



Efficient network coding solutions for limiting the effect of packet loss

Samih Abdul-Nabi, Ayman Khalil, Philippe Mary, Jean-François H  lard

► To cite this version:

Samih Abdul-Nabi, Ayman Khalil, Philippe Mary, Jean-Fran  ois H  lard. Efficient network coding solutions for limiting the effect of packet loss. EURASIP Journal on Wireless Communications and Networking, 2017, 2017 (1), 10.1186/s13638-017-0817-3 . hal-01492978

HAL Id: hal-01492978

<https://hal.science/hal-01492978>

Submitted on 22 Mar 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destin  e au d  p  t et    la diffusion de documents scientifiques de niveau recherche, publi  s ou non,   manant des   tablissements d'enseignement et de recherche fran  ais ou   trangers, des laboratoires publics ou priv  s.

RESEARCH

Open Access



Efficient network coding solutions for limiting the effect of packet loss

Samih Abdul-Nabi^{1,2*}, Ayman Khalil¹, Philippe Mary² and Jean-François Héland²

Abstract

In traditional store and forward protocols, lost packets have no impact on the delivery of other transmitted packets. With network coding, the impact of a packet loss may affect the decoding of other transmitted packets thus affecting the entire process of communication between nodes. In this work, we propose a new network coding model that allows generating, coding, decoding and transmission activities on the packets. Based on this model, the impact of lost packets on buffering and the complexity at the receiving nodes is studied and two new mechanisms are proposed to allow the recovery of lost packets. Compared to traditional linear network coding protocol, our mechanisms provide a significant performance amelioration in terms of number of transmissions required to recover from packet loss.

Keywords: Network coding, Packet loss, Recovery, Decision model, Algorithm, Closure

1 Introduction

1.1 Motivations

Recent advancements in network coding (NC) implementations have allowed substantial throughput gain in wireless networks. The simple idea of combining several received or generated packets into one outgoing packet was the starting point for the NC paradigm [1, 2]. Since the publication of [1], researchers have concentrated on the use of NC for optimizing throughput in multicast networks [1, 3, 4] and for improving network reliability through random linear coding [5, 6]. However, only few published works have considered packet loss and packet loss recovery in the analysis of network coding performance [7, 8]. Figure 1 illustrates a NC example with one sender and three receivers over a network with a random packet erasure process. Three cases are considered, a reliable transmission in Fig. 1a, one different packet loss on each link in Fig. 1b, c. To recover the three packets P_1 , P_2 , and P_3 , the scheme in Fig. 1c using NC for the additional retransmission requires only one channel use instead of 3 channel uses with the scheme in Fig. 1b, which does not use NC. Recently, Instantly Decodable NC (IDNC) [9] with its fast decoding potential becomes of major interest for

real time applications. IDNC uses XOR coding operations and offers instantaneous packet decoding when enough information is available at the receiver.

Another popular NC scheme, known as linear network coding (LNC), has been applied to save packet transmission in wireless networks [3, 10, 11]. Using the example of Fig. 1, LNC consists of improving reliability by initially sending linearly independent combinations of the three packets. Suppose the network eraser rate is 20% for example and that the sender broadcasts four linear combinations of the three packets as follows:

$$\underbrace{\begin{pmatrix} R_1 \\ R_2 \\ R_3 \\ R_4 \end{pmatrix}}_R = \underbrace{\begin{pmatrix} \alpha_1 & \beta_1 & \gamma_1 \\ \alpha_2 & \beta_2 & \gamma_2 \\ \alpha_3 & \beta_3 & \gamma_3 \\ \alpha_4 & \beta_4 & \gamma_4 \end{pmatrix}}_A \underbrace{\begin{pmatrix} P_1 \\ P_2 \\ P_3 \end{pmatrix}}_P \quad (1)$$

where A is the coding matrix and P the packet vector. In this case, each receiving node can successfully decode all transmitted packets when receiving at least three of the linear combinations. However, the eraser rate is usually variable and unknown and hence the required number of transmissions is very difficult to predict. To guarantee the delivery of all packets, the number of transmitted linear combinations should be large enough to compensate for any loss, consequently, this can compromise the throughput benefit of NC. Thus, one of the main contributions in

*Correspondence: samih.abdulnabi@liu.edu.lb

¹Lebanese International University, Beirut, Lebanon

²INSA de Rennes, IETR UMR CNRS 6164, Rennes, France

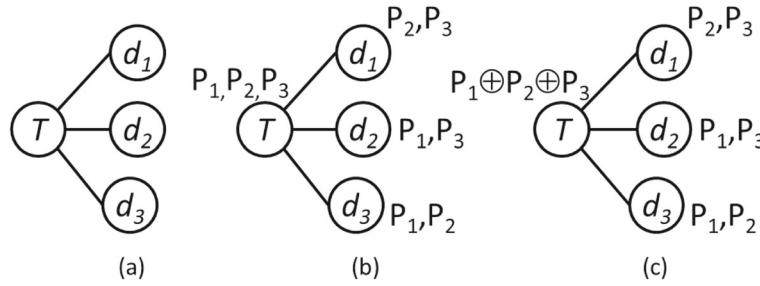


Fig. 1 Network coding example. **a** Channel uses, **b** six channel uses, and **c** four channel uses

the solution detailed later is that it allows, with an optimized number of transmissions, to decode all received combined messages independently of the loss rate.

In the traditional store and forward protocols, when a packet is lost, reliability is achieved by requesting the retransmission of that packet. However, with NC, the decoding of new packets relies on previously received ones. Thus, losing few packets might have a domino effect and decoding further packets could become impossible. This tradeoff between throughput gain with NC and decoding complexity was studied in [12] and [13], for instance, from a decoding delay point of view.

Motivated by this tradeoff and by the decoding feasibility at end nodes, a complete study on the impact of packet loss is conducted in this paper in the case of linear networks [14] and solutions are proposed to deliver all packets in a near optimal number of transmissions.

1.2 Background and prerequisites

Network model: In this paper, the adopted network model is represented as an undirected graph $G = (N, E)$ where the set of nodes N is divided into two exhaustive and mutually exclusive subsets, EN , the set of end nodes and IN , the set of intermediate (or middle) nodes. As illustrated in Fig. 2, A and B are the two end nodes and n_i $i = \{1, 2, \dots, N-2\}$ are the intermediate nodes. End nodes generate packets to be transmitted to each other via intermediate nodes responsible for coding and routing tasks. We denote by *packet* a newly generated message by an end node $\in EN$ and by *coded message* a combination of packets coded together using the XOR operation. Without the loss of generality, NC is assumed to be applied in $GF(2)$. The network in Fig. 2 can be the support of a video conference, data exchange, or other type of information.

The coded message has its own header identifying the two correlated addresses, the number of coded packets, a header for each packet, and some adaptive memory to help guiding the coded message to its destination as illustrated in Table 1. The payload of the coded message consists of all packets coded together by the XOR operation. By inspecting the adaptive memory, a node in the network can determine whether the message should be broadcasted to all neighboring nodes or unicasted to next hop, i.e., next node. The header of each packet in the coded message identifies its source, its destination, a flag that determines if it exists at destination, and its age.

Definition 1 [15] *The age of a packet is defined as the number of times the packet contributes to a coded message and the age of a coded message is defined as the age of the oldest packet in the coded message.*

Definition 2 [15] *The maturity of a network is defined as the highest allowed age for all the packets in the network.*

Packets reaching the maturity age are prohibited from further coding. The idea is to prevent packets from living forever in the network. Packets in a coded message are address-correlated and only those that are classified as address-correlated are allowed to be coded together [16, 17].

