Practical Source-Network Decoding

Gerhard Maierbacher
Instituto de Telecomunicações
Universidade do Porto, Portugal
gerhard@dcc.fc.up.pt

João Barros
Instituto de Telecomunicações
Universidade do Porto, Portugal
jbarros@fe.up.pt

Muriel Médard
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
medard@mit.edu

***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 conditional mean estimators, in the case of continuous sources), 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 messages) 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 [10] and, in general, joint decoding is required [5], this work aims at providing a computationally tractable decoding solution for correlated sources over 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 decoders (or conditional mean estimators, in case continuous sources are considered).

Starting with the problem of distributed source coding, Slepian and Wolf characterized in their landmark paper [11] the (minimal) achievable rates for the case where (two) correlated sources are to be encoded independently and communicated (over perfect channels) to a single sink. Csiszar showed in [4] that linear codes are sufficient when either (non-universal) maximum a-posteriori (MAP) decoders or even (universal) minimum entropy (ME) decoders are used at the sink. Subsequent research in this area yielded practical encoding and decoding solutions (mostly for a small number of sources), see e.g. [14] and references therein.

Ahlswede et al. considered in [1] the problem of communicating (uncorrelated) sources over a network to more than one sink and showed that the (maximum) achievable rate supported by the network (i.e. the maximum throughput) can be achieved by performing network coding. Koetter and Medard presented in [6] an algebraic framework for network coding based on linear codes giving rise to practical implementations.

For scenarios where correlated sources have to be communicated over a network the achievable rates have been derived by Song and Yeung in [12]. Ho et al. showed in [5] 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 by Coleman et at. [3] to reduce the complexity of ME decoders for growing block lengths, the complexity of both decoder types (MAP and ME) is generally not tractable for a large number of encoders. Therefore, Ramamoorthy et al. [10] 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 it is sufficient to use linear codes at the encoder side but also that we have to deal with high complexity at the decoder side. First attempts to provide practical coding solutions for joint SNC problems can e.g. be found in [13]. 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 on joint source-channel coding [2], [9]. The goal is to provide a feasible decoding solution for joint SNC problems for a possibly large number of encoders in form of a MAP decoder implementation (or a conditional mean estimator (CME) implementation, in the case of continuous sources) that exploits knowledge about the network topology as well as 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 how to describe the system statistically in Section III.
- *Decoding Model:* Using the statistical system representation, we construct a decoding model that can be used for an efficient decoder implementation in Section IV.
- *Iterative Decoder:* In Section V we describe how the derived decoding model can be used within an iterative decoding scheme based on the sum-product algorithm running on factor-graphs and show that, depending on the properties of the decoding model, the decoding complexity can be made to increase linearly with the number of nodes.
- *Proof-Of-Concept:* In Section VI provide a proof-of-concept in an form of a working decoder implementation, using the counter example for separation with three sources and two sinks as presented in [10].

We believe that this scheme is a valid first step towards practical joint SNC.

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 are denoted by bold letters and (if not stated differently) assumed to be column vectors, e.g. $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$. 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 $\mathbf{x}_{\mathcal{N}} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$. Similarly, we define the crossproduct, i.e. if $x_n \in \mathcal{X}_n$, $n = 1, 2, 3$, then $x_{\mathcal{N}} \in \mathcal{X}_{\mathcal{N}} = \mathcal{X}_1 \times \mathcal{X}_2 \times \mathcal{X}_3$.

This work was supported in part by the Fundação para a Ciêcia e Tecnologia (Portuguese Foundation for Science and Technology) under grant SFRH/BD/29918/2006 and in part by the European Commission under Grant FP7-INFSO-ICT-215252 (N-Crave Project). Part of this work was done while the first author was a visiting student at the Laboratory for Information and Decision Systems at the Massachusetts Institute of Technology.

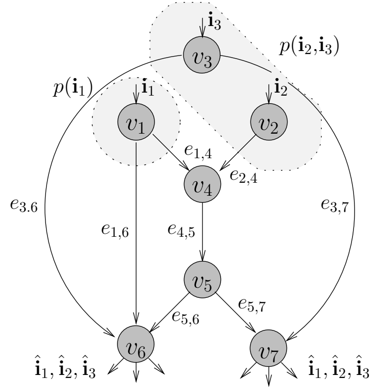


Fig. 1. Exemplary scenario with three sources and two sinks where separation does not hold as presented in [10].

