a PoA of 1.23).

This study shows the inefficiency of the Nash Equilibrium when our problem is modeled as a non-cooperative game, and this is caused by an undersupply of codecs within the multicast tree, due to the selfish behavior of the players. It is possible to obtain better placements, by making the agents less selfish by modeling our optimization problem as a cooperative game.

In cooperative games, players are able to cooperate and are grouped in coalitions, namely $C \subseteq Y$, whose cardinality, denoted as $|C|$, is equal to the number of players in the coalition; Y is called the grand coalition. If C is a coalition of players, then a function $v(C)$, often known as the characteristic function, describes the cost incurred by the agents in C if they cooperate; by convention, $v(\emptyset) = 0$. The scope of cooperative games is the efficient distribution of costs that a group of cooperating players has to pay. We indicate with $\bar{x} \in \Re^c$ the cost allocation vector for the players in a given coalition C (where c is the number of players in the coalition: $c = |C|$), and x_i represents the cost allocated to the ith player. Then, $x(C)$, with $C \subseteq Y$ indicating a given coalition of players, denotes the sum of all the costs allocated to the players:

$$x(C) = \sum_{i \in C} x_i \text{ for any } C \subseteq Y \tag{16}$$

A coalition can select a given cost allocation for its players, and a solution to the game is an allocation that is advantageous for all the members of the coalition, in a way that is not profitable for a player to leave the coalition in order to find a better allocation. The core indicates the set of all the possible fair solutions for a cooperative game, where the fairness is expressed by means of the following two conditions:

$$\text{core } = \{\bar{x} \in \Re^c : \forall C \subseteq Y, x(C) \leq v(C), x(Y) = v(Y)\}, \tag{17}$$

The first condition indicates that a cost allocation in the core is stable, *i.e.*, there is no subset of players that would have a better allocation by abandoning the coalition and acting on its own. On the other hand, the second condition states that a cost allocation in the core is efficient, *i.e.*, the cost allocated to the players is equal to the one when all the players cooperate. The key issue to be proven in cooperative games is that the core is not empty, which is NP-complete, and this is typically done by showing that the game is balanced or convex, by using proper theorems. Such aspect is of pivotal importance, since it can be considered as a sign that cooperation is attainable. In location games, it may be possible to have empty cores, but there are some conditions under which a game has a non-empty core. For a concrete example, the authors in [19] show that location games on a tree admit a non-empty core. The best, and unique, solution to a cooperative game is referred to as the Shapley Value, which is the cost allocation $\phi \in \Re^p$ that gives to each player the lowest possible cost than all the other possible coalitions, and consequent fair division of the grand-coalition costs among its participants. This value for the ith player is defined as follows:

$$\phi_i(Y, v) = \sum_{C \subseteq Y \setminus \{i\}} \frac{|C|!(|Y| - |C| - 1)!}{|Y|!} [v(C \cup i) - V(C)] \tag{18}$$

It is possible that such a value does not lie within the core [30], as possible in cooperative facility location games [19].

The traditional way of resolving such games by calculating the core or finding the Shapley value is based on centralized methods, where a mediator exists within the system. For a practical example, several solutions for the resolution of location games by means of greedy algorithms or LP relaxation of the problem are reviewed in [19,30]. Since they require a centralized resolution method, they are unsuitable to our case, but we are more interested to a distributed approach for the resolution of this cooperative game. Specifically, we have based our resolution approach on a different solution concept, known as Nash Bargaining Solution. The Bargain Set of a cooperative game is the set of cost allocations that implies lower costs than the best allocation when players do not collaborate, *i.e.*, the solutions verify the following condition:

$$BS = \{\bar{x} \in \Re^c : x(Y) < x_O(Y)\}, \tag{19}$$

where $x_O(Y)$ is the cost of the Nash Equilibrium in case of the codec placement modeled as a non-cooperative game (*i.e.*, the solution respecting Eqs. (10) and (11)). A Nash Bargaining Solution is the vector x in the Bargain Set that exhibits the best improvement than the optimal solution without cooperation:

$$NBS = \max_{x \in BS} \prod_{j \in J_x} x_{O,j} - x_j, \tag{20}$$

where J is the set of players that exhibit in x a cost lower than the one with the best solution in the non-cooperative game. Our resolution approach for the codec placement optimization intended as a cooperative game consists in an iterative bargaining, as illustrated in Algorithm 2:

