Correlation-Aware Reconstruction of Network Coded Sources

Eirina Bourtsoulatze, Nikolaos Thomos, and Pascal Frossard
Signal Processing Laboratory (LTS4), Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
{eirina.bourtsoulatze, nikolaos.thomos, pascal.frossard}@epfl.ch

***Abstract*—In this paper, we consider the problem of decoding network coded correlated data when the decoder does not receive sufficient information for exact decoding. We propose an iterative decoding algorithm based on belief propagation that efficiently exploits the data correlation and provides approximate reconstruction of the sources when conventional decoding methods fail. The dependencies among the sources are captured by means of a factor graph. A simple noise model is used in order to describe the pairwise source relationships. The decoding decision is based on MAP estimates that are inferred by message passing over the underlying factor graph. Performance evaluation of the proposed decoding algorithm on correlated data sets consisting of video sequences confirms the efficiency of the proposed algorithm. Simulation results show that high quality reconstruction can be achieved even if significant amount of network coded information is missing at the decoder.**

***Index Terms*—Network coding, correlated sources, message passing, factor graph.**

I. Introduction

Network coding has been extensively studied in the framework of distributed data delivery systems, where network coding can increase the throughput of networks. Theoretically, the min-cut max-flow bound can even be attained in a multicast setting if information is processed in the network instead of simply being relayed to the end-users. Network coding algorithms provide also robustness to losses. However, while many research efforts have focused on the design of network codes, only few works have addressed the problem of recovering the network coded data when the available network coded information is not sufficient for perfect decoding. In some applications like multimedia communications, information may still be useful even if data is not perfectly recovered. The inability of conventional decoding methods to provide approximate recovery of the sources diminishes the advantages of network coding in such cases.

Motivated by the demand for approximate decoding methods, we propose an iterative decoding algorithm based on message passing, which permits to reconstruct the source data when part of the network coded information is lost. Typically, prior information on the sources such as correlation can be exploited at the decoder to compensate for the missing network coded data. Our scenario consists of a set of correlated sources that transmit source symbols over a network where intermediate nodes perform randomized network coding. The network coded data is gathered at the receivers that provide an approximate reconstruction of the original source data when the network coded data is not sufficient for exact source recovery. We design a message passing algorithm that exploits prior information about sources in order to provide an approximate reconstruction of the data. The algorithm operates on a factor graph that encapsulates the network coding constraints and the source models and dependencies. Such graphical model permits to efficiently infer the approximate MAP (maximum a posteriori) probabilities that are used for decoding. Another important feature of the graphical representation is that it permits to unify in a single model constraints that are expressed in different domains, *i.e.*, Galois field and field of real numbers. Simulation results demonstrate that our proposed method efficiently exploits the prior information on the sources and permits data recovery with small approximation errors.

The first theoretical results on network coding for correlated sources were presented in [1], where the authors provide upper bounds on the probability of decoding error as a function of network parameters for two arbitrarily correlated sources in a general network. Early attempts to provide a practical implementation of joint source and network coding are described in [2]. However, the proposed scheme is sub-optimal and addresses only the case of two correlated sources related by a Binary Symmetric Channel (BSC). The design of optimal network codes for joint distributed source and network coding in a sensor network scenario is studied in [3]. The goal of the method is to optimize the tradeoff between the compression efficiency and the network robustness; the complexity of the solution in [3], however, grows exponentially with the size of the network. The first low-complexity approximate decoding algorithm [4] exploits the data similarity between the correlated sources in order to provide the missing information to the decoder. It is shown that there exists an optimal field size that leads to the highest approximate decoding performance. Another joint source-network decoding scheme for multiple description coded data is presented in [5]. The reconstruction of the data is formulated as a mixed integer quadratic programming and a good robustness to missing packets is observed. Most similar to our work is the framework in [6], where the authors propose a maximum a posteriori decoder based on the sum-product algorithm. However, the method in [6] requires a priori knowledge about the traversed network nodes and edges' transition probabilities, which is problematic in large scale networks. Contrarily, our scheme does not need any knowledge of the underlying network structure, since all the network constraints are reflected in the coding coefficients that

This work has been partly supported by the Swiss National Science Foundation, under grants 200021-138083 and PZ00P2-137275.

are transmitted along with the network coded data. Thus, the complexity of the algorithm does not depend on the network size and our method is suitable for any arbitrary network.

