Abstract—When correlated sources are to be communicated over a network to more than one sink, joint source-network coding is, in general, required for information theoretically optimal transmission. Whereas on the encoder side simple randomized schemes based on linear codes suffice, the decoder is required to perform joint source-network decoding which is computationally expensive. Focusing on maximum a-posteriori decoders (or, in the case of continuous sources, conditional mean estimators), we show how to exploit (structural) knowledge about the network topology as well as the source correlations giving rise to an efficient decoder implementation (in some cases even with linear dependency on the number of nodes). In particular, we show how to statistically represent the overall system (including the packets) by a factor-graph on which the sum-product algorithm can be run. A proof-of-concept is provided in the form of a working decoder for the case of three sources and two sinks.

I. INTRODUCTION

Motivated by the fact that in many important cases the separation between source and network coding fails [1] and, in general, joint decoding is required [2], this work aims at providing a computationally tractable decoding solution for correlated sources and general networks. In particular, the presented scheme exploits (structural) knowledge about the network topology as well as the source correlation within the system to allow for an efficient implementation of maximum a-posteriori (MAP) decoders (or, in case of continuous sources, decoders based on conditional mean estimation (CME)).

Starting with the problem of distributed source coding, the landmark paper [3] characterized the achievable rates for the case where (discrete-valued) correlated sources are to be communicated to a single sink. In [4] it was shown that linear codes are sufficient when either (non-universal) MAP decoders or even (universal) minimum entropy (ME) decoders are used at the sink. Subsequent research yielded practical encoding and decoding solutions (mostly for a small number of sources), see e.g. [5] and references therein.

The work in [6] considers the problem of communicating (uncorrelated) sources over a network to more than one sink and shows that the maximum network throughput can be achieved by performing network coding [6] for which a practical, algebraic framework based on linear codes was presented in [7].

For scenarios where correlated sources have to be communicated over a network, the achievable rates have been derived in [8] and in [2] it was shown that linear codes are sufficient to achieve those rates when either MAP or ME decoders are used at the sink. Although some attempts have been made to reduce the complexity of ME decoders for increasing block lengths, see e.g. [9], the complexity of both decoder types (MAP and ME) is generally not tractable for a large number of encoders. Therefore, [1] asked the question on whether the joint source-network coding (SNC) problem can be separated and showed that this, in general, is not the case. In summary this means that linear codes suffice at the encoder but also that the decoder becomes highly complex. First attempts to provide practical coding solutions for joint SNC can e.g. be found in [10]. Since those approaches are mostly of sub-optimal nature and only work for a small number of encoders, we look at the problem from a different perspective, building on previous work about joint source-channel coding [11], [12]. The goal is to provide a (MAP and CME) decoder implementation that is tractable for a possibly large number of encoders exploiting knowledge about the network topology and the source correlations. In particular this shall be achieved as follows:

- **Statistical System Representation:** After introducing the considered problem setup in Section II, we show in Section III how to describe the system statistically.
- **Decoding Model:** Using the statistical system representation, we construct in Section IV a decoding model that can be used for an efficient decoder implementation.
- **Iterative Decoder:** In Section V, we describe how the decoding model can be used within an iterative decoding scheme based on the sum-product algorithm and factor-graphs and show that, depending on the model’s properties, the decoding complexity might only increase linearly with the number of nodes.
- **Proof-Of-Concept:** Section VI provides a proof-of-concept in an form of a working decoder implementation, using the counter example for separation in [1].

We believe that this is a valid first step towards practical SNC.
 vertex by defining the minimum-cost subgraphs [13]. Each \( S \)
 vertex is assumed to be independent of \( I_2 \) and \( I_3 \). As discussed in [1], separation does not hold here in general.