A. Network Topology and Source-Terminal Configuration

We consider a network represented by the directed graph $\mathcal{G}=\{\mathcal{V},\mathcal{E}\}$ where $\mathcal{V}=\{v_n : n\in\mathcal{N}\}$ 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 transmission rates $R_{k,l}$ at all edges $e_{k,l}\in\mathcal{E}$ are either known beforehand or chosen adequately, e.g. by constructing the minimum-cost subgraphs [8] with respect to some cost function of interest.

Each vertex v_s within the network may or may not have access to the output of a (vector) source $\mathbf{i}_s$, $s\in\mathcal{S}$, where the set $\mathcal{S}\subseteq\mathcal{N}$ identifies the sources within the network. Furthermore, each vertex v_n, $n\in\mathcal{N}$, within the network may or may not be a sink, i.e. a node who wants to recover a subset of sources. The set of sink nodes is identified by the set $\mathcal{T}\subseteq\mathcal{N}$. In particular, we denote the subset of sources to be recovered at sink v_t, $t\in\mathcal{T}$, by the set $\mathcal{S}_t\subseteq\mathcal{S}$. The recovered (vector) sources shall be denoted as $\hat{\mathbf{i}}_s$.

Figure 1 shows a simple example network with $\mathcal{S}=\{1,2,3\}$, $\mathcal{T}=\{6,7\}$ and $\mathcal{S}_6=\mathcal{S}_7=\{1,2,3\}$ illustrating the used notation.

B. Source Model and Factorization

The output at each source $s\in\mathcal{S}$ is considered to be a vector of B discrete-valued source symbols such that $\mathbf{i}_s=(i_{s,1},i_{s,2},\ldots,i_{s,B})$ where the vector elements $i_{s,b}$ are drawn i.i.d. according to some probability mass function (PMF) such that $p(i_{s,b})=p(i_s)$ for $b=1,2,\ldots,B$, i.e. $p(\mathbf{i}_s)=\prod_{b=1}^{B}p(i_{s,b})$. The joint output of all sources $\mathcal{S}$ is given by the vector (of output vectors) $\mathbf{i}_\mathcal{S}=(\mathbf{i}_{s_1},\mathbf{i}_{s_2},\ldots,\mathbf{i}_{s_{|\mathcal{S}|}})$. Due to the i.i.d. nature of the source outputs $\mathbf{i}_s$, $s\in\mathcal{S}$, the probability of the joint output $p(\mathbf{i}_\mathcal{S})$ can be expressed by the product $p(\mathbf{i}_\mathcal{S})=\prod_{b=1}^{B}p(i_{\mathcal{S},b})$. As discussed later, it shall turn out useful if also the joint probability $p(\mathbf{i}_\mathcal{S})$ can be furthermore expressed (or approximated) by a product with factors $p(\mathbf{i}_\mathcal{A}|\mathbf{i}_\mathcal{B})$, $\mathcal{A}\subseteq\mathcal{S},\mathcal{B}\subseteq\mathcal{S}$. See Figure 1 for a brief example where $p(\mathbf{i}_1,\mathbf{i}_2,\mathbf{i}_3)$ factors into $p(\mathbf{i}_1)\cdot p(\mathbf{i}_2,\mathbf{i}_3)$ since, as assumed in [10], $\mathbf{i}_1$ is independent of $\mathbf{i}_2$ and $\mathbf{i}_3$.

C. Problem Statement

Under the system setup described above, the goal of this work is to show how to design an approximate MAP decoder (or a CME, in the case of continuous sources) that is capable of exploiting the network topology and the source factorization for an efficient implementation.[1] Focusing on the case of small block lengths[2] but a possibly large number of sources, we want to show that, depending on the degree of the SNC nodes and the properties of the source model, it is possible to design (optimal) decoders, whose complexity grows linearly with the number of considered sources, giving rise to practical joint source-network coding solutions.

[1]In cases where the factorization of $p(\mathbf{i}_\mathcal{S})$ is not approximated and (additionally) the *decoding model*, as introduced later in this work, can be represented by an acyclic factor-graph, the decoder will be optimal. Otherwise, we obtain a reasonably good approximation.

III. Statistical System Model

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

Considering the source model first, the statistical source representation directly follows from the problem statement presented before. The (vector) output $\mathbf{i}_s\in\mathcal{I}_s$ of the sources $s\in\mathcal{S}$ is considered to be a realization of the random variable (vector) $\mathbf{I}_s$ and the source statistics are represented by the joint probability $p(\mathbf{i}_\mathcal{S})$ which, as discussed before, might be expressed by a product with factors $p(\mathbf{i}_\mathcal{A}|\mathbf{i}_\mathcal{B})$, $\mathcal{A}\subseteq\mathcal{S},\mathcal{B}\subseteq\mathcal{S}$.