II. Framework

A. System setup

We consider the system illustrated in Fig. 1. Let $S_1, S_2, \ldots, S_N$ be a set of N correlated sources. The sources produce a set of discrete symbols $x_1, x_2, \ldots, x_N$ that are transmitted to the receivers through an overlay network, where intermediate nodes perform random linear network coding (RLNC). The source symbols belong to a finite alphabet $\mathcal{X} \subset \mathbb{Z}$, which is a subset of integer numbers and is common for all the sources. Without loss of generality, we assume that $\mathcal{X} = \{0, 1, \ldots, q-1\}$, where q is a prime or a power of a prime number.

Prior to transmission, the source symbols are mapped to values $\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_N$ in a Galois field of size q through a one-to-one mapping $\mathcal{F} : \mathcal{X} \rightarrow \text{GF}(q)$, such that

$$\hat{x}_n = \mathcal{F}[x_n],\ n = 1, \ldots, N,\ \hat{x}_n \in \text{GF}(q) \tag{1}$$

The source symbols are then transmitted to the receivers through the intermediate network nodes. The nodes perform RLNC and forward on the output links random linear combinations of the input symbols. Thus, the l-th network coded symbol $\hat{y}_l$ that reaches the receiver can be written as

$$\hat{y}_l = \sum_{i=1}^{N} a_{li} \hat{x}_i \tag{2}$$

where $a_{li} \in \text{GF}(q)$ and all the arithmetic operations are performed in the GF(q). The coding coefficients at every intermediate node are chosen randomly according to a uniform distribution from elements of the GF(q) and the global coding vectors $\boldsymbol{a}_l = (a_{l1}, a_{l2}, \ldots, a_{lN})$ are sent along with the network coded symbols to the receiver. In practice, in order to reduce the overhead introduced by the coding coefficients, several source symbols can be concatenated in a single packet and the same coding coefficient can be used for all the symbols within a packet.

The receiver collects a set of L network coded symbols represented by a vector $\hat{\boldsymbol{y}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_L)^T$ such that

$$\hat{\boldsymbol{y}} = A\hat{\boldsymbol{x}} \tag{3}$$

where A is a $L \times N$ matrix of coding coefficients $\{a_{li}\}$ and $\hat{\boldsymbol{x}} = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_N)^T$ is the vector of source symbols represented in GF(q). Given the vector of network coded symbols $\hat{\boldsymbol{y}}$ and the matrix of coding coefficients A, the decoder at the receiver provides an estimation $\hat{\boldsymbol{x}}^*$ of the vector $\hat{\boldsymbol{x}}$ which is subsequently mapped to the estimation $\boldsymbol{x}^*$ of the vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)^T$ of source symbols through the inverse mapping $\mathcal{F}^{-1} : \text{GF}(q) \rightarrow \mathcal{X}$, such that

$$x_n^* = \mathcal{F}^{-1}[\hat{x}_n^*],\ n = 1, 2, \ldots, N \tag{4}$$

where $\hat{x}_n^* \in \text{GF}(q)$ and $x_n^* \in \mathcal{X}$.

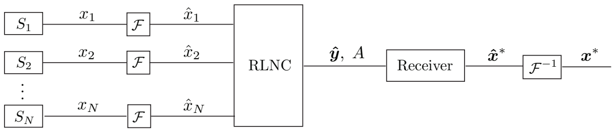


Fig. 1: Proposed framework.

If the matrix of coding coefficients A is full rank, the source symbols can be perfectly decoded. However, if the number of linearly independent network coded symbols is smaller than N, conventional decoding methods, such as Gaussian elimination, generally cannot recover any source symbol. In the rest of the paper, we will deal with the case where only a non full rank matrix A is available at the decoder. We design a decoding method that allows approximate reconstruction of the sources based on some prior information about source models and correlation.