Algorithm 2 Agent behaviour within a cooperative game, where find_Noncoop_Solution(·) is defined in Algorithm 1.

```
Require: τ, α, T_Nash
1:  Function run()
2:  isCodec ← find_Noncoop_Solution(T_Nash);
3:  while true do
4:    feedBacks ← collectFeedbacks();
5:    feedback ← 0,0;
6:    ρ ← getRedundancy(Parent);
7:    λ ← getMonitoredLoss();
8:    if isCodec != true then
9:      λ ← λ + max(feedBacks.second);
10:   end if
11:   if ρ < λ then
12:     feedback.first ← 1;
13:   else
14:     feedback.first ← 0;
15:   end if
16:   feedback.second ← (|λ − ρ|);
17:   sendMessage(Parent, feedback);
18:   if isCodec == true then
19:     δ ← 0;
20:     if sum(feedBacks.first) == 0 then
21:       δ ← min(feedBacks.second);
22:     else if sum(feedBacks.first) == feedBacks.length() then
23:       δ ← max(feedBacks.second);
24:     end if
25:     ρ ← ρ + δ;
26:   end if
27: end while
28: EndFunction
```

- At the beginning, each player runs the Algorithm 1 (row 2); after a timeout, the iterative bargaining is started.
- At each iteration, a player receives from the nodes placed in the lower layer of the multicast tree a set of feedbacks (row 4), and at the same time it sends a similar feedback to the nodes in the higher layer (row 17). Such a feedback is characterized by two elements:
  - a single bit expressing its satisfaction with respect to the received redundancy (*i.e.*, the increment of the generated redundancy (1), or the reduction of such a redundancy (0), as shown at rows 11–15);

– a non negative integer indicating how much the applied redundancy should be incremented or decremented. Such an integer is the difference between the experienced loss pattern and the received one (row 33). If the agent is not a codec, also the loss patterns communicated by the agents in the lower layer are considered in such a computation (rows 8–10).

- A player collects all the received feedbacks and updates the applied redundancy degree according to the following rule (row 25): $\rho(i) = \rho(i-1) + \delta$, where δ is equal to
  - the minimum of the integers in the received feedbacks, if having all the bits equal to 0 (rows 20–21);
  - the maximum of the integers in the received feedbacks, if having all the bits equal to 1 (rows 22–24);
  - 0, otherwise (row 19).

There is no guarantee that such algorithm can bring to the social optimum, but, by construction, it is guaranteed to obtain a better solution than the one given by the Nash Equilibrium.

3.3. Using Bayesian games

All the above presented distributed algorithms assume that the data needed for their convergence towards a given solution is a common knowledge among the participants to the game. Specifically, the messages exchanged during the algorithm are received by the intended destinations and the determination of the loss patterns is accurate. This is a strong and unrealistic assumption, since as long as the notifications may be lost by the network, also these messages can be dropped; moreover, the observation of the experienced loss rate is not accurate and strongly varies over time. The game theory provides tools to deal with this implausible assumption of common knowledge by means of the so-called Bayesian games [32]. In such games, the players have a partial or imperfect knowledge of the relevant data of the game and have to take decisions based on the possessed knowledge and on the beliefs about what they do not know or are uncertain about. More formally, we can consider the Harsanyi's model [22] for Bayesian games, where each player is characterized by a given type, denoted as θ_i, representing its private data, and a joint probability distribution over the types of other players, denoted as $p(\theta_{-i})$, where $\theta_{-i} = \{\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_n\}$, representing the beliefs of the player on the incomplete information of the game by the given player. Such a joint probability distribution over the types assumed by the other players is known a-priori, and is used to compute the conditional distribution $p(\theta_{-i}|\theta_i)$ by using the Bayes rule, from which the name of Bayesian games comes. Such a conditional probability distribution is used to determine the expected cost of a given strategy s_i' done by the ith player of type θ_i:

$$C(s_i', s_{-i}(\cdot), \theta_i) = \sum_{\Theta_{-i}} p(\theta_{-i}|\theta_i) C_i(s_i', s_{-i}(\theta_{-i}), \theta_i, \theta_{-i}). \tag{21}$$