Routing rules: Figure 3 clarifies the setting of the “Exist at dest.” flag. At iteration l , the routing table of node n_i shows that the next hop for packet P_A is n_j , and the routing table of node n_k shows that the next hop for packet P_B is n_j . When n_i receives back P_A at iteration $l + 1$ within a coded message it knows that a copy of P_A is in its way to

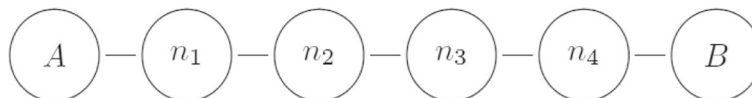


Fig. 2 A network with four intermediate nodes

Table 1 Routing concept

(a) Coding at node n_i		Updates at node n_{i+1}	
Unicast:	False	Unicast:	True
Addr1:	A	Addr1:	A
Addr2:	B	Addr2:	B
Coming from:	n_i	Coming from:	n_{i+1}
Nb. of packets:	2	Nb. of packets:	2
ID:	123	ID:	123
Source:	A	Source:	A
Exist at dest.:	False	Exist at dest.:	False
Next hop:	n_{i+1}	Next hop:	n_{i+2}
ID:	124	ID:	124
Source:	B	Source:	B
Exist at dest.:	False	Exist at dest.:	True
Next hop:	n_{i-1}	Next hop:	n_i

destination thus the “Exist at dest.” flag is set. Similarly, node n_k sets the “Exist at dest.” flag of packet P_B .

Table 1 shows the header of a newly coded message at node n_i . The created message is broadcasted to neighboring nodes (unicast is unset) and both packets in the coded message are unknown to their destinations A and B, respectively, as shown in Table 1a. When the coded message reaches the node n_{i+1} , the header is updated by the node server and the “Exist at dest.” flag of the packet originated by node B is set as shown in Table 1b. Thus, the message continue its way to the destination using unicast. The reader is referred to [16] for more details about routing coded messages in NC. The “Exist at dest.” flag, initially set to false, is updated by the routing nodes and set to true whenever a packet is received back within a coded message after being sent to its destination. For decodability purposes, the NC scheme guarantees that, in any coded message, at most one, packet is unknown by each destination.

Impact of losing a packet: Figure 4 shows a detailed trace of packets exchanged between end nodes A and B where node A generates packets a_i $i = 1, 2, \dots$ and node B generates packets b_i $i = 1, 2, \dots$. A maturity value of 3 is set and each packet is upper-indexed by its age. In this

figure, solid arrows indicate a unicasting activity to the next hop and dashed arrows indicate a broadcasting activity of newly coded messages. It is clearly observed that the multiplicity of a packet, given for example $a1$, is limited and controlled by the use of aging where the age is visible as superscript on each packet. Packet $a1$ generated at iteration 1 is removed from all nodes of the network at iteration 7.

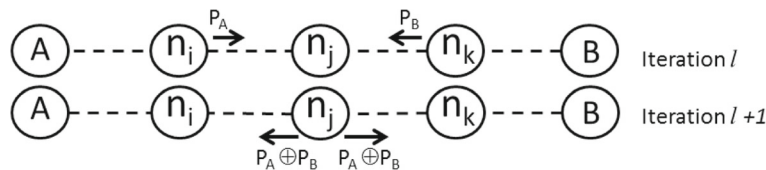
In this context, we are mainly interested in the impact of losing a coded message, for example $(a1 \oplus b2)$ between nodes n_2 and B at iteration 3, on the decoding capability at node B. Failing to receive and decode $a1$, node B will successively receive undecodable packets.

1.3 Related works

In traditional communication protocols, reliability relies on retransmissions, and end to end reliability is assured by the transport layer of the OSI layered communication stack. Moreover, hop to hop reliability is assured by the PHY and MAC layers by retransmitting unacknowledged packets. Recently, hop by hop transport layer reliability protocols have been developed where intermediate nodes store and recover packets to deliver data more efficiently [18, 19]. With NC, reliability and throughput gain are achieved through linear NC [3, 10] where several packets are linearly combined together and broadcasted into the network. Many works in the literature have dealt with the benefits of NC from the reliability point of view [6, 20].

In [6], authors have presented analytical and numerical results for the performance of end-to-end and link-by-link reliability mechanisms based on automatic repeat request (ARQ), forward error correction (FEC) and NC in a tree topology. Using an access point topology with k receivers, they demonstrated that the reliability gain of NC compared to ARQ is of $\Theta(\log k)$. They showed also that allowing intermediate nodes to recover lost packets remains in $\Theta(\log k)$ while further minimizing the number of transmissions. However, it should be noted that buffering packets in intermediate nodes requires decoding coded messages which might be costly. In our proposed solution, decoding only occurs at end nodes while intermediate nodes are only allowed to perform coding activity to minimize the number of transmissions.

Due to lossy wireless channel conditions, recent works have focused on loss recovery taking into consideration

**Fig. 3** “Exist at dest.” flag

Iteration	A	n_1	n_2	B
1	$a1^0$			$b1^0$
2	$a2^0$	$a1^0$		$b1^0$ $b2^0$
3	$a3^0$	$a2^1 \oplus b1^1$	$a1^1 \oplus b2^1$	$b3^0$
4	$a4^0$	$a3^1 \oplus a1^2 \oplus b2^2$	$a2^2 \oplus b1^2 \oplus b3^1$	$b4^0$
5	$a5^0$	$a4^1 \oplus a2^3 \oplus b1^3 \oplus b3^2$	$a3^2 \oplus a1^3 \oplus b2^3 \oplus b4^1$	$b5^0$
6	$a6^0$	$a5^0$ $a3^2 \oplus a1^3 \oplus b2^3 \oplus b4^1$	$a4^1 \oplus a2^3 \oplus b1^3 \oplus b3^2$	$b5^0$ $b6^0$
7	$a7^0$	$a6^1 \oplus b5^1$	$a5^1 \oplus b6^1$	$b7^0$
8	$a8^0$	$a7^1 \oplus a5^2 \oplus b6^2$	$a6^2 \oplus b5^2 \oplus b7^1$	$b8^0$
9	$a9^0$	$a8^1 \oplus a6^3 \oplus b5^3 \oplus b7^2$	$a7^2 \oplus a5^3 \oplus b6^3 \oplus b8^1$	$b9^0$
10	$a10^0$	$a9^0$ $a7^2 \oplus a5^3 \oplus b6^3 \oplus b8^1$	$a8^1 \oplus a6^3 \oplus b5^3 \oplus b7^2$	$b9^0$ $b10^0$
11	$a11^0$	$a10^1 \oplus b9^1$	$a9^1 \oplus b10^1$	$b11^0$
12	$a12^0$	$a11^1 \oplus a9^2 \oplus b10^2$	$a10^2 \oplus b9^2 \oplus b11^1$	$b12^0$
13	$a13^0$	$a12^1 \oplus a10^3 \oplus b9^3 \oplus b11^2$	$a11^2 \oplus a9^3 \oplus b10^3 \oplus b12^1$	$b13^0$
14	$a14^0$	$a13^0$ $a11^2 \oplus a9^3 \oplus b10^3 \oplus b12^1$	$a12^1 \oplus a10^3 \oplus b9^3 \oplus b11^2$	$b13^0$ $b14^0$

Fig. 4 Delivery of packets in a network with no loss

the overhead on the network due to retransmissions. In [21], authors have proposed *Codecast* protocol suitable to low-loss, low-latency constraint applications such as video streaming. Their work is based on random NC [3] implementing localized loss recovery and path diversity with very low overhead. They demonstrated throughput simulations that *codecast* achieves a near-perfect packet delivery ratio while maintaining at the same time lower overhead than conventional multicast. Routing and route selection have a major impact on reliability and time delivery.

To reduce loss impact, routing selection mechanisms have been investigated. In [22], overlay routing networks have been proposed to monitor communication between nodes in order to improve routing of packets and provide loss reduction over traditional routing algorithms. Opportunistic routing [23–25] dynamically selects paths based on loss conditions. However, in [26], authors conducted a comparison of best-path routing and opportunistic routing and found that the benefit of opportunistic routing is much less than commonly believed. It should be well noted that these mechanisms introduce latency on the overall communication time and overhead on the routing nodes. The mechanisms we are proposing are network independent and require no probe from the network.

In the majority of cases, NC is considered as block coding where packets can only be decoded in batches [27]; a packet needs to wait for the arrival of other packets before being able to be decoded. While NC increases throughput and lowers congestion, block decoding may cause major delays and reduction in the quality of service (QoS). Recent works have focused on QoS and proposed

methods to reduce the average waiting time in the buffer of the intermediate nodes [28, 29]. Our work differs from existing works in the sense that we use NC for data exchange activity and not for multicast purposes. With address-correlated NC protocol, no delay is engendered by the coding activity and no packet is delayed waiting for a match. Moreover, instantaneous decoding at end nodes is guaranteed when packets arrive in order. Decoding remains assured in real networks where messages arrive unordered.