B. Source and correlation model

The symbol x_n produced by the n-th source can be regarded as a realization of a discrete random variable X_n. Thus, we represent the source S_n by the random variable X_n with probability mass function $f_n(x) : \mathcal{X} \rightarrow [0, 1]$. Equivalently, $\hat{x}_n$ is the realization of the discrete random variable $\hat{X}_n$ that takes values in GF(q) and has the probability mass function $\hat{f}_n(\hat{x}) : \{0, 1, \ldots, q-1\} \rightarrow [0, 1]$, such that

$$\hat{f}_n(\hat{x}) = f_n(\mathcal{F}^{-1}(\hat{x})),\ \hat{x} \in \text{GF}(q) \tag{5}$$

The correlation between any pair of sources S_k and S_n $(k < n)$, is modeled as correlation noise E_{kn}

$$X_k = X_n + E_{kn} \tag{6}$$

Thus, the signal generated by source S_n can be viewed as a noisy version of the signal generated by source S_k. For the sake of simplicity, we replace the double index kn with the single index i and use E_i instead of E_{kn} throughout the paper. By definition, the random variable E_i is a discrete random variable whose realization e_i takes values in the set $\mathcal{E} = \{-q+1, \ldots, 0, \ldots, q-1\} \subset \mathbb{Z}$. We denote as $g_i(e) : \mathcal{E} \rightarrow [0, 1]$ the probability mass function of the random variable $E_i,\ i = 1, 2, \ldots, M$, where M is the number of correlated source pairs.

The number of correlated source pairs depends on the considered application. In general, the pairwise correlation dependencies between the sources are captured with the help of a $M \times N$ matrix C, such that

$$\boldsymbol{e} = C\boldsymbol{x} \tag{7}$$

where $\boldsymbol{e}$ is the realization of the random vector $E = (E_1, E_2, \ldots, E_M)^T$ of correlation noise terms and $\boldsymbol{x}$ is the vector of source symbols. It holds that $rank(C) \leq min(M, N)$. Whenever $rank(C) < M$, any $M - rank(C)$ correlation noise terms can be expressed as linear combinations of the remaining correlation noise terms. Thus, there exists a $(M - rank(C)) \times M$ matrix H, such that

$$H\boldsymbol{e} = \boldsymbol{0} \tag{8}$$

C. Source reconstruction

Upon receiving a set of L network coded symbols and the matrix A of coding coefficients, the goal of the decoder is to provide a good reconstruction $\boldsymbol{x}^*$ of the source symbols $\boldsymbol{x}$. When $rank(A) = N$, sufficient information is available for exact recovery of the source symbols by means of Gaussian elimination for example. However, when $rank(A) < N$, exact reconstruction methods are not applicable, since part of the information required for exact reconstruction is missing. In this case, the source symbol x_n can be approximated by the MAP estimate

$$x_n^* = \arg\max_x f_n(x/\hat{\boldsymbol{y}}) \tag{9}$$

where $f_n(x/\hat{\boldsymbol{y}})$ is the MAP probability mass function of the random variable X_n conditioned on the observed vector of network coded symbols $\hat{\boldsymbol{y}}$.

However, the exact inference of the MAP probability mass functions $f_n(x/\hat{\boldsymbol{y}})$ has exponential complexity and is known to be NP-hard [7]. Moreover, in many applications the decoder does not have access to the exact priors and joint probability mass functions of the correlated sources. Since we are not interested in the exact computation of $f_n(x/\hat{\boldsymbol{y}})$, but rather in a good approximation of these functions that would yield the correct reconstruction of the source symbols, the exact inference can be replaced by approximate inference on an appropriate graphical model that captures all the dependencies between the variables.

In the following section, we provide a detailed description of the way we construct the graphical model that incorporates the knowledge that is available at the decoder. We also present our message passing algorithm that allows to approximate the marginal conditional probability mass functions of the variables and obtain an approximate reconstruction of the sources.

III. Iterative decoding via message passing

We now present our iterative message passing algorithm for decoding of network coded correlated data. The algorithm operates on a factor graph that captures the relationships between the variables and uses prior information about the sources in order to provide an approximate reconstruction of the data. We build a belief propagation algorithm for solving the approximate decoding problem from Eq. (9). Belief propagation [8] is an efficient algorithm for performing (approximate) inference and computing marginal distributions in graph-based models. In the following, we will assume that apart from the vector $\hat{\boldsymbol{y}}$ and the matrix A that are available at the decoder, the decoder is aware of the probability mass functions $\hat{f}_n(\hat{x})$ and $g_i(e)$, as well as of the matrix C that represents the variables' dependencies.

