Distributed coding using punctured quasi-arithmetic codes for memory and memoryless sources

Simon Malinowski
IRISA/Univ. of Rennes
Simon.Malinowski@irisa.fr

Xavi Artigas
Technical University of Catalunya
xavi@gps.tsc.upc.edu

Christine Guillemot
IRISA/INRIA
Christine.Guillemot@irisa.fr

Luis Torres
Technical University of Catalunya
luis@gps.tsc.upc.edu

***Abstract*— This paper considers the use of punctured quasi-arithmetic (QA) codes for the Slepian-Wolf problem. These entropy codes are defined by finite state machines for memoryless and first-order memory sources. Puncturing an entropy coded bit-stream leads to an ambiguity at the decoder side. The decoder makes use of a correlated version of the original in order to remove this ambiguity. A complete DSC scheme based on QA encoding with side information at the decoder is presented. The proposed scheme is adapted to memoryless and first-order memory sources. Simulation results reveal that the proposed scheme is efficient in terms of decoding performance for short sequences compared to well-known DSC using channel codes.**

I. Introduction

Distributed source coding (DSC) addresses the problem of compressing correlated sources by encoding them separately and jointly decoding them. DSC is mainly applied in sensor networks, where several sensors measure a given signal and provide these correlated measures to a base station, which decodes them jointly. Recently, DSC has also been applied to video compression by exploiting the temporal correlation between consecutive images in a video sequence [1]. This correlation between consecutive images is used at the decoder side. DSC theory is based on the Slepian-Wolf theorem established in [2]. This theorem states that, even if the encoders of X and Y do not communicate with each other, lossless compression of X and Y can be achieved if the rates R_X and R_Y satisfy $R_X + R_Y \geq H(X,Y)$, $R_X \geq H(X \mid Y)$, and $R_Y \geq H(Y \mid X)$, provided that X and Y are decoded jointly. Later, this result was extended in [3] in order to compute rate-distortion bounds. In this paper, we focus on the so-called *asymmetric* Slepian-Wolf problem, in which the second source Y is encoded at its entropy rate $H(Y)$ and the first one X at the rate $H(X|Y)$. At the decoder side, X is estimated using its compressed version and the side information Y. The practical applications of the asymmetric Slepian Wolf problem mostly use capacity-approaching channel codes in order to compress the original message. Regular turbo codes are for instance used in [4][5][6], whereas low density parity check codes are used in [7]. Recently, irregular turbo codes have been applied to the Slepian-Wolf problem in [8].

Entropy codes have also been used recently for the Slepian-Wolf problem. The idea behind using source codes for that issue is to exploit their high compression capability together with their capability to exploit the source memory. In [9], the authors have designed Huffman and arithmetic codes for multilevel sources. Two approaches based on overlapped arithmetic and quasi-arithmetic codes have also been developed in parallel in [10] and in [11]. The overlapping mechanism introduced in the arithmetic encoding process induces some ambiguity in the encoded bit-stream. The decoder makes use of the correlated source in order to remove this ambiguity. This technique reveals better performance than the techniques based on channel codes for short sequences. Using entropy codes for DSC presents many interests:

- In contrast with entropy codes, channel codes are efficient only for long sequences (typically more than 10^5 symbols).
- Entropy codes are better suited to take into account the source probabilities and the memory of the source.

In this paper, we propose an alternative DSC scheme based on QA codes. We first consider memoryless sources. It is shown in [12] that QA codes adapted to memoryless sources can be represented by finite state machines (FSM). Hence, an optimal BCJR algorithm [13] can be applied at the decoder side, using the state model proposed in [14]. The branch metrics of the BCJR algorithm are modified so that the side information, coming from the correlated version of the original message, is exploited in the decoding process. The FSM are then extended to account for the realization of the previous symbol. The performance of punctured QA code for Slepian-Wolf coding has been assessed against the one obtained with turbo codes, for both memory and memoryless sources.

II. Description of the proposed DSC scheme

The proposed DSC scheme is represented in Fig.1. Let $\mathbf{S} = S_1, \ldots, S_{L(\mathbf{S})}$ be a source sequence of length $L(\mathbf{S})$ taking its value into the binary alphabet $\mathcal{A} = \{a, b\}$. The probability of the more probable symbol (MPS), $\mathbb{P}(a)$ will be denoted p in the following. Note that, for sake of clarity, only binary sources are considered in this article, but the whole method can be applied to non-binary sources. The symbol sequence $\mathbf{S}$ is encoded using a QA code, producing the bit-stream $\mathbf{X} = X_1, \ldots, X_{L(\mathbf{X})}$ of variable length $L(\mathbf{X})$. In order to reach a targeted overall rate, some bits of $\mathbf{X}$ are punctured. The resulting bit-stream is sent over an ideal channel. At the decoder side, a BCJR algorithm [13] is applied. This algorithm makes use of the received bits of $\mathbf{X}$, together with an additional side

