On the Performance of Network Coding in Multi-Resolution Wireless Video Streaming

Steluta Gheorghiu   Luísa Lima   Alberto Lopez Toledo   João Barros   Muriel Médard

***Abstract*— Scalable video can be used to provide video streaming reliably to an heterogeneous set of receivers with different subscription levels. However, the performance of such schemes can be highly affected by scheduling constraints and unreliable feedback. Network coding, on the other hand, has been shown to reduce scheduling and prioritization problems and to perform well in wireless scenarios with perfect feedback. Motivated by this observation, we implement and analyze a system architecture for network coding-based multiresolution video streaming in a wireless environment. In contrast to existing work, we take into account realistic feedback, where the control packets are sent over the same unreliable channel as data packets, and compare it to the case of perfect feedback, where the server has perfect knowledge of the state of the buffer at every receiver. We provide an evaluation of the system via simulation and show that even in highly volatile environments, a network coding-based scheme with limited and unreliable feedback can achieve a good performance.**

I. Introduction

Although layered coding [1] can ensure graceful degradation in the presence of packet losses and differentiated service provision to distinct users, its conjugation with network coding in a wireless setting has not been yet fully explored.

As an example, consider the scenario shown in *Fig. 1*, in which nodes A, B and C are interested in a video stream served by node S, but they have paid for different video qualities (different layers of a multi-resolution video stream). Node S can connect to the receivers through three relay nodes in wireless range, but with poor channel quality. Due to the unreliability of the wireless medium, S needs to retransmit the lost packets using the feedback received from A, B and C to ensure reliable video transmission. This task becomes even more challenging when feedback messages do not arrive in a reliable manner. Moreover, the relays need to synchronize and schedule transmissions to ensure every receiver gets all the packets without duplicates. However, layered coding is likely

L. Lima (luisalima@dcc.fc.up.pt) is with the Instituto de Telecomunicações (IT) and the Department of Computer Science, Faculdade de Ciências da Universidade do Porto, Portugal. J. Barros (jbarros@fe.up.pt) is with the Instituto de Telecomunicações (IT) and the Departamento de Engenharia Electrotécnica e de Computadores, Faculdade de Engenharia da Universidade do Porto, Portugal. M. Médard (medard@mit.edu) is with the Research Laboratory of Electronics at the Massachusetts Institute of Technology (MIT RLE). A. Lopez Toledo (alopezt@tid.es) and S. Gheorghiu (steluta@tid.es) are with Telefonica Research, Barcelona, Spain. This work was carried out with assistance of financial support from the European Community under grant FP7-INFSO-ICT-215252 (N-Crave Project) and from the Fundação para a Ciência e Tecnologia (Portuguese Foundation for Science and Technology) under grant SFRH/BD/24718/2005. A. Lopez Toledo is supported by the Institució Catalana de Recerca i Estudis Avançats (ICREA). Telefonica Research participates in Torres Quevedo subprogram (MICINN), cofinanced by the European Social Fund, for Researchers recruitment.

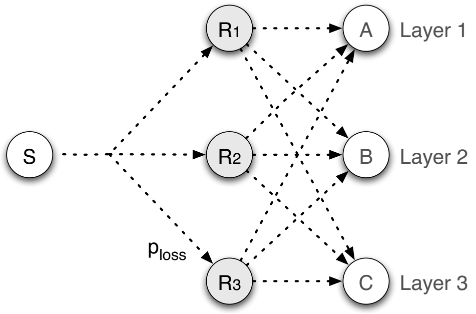


Fig. 1. A source S streams video to 3 sink nodes A, B and C through relay nodes R_1, R_2 and R_3 in a wireless setting. The probability of dropping a packet in each link (in dashed) is p_{loss}. The sinks subscribed for different video quality, thus one must devise mechanisms to ensure reliable delivery over the wireless medium.

to yield prioritization and scheduling problems in wireless scenarios. For instance, [2] has shown that even the apparently simple prioritization of the base layer is not a trivial task. Under this scenario, video quality can be decreased, with video frames not delivered in a timely fashion and therefore skipped.