For the decoder implementation we will use a graphical model representing the statistical dependencies within the network. Those graphical models consist of a set of *variable nodes*, representing the random variable within the system, that are connected via *function nodes*, representing the statistical dependencies between the connected random variable nodes. In Figure 2 e.g. the variable node representing the (vector) output $\mathbf{i}_a$, $a\in\mathcal{S}$, is depicted together with the function node $p_{\mathcal{A}|\mathcal{B}}$ representing the conditional probability $p(\mathbf{i}_\mathcal{A}|\mathbf{i}_\mathcal{B})$, $\mathcal{A}\subseteq\mathcal{S}$, $\mathcal{B}\subseteq\mathcal{S}$.

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.e. if $n\in\mathcal{S}$) and additional inputs from nodes $l\in\mathcal{L}$, $l\neq n$ and outputs to other nodes $k\in\mathcal{K}$,$k\neq n$. The source input $\mathbf{i}_n\in\mathcal{I}_n$ is assumed to be a realization of the random variable $\mathbf{I}_n$. Similarly, the inputs $\mathbf{y}_l\in\mathcal{Y}_l$ from the nodes $l\in\mathcal{L}$ is assumed to be a realization of the random variable $\mathbf{Y}_l$ and the outputs $\mathbf{x}_k\in\mathcal{X}_k$ from the nodes $k\in\mathcal{K}$ is assumed to be a realization of the random variable $\mathbf{X}_k$. Following this definition of the node inputs and outputs, in general, the encoding function $\varphi_{n,k}$ at node v_n with output v_k is given by the (deterministic) mapping $\varphi_{n,k}:\ \mathcal{I}_n\times\mathcal{Y}_\mathcal{L}\ \rightarrow\ \mathcal{X}_k$. Clearly, this mapping can also be described in a probabilistic fashion by employing the transition probabilities $p(\mathbf{x}_k|\mathbf{i}_n,\mathbf{y}_\mathcal{L})$ which are set equal to 1 or 0 depending if a certain input $\mathbf{i}_n$ and $\mathbf{y}_\mathcal{L}$ is mapped onto the output $\mathbf{x}_k$, or not, respectively.[3]

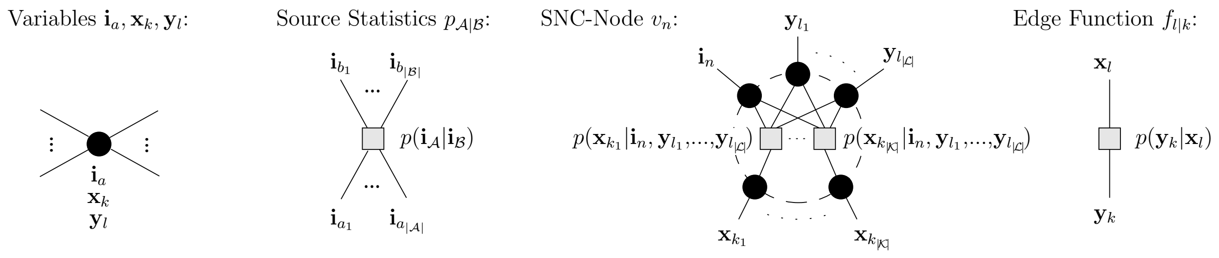
In Figure 2 the statistical model of the SNC nodes is depicted graphically using variable and function nodes.

In contrast to the most simplistic case, where the edges $e_{k,l}$ deliver the output $\mathbf{x}_l$ of node v_k perfectly as input $\mathbf{y}_k$ to node v_l, we allow that the edges represent a discrete memoryless channel $(\mathcal{X}_l,p(\mathbf{y}_k|\mathbf{x}_l),\mathcal{Y}_k)$ with input alphabet $\mathcal{X}_l$, output alphabet $\mathcal{Y}_k$ and transition probabilities $p(\mathbf{y}_k|\mathbf{x}_l)$.

In the graphical representation the edges $e_{k,l}$ with transition probabilities $p(\mathbf{y}_k|\mathbf{x}_l)$ are represented by the function nodes $f_{l|k}$, as shown e.g. in Figure 2. Using the statistical models of

[2]It worth pointing out that this restriction in some cases might not lead to direct drawbacks. Considering e.g. the important case of (random) linear network coding where coding is performed with symbols over the field $\mathbb{F}_{2^8}$ (i.e. 8 bits) the block lengths are short. Similarly, there exist distributed source coding schemes, e.g. those that rely on sending syndromes, that work with good performance for small block lengths. Furthermore, in contrast to universal decoders that require long block lengths, we fully use the source statistics within the design to compensate for the drawbacks raised by the small block length assumption.

