[Proceeding] Feedback-driven network coding for cooperative multimedia streaming

FEEDBACK-DRIVEN NETWORK CODING FOR COOPERATIVE VIDEO STREAMING

Attilio Fiandrotti, Valerio Bioglio, Enrico Magli

Dipartimento di Elettronica e Telecomunicazioni, Politecnico di Torino
attilio.fiandrotti@polito.it, valerio.bioglio@polito.it, enrico.magli@polito.it

ABSTRACT

In this work, we propose a feedback scheme to drive the packet recombination process at the network nodes in a collaborative Network Coding (NC) scenario. Our scheme addresses the issue of determining which symbols are more helpful at the receivers to recover the message and how to accordingly recombine the received packets at the intermediate nodes where the original symbols are not available. We experimentally demonstrate that our scheme increases the coding efficiency and reduces the computational complexity at the decoder in a video communication scenario without using explicit feedback messages.

Index Terms— Feedback, Network Coding, P2P, Video Streaming

1. INTRODUCTION

Network Coding (NC) [1] has attracted a lot of interest recently due to its ability to maximize the useful network throughput in multicast communications. In a classic NC scenario [2], a source node shares a message with multiple receivers in the network. The source node divides the message in input symbols and transmits linear combinations of the symbols to some intermediate nodes. The intermediate nodes relay linear combinations of the received packets to the receiver nodes until they have recovered the input symbols. After collecting enough independent packets, the receiver solves a system of linear equations and recovers the symbols and the message. In this work, we focus on a scenario where the nodes cooperate to distribute a message (a video content) by exchanging linear recombinations of the received packets. Wang *et al.* showed that NC enables robust video delivery and reduces the bandwidth requirements in an low-delay videoconferencing application [3]. Wang and Li [4] proposed a random-push video streaming protocol, showing that NC reduces the initial buffering delay improving the video quality.

Previous research demonstrated the benefits of feedback-based linear coding over pure random linear coding [5]. Sundararajan *et al.* [6] considered low-delay communications with online coding, where the source broadcasts linear combinations of the input symbols to multiple receivers. Each receiver signals to the source the list of the symbols it has not *seen* yet, i.e. symbols that are helpful to recover the message, with explicit feedback, and the source adjusts the encoding scheme accordingly. This scheme reduces the time required to recover the symbols at the receivers and enables to early-drop the symbols from the input queue of the source. Yang *et al.* [7] proposed to incorporate orthogonal coding to provide feedback in NC systems, where the feedback information is represented by the number of packet received by each node.

While existing feedback schemes focus on controlling the coding at the source, little attention has been devoted to design feedback schemes to control the packet recombination at the network due to the issues that must be addressed: i) the input symbols are usually unknown at the network nodes, which are constrained to recombine the received packets ii) the list of unseen symbols should be determined without increasing the computational complexity of the node iii) the feedback information should load as little as possible the network

In this paper, we propose a feedback scheme to drive the recombination of the received packets at the network nodes. Each node determines the subset of the input symbols that it has already seen by exploiting the packet decoding process without additional computations. Such information can be represented in a more compact way than the list of unseen symbols and can be embedded in the packets exchanged by the nodes avoiding explicit feedback. Finally, the network nodes exploit the feedback information restricting the recombinations to a subset of the received packets. The experiments show that our scheme increases the coding efficiency in a video conferencing scenario and reduces the computational complexity of the decoding process in a a video streaming application. while in this work we consider coding operations over $GF(2)$ due to the its low computational complexity [8], our findings could be in principle extended to coding operations over a generic $GF(q)$.