A. Factor graph representation

Fig. 2 illustrates the factor graph that captures the structure of the decoding problem. The graph is a bipartite graph that consists of variable nodes and check nodes. The variable nodes are the unknowns of our problem. The check nodes represent the constraints imposed on the variables they are associated with. The variable nodes can be further categorized into two types of nodes: the x-nodes that represent the source symbols (defined in GF(q)) and the e-nodes that represent the correlation noise variables (defined in $\mathcal{E} \subset \mathbb{Z}$). Similarly, there are three types of check nodes in the factor graph: the n-check nodes that describe the network coding constraints, the c-check nodes that link every pair of correlated source symbols to the corresponding correlation noise term and the h-check nodes that capture the dependencies between the noise variables.

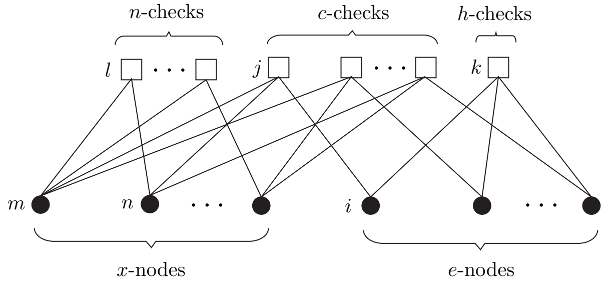


Fig. 2: Factor graph for the proposed decoding algorithm.

The connections between the check nodes and the variable nodes are built in the following way. Every n-check node corresponds to a network coded symbol. An edge between the l-th n-check node and the n-th x-node exists if the n-th source symbol participates in the l-th network coded symbol, *i.e.*, $a_{ln} \neq 0$. Every c-check corresponds to one pair of correlated sources. The j-th c-check is connected to the i-th e-node and two x-nodes m and n $(m < n)$, such that $c_{im} = -c_{in} \neq 0$. The number of c-check nodes is equal to the number of e-nodes. Finally, every h-check node represents the dependencies between the correlation noise variables. Thus, an edge between the k-th h-check node and the i-th e-node exists if the corresponding entry in the k-th row of matrix H is non zero, *i.e.*, $H_{ki} \neq 0$. Note that the matrix H can be easily computed if the matrix C is known. We will henceforth refer to a node that is connected to the node i as *neighbour* of i.

It is worth noting that one of the advantages of such a graphical representation is the unification of two different types of variables ($\hat{X}$ and E) defined in different domains (GF(q) and $\mathcal{E} \subset \mathbb{Z}$, respectively) in a single model.

B. Message passing algorithm

Table I summarizes the notation used throughout this section. The algorithm starts with the initialization step. At this step, the messages sent from the x-variable nodes and the e-variable nodes are initialized with the prior probability mass functions of the source symbols and correlation noise, respectively. All the messages from the check nodes are initialized to ones.

Once the initialization step is completed, the algorithm proceeds with the iterations during which messages are exchanged among the factor graph nodes and the beliefs are formed in the nodes. In particular, at every iteration of the algorithm, the messages $q_{nl}(a)$, $a \in \mathrm{GF}(q)$ are updated in the following way:

$$q_{nl}(a) = \alpha_{nl}\hat{f}_n(a) \prod_{j\in(\mathcal{L}(n)\cup\mathcal{J}(n))\backslash l} \mu_{jn}(a) \tag{10}$$

where

$$\mu_{jn}(a) = \begin{cases} r_{jn}(a), & \text{if } j \in \mathcal{L}(n) \\ t_{jn}(a), & \text{if } j \in \mathcal{J}(n) \end{cases} \tag{11}$$

They represent the probability (belief) that source symbol $\hat{x}_n$ has the value a, given the information obtained from all the neighbours other than the check node l.

Similarly, the messages $s_{ij}(b)$, $b \in \mathcal{E}$ represent the probability (belief) that the correlation noise term e_i has the value b, given the information obtained from all the neighbours other than the check node j. They are updated according to the following expression

$$s_{ij}(b) = \beta_{ij}g_i(b) \prod_{k\in(\mathcal{J}'(i)\cup\mathcal{K}(i))\backslash j} \nu_{ki}(b) \tag{12}$$

where

$$\nu_{ki}(b) = \begin{cases} u_{ki}(b), & \text{if } k \in \mathcal{J}'(i) \\ v_{ki}(b), & \text{if } k \in \mathcal{K}(i) \end{cases} \tag{13}$$

The normalization constants α_{nl} and β_{ij} are chosen such that $\sum_{a=0}^{q-1} q_{nl}(a) = 1$ and $\sum_{b=-q+1}^{q-1} s_{ij}(b) = 1$, respectively.