[3]Notice that this encoder formulation with given input and output alphabets directly allows for the description of (linear) blockcodes within the system which are sufficient for our purposes.



Variables $\mathbf{i}_a, \mathbf{x}_k, \mathbf{y}_l$:
$\mathbf{i}_a$
$\mathbf{x}_k$
$\mathbf{y}_l$
Source Statistics $p_{\mathcal{A}|\mathcal{B}}$:
$\mathbf{i}_{b_1}$
$\mathbf{i}_{b_{|\mathcal{B}|}}$
$p(\mathbf{i}_\mathcal{A}|\mathbf{i}_\mathcal{B})$
$\mathbf{i}_{a_1}$
$\mathbf{i}_{a_{|\mathcal{A}|}}$
SNC-Node v_n:
$\mathbf{y}_{l_1}$
$\mathbf{i}_n$
$\mathbf{y}_{l_{|\mathcal{L}|}}$
$p(\mathbf{x}_{k_1}|\mathbf{i}_n, \mathbf{y}_{l_1},...,\mathbf{y}_{l_{|\mathcal{L}|}})$
$p(\mathbf{x}_{k_{|\mathcal{K}|}}|\mathbf{i}_n, \mathbf{y}_{l_1},...,\mathbf{y}_{l_{|\mathcal{L}|}})$
$\mathbf{x}_{k_1}$
$\mathbf{x}_{k_{|\mathcal{K}|}}$
Edge Function $f_{l|k}$:
$\mathbf{x}_l$
$p(\mathbf{y}_k|\mathbf{x}_l)$
$\mathbf{y}_k$

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

the network components (i.e. the source model, the network nodes and the edges) it is possible to construct (by simple replacement) a model representing statistically the overall system (including the paths traversed by the packets).

IV. Decoding Model

The statistical system model, as derived before, can not yet be used (at least directly) for decoding purposes. In the following we show how to construct a statistical decoding model that later can be used for an efficient decoder implementation. 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[4]) knowledge about which network nodes and edges were traversed by the received packets as well as the traversed nodes' coding functions and the traversed edges' transition probabilities or, equally sufficient, knowledge about the global transition probabilities between the source outputs and the received messages at the sink node.

A. Statistical Message Representation

For now we assume full knowledge about the traversed nodes and edges and the performed coding operations and transition probabilities. Using this knowledge, we are able to construct for each node v_t, $t \in \mathcal{T}$, and for each received packet a statistical model of the packet path within the network.[5] 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 (in the corresponding configuration) by $v_n^{(m_t)}$ and the traversed edges by $e_{k,l}^{(m_t)}$. Using the set of all such nodes $\mathcal{V}_t^{(m_t)}$ and the set of all such edges $\mathcal{E}_t^{(m_t)}$, we are able to construct a graph $\mathcal{G}_t^{(m_t)} = \{\mathcal{V}_t^{(m_t)}, \mathcal{E}_t^{(m_t)}\}$, representing the path of packet m_t within the network. Using the same models as described before, the graph components of $\mathcal{G}_t^{(m_t)}$ can then be represented statistically and we obtain the *packet model* for packet m_t.

Considering the graphical representation via variable and function nodes, we shall use the same notation with superscript (m_t) to indicate that the corresponding node belongs to packet m_t, e.g. $f_{l|k}^{(m_t)}$ or $\mathbf{y}_l^{(m_t)}$.

In many practical cases not all of the variables within the model might be relevant for the task at hand. Therefore, usually, the statistical model can be simplified considerably. Considering e.g. the graphical model the concatenation of several function nodes (together with the interconnecting variable nodes) might be replaced by a single, equivalent, function node. We shall refer to the original packet model as the *full* model and to the simplified packet model as *simplified* model. The function nodes in the simplified model shall be distinguished, in terms of notation, by the usage of a tilde and updated subscripts, e.g. $\tilde{f}_{\mathcal{L}|\mathcal{K}}^{(m_t)}$, $\mathcal{K} \subseteq \mathcal{N}$, $\mathcal{L} \subseteq \mathcal{N}$, and the variable nodes by updated subscripts, e.g. $\mathbf{y}_\mathcal{L}^{(m_t)}$. In Figure 3 the full and simplified packet model for sink node v_6 and received packets $m_6 = 1, 2, 3$ are shown for the example scenario presented before.

