Distributed Source Coding Using Symbol-Based Turbo Codes

Mahdi Zamani, Farshad Lahouti
School of Electrical & Computer Engineering, University of Tehran
{zamanym, lahouti}@ut.ac.ir

ABSTRACT
A simple but powerful scheme for distributed source coding based on the concept of binning and symbol-based turbo codes is proposed. The previous works on the compression with side information (SI) using turbo codes and the binning technique are focused on binary turbo codes. The source is considered to be binary or is converted to a binary stream. This conversion, however, reduces the redundancy that could be exploited by the compression algorithm. To achieve higher compression efficiency, we propose using a scheme based on a turbo decoder that decides over symbols rather than bits. The results demonstrate improved performance.

1. INTRODUCTION
According to Slepian-Wolf (SW) theorem [1] separate compression of two sources could be as efficient as their joint compression. Recently, several attempts have been made to construct practical and efficient schemes for distributed source coding [2-5]. Pradhan and Ramchandran [2] constructed a scheme based on channel codes and suggested to use syndromes for binning. Using a more powerful channel code for distributed source coding, results in a higher compression efficiency. As a result, a number of schemes based on turbo codes and LDPC codes have been suggested [3-7].

In many practical applications like sensor networks where there are many highly correlated sources, turbo codes are more suitable than LDPC codes. This is due to the simplicity of the encoder structure and syndrome computation in turbo codes [6].

In [3] and [4] a distributed compression scheme based on turbo codes (turbo DSC) is proposed. The design is based on using parity bits as opposed to syndrome bits. In [3] a bit to symbol and symbol to bit converter is also suggested for using a turbo code at the bit level.

To improve performance Liveris et. al. [5] construct a turbo DSC using syndromes and the binning technique. Their design is based on a modified turbo decoder that switches between a principal and a complimentary trellis. To reduce the complexity, recently, Tu, Li and Blum [6] presented a turbo DSC with channel code binning that use only a conventional turbo decoder.

In distributed source coding applications like sensor networks, we are dealing with distributed and correlated continuous sources of data. The previous works based on turbo codes assume the sources are binary or else they are converted to binary using a symbol to bit converter. However, this conversion reduces the redundancy that could be exploited by the compression algorithm. For the SW problem, non-binary codes are only used in [8]. However, the presented solution is not based on the binning concept, which is optimal [2]. Their construction is based on turbo codes using finite state machine (FSM) trellis. To achieve higher compression efficiency, we propose a scheme based on the binning technique and a turbo decoder that decides over symbols rather than bits. The results demonstrate improved performance.

The paper is organized as follows. Section 2 introduces the DSC problem and the binning technique. Section 3 presents our method for constructing the DSC with symbol-based turbo codes. Section 4 presents the simulation results. Finally, section 5 concludes the paper.

2. DSC PROBLEM AND BINNING APPROACH
2.1. Slepian-Wolf Theorem
Consider two correlated discrete memoryless sources $X$ and $Y$ which are encoded separately but decoded jointly. SW boundary is given by [1]:

$$R_x \geq H(X|Y), \quad R_y \geq H(Y|X),$$
$$R_x + R_y \geq H(X,Y)$$

(1)

where $R_x$ and $R_y$ are the rates for sources $X$ and $Y$, respectively.

If source $Y$ is encoded by $H(Y)$ bits and $X$ is encoded by $H(X|Y)$ bits, one corner point of SW boundary is achieved. The entire SW boundary is achieved by time sharing or code partitioning [9].

---

1 The material is based on research supported by the Iran Telecommunication Research Center (ITRC).
2.2. Binning Technique

As introduced in [10] and constructed in [2], DSC can be designed using the binning technique. The source space is divided into $2^{n-k}$ cosets using an $(n,k)$ channel code and each coset is represented by its syndrome. If source $X$ is represented by its syndrome using $n-k$ bits, a compression ratio of $n/(n-k)$ is achieved. At the decoder, the syndrome of $X$ indicates the coset in which $X$ is located and the nearest sequence to its SI, $Y$ in that coset is decoded as $X$. To model the correlation between $X$ and $Y$, it is assumed that $X$ and $Y$ are related through a BSC channel. The channel code is designed for this channel to decode $X$ using $Y$ and syndrome of $X$.

2.3. Computing the Syndrome in Convolutional and Turbo Codes

An $(n,k)$ convolutional code could be represented by its $k \times n$ generator matrix $G(D)$. We should find the $(n-k) \times n$ parity check matrix $H$ such that $GH^T = 0$. This matrix is referred to as the syndrome former (SF) since the syndrome is computed as $s = XH^T$, where $X$ is a sequence with length $n$. On the other hand inverse syndrome former (ISF) is found such that $H = H^T H^{-1}$. These SF-ISF pair is not unique for a given code.

Turbo codes are constructed with parallel concatenation of two recursive systematic convolutional (RSC) codes such that the systematic bits of the second constituent code is the interleaved version of those of the first constituent code. Now, if we find the ISF of them separately their systematic bits should be the interleaved or deinterleaved version of each other. Since the all zero systematic bits are invariant regardless of the interleaver type, we force the ISF to find a sequence which has an all zero systematic part [8]. It could be proved that we can find one and only one sequence which satisfies these conditions [8]. For the generator matrix $G(D)=[I, P]$ and parity check matrix $H(D)=[P^T, I_{n-k}]$, therefore, we consider $H^{-1}(D)=[0, I_{n-k}]^T$.