Network coding (i.e. algebraic mixings of packets in a network [3]) is regarded as a promising approach to tackle the problems above. Random Linear Network Coding (RLNC) is a completely distributed fashion to implement network coding protocols. Nodes draw several coefficients at random and use them to form linear combinations of incoming packets [4]. The resulting packet is sent along with the global encoding vector, which enables the receivers to decode by means of Gaussian elimination. The benefits of network coding for wireless communications have been shown in [5], [6] and [7]. Network coding can also minimize the decoding delay by using feedback [8], making it suitable for multimedia streaming [9], [10], [11].

Our main goal is to propose a RLNC-based system architecture for layered video streaming and perform a reality check of its performance in a high-loss scenario with unreliable feedback. The main features of this architecture are credit-based redundancy control and carefully matched video layers and network codes. The decoder uses an adaptive playing policy and feedback is used to limit the number of transmissions at the server. Based on a detailed simulation study we compare the performance of the proposed scheme with that of a traditional RLNC approach, as well as of an approach without network coding, when the receivers send *realistic feedback*. As a benchmark, we also show the performance of the schemes for the case of *perfect feedback*. For the case of *perfect feedback*, we consider that the control packets are not delayed or lost and the server has perfect knowledge about the state of the buffer at each receiver. The results are valid for a general multicast setting in which several heterogenous devices subscribe to multi-resolution streaming video in a lossy wireless network.

II. Network Coding for Video Streaming

We consider a wireless network where the source and relay nodes only have access to the identifiers of the sinks (e.g. the IP addresses), without centralized knowledge of the network topology or of the encoding functions. We also adopt the model of video layers from [12], illustrated in *Fig. 2*. Video data is divided into groups of pictures (GoPs) with a constant duration of 1 second. In the paper, we use the terms video segment and GoP interchangeably. The data is then encoded into L layers; each layer is divided into a fixed number of packets. Each layer is dependent on all previous layers (that is, layers $1, \ldots, (i-1)$ are necessary to decode layer i).

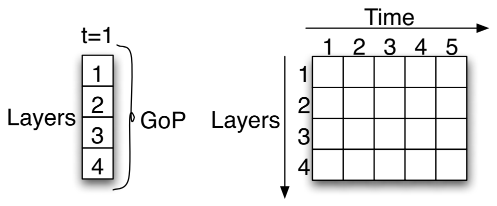


Fig. 2. Layer model. The video data is divided into groups of pictures (GoP) with the duration of 1 second. GoPs are then subdivided into layers.

We now introduce the proposed scheme and elaborate on its main properties.

A. Scheme Operation

For each GoP, the *source* generates an $n \times n$ lower-triangular matrix $\mathbf{A}$, in which n is the number of layers in the GoP. Matrix $\mathbf{A}$ is used for encoding at the source only. Each non-zero entry of $\mathbf{A}$ is an element a_{ij} chosen uniformly at random from all non-zero elements of the field $\mathbb{F}_q \backslash \{0\}$.

The GoP is then divided into vectors $\underline{b}_1 \ldots \underline{b}_x$, with the first symbol of each vector from layer 1, the next symbol from layer 2, etc. The number of vectors created[1] is $x = \lceil \text{size of GoP / n} \rceil$. The payload is then formed by vectors $\underline{c}_1, \ldots, \underline{c}_x$, where $\underline{c}_i = \mathbf{A}\underline{b}_i$. The packets are composed by the header (which includes the coefficient vector) and the payload. Note that, because of the nested structure of coding determined by the triangular matrix, a packet from layer 1 corresponds to the first line of matrix $\mathbf{A}$, a packet from layer 2 corresponds to the second line of matrix $\mathbf{A}$, etc, so that each packet of layer i includes packets from layers $i, i-1, \ldots 1$. Moreover, when encoding one packet of layer i with a packet of layer $j > i$, the resulting packet belongs to layer j.

The *relays* encode packets according to the rules of standard RLNC protocols. Relays identify the layer of a packet by looking at the coefficient vectors, and packets are mixed with packets of the same or lower layers only.

The *receivers* apply Gaussian elimination following standard RLNC.

[1]For clarity, we ignore inconsistencies regarding the proportion between the number of symbols in the layers.

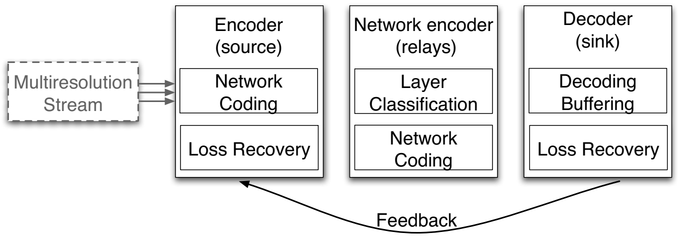


Fig. 3. Modules of a potential system implementation. Generation of a multiresolution stream is in dashed since it is an external entity to the evaluated system.