1.4 Contributions and paper organization

This work aims to study the impact of packet loss on the decoding capability and resource management. NC schemes where decoding only occurs at destination nodes are investigated. Undelivered packets have impact on some major resources especially buffer size and processing time.

Our contributions are summarized as follows:

- A new NC decision model is presented and implemented at each node of the network. The model is used to manage node queues in a way to avoid packet drop due to the limited queue capacity.
- The impact of packet loss on the decoding capability at the receiver and the decoding perturbation caused by each loss are investigated and analyzed.
- Two new recovery mechanisms initiated by end nodes are proposed to handle packet loss with NC. (i) The immediate retransmission request (IRR) mechanism that uses information extracted from the undecoded messages to request the retransmission of lost packets. (ii) The recovery from packet loss

mechanism (BCSD) with near optimal number of retransmissions.

The paper is organized as follows; the NC decision model to generate, route, code and decode messages is presented in Sections 2 and 3. In Section 4 different scenarios for packet loss are investigated and the impact on decoding capabilities and buffering requirements are recorded. Based on the results from Section 4, solutions are proposed in Section 5.

Simulations and results are presented in Section 6 and we conclude with Section 7.

2 A new NC decision model

2.1 General description of the NC decision model

In the considered network, each node consists of a server managing two queues; the receiving queue that holds messages waiting to be coded and the outgoing queue that holds messages waiting to be transmitted (Fig. 5). The receiving queue collects received messages to investigate coding opportunities for address correlated packets which are only allowed to be coded together. On the other hand, the outgoing queue holds messages ready to be transmitted into the network in broadcast or unicast modes. The presence of two queues facilitates the scheduling of tasks at the node and reduces the time needed to search for matching messages to be coded.

The state diagram of an intermediate node is shown in Fig. 6. In this diagram, the coding phase is represented by a link from the receiving queue to the server where two or more messages are selected from the receiving queue to be coded together and sent to the outgoing queue. Note that with aging, some mature messages are transferred from receiving to outgoing queue without being coded. The transmission of a message to a neighboring node is done through the server.

Similarly, the state diagram of an end node is shown in Fig. 7. In this diagram, the decoding phase is represented by a link to the server where one message is selected to be decoded. A message containing several packets can be decoded if a maximum of one of these packets does not exist at destination. When a packet is decoded, it is transferred to the upper layer. If the server is unable to decode a message, the message is returned to the receiving queue and further retries are performed later when

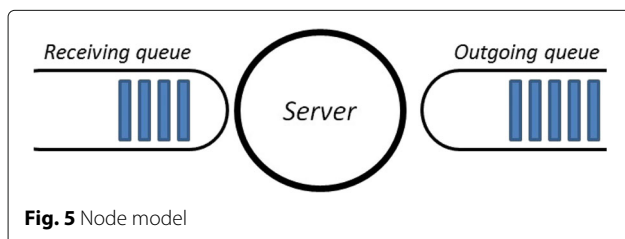


Fig. 5 Node model

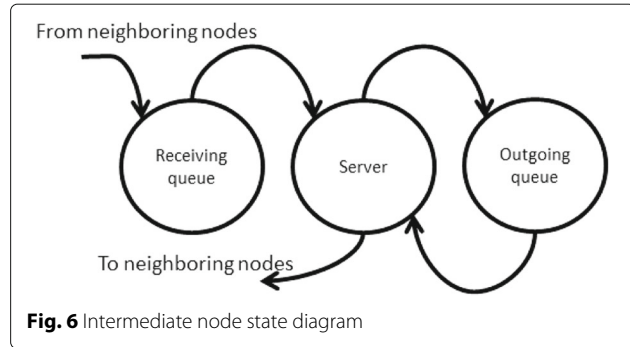


Fig. 6 Intermediate node state diagram

more received packets are available. The generation phase is represented by a link from the server which generates packets according to a model that will be described later. Finally, the transmission of a message to the neighboring node is done through the server.

The receiving queue is used to store all received messages and has a capacity denoted by C_{rq} . We also denote by $x_{rq}(t)$ the number of messages at time t waiting in the receiving queue. The outgoing queue holds messages ready to be transmitted to the network. We denote by C_{oq} and $x_{oq}(t)$ the capacity and the number of coded messages at time t in the outgoing queue, respectively. The server mainly performs one of the following activities: (1) packet generation denoted by g , coding activity denoted by c , (3) sending message denoted by s , and (4) decoding activity denoted by d . The possible activities at nodes are:

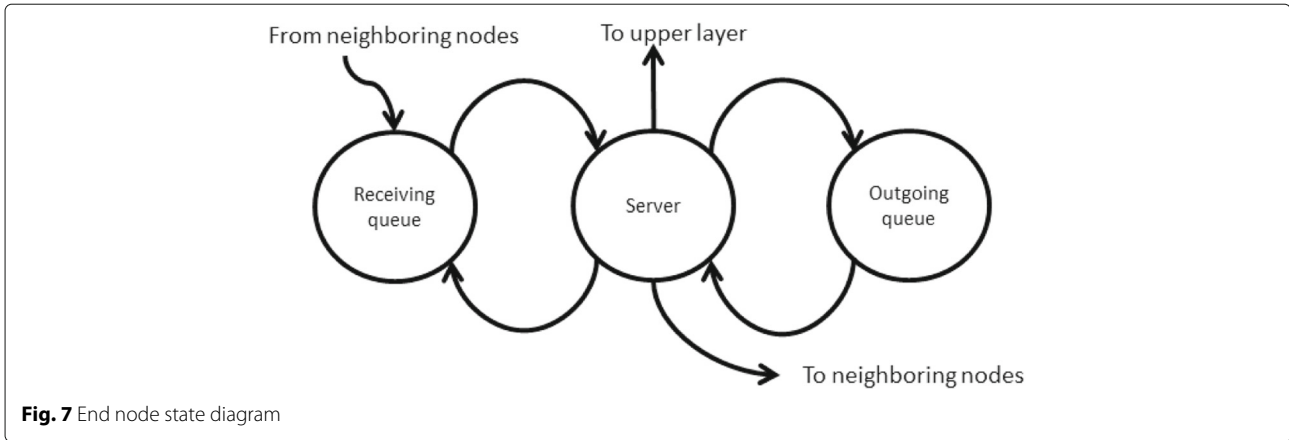
- Sending node activity set = $\{g, s\}$
- Intermediate node activity set = $\{c, s\}$
- Receiving node activity set = $\{d\}$

Note that, when a node plays more than one role, its set of activity is the union of the activity set associated to each role. For example, the set of activities at an end node is $\{g, s, d\}$.

A decision model is needed at each node to select the best activity. Two decision models are adopted for our implementation, a deterministic decision model for intermediate nodes and a probabilistic model for end nodes.

2.2 Deterministic model for intermediate nodes

In an intermediate node where the decision fluctuates between coding or sending messages, the selection of the activity to be performed largely depends on the number of messages in the queues. The decision model needs to constantly balance the occupation of both receiving and outgoing queues. When balanced, the decision model preserves QoS and favors early received messages in both queues to be processed. For that purpose, we propose an integer programming problem where a binary variable y_i is assigned to each possible activity with $i \in \{c, s\}$. y_i is 0 if the activity is not performed and 1 elsewhere. The



optimization problem handled at each intermediate node is described in (2).

$$\max_{y_c, y_s} y_c(t) \sum_{i=1}^{x_r(t)} m_{r_i}(t) + y_s(t) \sum_{i=1}^{x_o} m_{o_i}(t) \quad (2)$$

$$\text{subject to } y_c(t) + y_s(t) = 1 \quad (3)$$

$$y_c(t) \geq \frac{(x_r(t) - x_o(t)) - d_{ro}}{C_r} \quad (4)$$

$$y_s(t) \geq \frac{(x_o(t) - x_r(t)) - d_{or}}{C_o} \quad (5)$$

$$y_c, y_s \in \{0, 1\} \quad (6)$$

In this problem, d_{ro} (respectively, d_{or}) is the acceptable difference in number of messages between the receiving queue and the outgoing queue (respectively, between the outgoing queue and the receiving queue), and $m_{r_i}(t)$ is the amount of time message i has been in the receiving queue while m_{o_j} is the amount of time message j has been in the outgoing queue. The objective of (4) is to force $y_c(t)$ to take the value 1 whenever the difference between the number of messages waiting to be further coded and the number of messages waiting to be transmitted reaches or exceeds d_{ro} . Similar objective for (5) by forcing $y_s(t)$ to be 1 when the difference between the number of messages waiting to be transmitted and the number of messages waiting to be further coded reaches or exceeds the value d_{or} . Constraint (3) guarantees that the server performs exactly one task at a time. It should be noted that (4) and (5) are built to guarantee that only one of the binary variables $y_c(t)$ and $y_s(t)$ is forced to take the value 1, however, both variables are allowed to take the value 0 at the same time which explains the need for (3). In that case, the choice of activity, i.e., $(y_c, y_s) = (1, 0)$ or $(y_c, y_s) = (0, 1)$ is the choice maximizing (2). The objective function forces the selection of the activity that helps releasing messages with the largest waiting time.

