Practical source-network decoding

The MIT Faculty has made this article openly available. Please share how this access benefits you. Your story matters.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>As Published</td>
<td><a href="http://dx.doi.org/10.1109/ISWCS.2009.5285249">http://dx.doi.org/10.1109/ISWCS.2009.5285249</a></td>
</tr>
<tr>
<td>Publisher</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>Version</td>
<td>Final published version</td>
</tr>
<tr>
<td>Accessed</td>
<td>Sat Apr 13 09:48:23 EDT 2019</td>
</tr>
<tr>
<td>Citable Link</td>
<td><a href="http://hdl.handle.net/1721.1/59974">http://hdl.handle.net/1721.1/59974</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>Article is made available in accordance with the publisher's policy and may be subject to US copyright law. Please refer to the publisher's site for terms of use.</td>
</tr>
<tr>
<td>Detailed Terms</td>
<td></td>
</tr>
</tbody>
</table>
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. [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 e.g. by constructing the minimum-cost subgraphs [13]. Each $v_i$ to the output of a source (vector) $1_i$ denoted by the corresponding lowercase letters, e.g. $x_i$. Sets are denoted by capital calligraphic letters, e.g. $X$, where $|X|$ 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 $N = \{1,2,3\}$, then $x_N = \{x_1,x_2,x_3\}$ and $x_{|X|} = \{x_1,x_2,x_3\}$. Similarly, we define the crossproduct $X_N = X_1 \times X_2 \times X_3$.

Fig. 1. Exemplary network with three source nodes $v_1, v_2, v_3$ and two sinks $v_6, v_7$. The source symbols $1_i$ are 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 are denoted by bold letters, e.g. $x = (x_1, x_2, \ldots, x_N)$. Sets are denoted by capital calligraphic letters, e.g. $X$, where $|X|$ 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 $N = \{1,2,3\}$, then $x_N = \{x_1,x_2,x_3\}$ and $x_{|X|} = \{x_1,x_2,x_3\}$. Similarly, we define the crossproduct $X_N = X_1 \times X_2 \times X_3$.

We consider a network represented by the directed graph $G = (V,E)$, where $V = \{v_n : n \in N\}$ is the set of vertices (or nodes) $v_n$, uniquely identified by the indices $n \in N$, and $E$ is the set of directed edges $e_{k,l} = (v_k, v_l)$, where $k \in N$ identifies the parent vertex $v_k$ and $l \in N$, $l \neq k$, identifies the child vertex $v_l$. We assume that the rates $R_{k,l}$ for all edges $e_{k,l} \in E$ 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 (vector) $i_s$, $s \in S$, where $S \subseteq N$ identifies the sources within the network. Similarly, each vertex $v_l$, $l \in T$, might be a sink node, i.e. a node which wants to recover the sources $S$, where $T \subseteq N$ identifies the sinks within the network. The recovered source (vector) is denoted as $\hat{i}_s$, $s \in S$. Fig. 1 shows a simple example network with $S = \{1,2,3\}$ and $T = \{6,7\}$, illustrating the notation.

The output $i_s = (i_{s,1}, i_{s,2}, \ldots, i_{s,L})$ of each source $s \in S$ is considered to be a length-$L$ vector of i.i.d. source symbols $i_{s,l}$ with probability mass function (PMF) $p(i_{s,l}) = p(i_l)$, $l = 1,2,\ldots,L$, i.e. $p(i_s) = \prod_{l=1}^{L} p(i_{s,l})$. The joint output of all sources $S$ shall be denoted by the vector (or 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 $p(i_{A,l}|B)$, $A \subseteq S, B \subseteq S$, where $p(i_{A,l}|B) = \prod_{l=1}^{L} p(i_{A,l}|B_l)$. See Fig. 1 for an example where $p(i_1,i_2,i_3)$ factors into $p(i_1) \cdot p(i_2) \cdot p(i_3)$.

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 representation is directly given by the PMF $p(i_{S}, S \subseteq N$, 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 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 $i_k$, $a \in A$, and the function node $p_{A|B}$, representing the conditional probability $p(i_{A}|i_{B})$, $A \subseteq S, B \subseteq S$.

We assume that the nodes within the network generally have full SNC coding capabilities. Each node $v_n$, $n \in N$, might have a source input $i_n \in I_n$ (i.e. if $n \in S$), additional inputs $v_l \in Y_l$ from the nodes $l \in L$, $l \neq n$, and outputs $x_k \in X_k$ to the nodes $k \in K$, $k \neq n$. The encoding function $\varphi_{n,k}$ at node $v_n$ with output $v_k$ is given by the (deterministic) mapping $\varphi_{n,k} : I_n \times Y_L \rightarrow X_k$, 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 $x_k$ of node $v_k$ 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 $X_l$, output alphabet $Y_k$ and transition probabilities $p(y_k|x_l)$. In the graphical representation, the edges $e_{k,l}$ are represented by the function nodes $f_{i_ki_l}$, 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 are 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 \( v^{(m_{t})} \) and \( e^{(m_{t})} \), 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. \( f^{(m_{t})} \) or \( y^{(m_{t})} \).

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. \( f^{\tilde{(m_{t})}} \) and \( y^{\tilde{(m_{t})}} \).

Fig. 3 shows the full and simplified packet model for sink node \( v_5 \) 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 \( p(\mathbf{i}_S) \) in order to obtain an overall model, the decoding model \( G_t \), including all statistical dependencies required for decoding. In particular, the models are combined via the source nodes \( v^{(m_{t})} \) for all \( s \in S \) by setting \( i_s^{(m_{t})} = i_s \) for \( m_t = 1, 2, \ldots, M_t \) and adding the source statistics \( p(\mathbf{i}_S) \).