Once the messages from the variable nodes to the check nodes have been computed, we update the messages in the opposite direction. The messages $r_{ln}(a)$, $a \in \mathrm{GF}(q)$ from the n-check nodes to the variable x-nodes represent the probability that the check l is satisfied if the value of the source symbol $\hat{x}_n$ is set to a, given the messages from all the variable x-nodes other than n. They are updated as:

$$r_{ln}(a) = \sum_{\{\hat{\boldsymbol{x}}:\hat{x}_n=a,\hat{y}_l=A_l\hat{\boldsymbol{x}}\}} \prod_{n'\in\mathcal{N}(l)\backslash n} q_{n'l}(\hat{x}_{n'}) \tag{14}$$

where A_l denotes the l-th row of the matrix A. The summation is performed over all the vectors $\hat{\boldsymbol{x}}$ that satisfy the condition on the l-th n-check node and have the value a at the n-th position. The product term represents the probability of one such vector given the probability distributions (beliefs) from the variable nodes. The quantity in Eq. (14) can be efficiently computed using the method provided in [9].

The messages $t_{jn}(a)$, $a \in \mathrm{GF}(q)$ from the c-check nodes to the variable x-nodes can be computed as

$$t_{jn}(a) = \sum_{\{(\hat{x}_m,e_i):e_i=c_{im}\hat{x}_m+c_{in}a\}} q_{mj}(\hat{x}_m)s_{ij}(e_i) \tag{15}$$

where $m \in \mathcal{M}(j)\backslash n, i \in \mathcal{M}'(j)$. Recall that every c-check node has two x-node and one e-node neighbours. Thus, the summation is done over all the pairs of values $(\hat{x}_m, e_i)$, such that the i-th row of the matrix C is satisfied for a given value a of the variable $\hat{x}_n$. Similarly, the messages $u_{ji}(b)$, $b \in \mathcal{E}$ from the c-check nodes to the variable e-nodes are:

$$u_{ji}(b) = \sum_{\{(\hat{x}_m,\hat{x}_n):b=c_{im}\hat{x}_m+c_{in}\hat{x}_n\}} q_{mj}(\hat{x}_m)q_{nj}(\hat{x}_n) \tag{16}$$

where $m, n \in \mathcal{M}(j)$ and the summation is performed over all the pairs of values $(\hat{x}_m, \hat{x}_n)$ such that the i-th row of the matrix C is satisfied for a given value b of the noise e_i.

TABLE I: Notation.

$q_{nl}(a)$	messages sent from the variable x-node n to the check node l
$s_{ij}(b)$	messages sent from the variable e-node i to the check node j
$r_{ln}(a)$	messages sent from the n-check node l to the variable x-node n
$t_{jn}(a)$	messages sent from the c-check node j to the variable x-node n
$u_{ji}(b)$	messages sent from the c-check node j to the variable e-node i
$v_{ki}(b)$	messages sent from the h-check node k to the variable e-node i
$\mathcal{L}(n) = \{l : a_{ln} \neq 0\}$	all the n-check nodes that are neighbours of the x-node n
$\mathcal{J}(n) = \{j : c_{jn} \neq 0\}$	all the c-check nodes that are neighbours of the x-node n
$\mathcal{J}'(i) = \{j\}$	the c-check node that is connected to the e-node i
$\mathcal{K}(i) = \{k : h_{ki} \neq 0\}$	all the h-check nodes that are neighbours of the e-node i
$\mathcal{N}(l) = \{n : a_{ln} \neq 0\}$	all the variable x-nodes that are neighbours of the n-check node l
$\mathcal{M}(j) = \{m : c_{jm} \neq 0\}$	all the variable x-nodes that are neighbours of the c-check node j
$\mathcal{M}'(j) = \{i\}$	the e-node that is connected to the c-check node j
$\mathcal{I}(k) = \{i : h_{ki} \neq 0\}$	all the variable e-nodes that are neighbours of the h-check node k

Finally, the messages from the h-check nodes to the variable e-nodes are calculated according to the following equation:

$$v_{ki}(b) = \sum_{\{\boldsymbol{e}:e_i=b,H_k\boldsymbol{e}=0\}} \prod_{i'\in\mathcal{I}(k)\backslash i} s_{i'k}(e_{i'}) \tag{17}$$

where H_k stands for the k-th row of the matrix H. The summation is done over all the vectors $\boldsymbol{e}$ such that $e_i = b$ and the k-th h-check node is satisfied.