Such a cost function allows us to define the solution of the game in terms of the Bayesian Nash Equilibrium, consisting in the strategy profile $s^*(\cdot)$, where for all players and relative types we have the satisfaction of the following condition:

$$s_i^*(\theta_i) \in arg\min_{s_i' \in S_i} \sum_{\Theta_{-i}} p(\theta_{-i}|\theta_i) C_i(s_i', s_{-i}(\theta_{-i}), \theta_i, \theta_{-i}), \tag{22}$$

Such an equation means that the strategy of each player must be the best reply to other players' strategies given its own belief, and deviation from the best reply strategy is not profitable. Such an equilibrium concept is considered an ex-post Nash equilibrium, since with a probability very close to one such a strategy profile represents a Nash equilibrium of the complete information version of the game [2]. Such an ex-post stability of Bayesian Nash Equilibria simplifies the demonstration of their existence: for Bayesian games with finite types, a Bayesian Nash equilibrium always exists, and the proof is similar to the proof of existence of a Nash Equilibrium in a finite game of complete information. Since we have proved that our game holds Nash equilibria, then its Bayesian formulation is characterized by Bayesian Nash equilibria. In addition, the Bayesian formulation of our game implies that the demands of other nodes in terms of experienced loss rate is not known or partially known. Therefore, the type of a player represents the experienced loss rate λ_i. On the contrary, the redundancy degree received by a given node is known, since such an indication is included in the header of the distributed notifications. Therefore, our Bayesian game has a finite type set, contained between 0 and the maximum experienceable loss rate within the network, so a Bayesian Nash equilibrium exists. Specifically, considering our game, the expected cost function of Eq. (21) can be formulated as follows:

$$\begin{aligned} C(s_i', s_{-i}(\cdot), \theta_i) = \sum_{\Theta_{U_i}} p(\theta_{1\_l}, \ldots, \theta_{u\_l}) \cdot (\alpha_i \cdot s_i' + +(\rho_i + \max_{j \in U_i} \theta_j) \cdot (1 - s_i')) = \alpha_i \cdot s_i' + \\ +(1 - s_i') \cdot (\rho_i + \sum_{\Theta_{U_i}} (p(\theta_{1\_l}, \ldots, \theta_{u\_l}) \max_{j \in U_i} \theta_j)), \end{aligned} \tag{23}$$

where θ_{j_l} indicates the type of the jth player at the lower level in the multicast tree than the ith player and contained in the U_i set. In Eq. (23), we have only a joint probability, since the conditional probability for the types of the other players, given the type of the ith player, is equal to the unconditional probability for the types of the other players, since the two events are stochastically independent. Given Eq. (23), the control behavior of the agents, formalized in Eq. (10), is changed

as follows:

$$\rho_i + \sum_{\Theta_{U_i}} (p(\theta_{1\_l}, \ldots, \theta_{u\_l}) \cdot \max_{j \in U_i} \theta_j) \leq \alpha. \tag{24}$$

Algorithm 1 has been modified in order to deal with incomplete information of the redundancy received by the *i*th agent and the loss rate experienced by children agents of the *i*th agent:

- The *i*th agent measures the losses along the paths towards the agents in the lower level of the multicast tree by pinging them (rows 6–10). From the statistics collected during these measurements, the agent is able to build the probability distribution over the types of the other agents:

$$p(\theta_j = \lambda) = x \in [0,1] : x = \frac{Measurements\ with\ \lambda\ losses}{Total\ Measurements}, \forall \lambda \in [0, M] \tag{25}$$

- The agent computes the Cartesian product of the type sets of the agents in the lower level and the joint probability for the elements resulting from this product (row 11).
- The rest of the algorithm operates similarly to Algorithm 1, by changing the control behavior considering the condition in Algorithm 24 (rows 13–27), with the only exception being the delivery of the message to the Parent in a reliable manner (row 23).