B. Constructing the Decoding Model

Considering the decoder at sink node v_t, $t \in \mathcal{T}$, we are able to combine the packet models represented by the graphs $\mathcal{G}_t^{(m_t)}$, $m_t = 1, 2, \ldots, M_t$, as well as the source model $p(\mathbf{i}_\mathcal{S})$ in order to obtain an overall model, describing the statistical dependencies within the system that are required for decoding. We shall refer to this model as the *decoding model* $\mathcal{G}_t$. In particular, we combine the packet models and the source model via the source nodes $v_s^{(m_t)}$ for all $s \in \mathcal{S}_t$ by setting $\mathbf{i}_s^{(m_t)} = \mathbf{i}_s$ for $m_t = 1, 2, \ldots, M_t$ and add the source statistics $p(\mathbf{i}_\mathcal{S})$ to the resulting model.[6]

How the packet models and the source model are combined to jointly form the decoding model is illustrated graphically in Figure 3 for the exemplary scenario presented before (and simplified packet models).

V. Iterative Decoding

The results so far present us with a model representing the statistical dependencies within the network that can be exploited for an efficient decoder implementation, as presented in the following.

A. Maximum a-Posteriori (MAP) Decoding (and Conditional Mean Estimation (CME))

Considering the decoder at sink node v_t, $t \in \mathcal{T}$, we are interested in recovering the source output (vector) $\mathbf{i}_s$ of all sources $s \in \mathcal{S}_t$. The decoder jointly uses the inputs $\mathbf{y}^{(m_t)}$ of all received packets $m_t = 1, 2, \ldots, M_t$[7] as well as the a-priori knowledge about the source statistics $p(\mathbf{i}_\mathcal{S})$ in order to produce the estimate $\hat{\mathbf{i}}_s$ of all sources $s \in \mathcal{S}_t$. Considering MAP decoding (which is optimal in case where the error probability has to be minimized), the decoder selects the estimates $\mathbf{i}_s$ for all $s \in \mathcal{S}_t$ as follows

$$\hat{\mathbf{i}}_s = \underset{\mathbf{i} \in \mathcal{I}_s}{\operatorname{argmax}} \; p(\mathbf{i}_s = \mathbf{i} | \mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(M_t)}). \tag{1}$$

The core of the problem is to derive the actual values for the conditional probabilities in (1). Using a similar approach as

[4]Sending this side information in the packet header clearly creates some overhead which needs to be discussed separately. For brevity we neglect this discussion here and assume that the decoder uses its a-priori knowledge about the network.

[5]It is important to point out that, in general, some nodes v_n and some edges $e_{k,l}$ might be traversed by a certain packet m_t not at all, once, or several times. Furthermore, the coding function at each node might vary for each packet m_t, e.g. depending on how many inputs where available at the coding instant.

[6]It is worth pointing out that connecting the models in this fashion is allowed since the source output $\mathbf{i}_\mathcal{S}$ (with joint statistics $p(\mathbf{i}_\mathcal{S})$) was initially used to generate all packets emitted at source node v_s, i.e. $\mathbf{i}_s^{(m_t)}$ at $v_s^{(m_t)}$ was the same for $m_t = 1, 2, \ldots, M_t$.

[7]Since the packet index m_t can be used to uniquely identify the packet $\mathbf{y}_\mathcal{L}^{(m_t)}$, no matter from which set of nodes $\mathcal{L}$ they were sent, the subscript shall be dropped in the following.