2. BACKGROUND: RANDOM NC OVER GF(2)

We model the network as a graph $G(V, E)$ where each vertex $V = \{\mathcal{N}_0, ..., \mathcal{N}_{|V|}\}$ of the graph is a node of the network. The source $\mathcal{N}_0$ divides the message to be distributed to the network nodes in a sequence $(M_0, ..., M_{N-1})$ of N input

symbols, where N is called *message size*. When the opportunity to transmit a packet arises, the source randomly combines the input symbols as $X = \sum_{i=0}^{N-1} g_i M_i$. The sum operator is the bitwise XOR ($\oplus$), as we consider coding operations over $GF(2)$. The vector $g = (g_0, ..., g_{N-1}) \in (GF(2))^N$ is known as *coding vector* and each of its elements g_i is drawn at random so that $\mathbb{P}\{g_i = 1\} = \frac{1}{2}$, $\forall\ i$. The *null* coding vector (i.e., $g_i = 0$ for any i) represents no useful information, so the source is constrained to avoid encoding packets with null coding vector ($g \neq \mathbf{0}$). Finally, the packet $P(g, X)$ composed by the encoded payload X and the corresponding coding vector g is sent to the network [9]. The network nodes store the received packets and relay a random linear combination thereof at each transmission opportunity. Assuming that a node has received R packets when a transmission opportunity arises, the node computes $X' = \sum_{i=0}^{R-1} c_i X^i$ and $g' = \sum_{i=0}^{R-1} c_i g^i$, where $c = (c_0, ..., c_{R-1})$ is drawn at random in $(GF(2))^R$, and the packet $P(g', X')$ is sent to the network. When a receiver has collected N linearly independent packets, it solves the system of linear equations corresponding to the coding vectors of the collected packets and recovers the message. Due to the random coding, not all the packets received by a node are linearly independent. Usually a node needs to collect $N' > N$ packets to recover the message. We define as *coding overhead* the figure $\epsilon = \frac{N'}{N} - 1$. When a node receives a packet P that is linearly independent from the previously received packets, P is said to be *innovative*.

The nodes receive encoded packets $P(g, X)$ that are decoded via *On-the-Fly Gaussian Elimination* (OFG) [10]. Other than distributing the computational complexity of message recovery over time as similar early-decoding algorithms, OFG enables the feedback mechanism that we propose in this paper. OFG solves a system of N linear equations $GM = C$, where G is the $N \times N$ *decoding matrix* that stores the coding vectors of the received packets as rows. C is the *payload vector*, containing the encoded payloads X and M are the input symbols M_i to recover. For the sake of simplicity, we describe only the operations on the received coding vectors g and on the G matrix. In the following, we indicate the i-th row of G as G_i and the j-th element of the i-th row of G as $G_{i,j}$. When all the elements of G_i are equal to zero, we define G_i *empty* and we write $G_i = \emptyset$. The OFG algorithm is divided in two stages, triangularization and diagonalization. The triangularization is described in Algorithm 1, which is executed every time a packet $P(g, X)$ is received and builds an upper-triangular G matrix. Let $s \in [0, N-1]$ be the position of the first element of g equal to one (i.e., $g_i = 0$ for $i < s$.) If $G_s = \emptyset$, g is inserted in G as G_s (line 4), otherwise g and G_s *collide* and are swapped (line 7). Then, if g is identical to G_s, P is not innovative and the algorithm ends (lines 9). Otherwise, g is XOR'ed with G_s and the while cycle iterates (line 11). After receiving N linearly independent packets, the input symbols M_i are recovered via backward substitution during the following diagonalization stage.

Algorithm 1 OFG, Triangularization

```
1:  while true do
2:    s ← position of 1-st element of g equal to 1.
3:    if G_s = ∅ then
4:      G_s ← g
5:      end
6:    else
7:      swap G_s and g
8:      if g = G_s
9:        end
10:     g ← g ⊕ G_s
11:   end if
12: end while
```