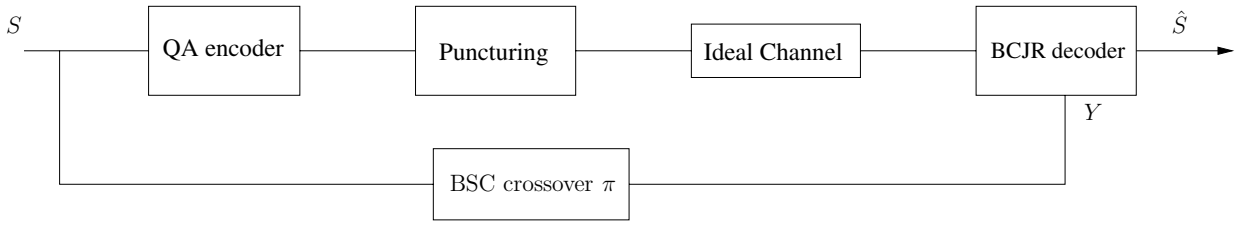

Fig. 1. The proposed DSC scheme

information $\mathbf{Y}$. This side information is obtained by passing $\mathbf{S}$ through a binary symmetrical channel of crossover probability π. In other words, $\forall i \in \{1, ..., L(\mathbf{S})\}$, $\mathbb{P}(\mathbf{Y}_i = \mathbf{S}_i) = 1 - \pi$. In the following, the notation $Y_m^{m'}$ will denote the sequence $Y_m, \ldots, Y_{m'}$. This scheme is explained in more detail in the rest of this section.

A. Quasi-arithmetic codes

It is shown in [12] that QA codes can be represented by two FSM, one for the encoding and one for the decoding. The way to build these FSM is also explained in the same paper. A QA code is defined for an integer N which represents the precision of the code (the initial interval is $[0, N]$) and for a given input source distribution. The way to create the automata of a QA code is detailed in [12] for memoryless sources. In the following, we will also consider first-order memory sources.

The construction of the encoding automaton representing a QA code for a first-order memory source is detailed in the following. A first order memory source is uniquely defined by the probability of the MPS $\mathbb{P}(a) = p$ and a correlation coefficient ρ_m ($\rho_m \in [-1, 1]$). Then, we have $\mathbb{P}(a \,|\, a) = 1 + (\rho_m - 1)(1 - p)$ and $\mathbb{P}(b \,|\, b) = 1 + (\rho_m - 1)p$. The design of QA codes for first order memory sources is inspired from the technique of [12]. The initial interval of the QA encoder automaton is set to $[0, N]$. Two new states are computed for symbols a and b by partitioning the interval $[0, N]$ according to p and $1-p$ respectively. The previous symbol for each new state is included in the state in order to account for the memory of the source. For every state that is created, the interval representing the state of the encoder is partitioned according to the probabilities $\mathbb{P}(a \,|\, a)$ and $\mathbb{P}(b \,|\, a)$ if the previous symbol of the current state is a and to $\mathbb{P}(a \,|\, b)$ and $\mathbb{P}(b \,|\, b)$ if the previous symbol of the current state is b. Once every new state has been visited, the encoding automaton representing the QA code is created. The encoding automaton for $p = 0.8$ and $\rho_m = 0.3$ is depicted in Fig.2. This automaton has been built according to the technique described above. The initial state of this automaton is state 0, corresponding to the initial interval $[0, 8]$ (this state does not contain the memory of the previous symbol). States $1\,(a)$ and $2\,(b)$ have been obtained by partitioning the initial interval according to p and $1 - p$ respectively. Note that the symbol in parenthesis in the labelling of a state corresponds to the memory of the previous symbol. Hence, a state labeled with (a) can only be reached if the previous realization of the source is a. Then, the probabilities $\mathbb{P}(a \,|\, a)$ and $\mathbb{P}(b \,|\, a)$ have been used to partition state $1\,(a)$ and the probabilities $\mathbb{P}(a \,|\, b)$ and $\mathbb{P}(b \,|\, b)$ to partition

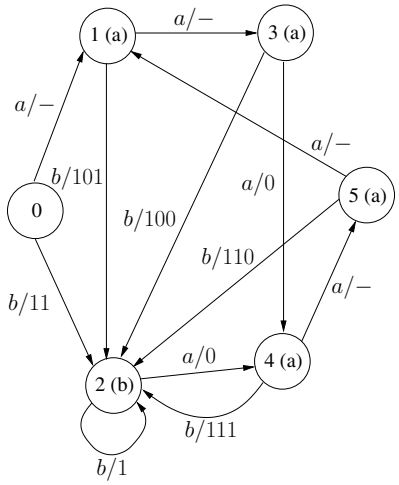

Fig. 2. Encoding automaton for a QA code with memory. This automaton is designed for $p = 0.8$ and $\rho_m = 0.3$

state $2\,(b)$. The same operations are repeated until the graph is completed.