II. PROBLEM SETUP

We start by introducing our notation. Random variables are
denoted by capital letters, e.g. \( X \), where its realizations are
denoted by the corresponding lowercase letters, e.g. \( x \). Vectors
do notated by bold letters, e.g. \( x = (x_1, x_2, \ldots, x_N) \). Sets
are denoted by capital calligraphic letters, e.g. \( \mathcal{X} \), where \( |\mathcal{X}| \)
denotes the set’s cardinality. We follow the convention, that
variables (or variable vectors) indexed by a set denote a set of
variables (or a set of variable vectors), e.g. if \( \mathcal{N} = \{1, 2, 3\} \),
then \( x_{\mathcal{N}} = \{x_1, x_2, x_3\} \) and \( x_{\mathcal{X}} = \{x_1, x_2, x_3\} \). Similarly,
we define the crossproduct \( X_{\mathcal{X}} = X_1 \times X_2 \times X_3 \).

We consider a network represented by the directed graph
\( G = (V, \mathcal{E}) \), where \( V = \{v_n : n \in \mathcal{N}\} \) is the set of vertices
(or nodes) \( v_n \), uniquely identified by the indices \( n \in \mathcal{N} \), and
\( \mathcal{E} \) is the set of directed edges \( e_{k,l} = (v_k, v_l) \), where \( k \in \mathcal{N} \)
identifies the parent vertex \( v_k \) and \( l \in \mathcal{N}, l \neq k \), identifies the
child vertex \( v_l \). We assume that the rates \( R_{k,l} \) for all edges
\( e_{k,l} \in \mathcal{E} \) are either known beforehand or chosen adequately,
e.g. by constructing the minimum-cost subgraphs [13]. Each
vertex \( v_n \) within the network may, or may not, have access to the output of a source \( (v_s, s) \), \( s \in \mathcal{S} \), where \( \mathcal{S} \subseteq \mathcal{N} \)
identifies the sources within the network. Similarly, each
vertex \( v_l, t \in \mathcal{T} \), might be a sink node, i.e. a node who wants
to recover the sources \( s \), where \( \mathcal{T} \subseteq \mathcal{N} \) identifies the sinks
within the network. The recovered source (vector) is denoted
as \( \hat{v}_s \), \( s \in \mathcal{S} \). Fig. 1 shows a simple example network with
\( \mathcal{S} = \{1, 2, 3\} \) and \( \mathcal{T} = \{6, 7\} \), illustrating the notation.

The output \( i_s = (i_{s,1}, i_{s,2}, \ldots, i_{s,L}) \) of each source \( s \in \mathcal{S} \)
is considered to be a length-\( L \) vector of i.i.d. source symbols
\( i_{s,l} \) with probability mass function (PMF) \( p(i_{s,l}) = p_i(s), l = 1, 2, \ldots, L \), i.e. \( p_i(s) = \prod_{l=1}^{L} \hat{p}(i_{s,l}) \). The joint output of all
sources \( S \) shall be denoted by the vector (of output vectors)
\( i_S = (i_{s,1}, i_{s,2}, \ldots, i_{s,n}) \). As discussed later, it shall turn out
useful if \( p(i_S) \) furthermore can be expressed (or approximated)
by a product with factors \( p_i(A_{l=1}|i_S), A \subseteq \mathcal{S}, B \subseteq \mathcal{S} \), where
\( p_i(A_{l=1}|i_S) = \prod_{l=1}^{L} p_i(A_{l=1}|i_S) \). See Fig. 1 for an example where
\( p(i_1, i_2, i_3) \) factors into \( p(i_1) \cdot p(i_2) \).

Under the described system setup, the goal is to show how
to design an approximate MAP (CME) decoder that is capable
of exploiting the network topology and the source factorization
for an efficient implementation. Focusing on the case of small
block lengths but a possibly large number of sources, we show
that, depending on network and source properties, it is possible
to design near-optimal decoders, whose complexity grows linearly
with the number of sources, giving rise to practical joint source-network-coding solutions.

III. STATISTICAL SYSTEM MODEL

The decoding concept presented in this work relies on a
statistical representation of the system components (i.e. of the
source model, the nodes and the edges).