3. FEEDBACK-BASED NETWORK CODING

3.1. Feedback Generation and Delivery

For any node $\mathcal{N}_r$, $r \in [1, |V|]$, we define the relative decoding matrix and payload vector as G^r and C^r respectively. For any $\mathcal{N}_r$, $k^r \leq N$ is the number of linearly independent packets collected by the node and it is equal to the rank of G^r. Let $m_k^r \leq k^r$ be the first empty row of G^r after receiving k^r independent packets, i.e. $G^r_{m_k} = \emptyset$ and $G^r_i \neq \emptyset$ for any $i < m_k^r$. Let $H^{m_k^r}$ be the linear subspace of $GF(2)^N$ spanned by the first m_k^r rows of G^r. In our scheme, the nodes exchange m_k^r as feedback information rather than the list of the seen symbols. The number of bits required to represent m_k^r is $log_2(N)$ bits, so $\mathcal{N}_r$ embeds m_k^r in any packet it transmits with little signaling overhead. Therefore, we assume that node $\mathcal{N}_t$, $t \neq r$, knows m_k^r before it encodes a packet $P(g, X)$ addressed to $\mathcal{N}_r$ as described below.

3.2. Feedback-Driven Recombinations at the Nodes

Let $\mathcal{N}_t$ be a network node that is given the opportunity to transmit a packet to the network and let us assume that $\mathcal{N}_r$ is selected as recipient. $\mathcal{N}_t$ recombines the rows of G^t, which are an (incomplete) basis for the linear space spanned by all the linear combination of the input symbols. $\mathcal{N}_t$ recombines the rows of G^t as $g = \sum_{i=0}^{N-1} c_i G_i^t$ and the payloads as $X = \sum_{i=0}^{N-1} c_i C_i^t$, where c_i is drawn as

$$\mathbb{P}\{c_i = 1\} = \begin{cases} 0 & \text{if} \quad i < m_k^r \\ 1/2 & \text{if} \quad i \geq m_k^r \end{cases} \tag{1}$$

with the constraint that $c \neq \mathbf{0}$. If $\mathcal{N}_t$ has received no feedback from $\mathcal{N}_r$ or $G_i^t = \emptyset$ for $i \geq m_k^r$, it assumes $m_k^r = 0$ which corresponds to the case of random linear coding discussed in Section 2. In the case where $\mathcal{N}_t$ has already recovered the symbols (or $\mathcal{N}_t$ is the source node), the elements of g are drawn as

$$\mathbb{P}\{g_i = 1\} = \begin{cases} 0 & \text{if} \quad i < m_k^r \\ 1/2 & \text{if} \quad i \geq m_k^r \end{cases} \tag{2}$$

still with the constraint $g \neq \mathbf{0}$. In general, if c is distributed following (1) then g is distributed following (2)

3.3. Coding Efficiency Evaluation

We model the probability that a packet encoded according to the proposed scheme is innovative and we compare over a random coding, feedback-less, reference. The packet $P(g, X)$ encoded by $\mathcal{N}_t$ as described above is innovative at $\mathcal{N}_r$ if g is linearly independent from the k^r non-empty rows of G^r. P is obviously linearly independent from the first m_k^r rows of G^r, i.e. rows G_i^r such that $i < m_k^r$. Let $H^{k^r - m_k^r}$ be the linear subspace of $GF(2)^N$ spanned by the $k^r - m_k^r$ rows G_i^r such that $i > m_k^r$. P is hence innovative if $g \notin H^{k^r - m_k^r}$, i.e. if g is different from all the $2^{k^r - m_k^r}$ elements in $H^{k - m_k^r}$. Recalling the constraint $g \neq \mathbf{0}$, P is innovative if g is different from the $2^{k^r - m_k^r} - 1$ elements in $H^{k - m_k^r}$ not counting the null vector $\mathbf{0}$. On the other hand, the number of the possible linear combinations of the input symbols such that $g_i \neq \mathbf{0}$ and $i \leq m_k^r$ is equal to $2^{N - m_k^r} - 1$. Therefore, the probability that P encoded according to our scheme is innovative at $\mathcal{N}_r$ is

$$\mathbb{P}_{inn}^{(m-FB)} = 1 - \frac{2^{k^r - m_k^r} - 1}{2^{N - m_k^r} - 1} \tag{3}$$