B. Puncturing

The encoding of $\mathbf{S}$ results in a bit-stream $\mathbf{X}$ of variable length $L(\mathbf{X})$. Let us denote by R_q the compression rate of the considered QA code. Then the average length of $\mathbf{X}$ is equal to $R_q \times L(\mathbf{S})$. Let a targeted compression rate be denoted by R_t, with $R_t < R_q$. In order to reach R_t, $\lceil (R_t - R_q) \times L(\mathbf{S}) \rceil$ bits in $\mathbf{X}$ have to be punctured. Different techniques exist in order to find the puncturing positions. In [15], the bits are inserted line by line into a square matrix, and punctured column by column. In our case, the technique leading to the best decoding result is the spread of the punctured bits inside the bit-stream. The punctured bits are separated by $\lfloor (R_t - R_q) \times L(\mathbf{S}) / L(\mathbf{X}) \rfloor - \delta$ bit positions, where $\delta = 1$ if $\lfloor (R_t - R_q) \times L(\mathbf{S}) / L(\mathbf{X}) \in \mathbb{N}$ and $\delta = 0$ otherwise . Note that if $(R_t - R_q) \times L(\mathbf{S}) > L(\mathbf{X})/2$ (i.e., if more than half of the bits in $\mathbf{X}$ have to be punctured), the above technique is used to compute the non-punctured positions. The interest of this technique is that the decoder only needs to know the interval between two punctured bits to recover the punctured positions (it is assumed that the first punctured bit is in the first position).

C. Soft decoding with side information

In this section, the decoding algorithm of punctured quasi-arithmetic codes with side information is detailed. As the side information is an information about the symbols, an

automaton sequential with respect to symbols is used. Let $\mathcal{I} = \{e_0, \ldots, e_d\}$ be the set of states of the QA automaton. For every transition t in the automaton, let b_t denote the sequence of bits output by t and $\overline{b_t}$ the length of b_t. In the proposed DSC scheme, the BCJR decoder takes as input the received bit-stream $\mathbf{X}$ (that contains punctured bits) and the side information $\mathbf{Y}$. In order to optimally exploit the side information on the symbols, a symbol clock based state model is used. This model is adapted from the model proposed in [14] for QA codes. It is defined by the pair of random variables $\mathcal{V}_k = (N_k, M_k)$, where N_k is the state of the QA automaton at the symbol clock instant k (i.e., $N_k \in \mathcal{I}$), and M_k represents the possible bit clock values at the symbol clock instant k. The transition probabilities on this model are given by:

$$\begin{aligned} &\forall (e_i, e_j) \in \mathcal{I} \times \mathcal{I}, \forall (m, m') \in \mathbb{N} \times \mathbb{N} \\ &\mathbb{P}(N_k = e_i, M_k = m' \,|\, N_{k-1} = e_j, M_{k-1} = m) = \\ &\begin{cases} \mathbb{P}(N_k = e_i \,|\, N_{k-1} = e_j) & \text{if } m' - m = \overline{b_t} \\ 0 & \text{otherwise,} \end{cases} \end{aligned} \tag{1}$$

where $\mathbb{P}(N_k = e_i \,|\, N_{k-1} = e_j)$ are deduced from the source statistics. This model allows us to integrate the side information brought by $\mathbf{Y}$ in the decoding trellis. To take into account this additional information, the branch metric $\gamma_k(N_k, M_k \,|\, N_{k-1}, M_{k-1})$ of a transition, triggered by symbol s, in the trellis is modified as follows :

$$\begin{aligned} &\gamma_k(N_k = e_i, M_k = m' \,|\, N_{k-1} = e_j, M_{k-1} = m) = \\ &\mathbb{P}(N_k = e_i, M_k = m' | N_{k-1} = e_j, M_{k-1} = m) \\ &\times \mathbb{P}(X_m^{m'} = b_t) \times \mathbb{P}(S_k = s \,|\, Y_k), \end{aligned} \tag{2}$$

where $\mathbb{P}(X_m^{m'} = b_t)$ is computed from $\mathbf{X}$ (taking into account the punctured bits in $\mathbf{X}$) and $\mathbb{P}(S_k = s \,|\, Y_k)$ is computed from the correlation factor π between $\mathbf{S}$ and $\mathbf{Y}$.