We employ the parallel schedule for the message update procedure, which implies that all the messages at the variable nodes are updated concurrently given the messages received from the check nodes at a previous stage. Similarly, all check nodes update and send their messages simultaneously. Other message passing schedules can be considered (*i.e.,* serial), however, they are not studied in this paper.

C. Decoding

At the end of each message passing round, the variable nodes form their beliefs on the marginal conditional probability mass functions of the variables $\hat{X}_n$ and E_i, and the values of the variables are tentatively set to

$$\hat{x}_n^* = \arg\max_a \hat{f}_n(a) \prod_{l\in\mathcal{L}(n)} r_{ln}(a) \prod_{j\in\mathcal{J}(n)} t_{jn}(a) \tag{18}$$

$$e_i^* = \arg\max_b g_i(b) \prod_{j\in\mathcal{J}'(i)} u_{ji}(b) \prod_{k\in\mathcal{K}(i)} v_{ki}(b) \tag{19}$$

If these values satisfy the constraints of the reconstruction, namely $\hat{\boldsymbol{y}} = A\hat{\boldsymbol{x}}^*$ and $\boldsymbol{e}^* = C\boldsymbol{x}^*$, where $\boldsymbol{x}^* = \mathcal{F}^{-1}[\hat{\boldsymbol{x}}^*]$, the solution is considered valid. In this case, the decoding stops and the decoder outputs the corresponding solution. If a valid solution is not found after the maximum number of iterations has been reached, the decoder declares an error and outputs the solution obtained at the last iteration.

IV. Simulation results

We now provide the experimental evaluation of the proposed decoding method. For our tests we use the QCIF format video sequences "Silent" and "Foreman". Each frame in the sequences corresponds to a 144×176 grayscale image with pixel values in the range $[0, 255]$. We extract N consecutive frames from each sequence and assume that the data transmitted by each source in Fig. 1 corresponds to a different video frame. In order to minimize the overhead induced by the network coding coefficients, the sequence of transmitted source symbols is divided into groups of symbols and the same coding operations are applied to all the symbols within a group.

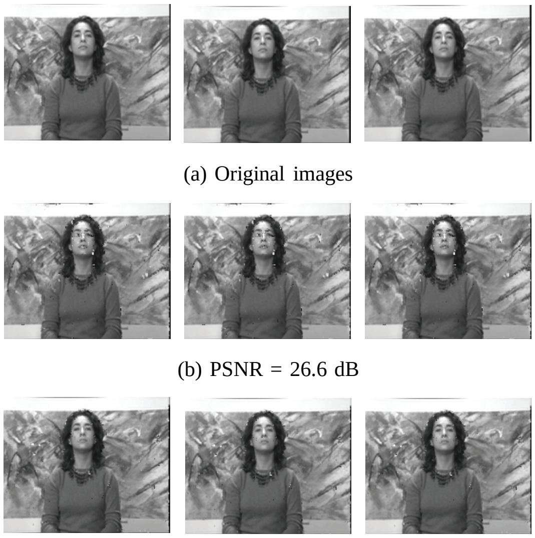
We conduct two sets of experiments. First, we consider the transmission of uncompressed images in pixel domain. In the second experiment, we simulate the transmission of DCT transformed images quantized with the JPEG standard quantization matrices.

A. Performance in pixel domain

In pixel domain, the pixel values generated by the sources are directly mapped to Galois field values and transmitted over the network to the receiver. At the receiver, the network coded symbols are decoded with the proposed iterative message passing algorithm and mapped back to pixel values.

The temporal correlation $g_i(e)$ between the frames in the video sequences is modeled with a zero mean discrete Laplacian distribution [10]. This model is similar to the one used in Distributed Video Coding (DVC) [11], where the correlation noise between the original frame and the side information is commonly considered to have Laplacian distribution (both in pixel and DCT transform domains).