The previous solution is based on the concept of best reply within the context of a game with incomplete information. Therefore, Algorithm 3 has a non-cooperative perspective in resolving the game. As mentioned above, solutions of a

Algorithm 3 Agent behaviour within a non-cooperative Bayesian game.

```
Require: τ, α
1: Function run()
2: find_Noncoop_Bayes_Solution(∞);
3: EndFunction
4: Function find_Noncoop_Bayes_Solution(TermCond)
5: isCodec ← false
6: Δ(U) ← 0;
7: for j ∈ U do
8:   Δ(Θ_j) ← collectLosses(j);
9:   Δ(U) ← Δ(Θ_j);
10: end for
11: P(Δ(U)) ← x(Δ(U));
12: t ← 0;
13: while t < TermCond do
14:   wait(τ);
15:   if isCodec == true then
16:     ρ ← max(Δ(U), P(Δ(U)));
17:   else
18:     ρ ← getRedundancy(Parent);
19:     if ρ + max(Δ(U), P(Δ(U)) > α then
20:       isCodec ← true
21:       ρ ← max(Δ(U), P(Δ(U));
22:     else
23:       rel_sendMessage(Parent, ρ + max(Δ(U), P(Δ(U)));
24:     end if
25:   end if
26:   t ← t +1
27: end while
28: return isCodec;
29: EndFunction
```

non-cooperative games are inefficient with respect to the optimal value of the functions they aim to optimize, and such a difference is the Price of Anarchy. This holds also for games with incomplete information, so the more efficient resolution approach is to model our problem in terms of a cooperative game with incomplete information and to find the Nash Bargaining Solution under incomplete information [17]. We deal with such a game like in the case of the Nash Bargaining Solution expressed in Eq. (20) and resolved with Algorithm 2:

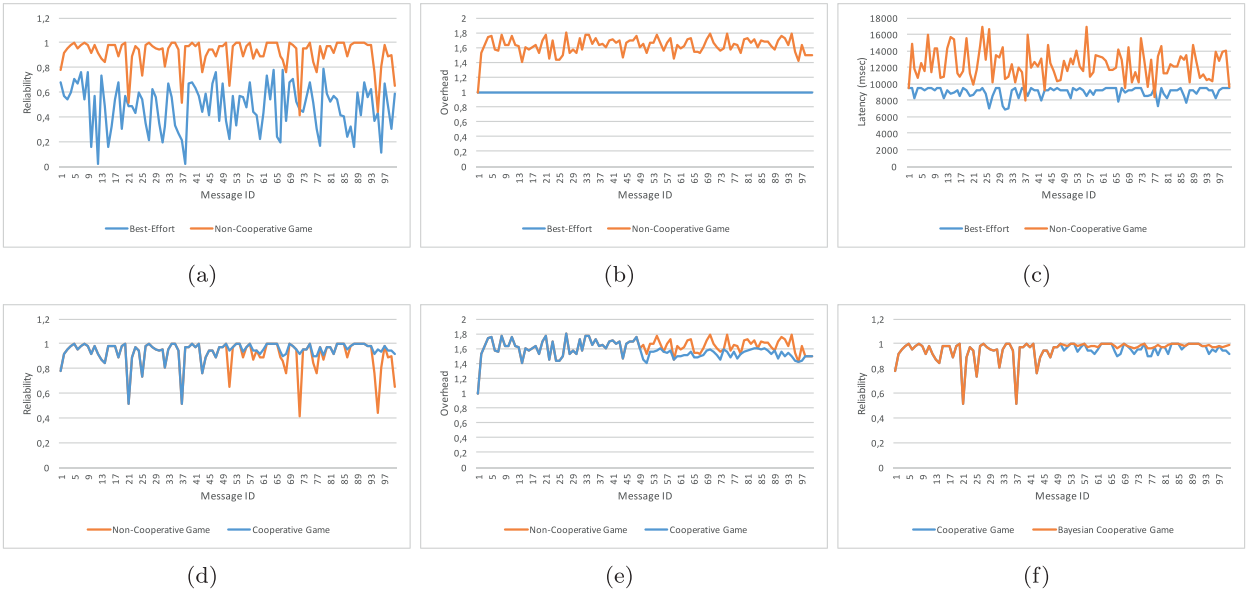

Fig. 5. Comparison of the achievable quality of the proposed solutions.