In a reference case where no feedback is available, $\mathcal{N}_t$ randomly encodes the input symbols (source node) or the packets collected so far (network node) as described in Section 2. The probability that the coded packet P is innovative at $\mathcal{N}_r$ can be derived from (3) posing $m_k^r = 0$ and is equal to

$$\mathbb{P}_{inn}^{(noFB)} = 1 - \frac{2^{k^r} - 1}{2^N - 1}. \tag{4}$$

We have $\mathbb{P}_{inn}^{(m-FB)} \geq \mathbb{P}_{inn}^{(noFB)} \; \forall \; m_k^r$ ($m_k^r \leq k^r$ by definition), so our feedback scheme increases the probability to encode an innovative packet.

3.4. Computational Complexity Evaluation

We model the computational complexity C_D of recovering the message at node $\mathcal{N}^r$ for the proposed and the feedback-less reference schemes. C_D is measured in terms of average number of XOR operations performed by the OFG to recover the message. C_D is the sum of a component related to the OFG triangularization stage, $C_{D,t}$, and a component related to the diagonalization stage, $C_{D,d}$ (i.e., $C_D = C_{D,t} + C_{D,d}$). $C_{D,d}$ depends on the number of non-zero elements of G^r at the end of triangularization stage and is independent from the considered feedback scheme. As at the beginning of this stage G^r is upper triangular and $\mathbb{P}\{G_{i,j}^r = 1\} = \frac{1}{2}$ for $j > i$ ($G_{i,j}^r = 1$ if $i = j$), the average number of non-zero elements of G^r is $\frac{N^2}{4}$. On the contrary, $C_{D,t}$ depends on the number of collisions performed to insert each packet received by node $\mathcal{N}_r$ in G^r at line 10 of Algorithm 1. Such figure depends on the number of non-empty rows of G^r that could be XORed with the coding vector of the received packet. On the average, each non-empty row of G^r collides with the received packet with probability $\frac{1}{2}$. Since the new packet coding vector does not collide with the first m_k^r rows of G^r, it is inserted in G^r after $\frac{k^r - m_k^r}{2}$ XOR operations, where $m_k^r = 0$ for the feedback-less reference scheme. So, the computational complexity of the triangularization stage for the proposed feedback-based coding scheme is

$$C_{D,d}^{(m_k - FB)} = \sum_{k^r=0}^{N-1} \frac{k^r}{2} - \sum_{k^r=0}^{N-1} \frac{m_k^r}{2} = C_{D,d}^{(noFB)} - \sum_{k^r=0}^{N-1} \frac{m_k^r}{2}, \tag{5}$$

where $C_{D,d}^{(noFB)}$ is the complexity of the reference scheme without feedback. We have $C_D^{(m_k - FB)} \leq C_D^{(noFB)} \; \forall \; m_k^r$, proving that our scheme reduces the computational complexity of recovering the message by reducing the number of operations needed to insert a new row in the decoding matrix.

4. EXPERIMENTAL EVALUATION

We experiment in two different scenarios, i.e. low-delay video conferencing and live video streaming. In the former scenario, the constraints on the decoding latency impose to reduce the message size N, which yields high coding overhead. In the latter scenario the latency constraints are looser and allow higher N, which reduces the coding overhead however at the price of higher decoding complexity. We evaluate the impact of our scheme on the coding overhead in the videoconferencing scenario and on the decoding complexity in the streaming scenario.