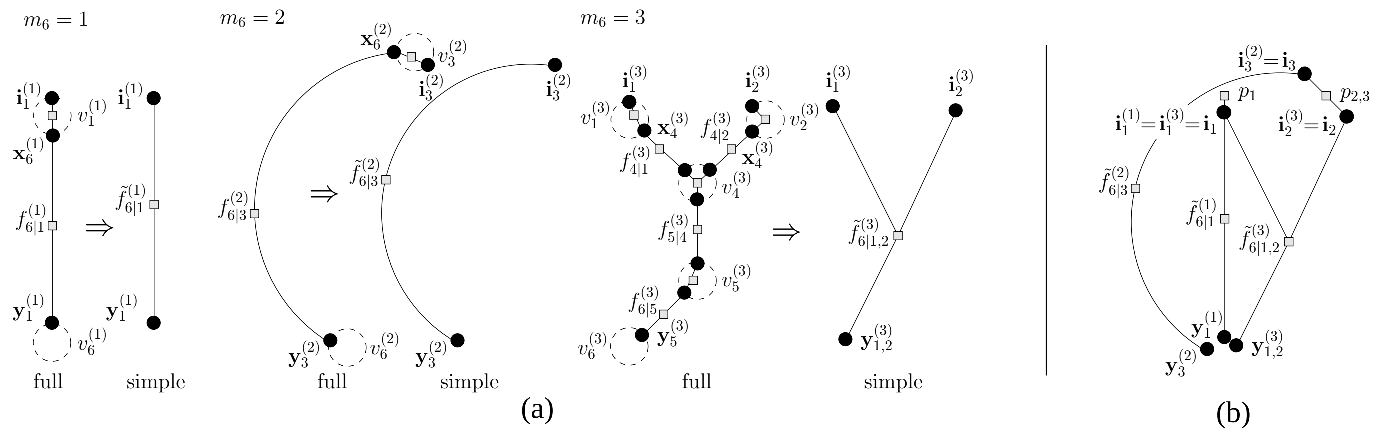


Fig. 3. (a) Using the received packets $m_6 = 1, 2, 3$ at sink node v_6, the full and the simplified packet models $\mathcal{G}_6^{(m_6)}$ can be constructed. (b) The (simplified) decoding model $\mathcal{G}_6$ is then obtained by connecting the (simplified) message models $\mathcal{G}_6^{(m_6)}$, $m_6 = 1, 2, 3$, together with the source model (consisting of the function nodes p_1 and $p_{2,3}$).

e.g. in [2], [9] it can be shown that the conditional probability can be derived via the marginalization

$$p(\mathbf{i}_s{=}\mathbf{i}|\mathbf{y}^{(1)},\mathbf{y}^{(2)},\ldots,\mathbf{y}^{(M_t)}) = \frac{1}{\gamma}\sum_{\mathbf{i}_\mathcal{S}\in\mathcal{I}_\mathcal{S}:\mathbf{i}_s=\mathbf{i}} p(\mathbf{i}_\mathcal{S},\mathbf{y}^{(1)},\mathbf{y}^{(2)},\ldots,\mathbf{y}^{(M_t)}), \tag{2}$$

where $\gamma = 1/p(\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(M_t)})$ can be seen as normalization constant and

$$p(\mathbf{i}_\mathcal{S}, \mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(M_t)}) = p(\mathbf{i}_\mathcal{S}) \cdot p(\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(M_t)}|\mathbf{i}_\mathcal{S}). \tag{3}$$

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

It is worth pointing out that it might be possible (depending on the source and decoding 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 decoding model derived before describes a (valid) factorization of the factors in (3) and its graphical representation is often called a *factor-graph*, see e.g. [7]. Employing the *sum-product algorithm*, also see [7], which runs on the factor-graph, the (global) marginalization in (2) can be performed via (local) marginalizations and giving rise to an efficient calculation. In particular this is achieved by running an appropriate *message passing* algorithm[8] 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 $p(\mathbf{i}_s = \mathbf{i}|\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(M_t)})$ is obtained simultaneously for all $\mathbf{i} \in \mathcal{I}_s$ and all $s \in \mathcal{S}_t$.

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 $\mathcal{S}_t = \mathcal{S} = \mathcal{N}$ for all $t \in \mathcal{T}$. We set $N = |\mathcal{N}|$ and define the following parameters concerning the graphical decoding model (not simplified): The *maximum alphabet size* $S = \max_{n\in\mathcal{N}} \{\max\{|\mathcal{I}_n|, |\mathcal{X}_n|, |\mathcal{Y}_n|\}\}$, 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 $\mathcal{O}(1)$, then the complexity of calculating a single message at any function node is of $\mathcal{O}(\mathcal{S}^{d_f})$ and the complexity of calculating a single message at any variable node is of $\mathcal{O}(d_v S)$, similar as e.g. derived in [2], [9]. Considering the case where the factor-graph is cycle-free, an adapted version of the efficient forward-backward algorithm [7] can be employed, and it can be shown that the complexity of calculating all messages at the function nodes is of $\mathcal{O}(N_f d_f \mathcal{S}^{d_f})$, that the complexity of calculating all messages at the variable nodes is of $\mathcal{O}(N_v d_v^2 S)$ 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 $\mathcal{O}(TN_f d_f \mathcal{S}^{d_f})$ and $\mathcal{O}(TN_v d_v^2 S)$ for the function and variable nodes, respectively, where $T \gg 1$ denotes the maximum number of iterations 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 $\mathcal{O}(N)$, (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_n and their degree, and (d) d_v depends on the topology of 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 model (exponential dependency on 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 aim for a large number of function nodes with a small degree rather than a small number with a large degree, i.e. we always should exploit the structural properties of the (full) decoding model to obtain a decoder with low complexity.