- Agents start by finding the non-cooperative solution with incomplete information by running Algorithm 3 until reaching the convergence;
- A series of feedbacks is computed and exchanged in a reliable manner among the agents;
- The applied redundancy is adjusted based on the feedbacks collected by the codes according to the rule at row 25 in Algorithm 2.

4. Experimental evaluation

The scope of this section is to present experimental results that study the reliability and overhead (*i.e.*, the traffic generated by the approach in terms of number of packets per link) of the proposed strategic placement of FEC codecs. To achieve this aim, we implemented our solution by using the OMNET++[2] simulator, and decided to not use any real wide-area networks, such as PlanetLab, due to the uncontrollable loss patterns that can make the obtained results non reproducible [7]. The used workload is characterized by exchanged messages with a size of 23 KB, the publication rate of 1 message per second and the total number of nodes equal to 256. The network behavior has 50 *ms* as link delay, and 0.02 as PLR, based on a measurement campaign described in [7], with message losses not independent as proved by [31]. We have assumed that the coding and decoding times are equal to 5 *ms* and 10 *ms*, respectively. We have also considered the block size equal to 1472 bytes, that is the size of the MTU in Ethernet, so that a single event is fragmented in 16 blocks. Finally, without loss of generality, we considered a system with 1 publisher and 39 subscribers, all subscribed to the same topic, *i.e.*, members of the same multicast tree. We have published 1000 events for each experiment, executed each experiment 3 times and reported the average results.

The first experiments involved 64 nodes, and has the objective of comparing the quality obtained by the non-cooperative and the cooperative game with the case of no coding applied within the communication protocol. In the Fig. 5(a), we have the comparison of the reliability of the multicast service without any recovery solution and the one where the codec placement is modeled as a non-cooperative game. Our solution allows the protocol to achieve a higher degree of successfully-delivered notifications to the interested subscribers; specifically, we have an average of 92% of published notifications that have been received correctly, against the average of 47% when no recovery is applied in our simulations. We can notice an increase of the success rate around the 6-th message, i.e., when the game is started. Such a reliability gain is obtained by sending more messages carrying out the redundancy obtained by encoding the data to be disseminated, as evident in the Fig. 5(b), where in average our solution needs 63% more messages than the case with no recovery. Another cost of our coding strategy is an increase of the delivery latency, as shown in the Fig. 5(c), passing from an average of 8978,49 ms to deliver a notification to an average of 12351,93 ms when our coding strategy is applied.

As we have previouslymentioned, a cooperative formulation allows achieving a better solution than a non-cooperative one, and the rest of the figure empirically proves such a statement. Specifically, we let our cooperative approach to start

[2] www.omnetpp.org.