B. Scheme Properties

Note that traditional RLNC [4], [13] mixes all packets by using a full square matrix. This, however, is not suitable for layered coding, since it is not possible to extract individual layers unless one matrix is used for each layer. The triangular matrix coding effectively mixes the layers, allowing for differentiated recovery of successive layers by nodes with different access levels, while relying on the dissemination of lower-level packets to achieve the resilience necessary for higher-level packets to be delivered in a timely fashion. Moreover, the triangular matrix form provides priority to the base layer, as all upper layer packets contain the base layer. Thus, the common prioritization and scheduling of lower layers is solved in a natural way. In *Section IV* we compare this scheme with traditional RLNC addressing scheduling and prioritization issues.

The choice of a triangular matrix further meets an important requirement: it allows us to remove the arbitrary delay introduced by the typical RLNC full-matrix at the source, since the source can code packets as soon as they are generated and does not have to wait for the end of the generation to send them. Note that if the triangular structure of the matrix is preserved through the network, the sinks are able to decode on-the-fly by means of forward substitution (if there are no losses in the network). Although an alternative strategy – *online network coding* [14] – has been shown to provide the same advantage, it requires full feedback at intermediate nodes. Furthermore, it is not clear how to combine it with layered coding without adding to the scheduling and synchronization problems.

III. System Setup

This section introduces the practical scheme that is used for evaluation in *Section IV*. The system architecture is shown in *Fig. 3*. We now discuss each of its components.

A. Source Encoder

The *source encoder* includes *loss recovery* and *network coding* modules. An important aspect of the encoder is the rate at which intermediate nodes generate and send linear combinations to the receiver. If a relay generates and forwards a linear combination every time he receives an innovative packet from the server, then many redundant packets may arrive at the destination. To solve this issue, the server generates a credit for each coded packet, which is further assigned to one of the intermediate relays [13]. Next, only the relay who receives also the credit associated with the packet is allowed to send a linear combination.

After transmitting a complete generation, and before streaming the next one, the server starts the loss recovery process. To recover lost packets, the server sends redundant linear combinations for each layer, mixing all packets of the layer. This process continues until all the receivers for that layer can decode or the server has another segment to stream.

B. Network (Relay) Encoder

The *network encoder* is a component of the wireless relays of the network and includes *layer classification* and *network coding*. As mentioned in *Section II*, packets of layer l should only be combined with packets of lower layers, i.e. , $l, l-1, \dots, 1$. This is done in order to maintain the diversity of layers in the network, because when combining a packet of layer l with layer $l+1$, the layer of the resulting packet is $l+1$. After classifying the packet, a relay generates and forwards a linear combination if he received the credit assigned to that packet.

C. Decoder

The *decoder* is a component of the receiver that includes *decoding and buffering* and *feedback*. When enough packets are received, the receiver performs Gaussian elimination to decode the packets.

Since, in the presented scheme, relay nodes perform coding on the packets of the same (and lower) layers, the shape of the triangular matrix sent by the source is not kept through the network. Thus, a received packet, even if innovative in terms of rank, might not be decodable immediately and should be stored at a decoding buffer at the receiver. This decoding buffer takes into account the maximum allowable delay of the video stream, similar to the play buffer at the receivers, and will preemptively flush the current undecoded packets if the delay requirement is not met. Once a full layer is decoded, it is stored in the playback buffer.

A node starts the playback once it decodes a number of segments in the lowest quality. If a frame is not received until the time of playback, then it is discarded and the subsequent frame is played instead. Likewise, if the frame is available in a lower quality, it is played in a lower quality than the one the node has access to. At timestep k the node plays segment k in the quality in which it is available. If the segment was not decoded (not even in the lowest quality), then the node stops the playback process and starts buffering. If after some buffering timeout, the node decodes segment k, then it plays it in the quality in which it is available; otherwise, the node skips segment k and plays the next one.

D. Feedback mechanism

We consider a system with minimal feedback, in order to free the wireless channels from unnecessary transmissions. The receivers send positive feedback to the server whenever they decode a segment in the desired quality. For example, a layer 3 receiver sends a unique feedback packet when it has decoded layers 1, 2 and 3.