Looking at the source model first, the statistical source represent-
dation is directly given by the PMF \( p(i_S), S \subseteq \mathcal{N} \), and its
underlying factorization, as described in the previous section.
Since our decoder relies on graphical models describing the
system’s statistics, we transduce the source model into its
graphical version which consist of a set of variable nodes,
representing the random variables within the system, and a set
of interconnecting function nodes, representing the statistical
dependencies between the connected random variables. Fig. 2
shows an exemplary variable node, representing the source
output \( i_s \), \( a \in \mathcal{S} \), and the function node \( p_{A|B} \), representing
the conditional probability \( p(i_A|i_B), A \subseteq \mathcal{S}, B \subseteq \mathcal{S} \).

We assume that the nodes within the network generally
have full SNC coding capabilities. Each node \( v_n, n \in \mathcal{N} \), might have a source input \( i_n \in \mathcal{I}_n \) (i.e. if \( n \in \mathcal{S} \)), additional
inputs \( y_l \in \mathcal{Y}_l \) from the nodes \( l \in \mathcal{T}, l \neq n \), and outputs
\( x_k \in \mathcal{X}_k \) to the nodes \( k \in \mathcal{K}, k \neq n \). The encoding
function \( \varphi_{n,k} \) at node \( v_n \) with output \( y_k \) is given by the
deterministic mapping \( \varphi_{n,k} : \mathcal{I}_n \times \mathcal{Y}_l \rightarrow \mathcal{X}_k \),
which can be described in a probabilistic fashion by the transition
probabilities \( p(x_k|i_n, y_l) \) chosen to be in \{0, 1\}. Fig. 2 shows the
graphical representation of an exemplary SNC node.

In contrast to the most simplistic case, where the edges
\( e_{k,l} \) deliver the output \( x_l \) of node \( v_l \) perfectly as input \( y_k \)
to node \( v_l \), we allow that the edges represent a discrete
memoryless channel \( (X_l, p(y_k|x_l), Y_k) \) with input alphabet
\( \mathcal{X}_l \), output alphabet \( \mathcal{Y}_k \) and transition probabilities \( p(y_k|x_l) \).
In the graphical representation, the edges \( e_{k,l} \) are represented
by the function nodes \( f_{l|k} \), as illustrated in Fig. 2.

Using the statistical representations of the network components
(i.e. of the source model, the network nodes and the edges), we it is able to construct (by replacement) a model
of the overall system (including the traversed packet paths).

IV. DECODING MODEL

Since the derived system model can not yet be used directly
for decoding, we now show how to construct the decoding
model. We assume that decoder at the considered sink node
has (a) access to the received packets, (b) a-priori knowledge
about the source statistics and (c) a-priori (or transmitted, e.g.
in the packet header) knowledge about the paths traversed by
the packets (in particular about the traversed nodes’ and edges’
transition probabilities) or, equally sufficient, knowledge about
the global transition probabilities between the source outputs
and the received packets.
A. Statistical Packet Representation

For now, we assume full knowledge about the traversed nodes and edges. Using this knowledge, we are able to construct for each sink node \( v_t, t \in T \), and for each received packet a statistical model of the packet’s path within the network. For referencing purposes, we introduce the running index \( m_t = 1, 2, \ldots, M_t \), uniquely identifying the incoming packets at node \( v_t \), and denote the traversed nodes and edges as \( v_t^{(m_t)} \) and \( e_t^{(m_t)} \), respectively. Using the set of all such nodes, \( V_t^{(m_t)} \), and the set of all such edges, \( E_t^{(m_t)} \), we are able to construct a graph \( G_t^{(m_t)} = \{ V_t^{(m_t)}, E_t^{(m_t)} \} \), representing the path of packet \( m_t \) within the network. Using the same techniques as before, the components of \( G_t^{(m_t)} \) can be represented statistically and we obtain the packet model for packet \( m_t \). In the graphical model, we also shall use the superscript \( (m_t) \) to indicate the nodes’ packet index, e.g. \( f_{ik}^{(m_t)} \) or \( \hat{y}_t^{(m_t)} \).