Fig. 1 shows the SF and ISF generators of a turbo code. In this figure, $X_s$ indicates the systematic bits, and for the constituent codes 1 and 2 respectively, $X_1$ and $X_2$ indicate the parity bits, $s_1$ and $s_2$ are the syndromes, and $H_1$ and $H_2$ denote the parity check matrices. We use $\Pi$ to indicate the interleaving operation. We have $X = [X_s, X_1, X_2]$ and $s = [s_1, s_2]$.

2.4. Encoding and Decoding of DSC

In an asymmetric DSC scheme, the side information $Y$ is sent using $H(Y)$ bits to the decoder. Decoder also receives the syndrome of source $X$ i.e., $s$. It, then, produces $Z(s)$, a sequence with syndrome $s$, at the output of ISF with input $s$. Next, the word $Y \oplus Z(s) = X(s) \oplus n \oplus Z(s) = W(0) \oplus n$ is decoded using the turbo decoder. Because the syndrome of $W$ is zero, it is a codeword of the channel code and $W+n$ can be decoded and $W$ can be found. Finally, the signal is decoded as $\hat{X} = W \oplus Z$, leading to a decoding error equivalent to only that of the channel decoder.

3. DSC USING SYMBOL-BASED TURBO CODES

3.1. System Model and Motivation

Fig. 2 shows the block diagram of the system under consideration. The sources $X$ and $Y$ are correlated. They are first quantized to $X_q$ and $Y_q$, respectively and then encoded as discussed.

If the quantized symbols are converted to bits the corresponding dependency in terms of mutual information is reduced. Some examples are shown in Table 1. The values are for a first-order linear model,

$$ Y = \alpha X + n $$

(2)

where $\alpha$ is a constant coefficient and $X$, and $n$ are zero mean independent Gaussian random variables with variances $\sigma_x^2$ and $\sigma_n^2$, respectively. The correlation coefficient of $X$ and $Y$ is then given by

**Figure 1.** SF and ISF of turbo codes.

**Figure 2.** System model.
TABLE I
MUTUAL INFORMATION BETWEEN SYMBOLS AND BITS OF SOURCES

<table>
<thead>
<tr>
<th>$\rho$</th>
<th>between symbols of two sources</th>
<th>between symbols of two sources per bit</th>
<th>between bits of two sources</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.98</td>
<td>1.244</td>
<td>0.622</td>
<td>0.618</td>
</tr>
<tr>
<td>0.88</td>
<td>0.7082</td>
<td>0.3541</td>
<td>0.3150</td>
</tr>
<tr>
<td>0.706</td>
<td>0.3662</td>
<td>0.1831</td>
<td>0.1395</td>
</tr>
<tr>
<td>0.503</td>
<td>0.1638</td>
<td>0.0819</td>
<td>0.0528</td>
</tr>
</tbody>
</table>

$\rho(X,Y) = \frac{1}{\sqrt{1+(\sigma_x/\sigma_y)^2}}$ (3)

The values in Table 1 are for a four level Llyod-Max quantizer.

As discussed in section 3.3, a decoder is designed that operates based on symbols as opposed to bits. This allows exploiting the dependency at the symbol level resulting improved performance.

3.2. Model of Dependency used at the Decoder

Consider a four level scalar quantizer designed for a Gaussian source. The quantization regions for the unit power case, are given by $(-\Delta,0],[0,\Delta)$ and $(\Delta,\infty)$. If the correlation between $X_a$ and $Y_a$ is assumed as a virtual quaternary channel, the transition probabilities of this channel are given by:

$$P(Y_{i,j} | X_{i,j}) = \frac{P(X_{i,j},Y_{i,j})}{P(X_{i,j})} = \int_{l_0}^{l_1} \int_{l_2}^{l_3} e^{-\frac{1}{2} (\rho^2 + 2 \eta \Delta)^2} d\rho d\eta \frac{2 \pi}{\sqrt{1-\rho^2}} \delta(Q(l_i) - Q(l_{i+1}))$$

where $l_0 = -\infty$, $l_1 = -\Delta$, $l_2 = 0$, $l_3 = \Delta$, $l_4 = \infty$ and

$$Q(t) = \frac{1}{\sqrt{2\pi}} \int_t^{\infty} e^{-\frac{x^2}{2}} dx$$

These integrals can be solved numerically, resulting the following transition probability matrix for the virtual quaternary channel:

$$P = \begin{bmatrix}
1 - e_1 - e_2 - e_3 & e_1 & e_2 & e_3 \\
e_1 & 1 - e_1 - e_2 - e_3 & e_2 & e_3 \\
e_2 & e_1 & 1 - e_1 - e_2 - e_3 & e_3 \\
e_3 & e_1 & e_2 & 1 - e_1 - e_2 - e_3
\end{bmatrix}$$