When the server receives a feedback packet from a layer l receiver for segment k, it updates the information for the loss recovery process as follows. If all the receivers of layer l have decoded, then the server will send redundant packets for layer $l+1$, provided that l is not the highest layer, i.e. $l < L$, where L denotes the number of layers in the system. If l is the highest layer in the system $l = L$, then the server moves with the error recovery at the next segment, that is it will send redundant packets for layer 1 of segment $k+1$.

Finally, the feedback packets are protected against loss in the unreliable medium by *hop-by-hop retransmissions*. Whenever a node N_1 receives a feedback packet from node N_2, N_1 sends an *ack* to N_2. If the *ack* does not arrive within a timeout τ, then N_2 retransmits the feedback packet.

IV. Evaluation

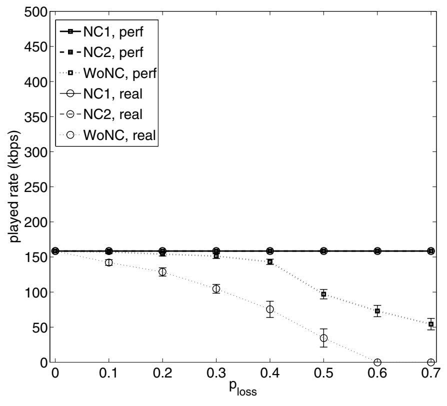
We evaluate the performance of the protocol described in *Section III* via simulation in the multi-hop multi-path scenario from *Fig. 1*, in which the server S sends video to 3 heterogenous receivers A, B and C, through relays R_1, R_2 and R_3, over wireless links. In this section we will show the performance of the scheme in terms of throughput and robustness to losses, and its ability to deliver quality video to a heterogeneous set of receivers, when the feedback from the receivers is *realistic*, i.e. it is sent over unreliable links.

We compare the layered network coding model (*scheme NC1*) with standard RLNC (*scheme NC2*) and an implementation without network coding (*scheme WoNC*). In *scheme NC2* the server sends a different stream for every layer. Each segment is encoded in different qualities, using a full coefficient matrix for each layer. Relay nodes perform RLNC operations on the received packets that belong to the same generation and to the same or lower layers. In this case, since a sink of layer L needs to receive a full-rank matrix for layers $1, 2, \dots L$, sinks acknowledge each layer that they decode. Error recovery is similar to *scheme NC1*. In *scheme WoNC*, the server sends the native packets without coding them and the intermediate nodes just forward uncoded packets normally. The sinks send as feedback the *ids* of the packets they received. If some packets are lost, the server retransmits them.

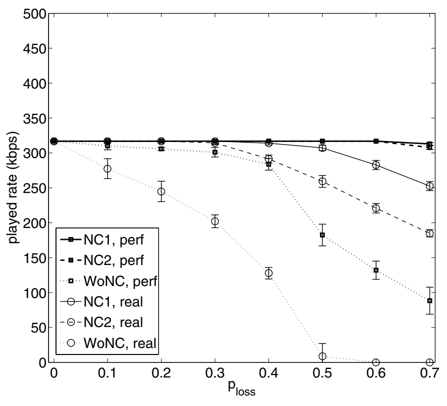
As a benchmark, we also show the performance of the schemes for the case of *perfect feedback*. For the *perfect feedback* case, we consider the control packets are not delayed or lost and the server has perfect knowledge about the state of the buffer at each receiver.

A. Simulation Setup

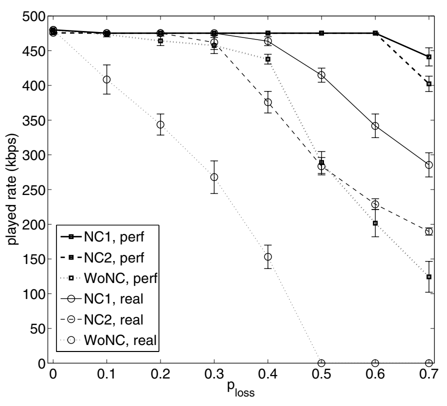
We use the ns-2 simulator 2.33 [15], with the default random number generator for this version. The network coding libraries are independently programmed. The video stream is a constant bit rate traffic over UDP, where the server is streaming at 480 kbps during 100 seconds. Each layer has a fixed size of 20 packets and we consider 3 layers for the system, which yields a generation of 60 packets, corresponding to 1 second of video. The packet size is 1000 bytes. As a propagation model, we use *two-ray ground* and we consider