VI. Proof-of-Concept

Considering the counter example with three sources and two sinks presented in [10], we focus on the system setup depicted in Figure 1. To provide some first results, we consider the case of block length $B = 1$, i.e. where each each source symbol is encoded and transmitted separately. In our source model the (discrete-valued) source symbols I_s are the quantized

[8]For factor-graphs without cycles the efficient *forward-backward* algorithm can be employed, see [7].

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 [10], 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 $U_1 \sim \mathcal{N}(1, 0)$ and $(U_2, U_3) \sim \mathcal{N}(\Sigma, \mu)$ with $\Sigma = \left[\begin{smallmatrix} 1 & \rho \\ \rho & 1 \end{smallmatrix}\right]$, $\mu = (0,0)^T$ and correlation coefficient ρ. We are able to express this correlation model of the sources I_1, I_2 and I_3 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 $H(I_2, I_3) = h + \epsilon$ where h (under afore mentioned assumptions) depends of the chosen quantizer and ϵ (additionally) on the correlation coefficient ρ.

After introducing the constants R_h, R_ϵ, $R_{h+\epsilon}$ and using the results in [12], it can be easily verified that the rates $R_{1,6} = R_{1,4} = R_{2,4} = R_{3,7} = R_{5,6} = R_h \geq h$, $R_{3,6} = R_\epsilon \geq \epsilon$ and $R_{4,5} = R_{5,7} = R_{h+\epsilon} \geq h + \epsilon$ are admissible. Following this results, we choose $R_h = \lceil h \rceil$, $R_\epsilon = \lceil \epsilon \rceil + \delta$ and $R_{h+\epsilon} = \lceil h + \epsilon \rceil + \delta$, where δ corresponds to some additional rate we might be willing to utilize to improve the decoding results.

Considering the system setup in Figure 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_ϵ. 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 $R_{h+\epsilon}$.[9] Node v_5 simply corresponds to a bijective mapping from the input to each output.

For our numerical results, we choose a 8 level Lloyd-Max quantizer leading to $h = 2.83$ [bit] and choose $\epsilon = 0 \cdot h, \frac{1}{3} \cdot h, \frac{2}{3} \cdot h, h$ corresponding to $\rho = 1, 0.988, 0.881, 0$ (derived by experiment). We used 10^6 samples for each source and each simulation. We quantify the performance of decoder t decoding the discrete-valued source I_s to form the reconstruction value $\hat{I}_{s,t}$ in terms of the error probability $P_{s,t}$ and, similarly, we consider the output signal-to-noise ratio $\text{SNR}_{s,t} = -10 \log_{10} E\{(U_s - \hat{U}_{s,t})^2\}$ in [dB] to evaluate the performance in the case of continuous-valued sources, $s = 1,2,3$, $t = 6,7$.

Numerical results for the presented setup and several values of δ are summarized in Table I. In Case (a) where U_2 and U_3 are fully correlated (i.e. $u_2 = u_3$) and Case (d) where U_2 and U_3 are statistically independent, we obtain optimal results with a error probability $P_{s,t} = 0$ and an output $\text{SNR}_{s,t} = 14.6$ [dB] (distortion of Lloyd-Max quantization alone). The optimality of the results is expected, since in each of those cases the system degrades and can be represented by an equivalent system that does not require source correlations for decoding. For Case (b) and (c) we need the correlations for decoding. We observe that already for $\delta = 0$ we obtain reasonably good performance by our joint source-network decoding approach. Furthermore, considering Case (b1), (b2), (c1) and (c2), we observe that if we are willing to increase the transmission rate by a small amount $\delta > 0$ then the overall performance improves rapidly, which underlines the capabilities of the decoder to effectively exploit additional redundancy (in the received packets) to improve the overall decoding result.

[9]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 $B > 1$.

TABLE I

ϵ [bit] (ρ)	0 (1)	$\frac{1}{3}h = 0.942$ (0.988)		
δ [bit]	0	0	0.585	1
R_h [bit]	3	3	3	3
R_ϵ [bit]	0	1	1.59	2
$R_{h+\epsilon}$ [bit]	3	4	4.59	5
$P_{1,6} = P_{2,6}$ [bit]	0	0	0	0
$P_{3,6}$ [bit]	0	82.6e-3	43e-6	18e-6
$P_{1,7} = P_{2,7}$ [bit]	0	65.8e-3	43e-6	11e-6
$P_{3,7}$ [bit]	0	0	0	0
$\text{SNR}_{1,6}$ [dB]	14.6	14.6	14.6	14.6
$\text{SNR}_{2,6}$ [dB]	14.6	14.6	14.6	14.6
$\text{SNR}_{3,6}$ [dB]	14.6	12.6	14.6	14.6
$\text{SNR}_{1,7}$ [dB]	14.6	8.29	14.6	14.6
$\text{SNR}_{2,7}$ [dB]	14.6	12.8	14.6	14.6
$\text{SNR}_{3,7}$ [dB]	14.6	14.6	14.6	14.6
Case	(a)	(b)	(b1)	(b2)