2.3 Probabilistic model for end nodes

A binary variable y_i is assigned to each possible activity with $i \in \{d, g, s\}$. y_i can take the values 0 if the activity is not performed and 1 elsewhere. As for the case of intermediate nodes, the server can perform exactly one task at a time. We denote by $P(d)$, $P(g)$ and $P(s)$, respectively, the probability of performing a decoding ($y_d = 1$), a generation ($y_g = 1$), or a sending activity ($y_s = 1$).

The generation of packets is considered as an independent process from other activities and the decision of the activity to be undertaken at end nodes is performed in two phases. During the first phase, a decision is taken on whether or not the node should generate a packet. If no packets are to be generated, we move to the second phase to select, according to a stochastic decision model, the node activity to be performed.

The stochastic decision model is based on [30] where routing packets are given by some probability functions. For our model, we have the following equations:

$$P(d) = \frac{e^{-\beta x_{oq}(t)}}{e^{-\beta x_{oq}(t)} + e^{-\beta x_{rq}(t)}} \quad (7)$$

$$P(s) = \frac{e^{-\beta x_{rq}(t)}}{e^{-\beta x_{oq}(t)} + e^{-\beta x_{rq}(t)}} \quad (8)$$

where β is a control parameter and $P(d) + P(s) = 1$. In (7), the probability of decoding decreases when the number of messages in the outgoing queue increases favoring in this case the sending activity. Furthermore, a similar argument can be given to (8). However, even when the number of coded messages in the outgoing queue (respectively, receiving queue) exceeds the number of coded messages in the receiving queue (respectively, outgoing queue), the activity selection process remains stochastic and there is still some probability of selecting the decoding (respectively, sending) activity.

As noted in [30], if we set $\beta = 0$, both probabilities will be equal to 1/2 and decoding or sending are chosen randomly. Furthermore, if β grows infinitely, the limits of

$P(d)$ and $P(s)$ will be, respectively, 0 and 1 if $x_{oq}(t) > x_{rq}(t)$ thus creating a deterministic decision model where the activity is selected according to queues occupation.

Note that, a delayed message at an intermediate node has a domino effect on the delivery of other messages. This imposes a QoS requirement to avoid such effect which is assured by the deterministic model. In revenge, delayed messages at end nodes do not have such effect and the probabilistic model reveals to be flexible and requires less computational time.

3 Activity selection algorithms

Models presented in Section 2 are used to propose two-low cost algorithms to select the best activity to be performed at intermediate and end nodes, respectively. These algorithms are initiated each time the node is requested to select and complete a task. The activity selection at the intermediate node is shown in Algorithm 1.

Algorithm 1 Activity selection algorithm: intermediate nodes

```

1: if  $\frac{(x_r(t) - x_o(t)) - d_{ro}}{C_r} > 0$  then
2:   return Coding
3: else if  $\frac{(x_o(t) - x_r(t)) - d_{or}}{C_o} > 0$  then
4:   return Sending
5: else
6:    $delay_r = 0$ 
7:   for  $i = 1$  to  $x_r(t)$  do
8:      $delay_r = delay_r + (t - \text{arrival time of } i)$ 
9:   end for
10:   $delay_o = 0$ 
11:  for  $i = 1$  to  $x_o(t)$  do
12:     $delay_o = delay_o + (t - \text{arrival time of } i)$ 
13:  end for
14:  if  $delay_r > delay_o$  then
15:    return Coding
16:  else
17:    return Sending
18:  end if
19: end if

```

In this algorithm, steps 1 and 3 represent constraints (4) and (5) in the optimization problem presented in (2) and select the activity to perform depending on the occupation of the receiving and outgoing queues. When queues are almost balanced, steps 6 and 17 attempt to maximize (2) helping late packets to be released from the queues and improving QoS.

The activity selection at end nodes is presented in Algorithm 2. In this algorithm, step 2 independently selects the generation activity with a certain predefined

Algorithm 2 Activity selection algorithm: end nodes

```

1:  $P(\text{generation}) = \text{random}(0, 1)$ 
2: if  $P(\text{generation}) \leq (\text{Generation-threshold})$  then
3:   return Generation
4: end if
5:  $P(\text{decoding}) = \frac{e^{-\beta x_{oq}(t)}}{e^{-\beta x_{oq}(t)} + e^{-\beta x_{rq}(t)}}$ 
6: if  $P(\text{decoding}) \leq \text{random}(0, 1)$  then
7:   return Decoding
8: else
9:   return Sending
10: end if

```

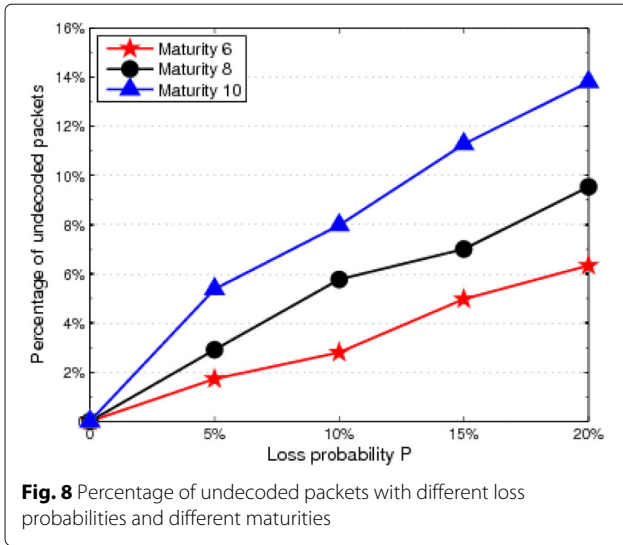
probability. If no new packets are generated, the code between lines 5 and 10 give higher probability to the activity that balances the receiving and outgoing queues.

4 Packet loss simulation

Algorithms presented in Section 3 are validated by running simulations on a linear network of 6 nodes, i.e., with four intermediate nodes as shown in Fig. 2. Statistics are recorded about the loss impact on decoding opportunity and buffering time at end nodes. For this purpose, the new NC decision model presented in Section 2 is used to built a specific application to simulate the network layer at each node of the network. As example, only the hop $n_3 \rightarrow n_2$ is considered with loss and the packet loss is modeled as an independent and identical Bernoulli distributed process. The contribution of this section is of twofold; first, to understand the correlation between the packet loss and the number of undecoded packets at the receiver, and second, to aid to establish a recovery process to minimize the number of retransmissions.

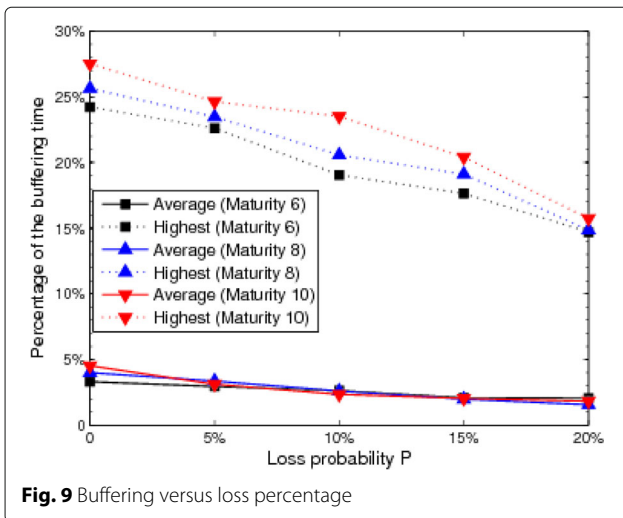
4.1 Correlation between lost and undecoded messages

In this simulation, the number of packets exchanged between A and B is set to be 10,000 packets. During the same simulations, the impact of aging is also observed [15]. The results of simulation for each specific percentage of packet loss are presented in Fig. 8. The percentage of undecoded packets corresponds to the percentage of different packets that have been identified in the header of the received messages without being decoded. The simulation is repeated for three different maturity values. Figure 8 shows the impact of the aging concept on the number of undecoded messages. As shown in the figure, the number of undecoded packets at the receiving node is proportional to the maturity. This is justified by the fact that when maturity increases, coded packets live longer in the network and are further coded with other packets leading to an increase in the number of undecoded packets at the receiver. Moreover, the number of undecoded



packets at the receiver is finite and the system resumes receiving decodable packets after each loss.