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.

A. MAP (CME) Decoding

Considering the decoder at sink node \( v_t \), \( t \in T \), we are interested in recovering the source output \( \mathbf{i}_S \) of all sources \( s \in S \). The decoder jointly uses the inputs \( y^{(m_{t})} \) of all received packets \( m_t = 1, 2, \ldots, M_t \) as well as the a-priori knowledge about the source statistics \( p(\mathbf{i}_S) \) in order to produce an estimate \( \hat{\mathbf{i}}_s \) 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 \( \hat{\mathbf{i}}_s \) for all \( s \in S \) as follows

\[
\hat{\mathbf{i}}_s = \arg \max_{i_{S|t}} p(\mathbf{i}_S = i_{S|t} | y^{(1)}, y^{(2)}, \ldots, y^{(M_t)}).
\]

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

\[
p(\mathbf{i}_S = i_{S|t} | y^{(1)}, y^{(2)}, \ldots, y^{(M_t)}) = \frac{1}{\gamma} \sum_{i_{S|t}} p(\mathbf{i}_S, y^{(1)}, y^{(2)}, \ldots, y^{(M_t)}),
\]

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

\[
p(\mathbf{i}_S, y^{(1)}, y^{(2)}, \ldots, y^{(M_t)}) = p(\mathbf{i}_S) \cdot p(y^{(1)}, y^{(2)}, \ldots, y^{(M_t)} | i_{S}).
\]

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

\[1\text{Connecting the models in this fashion is allowed since the same source output } \mathbf{i}_S \text{ was originally used to generate the packets } m_t = 1, 2, \ldots, M_t.\]

\[2\text{Since the packet index } m_t \text{ can be used to uniquely identify the packet } y^{(m_{t})}, \text{ no matter from which set of nodes } L \text{ they originate, the subscript shall be dropped in the following.}\]
rise to an efficient calculation. In particular this is achieved by running an appropriate message passing algorithm\(^3\) 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 \(p(u_1)\cdot p(u_2, u_3)\) where \(U_1 \sim N(1, 0)\) and \((U_2, U_3) \sim N(\Sigma, \mu)\) with \(\Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}\), \(\mu = (0, 0)^T\) and correlation coefficient \(\rho\). We are able to express this correlation model of the sources \(I_1, I_2\) and \(I_3\)

---

\(^3\)For factor-graphs without cycles, the efficient forward-backward algorithm can be employed, see [14].
TABLE I

<table>
<thead>
<tr>
<th>δ [bit]</th>
<th>ρ [bit]</th>
<th>$h = 0.942 (0.988)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{b_{0}}$</td>
<td>0</td>
<td>0.585</td>
</tr>
<tr>
<td>$R_{e_{b}}$</td>
<td>0</td>
<td>0.585</td>
</tr>
<tr>
<td>$R_{b_{0}+}$</td>
<td>0</td>
<td>0.585</td>
</tr>
<tr>
<td>$P_{b_{0}=0}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$P_{b_{0}=1}$</td>
<td>0.926</td>
<td>0</td>
</tr>
<tr>
<td>$P_{b_{1}=0}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$P_{b_{1}=1}$</td>
<td>0.926</td>
<td>0</td>
</tr>
<tr>
<td>$P_{b_{2}}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$P_{b_{3}}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$\text{SNR}<em>{b}\text{,b}</em>{0}$</td>
<td>14.6</td>
<td>14.6</td>
</tr>
<tr>
<td>$\text{SNR}<em>{b}\text{,b}</em>{1}$</td>
<td>14.6</td>
<td>14.6</td>
</tr>
<tr>
<td>$\text{SNR}<em>{b}\text{,b}</em>{2}$</td>
<td>14.6</td>
<td>14.6</td>
</tr>
<tr>
<td>$\text{SNR}<em>{b}\text{,b}</em>{3}$</td>
<td>14.6</td>
<td>14.6</td>
</tr>
<tr>
<td>Case</td>
<td>(a)</td>
<td>(b1)</td>
</tr>
</tbody>
</table>

in terms of (conditional) entropies such that $H(I_{1}) = H(I_{2}) = H(I_{3}) = h$, $H(I_{1}I_{2}) = H(I_{2}I_{3}) = \epsilon$ and $H(I_{2}, I_{3}) = h + \epsilon$ where $h$ (under afore mentioned assumptions) depends of the chosen quantizer and $\epsilon$ (additionally) on the correlation coefficient $\rho$.

After introducing the constants $R_{b_{0}}, R_{e_{b}}$ and using the results in [8], it can be easily verified that the rates $R_{b_{0}} = R_{b_{1}} = R_{b_{2}} = R_{b_{3}} = R_{e_{b}} = \rho > 0$, and $R_{b_{0}} = R_{b_{1}} = R_{b_{2}} = R_{b_{3}} = \rho + \epsilon$ are admissible. Following this results, we choose $R_{b_{0}} = \lceil h \rceil$, $R_{e_{b}} = \lceil \epsilon \rceil + \delta$. In conclusion, we observe that we are willing to increase the transmission rate by a small amount $\delta > 0$ and that the overall performance improves rapidly, which underlines the capabilities of the decoder to effectively exploit additional redundancy (in the received packets) to improve the decoding result.

## Numerical Results

Numerical results for the presented setup and several values of $\delta$ 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 $P_{e_{t}} = 0$ and an output SNR $\text{SNR}_{s,t} = 14.6$ [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) with $\delta = 0$, we need the correlations for decoding. We observe that our joint source-network decoding approach already achieves reasonably good performance for $\delta = 0$. 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.

REFERENCES