(a) Layer 1



(b) Layer 2



(c) Layer 3

Fig. 4. Played rate in function of loss probability p_{loss}, for the scheme described in *Section II* (*NC1*), three streams with network coding (*NC2*) and without network coding (*WoNC*). The circles on the curves denote the case of *realistic feedback* and the squares denote the case of *perfect feedback*.

the loss probability p_{loss} as a simulation parameter. Since it was shown that RTS/CTS has a negative impact on the performance, we disable it for all experiments. In order to simulate heavy loss conditions, we also disable MAC layer retransmissions. The rate at the MAC layer is 11 Mbps.

The receivers start to playback the video stream once they have decoded at least 5 segments of the lowest quality. The buffering timeout for a segment that has not been decoded until its playback deadline arrives is set to 1 second. We fit the timeout for feedback retransmissions τ through experiments and set it to 200 ms. In order to take full advantage of the broadcast nature of the wireless medium, the relays listen to transmitted packets in promiscuous mode.

We consider the following metrics: (i) *played rate* at the receivers, (ii) *initial buffering delay*, the time interval from receiving the first packet to the beginning of the playback, (iii) *decoding delay*, the time elapsed from receiving the first packet of a a segment until that segment is decoded, (iv) *skipped segments*, percentage of segments skipped at playback, (v) *lower quality segments*, percentage of segments played in lower quality than the one requested, (vi) *playback quality*, average quality in which each segment is played and (vii) *load on the server*, defined as the ratio between the total rate sent by the server and the streaming rate. In all plots, each point is the average of 10 runs and the vertical lines show the standard deviation. When the behavior is very similar for all 3 layers, we only show the plot for layer 3. The behavior for layers 1 and 2 is always slightly better, since layer 3 receivers need to receive more packets than lower layer nodes.

B. Results

Fig. 4 shows the rate played by each receiver vs. loss probability. When *feedback is perfect*, *Scheme NC1* and *Scheme NC2* achieve the maximum played rate for each layer due to the inherent reliability of network coding, even if the probability of loss increases. The performance of *Scheme WoNC* decreases as p_{loss} increases because the receivers send less feedback and the server does fewer retransmissions, thus the lost packets are not recovered. For the case of *realistic*

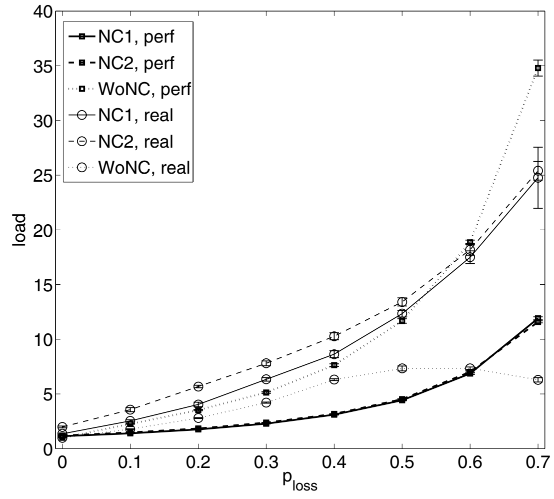


Fig. 5. The load on the server in function of the loss probability p_{loss}.

feedback, the best performance is achieved by *Scheme NC1*. *Scheme NC2* is more affected by losses because it needs more feedback. As explained above, each receiver sends a control packet when it decodes each layer allowing the server to progress with the loss recovery process, thus this scheme is more sensitive to feedback packets being lost. *Scheme WoNC* performs even worse in this case, because as the loss increases fewer data packets that trigger the feedback mechanism arrive at destinations. Next, these feedback packets may not reach the server due to the unreliable medium, and the server does not retransmit lost packets.