In Fig. 9, the buffering of received packets at the destination nodes is studied. Destination nodes need to buffer sent and received packets so that they will be used in the decoding process. Buffering is costly since it requires significant memory space. Buffering time of a packet is computed as the laps of time between the moment the packet was received and the last time the packet was used for decoding other packets. The figure gives the percentage of the buffering time compared to the total communication time versus the loss probability P , for three different maturity values. Furthermore, for each case, the highest values and the average values are represented. These results show that the buffering time decreases when the loss percentage increases. This is justified by the fact that less decoding occurs when more coded messages are lost. Moreover,



the buffering also varies with maturity. When maturity is increased, more buffering time is required since packets live longer in the network.

4.2 Decoding opportunity

In this section, we study the multiplicity of the lost packets in the network and its impact on the decoding process. By multiplicity, we mean the number of copies of the packet that exist in the network. Recall that, when a packet is coded with others, the coded message is broadcasted into the network creating duplicates of these coded packets. When a packet is lost, other copies of that packet still exist in the network creating undecoding opportunities at the receiver.

To clarify how end nodes can receive undecodable messages, consider the example shown in Fig. 10. In this example, a coded message m is created and broadcasted at iteration l by node n_j (shown in the figure as m_1 and m_2 which are identical). Consider now that message m_1 is lost and m_2 reaches node n_i . All packets in m_2 originated from A are flagged as exist at destination ("Exist at dest." flag set to true). Node n_i uses message m_2 in further coding activity and broadcasts at iteration $l+1$ a new coded message $m' \oplus m_2$. Node B cannot decode $m' \oplus m_2$ since it holds two packets originated by A and unknown to B .

Let us denote by $\mu \in \mathbb{N}^+$ the maturity age of the messages. Theorem 4.1 sets a boundary on the dissemination of each packet in the network.

Theorem 4.1 [15] *Given a network $G = (N, E)$ and a maturity scalar μ in $\mathbb{N}^+$. The number of coded messages in the network containing a specific packet is bounded by 2^μ .*

Note that, due to the linear nature of the network, each end node receives a maximum of $\frac{2^\mu}{2}$ messages containing the same packet. This upper bound on the multiplicity of a packet is used in the following theorem to estimate the number of undecoded messages when a packet is lost.

Theorem 4.2 *Given a network $G = (N, E)$ and a maturity scalar μ in $\mathbb{N}^+$. When a packet loss occurs in the direction of end node E , the estimated number of undecoded messages that can be received by E as a result of that loss is given by*

$$\frac{1}{\mu + 1} \sum_{i=2}^{\mu} 2^{\mu-i} \quad (9)$$

Proof If a packet in the direction of E is lost at age 0, i.e., before being coded, no other copies of that packet exist in the network. Consequently, no undecodable messages are received by E . On the other hand, if a packet is lost in the direction of E at maturity age, the other

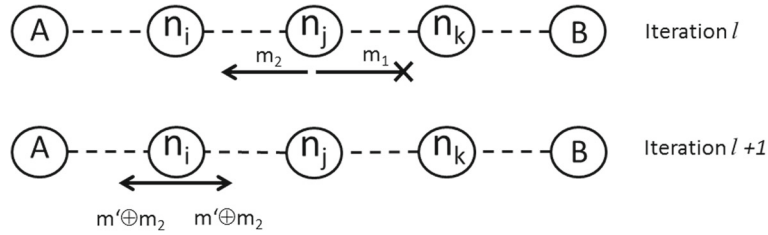


Fig. 10 Lost packet discovery

copy of that packet is unicasted in the opposite direction. Consequently, no undecodable messages are received by E . Let P^a be a packet with age a , and N_{P^a} the number of messages received by E that are created starting from P^a . According to Theorem 4.1, $N_{P^a} = \frac{2^{\mu-a}}{2}$. Referring to the coding graph presented in [15], we assume that in a network with maturity μ , there is an equal probability that a lost packet has an age between 0 and μ . The total number of undecoded messages that can be received by E is:

$$\frac{1}{\mu+1} \sum_{i=1}^{\mu-1} N_{P^i} = \frac{1}{\mu+1} \sum_{i=1}^{\mu-1} \frac{2^{\mu-i}}{2} = \frac{1}{\mu+1} \sum_{i=1}^{\mu-1} 2^{\mu-(i+1)}$$

By taking $i+1 = j$, we can write

$$\frac{1}{\mu+1} \sum_{j=2}^{\mu} 2^{\mu-j}$$

□

As an example, a packet loss might lead to receiving four undecodable messages when maturity is set to 6, after which the system resumes receiving decodable messages.

Another important factor that helps in reducing the number of received undecodable messages is related to the model presented in Section 2. Recall that the model allows packets to be transmitted without coding and never waits for a match. A simulation is performed to count the number of packets that are traveling through the network without being coded. Figure 11 shows the number of uncoded messages in both queues of each node of the network. End node generates uncoded messages while they receive the lowest percentage of uncoded messages. There is, however, a significant percentage of uncoded packets floating between the nodes of the network. This is in fact due to the following factors:

- Aging: preventing packets from being further coded.
- Decision model: forcing packets to be transmitted without waiting for a match.

Analysis in this section shows an interesting characteristic of NC when it comes to lossy network. The intensive

use of NC, represented by a large maturity value, keeps duplicates of a lost packet in the network and the number of undecodable packets degenerate. One of the advantages of aging is that it helps creating mature messages in the network thus preventing coupling with duplicates of lost packets.

5 Proposed mechanisms

Two new mechanisms are detailed in this section. To further evaluate the performance of these mechanisms, a revised version of the LNC protocol is also presented in this section.

5.1 Immediate retransmission request

The end node receiving an undecodable message is able to identify, by reading the header of the undecodable message, those packets that are flagged as “exist at dest.” but are not in its buffered packets for decoding purpose. Without waiting for a retransmission event from the upper layer (i.e., transport layer) a request of retransmission is initiated by the NC protocol for these packets. Note that the retransmission request initiated by the network layer of the receiving node is answered by the network layer of the sending node since it buffers also transmitted packets for

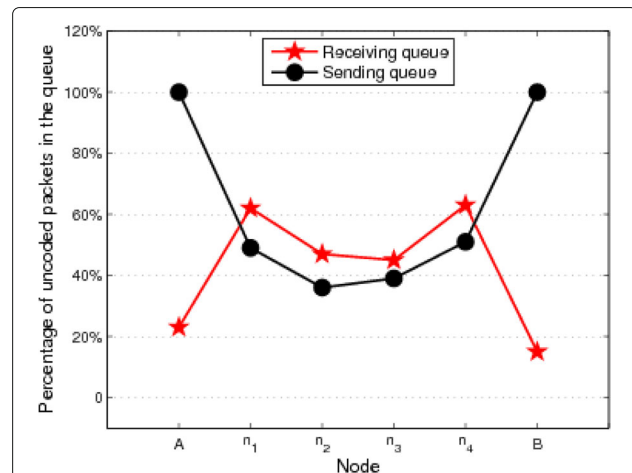


Fig. 11 Percentage of uncoded messages at each node

decoding purposes. These requesting and retransmission operations are transparent to the TCP protocol.

Due to delays and unordered arrival of messages at the receiver, one undecodable message is not a sign of packet loss. The receiving node suspects that a loss occurs when undecodable messages remain undecodable for a specific period of time, denoted by t_{time} (threshold on time) or when receiving a certain number of consecutive undecodable messages, denoted by t_{cup} (threshold on consecutive undecoded packets).