4.1. Two-Users Videoconferencing

We consider a videoconferencing scenario similar to [3] where two users exchange two video streams. Each video stream is encoded in H.264/AVC format at 1 Mbit/s and is subdivided in messages composed by independently decodable Groups of Pictures (GoPs). The nodes exchange packets where the encoded video payload is equal to 1250 bytes over a symmetric link with a 50 ms delay, accounting for the fact that in real applications the nodes receive slightly outdated feedback information. Each node transmits linear combinations of the input symbols of one message of the outgoing video stream and receives coded packets for one of the messages of the incoming stream at a time. Node $\mathcal{N}_r$ appends the m_k^r-value relative to the message being currently decoded to each packet of the outgoing stream. Node $\mathcal{N}_t$ encodes the input symbols of the outgoing messages following (2) exploiting the m_k^r-feedback provided by $\mathcal{N}_r$ (i.e. no explicit feedback is used). Moreover, we consider the case where $\mathcal{N}_r$ appends the m_k^r-value only every f transmitted packets to $\mathcal{N}_t$ to assess how the feedback frequency affects the performance of our scheme. As a reference, we consider a feedback-less scheme

(*No-FB*) where the nodes do not embed any feedback information in the transmitted packets. As in a videoconferencing application the end-to-end latency should be constrained as much as possible, we experiment with messages of 13, 26, and 51 symbols, which correspond to a decoding delay of 0.125, 0.25 and 0.5 seconds respectively.

Figure 1 shows the coding overhead in terms of required extra bandwidth as a function of the message size N and the feedback period f and the inherent tradeoff between coding efficiency and decoding delay. Our scheme produces the most significant gains when N is small and the coding efficiency is low, which is the case of most interest in low-delay communications. If feedback is provided with each transmitted packet (f=1), the coding overhead drops from about 60 to 20 kbit/s with a 40 kbit/s (66%) gain over the reference. Even if feedback is provided every 2 and 4 transmitted packets (f=2 and f=4), our feedback scheme reduces the coding overhead by, respectively, 25 and 20 kbit/s (50% and 33% less overhead than the reference). The maximum signaling bandwidth required to embed the m-feedback is around 800 bit/s for the case $f = 1$, which corresponds to 0.08% of the bandwidth of the test video. This figure further reduces to 400 and 200 bit/s (0.04% and 0.02% of the video bandwidth) for the cases $f = 2$ and $f = 4$, respectively.

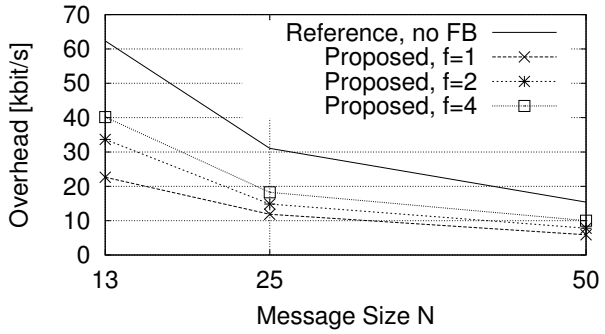

Fig. 1. Coding overhead in videoconferencing scenario as a function of message size N and feedback period f. Feedback improves coding efficiency in low-delay applications.

4.2. Multiple-Users Video Streaming

We now consider a video streaming scenario where one source node distributes a live stream to 10 cooperating receivers organized using the NC-based P2P video streaming protocol described in [11]. The network nodes are organized into a non acyclic, unstructured, mesh of cooperating *peers* that exchange linear combinations of the received packets. The source node divides a video stream encoded at 1 Mbit/s in messages of 50, 100 or 200 symbols N, which yields decoding latencies of 0.5, 1 or 2 seconds respectively. The source node randomly selects one of the network nodes at each transmission opportunity and transmits random linear combinations of the input symbols (i.e., the source does not receive feedback by the network nodes). Each time a transmission opportunity arises for a node, it selects one of its peer nodes and relies on our feedback scheme to select which rows of the G matrix to recombine. Let $\mathcal{N}_r$ be the node drawn by $\mathcal{N}_t$ as the recipient of the packet to be transmitted. At the moment of transmitting the packet, $\mathcal{N}_t$ recombines the rows of G^t as shown in (1) exploiting the m_k^r-feedback provided by $\mathcal{N}_r$. Finally, before transmitting the packet, the node appends the m-value to it to provide the feedback information required by its peers. Figure 2 shows how feedback scheme reduces the computational complexity C_D of recovering the message. As for any Gaussian Elimination-like algorithm, the number of XOR operations between rows of G is quadratic with the message size N. The figure shows that our scheme reduces the decoding complexity by nearly 50% by reducing the number of XOR operations performed during the triangularization stage of the OFG algorithm. While the encoding overhead is less of a problem that in the previous scenario, our scheme still reduces the coding overhead up to 10%. The signaling bandwidth required to embed the m-feedback information in the transmitted packets is below 1 kbit/s, that is less than 0.1% of the test video bandwidth.