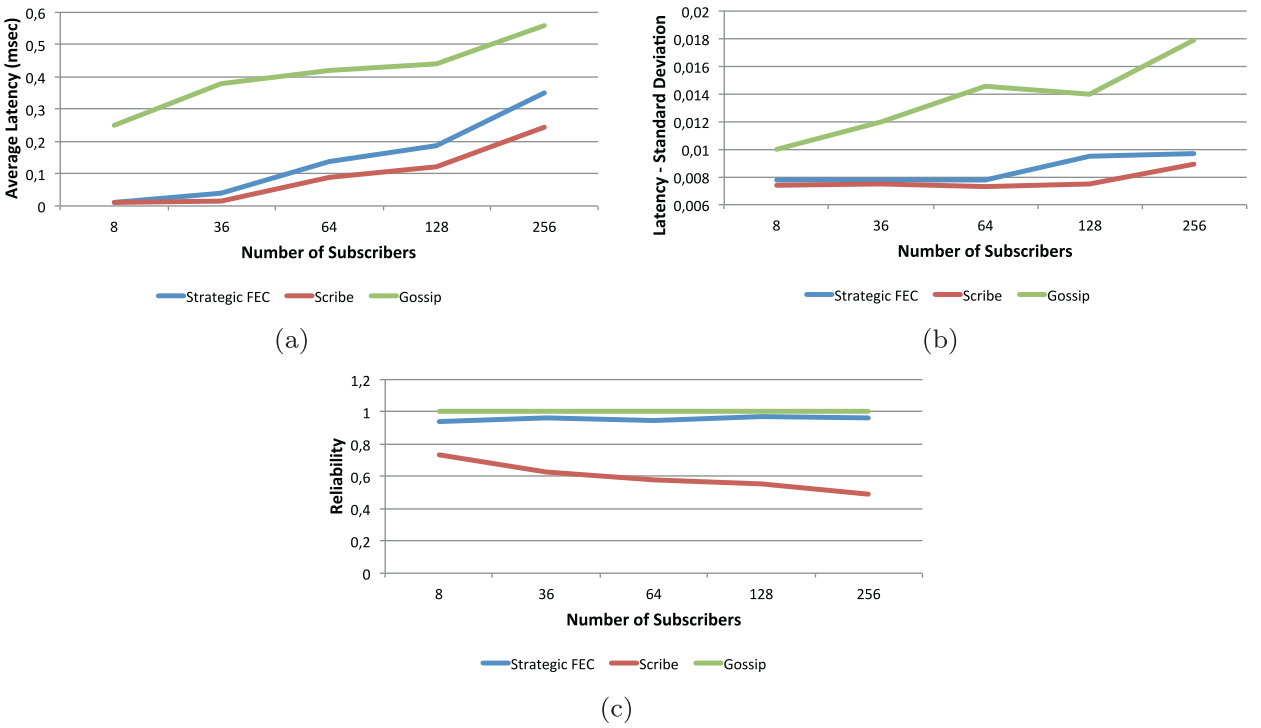

Fig. 6. Comparison of the best proposed solution with the gossiping and the best-effort approach.

at the 50th message, and in fact, we notice an improvement from such a point. Fig. 5(d) compares the achievable success rate of the non-cooperative formulation of our solution and the cooperative one; we can notice that the trend relative to the cooperative game does not have the high spikes of the non-cooperative one, and the average reliability is equal to the 96.14% of notifications that have been successfully delivered during the execution of the cooperative game, which is higher than the one for the non-cooperative game, which was equal to 90.76% in the same execution period. Moreover, the overhead of the solution in terms of the exchanged messages needed to deliver a notification and tolerate the network losses is reduced, having an average of 52.68% of more messages than the case with no recovery, which is lower than the one seen for the non-cooperative game, which is equal to 63.2%. If we consider the latency, we have not seen any considerable variation between the two different solutions. Those considerations allow us to state that the non-cooperative formulation allows obtaining a placement with a lower quality than the cooperative formulation, and this is not surprising. We conclude such a comparison by studying how the uncertainty of the monitored loss patterns affects the obtainable solutions and how a Bayesian formulation can be helpful. Fig. 5(f) illustrates that the solution of a Bayesian formulation of the cooperative game is superior in the quality of the achievable solution than the deterministic formulation in terms of the achievable success rate. In fact, in average, the Bayesian formulation obtains 98.43% of the published notifications that have been successfully delivered, which is higher than 96.14% of notifications achieved by the deterministic formulation. Such a reliability improvement implies a similar worsening of the overhead, equal to about 5.2% of more messages exchanged to correctly disseminate the published notifications.

We have also compared our approach, in its Bayesian cooperative formulation, with the delivery quality achievable with Scribe (with no reliability means) and with Scribe equipped with the Gossip strategy, when varying the number of nodes within the multicast tree. As it is possible to notice in Fig. 6(c), both our approach and Gossip are able to increase the reliability of the delivery protocol offered by Scribe, where Gossip is more reliable due to its retransmission behavior. From the rest of Fig. 6(c), we can notice that the higher reliability of the Gossip is obtained at the cost of a reduced performance, as it has also been observed in [11], while our approach results in a performance comparable to the one of Scribe with no reliability.