The IRR algorithm is designed to take into consideration the previously mentioned concerns. The IRR algorithm is shown in Algorithm 3 where UM denotes an undecoded message. The algorithm works as follows: initially, consecutive undecoded packets (cup) is set to 0, request retransmission (rr) is set to false and the list of requested packets (l_{rp}) is empty. The algorithm then iterates on each entry in the list of undecoded messages. Lines 4 to 11 are dedicated to identify consecutive undecoded messages and a request for retransmission is initiated if cup reaches a predefined threshold. Lines 12 to 14 investigate the buffering time of each undecoded message and request retransmission if a threshold on time is reached. The request for packet retransmissions are sent in the header of newly generated packets or immediately if no new packets are available for transmission. Since no buffering of packets is performed in the intermediate nodes, the request for retransmission is sent to the generator nodes. The algorithm is initiated each time the node is about to perform a decoding activity.

Algorithm 3 Immediate retransmission request

```

1:  $cup=0, i=1, rr=false, rp = \emptyset$ 
2: while  $i \leq \text{number of UM}$  do
3:   Add to  $rp$  the undelivered packets in  $UM(i)$ 
4:   if  $UM(i)$  and  $UM(i-1)$  are consecutive then
5:     Increment  $cup$ 
6:     if  $cup \geq t_{cup}$  then
7:        $rr = true$ 
8:     end if
9:   else
10:     $cup=0$ 
11:   end if
12:   if Current time - Receiving time of  $UM(i) \geq t_{time}$  then
13:      $rr = true$ 
14:   end if
15:   Increment  $i$ 
16: end while
17: if  $rr = true$  then
18:   Request all needed packets in  $rp$ 
19: end if

```

IRR differs from the ARQ of the end to end transport layer in that IRR is faster to identify lost packets and to initiate a retransmission request. However, it should be noted that IRR does not replace ARQ. As shown in the proof of Theorem 4.2, a packet might be lost without decoding consequences at the receiving nodes. Such a loss can only be detected and requested by the transport layer using ARQ.

The simulation results of IRR are shown in Fig. 12. The simulation ran with the three values of maturity and with the same conditions than in Fig. 8 of Section 4. Figure 12 shows that the number of requested packets increases with maturity for the same packet loss percentage. Furthermore, the number of retransmissions compared to the number of received undecodable messages is presented in Table 2 for two loss percentages, where the undecoded and the requested columns are extracted from the results of Figs. 8 and 12, respectively. Table 2 reveals a reduction in the percentage of requested packets when maturity increases. This is justified by the fact that, with larger maturity, a loss generates more undecodable messages and a retransmission helps also decoding more packets.

5.2 Recovery from packet loss

With NC, recovery from packet loss differs from traditional retransmission protocols since packets are interconnected together by the mean of coded messages. We showed that losing one packet might lead the way to lose additional packets due to the nature of the coded messages and of the decoding process. For this reason, it is obvious that an efficient algorithm is needed in order to minimize the number of retransmissions.

Due to NC, end nodes receive multiple copies of the same packet as shown previously in Fig. 4. When a loss occurs on a link, a copy of a packet is lost but other

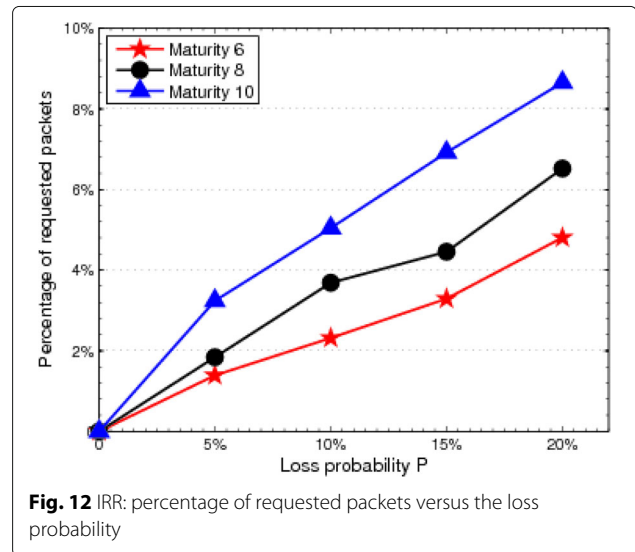


Table 2 Percentage of retransmission versus undecoded

Maturity	Loss probability P					
	10%			20%		
	Undecoded	Requested	Percentage	Undecoded	Requested	Percentage
6	280	205	73.2%	634	486	76.7%
8	577	371	64.3%	954	615	64.5%
10	798	503	63.0%	1381	865	62.6%

copies might still be present in the network. End nodes receive messages that they are unable to decode but might contain such copies. In what follows, we propose algorithms that request the minimal number of retransmissions in order to retrieve lost packets. Note that retransmitted packets can be coded with newly generated packets leading to further reduction in the number of retransmissions.

Before continuing with the recovery process, we give some definitions and then propose a mechanism to identify the packets to be requested from the sender in order to recover all lost packets with minimal retransmissions.

5.2.1 Closure and covering sets

Let S be a set of packets and C be a set of coded messages.

Definition 3 When XOR operation is used to code packets together, the **simple closure** of S , denoted by S_C^+ , is the set of packets that can be decoded from C using packets in S .

To better understand the simple closure definition, we illustrate the following example.

Let $S = \{p_1, p_2, p_3\}$ and $C = \{p_1p_4, p_1p_3p_5, p_4p_5p_7, p_5p_6p_7, p_6p_8p_9, p_7p_8p_9\}$.

To build the simple closure of S , we proceed as follows:

$$\begin{aligned}
 S_C^+ &= \{p_1, p_2, p_3\} \\
 S_C^+ &= \{p_1, p_2, p_3, p_4\} && \text{using } p_1p_4 \\
 S_C^+ &= \{p_1, p_2, p_3, p_4, p_5\} && \text{using } p_1p_3p_5 \\
 S_C^+ &= \{p_1, p_2, p_3, p_4, p_5, p_7\} && \text{using } p_4p_5p_7 \\
 S_C^+ &= \{p_1, p_2, p_3, p_4, p_5, p_7, p_6\} && \text{using } p_5p_6p_7
 \end{aligned}$$

Definition 4 When linear combination is used to code packets together, the **complex closure** of S , denoted by S_C^{++} , is the set of packets that can be decoded, by the mean of Gaussian Elimination, from C using packets in S .

Going back to the previous example, the complex closure of S , is denoted by: $S_C^{++} = S_C^+ \cup \{p_8, p_9\}$ where p_8 and p_9 can be obtained by solving the system:

$$\begin{cases} \alpha_1 p_8 + \alpha_2 p_9 = R_1 \\ \beta_1 p_8 + \beta_2 p_9 = R_2 \end{cases}$$

where R_1 and R_2 are the received coded packets and $(\alpha_1, \alpha_2)^t$ and $(\beta_1, \beta_2)^t$ are the corresponding coding vectors normally included in the header of the received coded packets.

Definition 5 S is a **covering set** of C if the closure of S includes all packets that are part of the coded messages in C .

Going back again to the previous example, we see that S_C^{++} is a covering set of C but S_C^+ is not.

Definition 6 A **basic covering set** of C is a covering set of C with minimal cardinality (having the least number of packets).

5.2.2 Recovery mechanism

The recovery process starts when the receiving end node receives a message that cannot be decoded, i.e., it has more than one unknown packet. At that time, the end node expects to receive additional undecodable packets. Each received undecodable message is XOR-ed with all known packets and the resulting chunk of undecoded packets XOR-ed together is added to an originally empty set C .