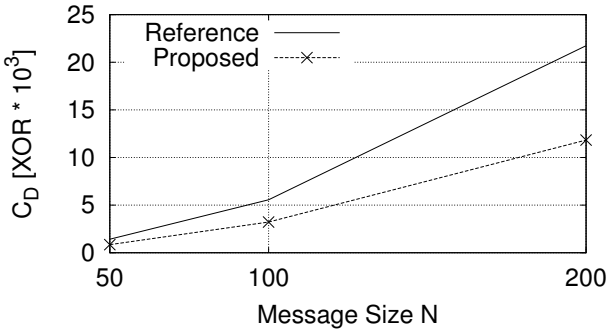

Fig. 2. Computational complexity of recovering the message as a function of the message size N is reduced by a factor of two by our feedback scheme.

5. CONCLUSIONS

We proposed a feedback scheme to control the packet recombination process at the nodes of the network where the input symbols are often not available. The experimental evaluation shows consistent gains in terms of coding efficiency in a videoconferencing application and computational complexity in video streaming scenario with respect to a reference scheme where no feedback is employed.

6. ACKNOWLEDGMENTS

Part of this work was supported by PRIN ARACNE, a research project funded by the Italian Ministry of Education and Research.

7. REFERENCES

[1] Christina Fragouli, Jean-Yves Le Boudec, and Jörg Widmer, "Network coding: an instant primer," *SIGCOMM Computer Community Review*, vol. 36, no. 1, pp. 63–68, Jan. 2006.

[2] Rudolf Ahlswede, Ning Cai, Shuo-Yen Robert Li, and Raymond W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[3] H. Wang, Y. R. Chang, and C. C. J Kuo, "Wireless multy-party video conferencing with network coding," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2009, pp. 1492–1495.

[4] M. Wang and B. Li, "R2: Random push with random network coding in live peer-to-peer streaming," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 9, pp. 1655–1666, 2007.

[5] Christina Fragouli, Desmond S. Lun, Muriel Médard, and Payam Pakzad, "On feedback for network coding," in *Proceedings of the Annual Conference on Information Sciences and Systems (CISS)*, 2007, pp. 248–252.

[6] Jay Kumar Sundararajan, Devavrat Shah, and Muriel Médard, "Arq for network coding," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2008, vol. abs/0802.1754, pp. 1651–1655.

[7] J. Yang, B. Dai, B Huang, and S. Yu, "Orthogonal feedback scheme for network coding," *Journal on Network and Computer Applications*, vol. 34, pp. 1623–1633, 2011.

[8] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "Xors in the air: Practical network coding," in *Proceedings of the ACM SIGCOMM*, Sep 2006, pp. 243–254.

[9] P.A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proceedings of the Annual Allerton Conference on Communication Control and Computing*. The University; 1998, 2003, vol. 41, pp. 40–49.

[10] V. Bioglio, M. Grangetto, R. Gaeta, and M. Sereno, "On the fly gaussian elimination for LT codes," *IEEE Communications Letters*, vol. 13, no. 12, pp. 953–955, 2009.

[11] A. Fiandrotti, A. M. Sheikh, and E. Magli, "Towards a P2P videoconferencing system based on low-delay network coding," in *Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, Aug. 2012.