We can see in *Fig. 5* that the load on the server grows exponentially as the loss probability increases (for the case of *perfect feedback*). The network coding schemes must send less coded packets to recover losses, because a linear combination may recover different losses at different receivers, while for *Scheme WoNC* the server needs to retransmit exact packets. The exponential behavior of the load on server is similar for *realistic feedback* for *Scheme NC1* and *Scheme NC2*. Note that with *Scheme NC2* the server sends more packets than with *Scheme NC1* due to its sensitivity to lost feedback. Consider for example that all receivers decoded layer 1, but the server received only the feedback from nodes A and B, then the server will continue to send redundant packets for layer 1 that are not needed anymore. For *scheme WoNC* the load is significantly lower because the server retransmits packets only when it receives feedback from the receivers. Since fewer data packets reach the destinations, then fewer feedback packets need to be sent (which at their turn may be lost in the channel). Thus, in *scheme WoNC* the server sends less packets overall.

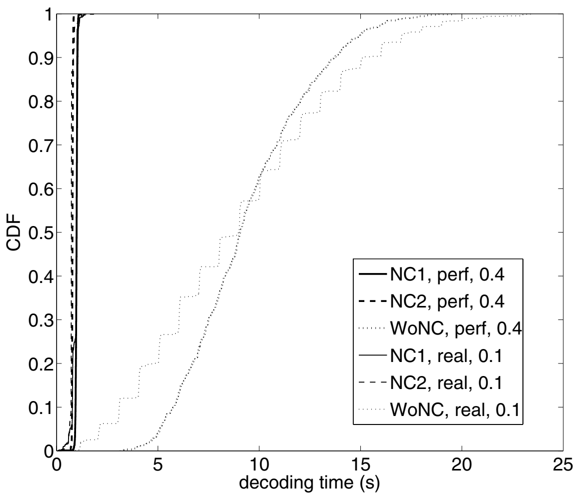


Fig. 6. CDF of decoding delay for layer 3, for *perfect feedback* and loss probability $p_{loss} = 0.4$, and for *realistic feedback* and $p_{loss} = 0.1$.

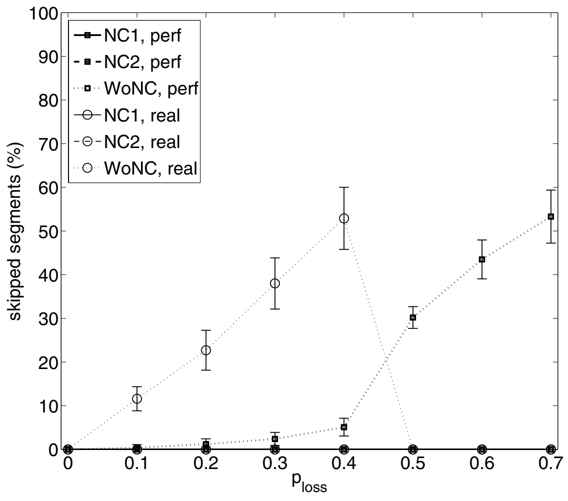


Fig. 7. The percentage of skipped segments with the probability of loss, p_{loss}, for layer 3, for *perfect feedback* and for *realistic feedback*.

Fig. 6 shows that the network coding approaches are able to decode segments within a second, for both *perfect* and *realistic feedback*, as the server sends redundant linear combinations in a feed-forward manner. *Scheme WoNC* needs a longer decoding time, because the server waits for the feedback before retransmitting. Note that for the *real* case, the decoding delay evolves in a ladder, due to the fact that feedback packets are lost and retransmitted hop-by-hop.

Figs. 7 and *8* show the percentage of segments that are skipped and played in lower quality, respectively, for layer

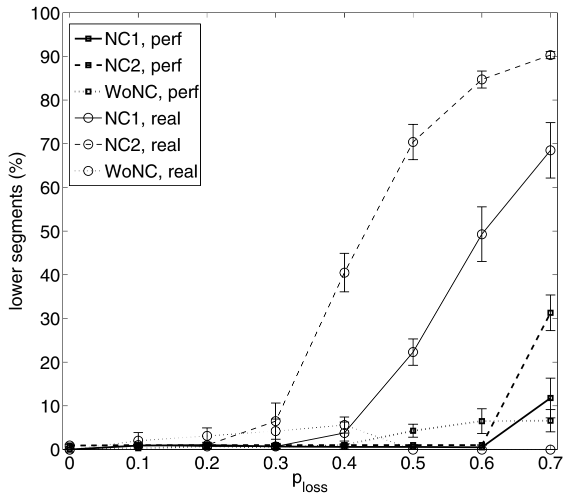


Fig. 8. The percentage of segments played in lower quality in function of the probability of loss p_{loss} for layer 3, for *perfect feedback* and for *realistic feedback*.