The gathering period of undecodable chunks of messages is controlled by a timer that starts immediately after the reception of the first undecodable coded message. As stated before, undecodability might be caused by unordered arrival of messages and the problem might be solved by itself when more messages arrive to the end node. The duration of the timer is an application-specific parameter based on the communication medium and the type of application. It can vary from milliseconds in the case of instant messaging where messages need to be delivered in order to the upper layer, to seconds in the case of file exchange where only the complete file needs to be ordered. When the timer times out, the recovery mechanism is launched. The recovery mechanism consists of finding a basic covering set and packets in that covering set are requested from the sender. The Basic Covering Set Discovery (BCSD) algorithm is presented in Algorithm 4. The algorithm follows a greedy mechanism that tries, within a reasonable amount of time, to identify a basic covering set for all chunks of messages in C . The algorithm

starts by building the incident matrix where columns identify the packets in C and rows represent undecoded chunks of messages sorted by receiving time. For the set $C = \{p_1p_4, p_1p_3p_5, p_4p_5p_7, p_5p_6p_7, p_6p_8p_9, p_7p_8p_9\}$ the corresponding incident matrix M is

$$M = \begin{matrix} & p_1 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 \\ \begin{matrix} p_1p_4 \\ p_1p_3p_5 \\ p_4p_5p_7 \\ p_5p_6p_7 \\ p_6p_8p_9 \\ p_7p_8p_9 \end{matrix} & \begin{pmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & 1 & \\ & & & & & 1 & & \\ & & & & & & 1 & 1 \\ & & & & & & & 1 & 1 \end{pmatrix} \end{matrix}.$$

The algorithm then iterates on the incident matrix until the covering set is built. At each iteration, the column with the highest number of 1's is selected (i.e., highest occurrence in the undecoded messages) and the corresponding packet is added to the covering set B . Rows with one entry are then removed together with the column of the entry since the corresponding packet can be decoded. The algorithm stops when the incident matrix becomes empty.

By applying BCSD to the incident matrix M , the basic covering set found is $S_C^+ = \{p_5, p_7, p_8\}$.

Algorithm 4 Basic covering set discovery (BCSD)

```

1: Build the incident matrix
2:  $B$  an empty set
3: repeat
4:    $i = 1$ 
5:    $maxj = 0$ 
6:    $max1 = 0$ 
7:   for  $j = 1$  to  $NbColumns$  do
8:     if sum of 1's in column  $j > max1$  then
9:        $maxj = j$ 
10:     $max1 = \text{sum of 1's in column } j$ 
11:   end if
12: end for
13: add  $maxj$  to  $B$ 
14: Remove column  $maxj$ 
15: repeat
16:   if there is a row with one entry then
17:     Remove the row and column of that entry
18:   end if
19: until No more columns can be removed
20: until Matrix is empty
21: Request packets in  $B$  from the sender

```

5.3 Revised linear network coding

To make a fair comparison with both mechanisms presented earlier, a revised version of the LNC mechanism is introduced in this section. The idea of the revised

LNC protocol is to use LNC for unicast and to send enough packets through a lossy network that uses the new NC decision models for intermediate and end nodes, in order to:

- overcome the loss of packets in the links of the network,
- overcome the undecodability of packets at the end nodes.

The revised LNC protocol uses block coding as follows: each sending node groups b packets together and sends b_Δ linearly independent combinations of the packets in each group. The cardinality b of each block and the number of combinations b_Δ are determined according to Theorem 4.2 and according to the loss percentage between the end nodes of the network given by

$$P_n = 1 - (1 - P)^{N-1}, \quad (10)$$

where N is the total number of nodes and P is the probability of loss in a link.

Once P_n is identified, the sending node needs to send $b_\Delta = \left\lceil \frac{b}{1-P_n} \right\rceil$ independent linear combinations of the packets of each block to guarantee the delivery of the packets in the block. In order to minimize the number of transmissions, the cardinality of the block should be selected to satisfy $\frac{b}{1-P_n} = \left\lceil \frac{b}{1-P_n} \right\rceil$.

In addition, each loss in the network leads, by Theorem 4.2, to have $U_n = \frac{1}{\mu+1} \sum_{i=2}^{\mu} 2^{\mu-i}$ undecodable messages that might reach the destination. With P_n as loss percentage in the network, $U_n \times P_n$ undecodable messages can reach the end node. To bypass the undecodability problem, b_Δ is augmented by $P_n \times b_\Delta \times U_n \times (1 - P_n)$.

6 Simulation and results

In this section, both IRR and BCSD algorithms are implemented and tested. Simulations are run on linear network of six nodes with different loss probabilities on any link. Results are compared to both traditional store and forward with ARQ protocols and to the revised LNC protocol presented in Section 5.3. Along the simulations, 10,000 packets are exchanged between both end nodes of the network. Algorithms are compared according to the total number of transmissions needed to deliver all the exchanged packets between the end nodes. In the store and forward protocol, the transport layer at the end nodes automatically claims unacknowledged packets. The total number of recorded transmissions to reliably deliver all the packets is recorded. In a reliable network of six nodes, five transmissions are needed to deliver each packet from one end node to another. For our algorithms, two series of runs are performed and compared to the other solutions. The first one with NC using IRR and the second one with NC using BCSD.

Results are presented in Fig. 13. Traditional store and forward algorithm starts with a total of 50,000 transmissions on a reliable network and ends with an increase of 34% when the loss probability reaches 20%.

Also on a reliable network, revised LNC, IRR, and BCSD start with almost the same number of required transmissions to deliver all the packets. When the loss probability increases, the number of transmissions with the revised LNC increases about 56% when the loss probability reaches 20%. Both NC with IRR and NC with BCSD show an important saving in the number of transmissions compared to the revised LNC. With a loss probability of 20%, NC with BCSD represent a saving of 22% in the number of transmissions over the revised LNC. IRR lightly increases the number of transmissions compared to BCSD.

Figures 14 and 15 show, respectively, the total delivery time and the average delivery time for the three competing mechanisms. The total delivery time is the time elapsed between the transmission of the first packet and the reception of the last one while the delivery time of a packet is computed as the time between the first transmission of the packet and its delivery. In these simulations, each transmission from one node to an adjacent one requires 1 ms of time. As shown in Fig. 15, the revised LNC overcomes the proposed mechanisms when it comes to average delivery time. This is justified by the fact that no retransmission is requested. With the revised LNC, the slow increase in the average delivery time is due to the time needed at the receiver to gather enough messages of each block in order to decode the packets of the block. As for our proposed mechanisms, IRR bypasses BCSD when it comes to average delivery time, since IRR reacts immediately when a loss is detected while BCSD waits longer to gather undecoded messages and reduces the size of the covering set; thus, the number of retransmissions.

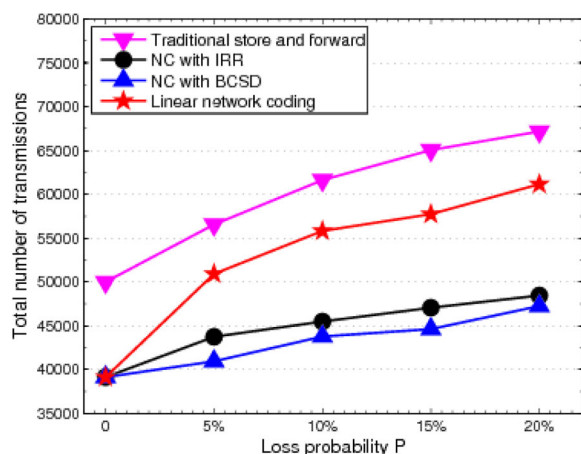


Fig. 13 Total number of transmissions required to deliver 10,000 packets

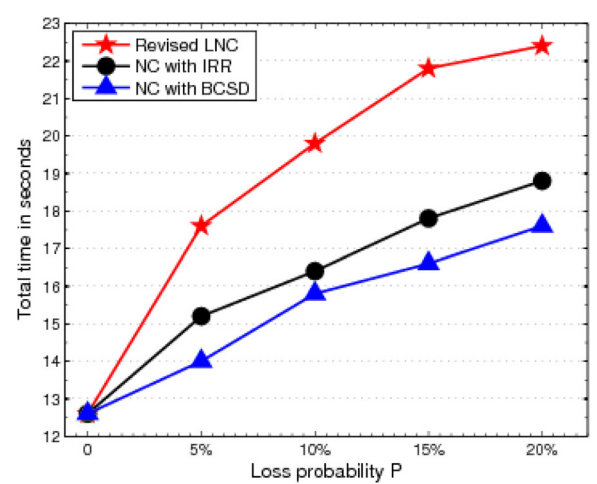


Fig. 14 Total delivery time of the 10,000 packets exchanged

7 Conclusions

This paper deals with the impact of packet loss in NC. First of all, a new decision model has been presented that prioritizes the tasks that should be performed by each node of the network. The model balances the receiving and the sending queues of the node and manages to guarantee QoS by releasing late packets to destination. The new model has been used to study packet loss effect on the decoding process at end nodes. Simulations have shown that with the proposed model and with the use of aging, any perturbation caused by losing packets is finite in time and the receiving nodes receive a limited number of undecoded packets. The number of undecoded packets at end nodes has been analytically and experimentally studied under the effect of aging and NC activities and two new mechanisms have been proposed to improve NC performance in lossy networks. The first mechanism, called