Usually, the statistical model can be simplified considerably. Considering e.g. the graphical model, the concatenation of several function and variable nodes might be replaced by a single, equivalent, function node. The function nodes in this simplified model shall be distinguished, in terms of notation, by the usage of a tilde and updated subscripts, e.g. \( \tilde{f}_{ik} \) and the variables by updated subscripts, e.g. \( \tilde{y}_t \).

Fig. 3 shows the full and simplified packet model for sink node \( v_b \) and packet \( m_4 = 1, 2, 3 \) for the previous example.

B. Constructing the Decoding Model

Considering the decoder at sink node \( v_t, t \in T \), we are able to combine the packet models \( G_t^{(m_t)} \), \( m_t = 1, 2, \ldots, M_t \), as well as the source model \( p(s) \) in order to obtain an overall model, the decoding model \( \tilde{G}_t \), including all statistical dependencies required for decoding. In particular, the models are combined via the source nodes \( v_s^{(m_t)} \) for all \( s \in S \) by setting \( \tilde{y}_s^{(m_t)} = \hat{y}_s \) for \( m_t = 1, 2, \ldots, M_t \) and adding the source statistics \( p(s) \).

How the packet models and the source model are combined to jointly form the decoding model is illustrated in Fig. 3 for the previous example (and simplified packet models).

V. ITERATIVE DECODING

The results presented so far provide us with a statistical description of the overall system that can be exploited for an efficient decoder implementation, as shown below.

                                 Variables \( i_a, x_k, y_l \): Source Statistics \( p_{AB} \): SNC-Node \( v_n \): Edge Function \( f_{ik} \):
\[ i_a \quad \begin{array}{ccc} x_k & y_l \end{array} \]
\[ p(i_a|k) \]
\[ x_k \quad \begin{array}{ccc} p(x_k|i_a, y_l, \ldots, y_{l'}, \ldots) & p(x_k|i_a, y_l, \ldots, y_{l'}) \end{array} \]
\[ p(y_k|x_k) \]

Fig. 2. Statistical model of the system components represented graphically via interconnected variable and function nodes.

A. MAP (CME) Decoding

Considering the decoder at sink node \( v_t, t \in T \), we are interested in recovering the source output \( i_s \) of all sources \( s \in S \). The decoder jointly uses the inputs \( y_s^{(m_t)} \) of all received packets \( m_t = 1, 2, \ldots, M_t \) as well as the a-priori knowledge about the source statistics \( p(s) \) in order to produce an estimate \( \hat{i}_s \) of all sources \( s \in S \). Considering MAP decoding (which is optimal in case where the error probability has to be minimized), the decoder selects the estimates \( \hat{i}_s \) for all \( s \in S \) as follows
\[
\hat{i}_s = \arg \max_{i_s} p(i_s | y_1^{(1)}, y_2^{(2)}, \ldots, y_{M_t}^{(M_t)}).
\]

The core of the problem is to derive the actual values for the conditional probabilities in (1). Using a similar approach as e.g. in [11], [12], it can be shown that the conditional probability can be derived via the marginalization
\[
p(i_s = i | y_1^{(1)}, y_2^{(2)}, \ldots, y_{M_t}^{(M_t)}) = \frac{1}{\gamma} \sum_{i_s \in \mathcal{I}_s} p(i_s, y_1^{(1)}, y_2^{(2)}, \ldots, y_{M_t}^{(M_t)}),
\]
where \( \gamma = 1/p(y_1^{(1)}, y_2^{(2)}, \ldots, y_{M_t}^{(M_t)}) \) can be seen as normalization constant and
\[
p(i_s, y_1^{(1)}, y_2^{(2)}, \ldots, y_{M_t}^{(M_t)}) = p(s) \cdot p(y_1^{(1)}, y_2^{(2)}, \ldots, y_{M_t}^{(M_t)} | s).
\]

Considering the case of CME, the decoding operation reduces, similar as before, to the problem of calculating the conditional probabilities (2), which is not discussed separately here.

It is worth pointing out that it might be possible (depending on the source and packet models) to express the factors in (3) themselves by a product, which can be exploited for an efficient implementation of the marginalization in (2), as outlined in the next section.