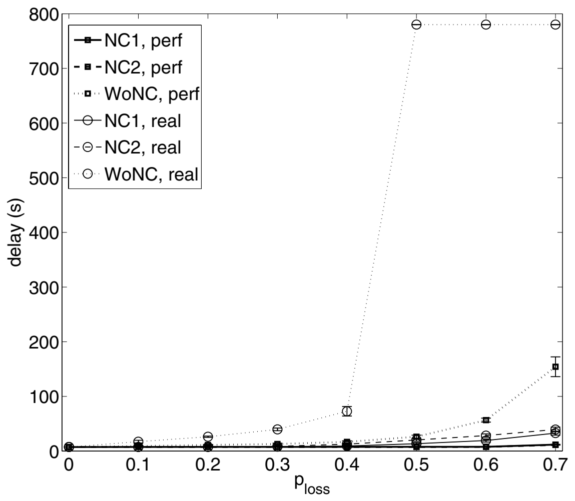


Fig. 9. Initial buffering delay with the loss probability p_{loss}, for layer 3.

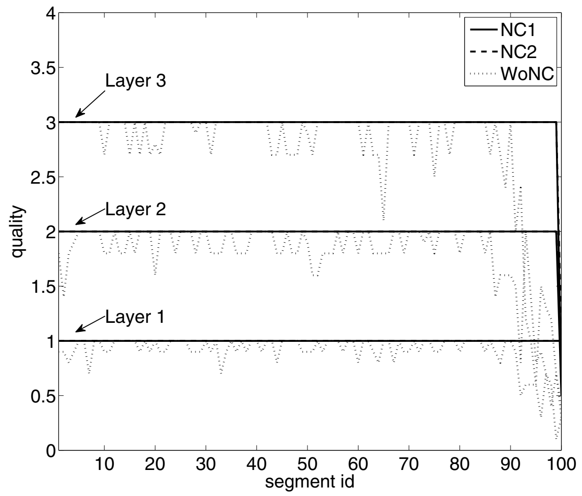


Fig. 10. Played quality for the case of *perfect feedback* and $p_{loss} = 0.4$.

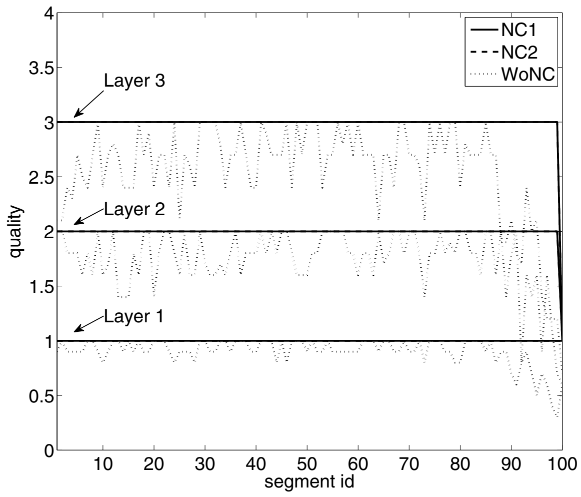


Fig. 11. Played quality for the case of *realistic feedback* and $p_{loss} = 0.1$.

3. The results for the other layers are similar and we omit them due to lack of space. Note that with network coding, no segments are skipped for any layers, neither for *perfect feedback*, nor for *realistic feedback*. For *perfect feedback* with *Scheme WoNC* the receiver starts to skip segments as the p_{loss} increases beyond 0.4 because it is not able to decode in a timely manner. For *realistic feedback* the percentage of skips increases with the probability of loss up to the point of $p_{loss} = 0.5$ where the receiver is unable to decode and play anything (see also *Fig. 4(c)*). Consequently it does not skip any segment either. Note that no segments are played in lower quality with *Scheme NC1* and *Scheme NC2* in the case of *perfect feedback*. However, for *realistic feedback* the percentage of lower quality segments increases with the probability of loss. With *Scheme WoNC* the receiver plays very few packets in lower quality because the packets retransmitted by the server do not arrive in due time, thus most of them are skipped.

We can see in *Fig. 9* that for *perfect* and *realistic feedback*, the receivers buffer a shorter time before starting the playback for both *Scheme NC1* and *Scheme NC2*. The initial buffering delay grows slowly with the probability of loss, because a single network coded packet can recover multiple losses. For *scheme WoNC* with *realistic feedback*, when losses are high the receivers are not able to decode anything, thus they never start to play the file.