Now, consider the case where the symbols are converted to bits using Gray index assignment. The bit representation of $X$ and $Y$ are given by $X_b$ and $Y_b$, respectively. The transition probabilities of the virtual binary channel modeling the dependency at the bit level are given by:

$$P_b = \begin{bmatrix}
1 - \epsilon_1 - \epsilon_2 - \epsilon_3 & \epsilon_1 & \epsilon_2 & \epsilon_3 \\
\epsilon_1 & 1 - \epsilon_1 - \epsilon_2 - \epsilon_3 & \epsilon_2 & \epsilon_3 \\
\epsilon_2 & \epsilon_1 & 1 - \epsilon_1 - \epsilon_2 - \epsilon_3 & \epsilon_3 \\
\epsilon_3 & \epsilon_1 & \epsilon_2 & 1 - \epsilon_1 - \epsilon_2 - \epsilon_3
\end{bmatrix}$$

3.3. Symbol-Based Turbo Codes

As discussed, the system presented in Fig. 2 hold the promise of a higher compression rate due to a symbol-based turbo decoder module.

Bingeman and Khandani [11] proposed a method which decodes a binary turbo coded signal over a merged trellis, aiming at combining coding and modulation. They presented a method which parses the input block into symbols and interleaves on a symbol-by-symbol basis. In this scheme, the stages of original trellis are merged and the interleaving is performed over symbols as opposed to bits. Fig. 3 shows the trellis diagram when two stages are merged.

We use the same concept of merged trellis for decoding at the symbol level. Now, at the decoder the conditional probabilities are computed over symbols rather than bits. For example, when two stages of the trellis are merged, the probabilities are given by $P(A | B)$ exactly, as opposed to $P(a_i | b_j)$ where $a_i a_j$ and $b_i b_j$ are the bit representations of symbols $A$ and $B$, respectively. As discussed, this results in improved decoding performance.

In this scheme, if we have $M$ states, at most $\log_2 M$ stages could be merged and we can have at most $M$ symbols. There are $M$ branches leaving each state in a $\log_2 M$ -stage merged trellis diagram which correspond to the $M$ possible inputs. Then we choose $\nu \geq \log_2 M$, where $\nu$ is the number of memory elements [11].

The logarithm of log-likelihood ratio (LLR) values in the modified BCJR algorithm is defined as:

$$P_1 = P(Y_a = 0 | X_a = 1) = \frac{\eta_1(2\epsilon_1 + \epsilon_2 + 3\epsilon_3) + \eta_2(\epsilon_1 + \epsilon_2 + \epsilon_3)}{\eta_1^2 + \eta_2^2}$$

$$P_2 = P(Y_a = 1 | X_a = 0) = \frac{\eta_1(\epsilon_1 + \epsilon_2 + \epsilon_3) + \eta_2(2\epsilon_1 + 3\epsilon_2 + \epsilon_3)}{\eta_1^2 + \eta_2^2}$$

where $\eta_1 = P(-\Delta < X_a)$ and $\eta_2 = P(X < -\Delta)$. Therefore, in general the dependency between the corresponding bits of the two sources may be modeled by a binary asymmetric channel (BAC) as opposed to the popular BSC channel used in the previous works. As discussed next, in this work, since the proposed decoder operates at the symbol level, the true symbol-level transition probabilities are used.
1. \( LLR(d_i = k) = \log \left( \frac{Pr \{ d_i = k \mid Y \}}{Pr \{ d_i = 0 \mid Y \}} \right) \quad k = 0, 1, \ldots, M - 1 \) (7)

4. SIMULATION RESULTS

The performance of the proposed scheme is analyzed by simulation using a rate 1/3 parallel concatenated convolutional code with constituent codes 
\[ G_i = G_s \left[ \frac{1 + D + D^2 + D^3}{1 + D^2 + D^3} \right], \text{ length } 10^4 \text{ random interleaver and } 20 \text{ decoding iterations.} \]

and 20 decoding iterations. The correlation model is 
\[ Y = qX + n \text{ and a 4 level Llyod-Max quantizer is used.} \]

The presented scheme and the binary scheme of [6] are simulated. Fig. 4 demonstrates the bit error rate performance vs. the correlation coefficient between \( X \) and \( Y \). At \( \rho = 88.5\% \), the proposed scheme achieves lossless compression (BER=10^-6), while the binary scheme results in a BER of 10^-2. The proposed method is therefore, only 0.04 away from the theoretical limit. This indicates improved performance, when compared to a gap of 0.07 reported in [6] and 0.09 reported in [5], using identical setups. Fig. 5 presents the performance results in terms of reconstructed signal to noise ratio. These results also indicate that the proposed scheme achieves a noticeable performance improvement.

Our simulations using the setup of [8] indicate that the proposed scheme remains only about 0.035 away from the theoretical limit, while the symbol-based scheme of [8] reports a gap of about 0.05.

8. CONCLUSION

A scheme for asymmetric noiseless distributed source coding using symbol-based turbo codes is proposed. The suggested solution exploits higher level of dependency at the symbol level as opposed to the bit level and hence, results in improved performance. In the future works other quantization schemes may be considered.

9. REFERENCES