B. Factor-Graphs and the Sum-Product Algorithm

It is easy to see that the previously derived decoding model describes a (valid) factorization of the factors in (3) and that its graphical representation can be seen as a factor-graph, see e.g. [14]. Employing the sum-product algorithm [14], which can be run on the factor-graph, the (global) marginalization in (2) can be performed via (local) marginalizations giving

\[ \text{Since the packet index } m_t \text{ can be used to uniquely identify the packet } y_s^{(m_t)}, \text{ no matter from which set of nodes } \mathcal{L} \text{ they originate, the subscript shall be dropped in the following.} \]
rise to an efficient calculation. In particular this is achieved by running an appropriate message passing algorithm along the factor-graph and, depending on whether the message passing procedure terminates, or not (i.e. if the factor-graph is cycle-free, or not), the exact or an approximated value of $p(I_1 = i|y^{(1)}, y^{(2)}, \ldots, y^{(M)})$ is obtained simultaneously for all $i \in I_s$ and all $s \in S$.

C. Complexity Considerations

For the following discussion of the sum-product algorithm's decoding complexity, we consider (for the sake of simplicity) the case where $S = N$. We set $N = |N|$ and define the following parameters concerning the (non-simplified) graphical decoding model: The maximum alphabet size $S = \max_{n \in N} \{\max\{|I_n|, |X_n|, |Y_n|\}\}$, the number of function nodes $N_f$, the number of variable nodes $N_v$, the maximum degree of function nodes $d_f$, i.e. the maximum number of variable nodes (directly) connected to any function node, and, similarly, the maximum degree of variable nodes $d_v$.

If we assume that the complexity of elementary operations (additions, multiplications, look-ups, etc.) is of constant complexity, i.e. $O(1)$, then the complexity of calculating a single message at any function node is of $O(S^{d_f})$ and the complexity of calculating a single message at any variable node is of $O(S)$, similar as e.g. derived in [11], [12]. Considering the case where the factor-graph is cycle-free, an adapted version of the efficient forward-backward algorithm [14] can be employed, and it can be shown that the complexity of calculating all messages at the function nodes is of $O(N_f d_f S^{d_f})$, that the complexity of calculating all messages at the variable nodes is of $O(N_v d_v S)$ and that the overall complexity is given by their sum. For the case where the factor-graph has cycles, an iterative approach is required and the complexities derive to be of $O(T N_f d_f S^{d_f})$ and $O(T N_v d_v S)$ for the function and variable nodes, respectively, where $T \gg 1$ denotes the maximum number of iterations to be performed.

We observe the following: (a) Source and packet models that can be represented by trees lead to a maximum number of function nodes $N_f$ that is of $O(N)$, (b) $d_f$ depends on the connectivity of the nodes within the packet models (i.e. the number of packets that are jointly encoded) and the properties of the given (chosen) the source model, (c) $N_v$ depends on the number of network nodes $v_s$ and their degree and (d) $d_v$ depends on the node connectivity and the given (chosen) source model. We conclude that the decoding complexity, which is clearly governed by the function nodes, is strongly affected by the topology of the packet and source models (exponential dependency on the node degree) and not by the number of nodes (only linear dependency on $N$). This in turn means that the decoding model used for the decoder implementation (which might be simplified, or not) should rather aim for a large number of function nodes with a small degree than a small number with a large degree, i.e. we always should exploit the structural properties of the system to obtain a decoder with low complexity.