Figs. 10 and *11* show the average quality in which every segment is played, for the case of *perfect feedback* with $p_{loss} = 0.4$, and the case of *realistic feedback* with $p_{loss} = 0.1$, respectively. A skipped segment accounts as played in a quality equal to 0. Note that both network coding approaches show a high resilience to errors and the video file is constantly played in the desired quality by each receiver, for both types of *feedback*, *perfect* or *realistic*. On the other hand, with *Scheme WoNC* the played quality is lower because the lost packets are not recovered in a timely manner. For the *realistic feedback* case, the control packets that are lost further deteriorate the played quality, such that for $p_{loss} = 0.1$, the played quality is similar to the one obtained for *perfect feedback* at $p_{loss} = 0.4$.

V. Conclusions and Further Work

We evaluated the performance of RLNC for layered video coding, by designing and implementing a specific system solution for a lossy wireless scenario. Our results show that network coding approaches perform better in high loss scenarios and mixing several layers yields higher performance gains. In particular, by generating the encoding matrix at the source in a specific shape and allowing intermediate nodes to perform the traditional network coding operations yet still prioritizing base layers of the video, we are able to achieve gains in buffering delay, percentage of skipped segments and variability of the quality played at the sinks (even if the feedback is not perfect). The implementation of such an architecture in a real scenario was shown to be feasible.

As part of our future work, we are considering the comparison of the RLNC approach with online network coding schemes, as well as selective discarding of video frames.

References

[1] U. Horn, K. Stuhlmüller, M. Link, and B. Girod, "Robust internet video transmission based on scalable coding and unequal error protection," *Signal Processing: Image Communication*, vol. 15, no. 1, pp. 77–94, 1999.

[2] J. Kritzner, U. Horn, M. Kampmann, and J. Sachs, "Priority Based Packet Scheduling with Tunable Reliability for Wireless Streaming," *Lecture Notes in Computer Science*, pp. 707–717, 2004.

[3] R. Ahlswede, N. Cai, S.Y.R. Li, and R.W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[4] T. Ho, M. Médard, R. Koetter, D.R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.

[5] C. Fragouli, D. Katabi, A. Markopoulou, M. Medard, and H. Rahul, "Wireless network coding: Opportunities & challenges," in *IEEE Military Communications Conference (MILCOM)*, 2007, pp. 1–8.

[6] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: practical wireless network coding," *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 243–254, 2006.

[7] J. Jin, B. Li, and T. Kong, "Is Random Network Coding Helpful in WiMAX?," in *IEEE 27th Conference on Computer Communications (INFOCOM)*, 2008, pp. 2162–2170.

[8] R. A. Costa, D. Munaretto, J. Widmer, and J. Barros, "Informed network coding for minimum decoding delay," in *Fifth IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, 2008.

[9] H. Seferoglu and A. Markopoulou, "Opportunistic network coding for video streaming over wireless," *Packet Video 2007*, pp. 191–200, 2007.

[10] N. Sundaram, P. Ramanathan, and S. Banerjee, "Multirate Media Streaming Using Network Coding," *Proc. 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sep*, 2005.

[11] P. Frossard, J.C. de Martin, and M. Reha Civanlar, "Media streaming with network diversity," *Proceedings of the IEEE*, vol. 96, no. 1, pp. 39–53, Jan. 2008.

[12] Z. Liu, Y. Shen, S. Panwar, K. Ross, and Y. Wang, "Using layered video to provide incentives in p2p live streaming," in *P2P-TV '07: Proceedings of the 2007 workshop on Peer-to-peer streaming and IP-TV*, New York, NY, USA, 2007, pp. 311–316, ACM.

[13] C. Gkantsidis, W. Hu, P. Key, B. Radunovic, P. Rodriguez, and S. Gheorghiu, "Multipath code casting for wireless mesh networks," in *Proceedings of the 2007 ACM CoNEXT conference*, 2007.

[14] J. K. Sundararajan, D. Shah, and M. Médard, "Feedback-based online network coding," *CoRR*, vol. abs/0904.1730, 2009.

[15] S. McCanne, S. Floyd, and K. Fall, "ns2 (network simulator 2)," http://www-nrg.ee.lbl.gov/ns/.