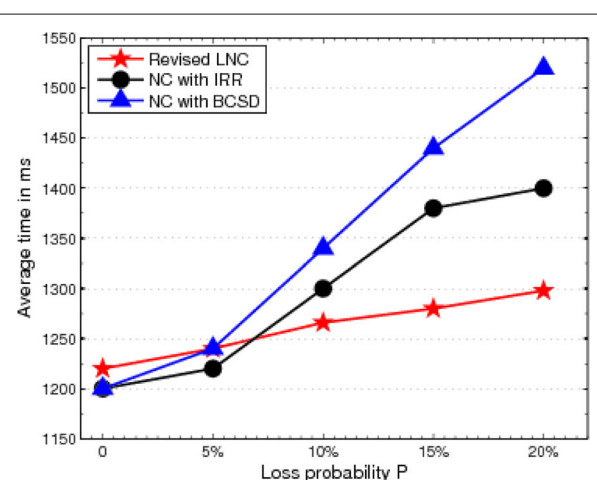


Fig. 15 Average delivery time

IRR, immediately requests lost packets when a packet loss occurs and the second mechanism, called BCSD, requests missing packets in a reduced number of transmissions. Simulation results have shown that in typical cases, both algorithms help reduce by 22% the number of transmissions compared to the revised LNC protocol. Finally, it should be noted that IRR and the revised LNC are more suitable to some applications like video conferencing where the average delivery time is needed to be reduced while BCSD is more convenient to file exchange where the total delivery time needs to be optimized.

Abbreviations

ARQ: Automatic repeat request; BCSD: Basic covering set discovery; FEC: Forward error correction; IRR: Immediate retransmission request; LNC: Linear network coding; NC: Network coding; QoS: Quality of service

Competing interests

The authors declare that they have no competing interests.

Received: 17 April 2015 Accepted: 30 January 2017

Published online: 20 February 2017

References

1. R Ahlswede, N Cai, SYR Li, RW Yeung, Network information flow. *IEEE Trans. Inform. Theory*. **46**(4), 1204–1216 (2000)
2. JK Sundararajan, D Shah, M Médard, M Mitzenmacher, J Barros, Network coding meets TCP: Theory and implementation. *Proceedings of the IEEE*. **99**(3), 490–512 (2011)
3. T Ho, M Médard, R Koetter, DR Karger, M Effros, J Shi, B Leong, A random linear network coding approach to multicast. *IEEE Trans. Inform. Theory*. **52**(10), 4413–4430 (2006)
4. A Agarwal, M Charikar, On the advantage of network coding for improving network throughput. *IEEE Information Theory Workshop*, 247–249 (2004)
5. T Ho, R Koetter, M Médard, DR Karger, M Effros, in *Proc. IEEE International Symposium on Information Theory ISIT, Yokohama, Japan, Jun 29 – July 4*. The benefits of coding over routing in a randomized setting, (2003), p. 442
6. M Ghaderi, D Towsley, J Kurose, in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*. Reliability gain of network coding in lossy wireless networks, (2008), pp. 2171–2179. *IEEE*
7. S Rayanchu, S Sen, J Wu, S Banerjee, S Sengupta, Loss-aware network coding for unicast wireless sessions: design, implementation, and performance evaluation. *ACM SIGMETRICS Perform. Eval. Rev.* **36**(1), 85–96 (2008)
8. DS Papaliopoulos, J Luo, AG Dimakis, C Huang, J Li, in *INFOCOM, 2012 Proceedings IEEE*. Simple regenerating codes: network coding for cloud storage, (2012), pp. 2801–2805. *IEEE*
9. S Sorour, S Valaee, Completion delay minimization for instantly decodable network codes. *IEEE/ACM Trans. Netw. (TON)*. **23**(5), 1553–1567 (2015)
10. S-Y Li, RW Yeung, N Cai, Linear network coding. *IEEE Trans. Inform. Theory*. **49**(2), 371–381 (2003)
11. M Effros, M Médard, T Ho, S Ray, D Karger, R Koetter, B Hassibi, Linear network codes: a unified framework for source, channel, and network coding. *DIMACS Series Discrete Math. Theor. Comput. Sci.* **66**, 197–216 (2004)
12. C Fragouli, D Lun, M Médard, P Pakzad, in *41st Annual Conference on Information Sciences and Systems. CISS'07*. On feedback for network coding (IEEE, 2007), pp. 248–252
13. RA Costa, D Munaretto, J Widmer, J Barros, in *IEEE MASS, Atlanta, Georgia, Sep. 2008*. Informed network coding for minimum decoding delay (IEEE, 2008), pp. 80–91
14. J Skulic, KK Leung, in *IEEE 23rd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), 2012*. Application of network coding in wireless sensor networks for bridge monitoring (IEEE, 2012), pp. 789–795
15. S Abdul-Nabi, A Khalil, P Mary, J-F Héland, Aging in network coding. *IEEE Wireless Commun. Lett.* **4**(1), 78–81 (2015). *IEEE*
16. S Abdul-Nabi, A Khalil, J-F Héland, in *Third International Conference on Communications and Information Technology (ICCIT)*. Routing coded messages in wireless networks (IEEE, 2013), pp. 295–299
17. S Abdul-Nabi, A Khalil, J-F Héland, in *9th International Wireless Communications and Mobile Computing Conference (IWCMC)*. Efficient network coding packet selection model for qos-based applications (IEEE, 2013), pp. 509–514
18. JF Mingorance-Puga, G Maciá-Fernández, A Grilo, NMC Tiglaio, in *6th EURO-NF Conference on Next Generation Internet (NGI)*. Efficient multimedia transmission in wireless sensor networks (IEEE, 2010), pp. 1–8
19. H Lee, Y Ko, D Lee, in *Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops. PerCom Workshops*. A hop-by-hop reliability support scheme for wireless sensor networks (IEEE, 2006), p. 5
20. C Fragouli, D Katabi, A Markopoulou, M Médard, H Rahul, in *Military Communications Conference, 2007. MILCOM 2007. IEEE*. Wireless network coding: opportunities & challenges, (2007), pp. 1–8. *IEEE*
21. J-S Park, M Gerla, DS Lun, Y Yi, M Médard, Codecast: a network-coding-based ad hoc multicast protocol. *IEEE Wireless Commun.* **13**(5), 76–81 (2006)
22. DG Andersen, AC Snoeren, H Balakrishnan, in *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*. Best-path vs. multi-path overlay routing (ACM, 2003), pp. 91–100
23. S Biswas, R Morris, Opportunistic routing in multi-hop wireless networks. *ACM SIGCOMM Comput. Commun. Rev.* **34**(1), 69–74 (2004)
24. K Zeng, W Lou, H Zhai, Capacity of opportunistic routing in multi-rate and multi-hop wireless networks. *IEEE Trans. Wireless Commun.* **7**(12), 5118–5128 (2008)
25. S Chachulski, M Jennings, S Katti, D Katabi, Trading structure for randomness in wireless opportunistic routing. **37**(4), 169–180 (2007). *ACM*
26. Z Zhong, S Nelakuditi, in *4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks. SECON'07*. On the efficacy of opportunistic routing (IEEE, 2007), pp. 441–450
27. W-L Yeow, AT Hoang, C-K Tham, Minimizing delay for multicast-streaming in wireless networks with network coding. *IEEE INFOCOM*, 190–198 (2009). *IEEE*
28. W Chen, KB Letaief, Z Cao, Buffer-aware network coding for wireless networks. *IEEE/ACM Trans. Netw.* **20**(5), 1389–1401 (2012)
29. E Drinea, L Keller, C Fragouli, Real-time delay with network coding and feedback. *Phys. Commun.* **6**, 100–113 (2013)
30. T Ohira, R Sawatari, Phase transition in a computer network traffic model. *APS, Phys. Rev. (E)*. **58**(1), 193–195 (1998)

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Immediate publication on acceptance
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com