VI. PROOF-OF-CONCEPT

Looking at the counter example with three sources and two sinks where separation does not hold [1], as depicted in Fig. 1, we focus on the case of block length $L = 1$, i.e. where each source symbol is encoded and transmitted separately. In our source model the (discrete-valued) source symbols $I_s$ are the quantized versions of a continuous-valued source samples $U_s$, $s = 1, 2, 3$, where the vector $(U_1, U_2, U_3)$ is distributed according to a multivariate Gaussian distribution. To emulate the scenario in [1], where $I_2$ and $I_3$ are correlated and $I_1$ is independent of them, the joint distribution of $(U_1, U_2, U_3)$ is chosen such that $p(u_1, u_2, u_3)$ factors into $p(u_1) \cdot (u_2, u_3)$ where $U_1 \sim N(1, 0)$ and $(U_2, U_3) \sim N(\Sigma, \mu)$ with $\Sigma = \left[ \begin{array}{cc} \rho & \mu \\ 0 & \end{array} \right]$, $\mu = (0, 0)^T$ and correlation coefficient $\rho$. We are able to express this correlation model of the sources $I_1, I_2$ and $I_3$.

---

3For factor-graphs without cycles, the efficient forward-backward algorithm can be employed, see [14].
in terms of (conditional) entropies such that $H(I_1) = H(I_2) = H(I_3) = h$, $H(I_2|I_3) = H(I_3|I_2) = \epsilon$ and $H(I_1, I_2) = h + \epsilon$ where $h$ (under aforementioned assumptions) depends of the chosen quantizer and $\epsilon$ (additionally) on the correlation coefficient $\rho$.

After introducing the constants $R_h, R_e, R_{h+\epsilon}$ and using the results in [8], it can be easily verified that the rates $R_1, R_4 = R_5 = R_6 = R_8 = h$, $R_3, R_6 = R_7 = R_8 + \epsilon$ and $R_4, R_5 = R_7 = R_8 + \epsilon + \delta$ are admissible. Following this results, we choose $R_h = [h]$, $R_e = [\epsilon] + \delta$ and $R_{h+\epsilon} = [h + \epsilon] + \delta$, where $\delta$ corresponds to some additional rate we might be willing to utilize to improve the decoding result.

Considering the system setup in Fig. 1, we choose the encoding functions at node $v_1, v_2$ and $v_3$ as bijective mappings in case the output rate is $R_h$ and as a surjective mapping in case the output rate is $R_e$. For node $v_4$ the coding function from the two inputs to the output corresponds to a mapping representing the modulo-$2^R_h$ addition of the input symbols which is then modified to obtain a output rate $R_h + \delta$. Node $v_5$ simply performs a bijective mapping.

For our numerical results, we choose a 8 level Lloyd-Max quantizer leading to $h = 2.83$ [bit] and choose $\epsilon = 0 \cdot h, 1/2 \cdot h, 3/2 \cdot h, h$ corresponding to $\rho = 1, 0.988, 0.881, 0$ (derived by experiment). We used $10^6$ samples for each

---

4 At this point we neglect a detailed description on how good mappings can be constructed (or selected) since the decoder works for all mappings meeting the above requirements. For our experiments, we choose deterministic mappings preserving the low-resolution information of the sources but our experiments indicate (as expected) that random mappings usually yield good results, especially when $L > 1$.