The discrete probability mass functions $\hat{f}_n(\hat{x})$ are estimated from the histograms of the images and are represented as vectors of q values. To further enhance the performance of the system in pixel domain, we take advantage of the smoothness and the spatial correlation that characterizes our data. We exploit this prior knowledge on the data to modify the models of the sources that we provide to the decoder. When a pixel is decoded independently of the neighbouring pixels, the actual probability mass functions of the sources are used. However, when one or more neighbouring pixels have been decoded, we replace the actual probability mass function with a sum of gaussian functions, which are centered at the neighbouring pixel values. With this simple modification we reduce the search space and significantly speed up the convergence of the algorithm without adding complexity in terms of additional nodes in the factor graph. We assume that the correlation information and the source distributions are available at the decoder as side information.



(a) Original images

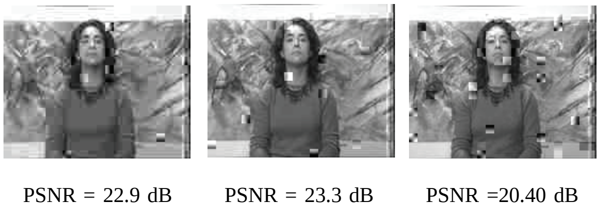
(b) PSNR = 26.6 dB

(c) PSNR = 27.7 dB

Fig. 3: (a) Original images represented with 8 bits per pixel and images represented with (b) 7 and (c) 6 bits per pixel after reconstruction with the proposed message passing algorithm.

In Fig. 3, we present illustrative results of decoding with the proposed message passing algorithm in pixel domain for the Silent QCIF sequence. The number of sources is set to $N = 3$. We assume that for every triplet of pixels only $L = 2$ network coded symbols are available at the decoder, *i.e.*, the decoding is performed with only $2/3$ of the information that is necessary for exact source reconstruction. The maximum number of iterations performed by the decoder for one triplet of source symbols is set to 500. Figs. 3b and 3c show the images that were generated by sources with 7 and 6 bits per source symbols, respectively, and reconstructed with the proposed decoding algorithm. The quantized images were obtained from the original images with pixel values in the range $[0, 255]$ (Fig. 3a) by discarding one and two less significant bits, respectively. The corresponding GFs, where network coding was performed, have size $q = 2^7$ and $q = 2^6$, respectively. We can observe that the proposed algorithm achieves high visual quality reconstruction despite the fact that 1/3 of the information required for exact decoding is lost. We can notice that fewer errors are present in the reconstruction of images represented with 6 bits compared to the images represented with 7 bits, where the majority of the errors appear close to the edges. This can be attributed to the fact that quantization increases the smoothness in terms of both spatial and temporal correlation. The results for the Foreman QCIF sequence are similar and are omitted due to lack of space.

The performance of the proposed decoding scheme with respect to the percent of correctly decoded pixels is given in Table II.A for Silent and Foreman QCIF sequences. The results are averages of 30 realizations. The quantization level n corresponds to the number of less significant bits discarded from the original non quantized images. We can see that the decoding performance is worse for the Foreman QCIF sequence which



PSNR = 22.9 dB PSNR = 23.3 dB PSNR =20.40 dB

Fig. 4: First decoded frame of the Silent sequence for different quality levels: Q = 10 (left), Q = 30 (middle) and Q = 60 (right).

is due to the higher motion, *i.e.,* lower temporal correlation compared to the Silent sequence. Moreover, for the Silent QCIF sequence, the performance in terms of the number of correctly decoded pixels degrades after reaching a maximum value as the number of discarded bits n increases. This is due to the fact that a very coarse quantization reduces the temporal and spatial correlation causing more decoding errors.

B. Performance in DCT domain

For transmission in DCT transform domain, an 8×8 two-dimesional Discrete Cosine Transform (DCT) is first applied on image blocks of 8×8 pixels. The DCT coefficients are then quantized using the JPEG standard quantization matrices and the resulting quantized coefficients are mapped to values in GF and transmitted over the network. At the receiver, the quantized coefficients are recovered from the network coded symbols and decompressed using the same quantization matrices. Finally, inverse DCT is applied to transform the images back to pixel domain.

The correlation noise between a pair of transformed images is modeled again with a zero mean discrete Laplacian distribution. However, contrarily to the pixel domain, in DCT domain the correlation noise is modeled for every DCT frequency band independently. Thus, for an 8×8 DCT transform, 64 parameters per pair of sources need to be estimated, one for every frequency band. For the low frequency bands, the distribution of the DCT coefficients $\hat{f}_n(\hat{x})$ is modeled with a discrete uniform distribution. For the high frequency bands, the probability mass function of the DCT coefficients is approximated with a gaussian distribution. Note that the spatial smoothness condition does not hold in the transform domain, so in this case every symbol is decoded independently of the others. We assume that the parameters of the correlation and source models are available as side information at the decoder.