5. Conclusions

We have presented an innovative approach to guarantee reliable and timely multicast communications in publish/subscribe services built on top of an application-level multicast infrastructure. Such an approach adopts forward error correction scheme, which adds redundancy to recover lost packets and does not get into the same timeliness limitations of approaches based on retransmission. On the other hand, we avoid the typical problems occurring with coding approaches, *i.e.*, either the drawback of a coarse-grain control on the redundancy degree or a strong performance overhead. Specifically,

we propose an optimal approach that places encoding functionality only in a selected subset of the interior nodes and the root of the tree. The problem of finding where to place the coding nodes has been formulated as an optimization problem, which has been resolved with game theory by comparing non-cooperative, cooperative and Bayesian formulations. Future work will focus on approaching such a problem with other possible solutions, so as to find the best ones in terms of achievable reliability and costs. Last, we planned to solve the codec placement optimization problem with an analytic method finding the optimal solutions and comparing them with the ones found within this, and future, investigation.

References

[1] J. Cardinal, M. Hoefer, Non-cooperative facility location and covering games, Theor. Comput. Sci. 411 (16–18) (2010) 1855–1876.
[2] G. Carmona, K. Podczeck, Games and economic behavior, Am. Econ. Rev. 74 (1) (January 2012) 418–430.
[3] C. Chekuri, J. Chuzhoy, L. Lewin-Eytan, J. Naor, A. Orda, Non-cooperative multicast and facility location games, IEEE J. Sel. Areas Commun. 25 (6) (2007) 1193–1206.
[4] Y. Chu, S.G. Rao, S. Seshan, H. Zhang, A case for end system multicast, IEEE J. Select. Areas Commun. (JSAC) 20 (8) (October 2002) 1456–1471.
[5] C. Diot, B. Levine, B. Lyles, H. Kassan, D. Balendiefen, Deployment numbers for the IP multicast services and architecture, IEEE Netw. Special Number Multicasting 14 (1) (2000) 78–88.
[6] A. Downs, An economic theory of political action in a democracy, J. Polit. Econ. 65 (2) (1957) 135–150.
[7] C. Esposito, Data Distribution Service (DDS) Limitations for Data Dissemination w.r.t. Large-Scale Complex Critical Infrastructures (LCCI), Mobilab Technical Report, March 2011. (www.mobilab.unina.it).
[8] C. Esposito, A. Castiglione, F. Palmieri, M. Ficco, A game-theoretic approach to network embedded fec over large-scale networks, Comput. Intel. Intel. Syst. Commun. Comput. Inf. Sci. 575 (January 2016) 353–364.
[9] C. Esposito, D. Cotroneo, A. Gokhale, D. Schmidt, Architectural evolution of monitor and control systems - issues and challenges, Int. J. Netw. Protocols Algo. 2 (3) (2010) 1–17.
[10] C. Esposito, D. Cotroneo, S. Russo, On reliability in publish/subscribe services, Comput. Netw. 57 (5) (April 2013).
[11] C. Esposito, M. Platania, R. Beraldi, Reliable and timely event notification for publish/subscribe services over the internet, IEEE/ACM Trans. Netw. 22 (1) (February 2014) 230–243.
[12] P. Eugster, P. Felber, R. Guerraoui, A.-M. Kermarrec, The many faces of publish/subscribe, ACM Comput. Surv. (CSUR) 35 (2) (June 2003) 114–131.
[13] P.T. Eugster, R. Guerraoui, A.-M. Kermarrec, L. Massoulié, Epidemic information dissemination in distributed systems, IEEE Comput. 37 (5) (May 2004) 60–67.
[14] EuroControl, EATMS operational concept document, ver. 1.1, 1998. EuroControl ODT Library.
[15] M. Ficco, G. Avolio, L. Battaglia, V. Manetti, Hybrid simulation of distributed large-scale critical infrastructures, in: Proceedings of the Int. Conf. on Intelligent Networking and Collaborative Systems (INCoS 2014), 2014, pp. 616–621.