---

**Table I**

| $\epsilon$ (bit) | $h$ (0.881) | $\delta$ (0.585) |
|------------------|-------------|-----------------|
| $R_h$ (bit)      | 3           | 3               |
| $R_e$ (bit)      | 2           | 2.32            |
| $R_{h+\epsilon}$ (bit) | 5       | 5.32            |
| $P_{1,7} = P_{2,7}$ [bit] | 2       | 282.6-3          |
| $P_{3,7}$ [bit]  | 0           | 4.50e-3          |
| $P_{1,6} = P_{2,6} = P_{3,6}$ [bit] | 0       | 430e-6          |
| $P_{1,5} = P_{2,5}$ [bit] | 0       | 46e-6           |
| $P_{1,4} = P_{2,4}$ [bit] | 0       | 0               |
| $P_{3,5}$ [bit]  | 0           | 0               |
| $SNR_{1,6}$ (dB) | 14.6       | 14.6            |
| $SNR_{2,6}$ (dB) | 14.6       | 14.6            |
| $SNR_{3,6}$ (dB) | 14.6       | 14.6            |
| $SNR_{1,7}$ (dB) | 14.6       | 8.29            |
| $SNR_{2,7}$ (dB) | 14.6       | 12.8            |
| $SNR_{3,7}$ (dB) | 14.6       | 12.8            |
| $\epsilon$ (bit) | $h$ (0.881) | $\delta$ (0.585) |
| $R_h$ (bit)      | 3           | 3               |
| $R_e$ (bit)      | 2           | 2.32            |
| $R_{h+\epsilon}$ (bit) | 5       | 5.32            |
| $P_{1,7} = P_{2,7}$ [bit] | 2       | 282.6-3          |
| $P_{3,7}$ [bit]  | 0           | 4.50e-3          |
| $P_{1,6} = P_{2,6} = P_{3,6}$ [bit] | 0       | 430e-6          |
| $P_{1,5} = P_{2,5}$ [bit] | 0       | 46e-6           |
| $P_{1,4} = P_{2,4}$ [bit] | 0       | 0               |
| $P_{3,5}$ [bit]  | 0           | 0               |
| $SNR_{1,6}$ (dB) | 14.6       | 14.6            |
| $SNR_{2,6}$ (dB) | 14.6       | 14.6            |
| $SNR_{3,6}$ (dB) | 14.6       | 14.6            |
| $SNR_{1,7}$ (dB) | 14.6       | 8.29            |
| $SNR_{2,7}$ (dB) | 14.6       | 12.8            |
| $SNR_{3,7}$ (dB) | 14.6       | 12.8            |

---

**References**

[1] A. Ramamoorthy, K. Jain, P. A. Chou, and M. Effros, “Separating distributed source coding from network coding,” *IEEE/ACM Trans. Netw.*, vol. 14, no. 6, pp. 2785-2795, 2006.

[2] T. Ho, M. Médard, M. Effros, and R. Koetter, “Network coding for correlated sources,” in *Proc. CIST’04*, Princeton, NJ, USA, Mar. 2004.

[3] D. Spielman and J. K. Wolf, “Noisyless coding of correlated information sources,” *IEEE Trans. Inform. Theory*, vol. IT-19, no. 4, pp. 471-480, 1973.

[4] I. Csiszar, “Linear codes for sources and source networks: Error exponents, universal coding,” *IEEE Trans. on Inform. Theory*, vol. 28, no. 4, pp. 585–592, Jul. 1982.

[5] Z. Xiong, A. D. Liveris, and S. Cheng, “Distributed source coding for sensor networks,” *IEEE Signal Processing Magazine*, vol. 21, no. 5, pp. 80–94, Sep. 2004.

[6] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, “Network information flow,” *IEEE Trans. Inform. Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[7] R. Koetter and M. Médard, “An algebraic approach to network coding,” *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, 2003.

[8] L. Song and R. W. Yeung, “Network information flow—multiple sources,” in *Proc. ISIT’01*, Washington, DC, USA, Jun. 2001.

[9] T. P. Coleman, M. Médard, and M. Effros, “Towards practical minimum-entropy universal decoding,” in *Proc. DCC’05*, Snowbird, UT, USA, Mar. 2005, pp. 33–42.

[10] Y. Wu, V. Stankovic, Z. Xiong, and S. Y. Kung, “On practical design for joint distributed source and network coding,” *IEEE Trans. on Inform. Theory*, vol. 55, pp. 1709–1720, 2009.

[11] J. Barros and M. Tuchler, “Scalable decoding on factor trees: a practical solution for wireless sensor networks,” *IEEE Trans. on Communications*, vol. 54, no. 2, pp. 284–294, Feb. 2006.

[12] G. Maierbacher and J. Barros, “Low-complexity coding and source-optimized clustering for large-scale sensor networks,” *ACM Trans. Sens. Netw.*, vol. 5, no. 3, pp. 3–12, 2009.

[13] A. Lee, M. Médard, K. Z. Haigh, S. Gowan, and P. Rube, “Minimum-cost subgraphs for joint distributed source and network coding,” in *Proc. NETCOD’07*, San Diego, CA, USA, Jan. 2007.

[14] F. R. Kschischang, B. Frey, and H.-A. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, 2001.