Results that demonstrate the visual quality of decoded frames in DCT domain are presented in Fig. 4. The simulation parameters are the same as for the pixel domain, *i.e.,* $N = 3$, $L = 2$ and the maximum number of iterations set to 500. The quality level Q determines the level of compression applied to the original images. Higher values of Q correspond to lower amounts of compression. As the quality level increases, the visual quality of decoded images tends to decrease because of the reduction in the amount of correlation. Table II.B presents the percent of correctly decoded DCT coefficients for different quality levels Q. It is worth noting that though the percent of correctly decoded DCT coefficients is in general higher compared to the percent of correctly decoded pixels, the achieved quality in terms of PSNR is lower than in the pixel domain, since one erroneously decoded DCT coefficient causes errors to all the pixels of the corresponding 8×8 block and leads to significant degradation of the PSNR.

TABLE II: Percent of correctly decoded symbols (%).

A. Pixel domain.

n	0	1	2	3	4	5	6
Silent	94.4	97.0	97.5	98.8	98.6	98.1	97.8
Foreman	65.1	71.7	78.7	85.2	89.8	91.5	93.2

B. DCT domain.

Q	10	20	30	40	50	60
Silent	99.8	99.6	99.7	99.4	99.0	99.4
Foreman	99.3	98.8	97.5	98.1	97.3	98.2

V. Conclusions

We have proposed an iterative message passing algorithm for decoding of network coded correlated data. The correlation of the source data is exploited at the decoder in order to compensate for the missing network coded information and to permit approximate recovery of the sources. The proposed algorithm operates on a factor graph that captures the network coding constraints as well as the correlation between the sources. We have demonstrated through simulations that the proposed decoding method is highly effective and is capable of providing a good approximation of the original sources even when 1/3 of the necessary network coded data is missing.

References

[1] T. Ho, M. Médard, M. Effros, and R. Koetter, “Network Coding for Correlated Sources,” in *Proc. of Conf. on Information Science and Systems*, Princeton, NJ, 2004.

[2] Y. Wu, V. Stankovic, Z. Xiong, and S. Y. Kung, “On Practical Design for Joint Distributed Source and Network Coding,” *IEEE Trans. Information Theory*, vol. 55, no. 4, pp. 1709–1720, Apr. 2009.

[3] X. Zhang and S. B. Wicker, “Robustness vs. Efficiency in Sensor network,” in *Proc. of Int. Symp. on Information Processing in Sensor Networks, ISPN’05*, Los Angeles, CA, USA, Apr. 2005.

[4] H. Park, N. Thomos, and P. Frossard, “Approximate Decoding Approaches for Network Coded Correlated Data,” available at `arXiv:1112.4210v1 [cs.NI]`.

[5] L. Iwaza, M. Kieffer, L. Liberti, and K. A. Agha, “Joint Decoding of Multiple-Description Network-Coded Data,” in *Int. Symp. on Network Coding, NETCOD’11*, Beijing, July 2011, pp. 1–6.

[6] G. Maierbacher, J. Barros, and M. Medard, “Practical source-network decoding,” in *Proc. of 6th Int. Symp. on Wireless Communication Systems, ISWCS’09*, Tuscany, 2009, pp. 283 –287.

[7] G. F. Cooper, “The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks,” *Artificial Intelligence*, vol. 42, no. 2-3, pp. 393 – 405, March 1990.

[8] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988.

[9] M. Davey and D. MacKay, “Low-Density Parity Check Codes over GF(q),” *IEEE Communications Letters*, vol. 2, no. 6, pp. 165 –167, June 1998.

[10] S. Inusah and T. J. Kozubowski, “A Discrete Analogue of the Laplace Distribution,” *Journal of Statistical Planning and Inference*, vol. 136, no. 3, pp. 1090 – 1102, March 2006.

[11] C. Brites and F. Pereira, “Correlation Noise Modeling for Efficient Pixel and Transform Domain Wyner-Ziv Video Coding,” *IEEE Trans. Circuits and Systems for Video Technology*, vol. 18, no. 9, pp. 1177 – 1190, Sept. 2008.