[16] M. Ficco, G. Avolio, F. Palmieri, A. Castiglione, An hla-based framework for simulation of large-scale critical systems, Concurr. Comput. 28 (2) (2016).
[17] F. Forges, E. Minelli, R. Vohra, Incentives and the core of an exchange economy: a survey, J. Math. Econ. 38 (1–2) (September 2002) 1–41.
[18] M. Ghaderi, D. Towsley, J. Kurose, Reliability gain of network coding in lossy wireless networks, in: Proceedings of the 27th Conference on Computer Communications, 2008, pp. 2171–2179.
[19] M. Goemans, M. Skutella, Cooperative facility location games, J. Algo. 50 (2) (2004) 194–214.
[20] O.M. Group, Data distribution service (DDS) for real-time systems, v1.2, 2007. OMG Document.
[21] T. Hansen, O. Telelis, On pure and (approximate) strong equilibria of facility location games, in: Internet and Network Economics, 2008, pp. 490–497. Lecture Notes in Computer Science 5385.
[22] J. Harsanyi, Games with incomplete information, Am. Econ. Rev. 85 (3) (June 1995) 291–303.
[23] H. Hotelling, Stability in competition, Econ. J. 39 (153) (1929) 41–57.
[24] X. Jin, W.P.K. Yiu, S.H.G. Chan, Loss recovery in application-layer multicast, IEEE Multimed. 15 (1) (January 2008) 18–27.
[25] A. Klose, A. Drexl, Facility location models for distribution system design, Eur. J. Oper. Res. 162 (1) (April 2005) 4–29.
[26] B.N. Levine, J.J. Garcia-Luna-Aceves, A comparison of reliable multicast protocols, Multimed. Syst. 6 (5) (September 1998) 334–348.
[27] N. Li, J. Marden, Designing games for distributed optimization, IEEE J. Sel. Top. Sig. Process. 7 (2) (April 2013) 230–242.
[28] S. Lin, D. Costello, M. Miller, Automatic-repeat-request error-control schemes, IEEE Commun. Mag. 22 (12) (December 1984) 5–17.
[29] D. Loguinov, H. Radha, Measurement study of low-bitrate internet video streaming, in: Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement, 2001, pp. 281–293.
[30] Y. Marinakis, A. Migdalas, P. Pardalos, Cost allocation in combinatorial optimization games, Pareto Optim. Game Theor. Equil. 17 (2008) 217–247.
[31] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, Y. Ganjali, C. Diot, Characterization of failures in an operational IP backbone network, IEEE/ACM Trans. Netw. (TON) 16 (4) (2008) 749–762.
[32] M. Osborne, A. Rubinstein, A Course in Game Theory, The MIT Press, 1994.
[33] J. Reese, Solution methods for the p-median problem: an annotated bibliography, Networks 48 (3) (October 2006) 125–142.
[34] L. Rizzo, L. Vicisano, RMDP: an FEC-based reliable multicast protocol for wireless environments, ACM SIGMOBILE Mob. Comput. Commun. Rev. (MC2R 98) 2 (2) (April 1998) 23–31.
[35] G. Tan, S.A. Jarvis, D.P. Spooner, Improving the fault resilience of overlay multicast for media streaming, IEEE Trans. Parallel Distrib. Syst. (TPDS) 18 (6) (June 2007) 721–734.
[36] A. Vetta, Nash equilibria in competitive societies, with applications to facility location, traffic routing and auctions, in: Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002, pp. 416–425.
[37] F. Wang, Z. Mao, J. Wang, L. Gao, R. Bush, A measurement study on the impact of routing events on end-to-end internet path performance, Comput. Commun. 36 (4) (October 2006) 375–386.
[38] M. Wu, S.S. Karande, H. Radha, Network-embedded FEC for optimum throughput of multicast packet video, J. Sig. Process. 20 (8) (September 2005) 728–742.
[39] W.-P.K. Yiu, K.-F.S. Wong, S.-H.G. Chan, W.-C. Wong, Q. Zhang, W.-W. Zhu, Y.-Q. Zhang, Lateral error correction for media streaming in application-level multicast, IEEE/ACM Trans. Multimed. (T-MM) 8 (2) (April 2006) 219–232.