ϵ [bit] (ρ)	$\frac{2}{3}h = 1.88$ (0.881)			$h = 2.83$ (0)
δ [bit]	0	0.322	0.585	0
R_h [bit]	3	3	3	3
R_ϵ [bit]	2	2.32	2.59	3
$R_{h+\epsilon}$ [bit]	5	5.32	5.59	6
$P_{1,6} = P_{2,6}$ [bit]	0	0	0	0
$P_{3,6}$ [bit]	28.2e-3	5.20e-3	430e-6	0
$P_{1,7} = P_{2,7}$ [bit]	23.7e-3	4.50e-3	468e-6	0
$P_{3,7}$ [bit]	0	0	0	0
$\text{SNR}_{1,6}$ [dB]	14.6	14.6	14.6	14.6
$\text{SNR}_{2,6}$ [dB]	14.6	14.6	14.6	14.6
$\text{SNR}_{3,6}$ [dB]	9.40	12.3	14.2	14.6
$\text{SNR}_{1,7}$ [dB]	9.17	13.0	14.5	14.6
$\text{SNR}_{2,7}$ [dB]	9.87	12.5	14.2	14.6
$\text{SNR}_{3,7}$ [dB]	14.6	14.6	14.6	14.6
Case	(c)	(c1)	(c2)	(d)

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network Information Flow. *IEEE Trans. Inform. Theory*, 46(4):1204–1216, 2000.

[2] J. Barros and M. Tuchler. Scalable decoding on factor trees: a practical solution for wireless sensor networks. *IEEE Transactions on Communications*, 54(2):284–294, Feb. 2006.

[3] T.P. Coleman, M. Medard, and M. Effros. Towards practical minimum-entropy universal decoding. In *Proceedings of the Data Compression Conference (DCC 2005)*, pages 33–42, Snowbird, UT, USA, March 2005.

[4] I. Csiszar. Linear codes for sources and source networks: Error exponents, universal coding. *IEEE Transactions on Information Theory*, 28(4):585–592, July 1982.

[5] Tracey Ho, Muriel Medard, Michelle Effros, and Ralf Koetter. Network coding for correlated sources. In *Proceedings of the Conference on Information Sciences and Systems (CISS)*, Princeton, NJ, March 2004.

[6] Ralf Koetter and Muriel Médard. An algebraic approach to network coding. *IEEE/ACM Trans. Netw.*, 11(5):782–795, 2003.

[7] F. R. Kschischang, B. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inform. Theory*, 47(2):498–519, 2001.

[8] A. Lee, M. Medard, K. Z. Haigh, S. Gowan, and P. Rubel. Minimum-cost subgraphs for joint distributed source and network coding. In *Proceedings of the Third Workshop on Network Coding, Theory, and Applications (NETCOD 2007)*, San Diego, CA, USA, January 2007.

[9] G. Maierbacher and J. Barros. Low-complexity coding and source-optimized clustering for large-scale sensor networks. Accepted for publication in the ACM Transactions on Sensor Networks. To appear in Vol. 5, Iss. 3, Aug. 2009. Available from `http://arxiv.org/abs/0809.1330`.

[10] Aditya Ramamoorthy, Kamal Jain, Philip A. Chou, and Michelle Effros. Separating distributed source coding from network coding. *IEEE/ACM Trans. Netw.*, 14(SI):2785–2795, 2006.

[11] D. Slepian and J. K. Wolf. Noiseless coding of correlated information sources. *IEEE Trans. Inform. Theory*, IT-19(4):471–480, 1973.

[12] Lihua Song and R.W. Yeung. Network information flow-multiple sources. In *Proceedings of the IEEE International Symposium on Information Theory*, Washington, DC, June 2001.

[13] Y. Wu, V. Stankovic, Z. Xiong, and S. Y. Kung. On practical design for joint distributed source and network coding. *IEEE Transactions on Information Theory*, 55:1709–1720, 2009.

[14] Zixiang Xiong, A.D. Liveris, and S. Cheng. Distributed source coding for sensor networks. *Signal Processing Magazine, IEEE*, 21(5):80–94, Sept. 2004.