The BCJR algorithm, is applied on the state model (N_k, M_k) defined above. For all state $v = (n, m), n \in \mathcal{I}, 1 \leq m \leq L(\mathbf{X})$ of the trellis, the following probability functions are computed:

$$\alpha_k(v) = \mathbb{P}(\mathcal{V}_k = v; \mathbf{Y}_1^k) \text{ and } \beta_k(v) = \mathbb{P}(\mathbf{Y}_{k+1}^{L(\mathbf{S})} \,|\, \mathcal{V}_k = v), \tag{3}$$

for $1 \leq k \leq L(\mathbf{S})$. Let us define, for every symbol s of the alphabet, the set $\Omega(s)$ of state pairs ($v = (n, m), v' = (n', m')$) in the decoding trellis, such that the input symbol of the transition from v to v' is equal to s. Then, the symbol marginal probabilities on the trellis defined by the states $\mathcal{V}_k$ are obtained by:

$$\begin{aligned} &\forall k \in [1, L(\mathbf{S})], \forall s \in \mathcal{A}, \\ &\mathbb{P}(S_k = s \,|\, X_1^{L(\mathbf{X})}; Y_1^{L(\mathbf{S})}) \\ &\propto \sum_{(v,v') \in \Omega(s)} \alpha_k(v)\, \beta_{k+1}(v')\, \gamma_{k+1}(v' \,|\, v), \end{aligned} \tag{4}$$

where $\gamma_{k+1}(v' \,|\, v)$ is calculated from Eqn.2.

The BCJR algorithm applied on this state model allows the computation of posterior symbol marginals, and hence the minimisation of the symbol error rate.

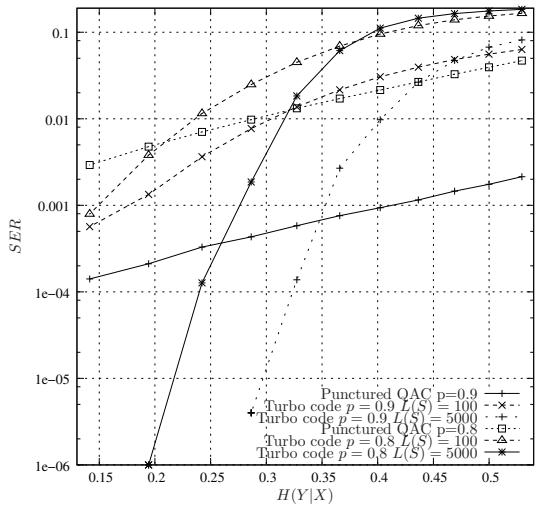

Fig. 3. Performance comparison of DSC schemes at an overall rate of 0.4 bps for memoryless sources

III. Simulation results

The DSC scheme described above has been applied to both memoryless sources and first-order memory sources, assuming that the correlated side information is available at the decoder. In both cases, we have chosen a fixed rate and computed the symbol error rates at the output of the decoder for different crossover probabilities. The results have been compared to turbo-code based DSC schemes.

In the first experiment, we have considered memoryless sources. Two different sources have been used, the first one with $p = 0.9$ and the second with $p = 0.8$. The entropy of these two sources is 0.4690 and 0.7219 respectively. The proposed DSC scheme has been applied for sequences of $L(\mathbf{S}) = 100$ symbols. The SER at the output of the decoder is averaged over 10^5 realizations. The average rate at the input of the ideal channel is set to 0.4 bits per input symbol. The QA code adapted to the first source ($p = 0.9$) and built according to the method of [12] for $N = 16$ has a compression efficiency of 0.48. It means that, in average, 8 bits of the encoded bit-stream have to be punctured to reach the overall rate of 0.4. The QA code adapted to the second source ($p = 0.8$) has a compression efficiency of 0.72. Hence, 32 bits in average are punctured at the output of this QA encoder. The symbol error rate at the output of the decoder for these two different sources are plotted for different values of the correlation with the side information $\mathbf{Y}$ (i.e., for different values of $H(Y \,|\, X)$) in Fig. 3. In this figure, we have also depicted the decoding performance of punctured turbo codes for the same simulation parameters. The considered turbo code is a parallel concatenation of two (21,37) octal convolutional codes. The interleavers of the turbo codes are randomly chosen, and 15 iterations have been computed in order to evaluate the decoding performance of these codes. Note that, the source probabilities ($p = 0.9$ or $p = 0.8$) have been accordingly integrated to the turbo decoder, as this additional information is also available when QA codes are used.

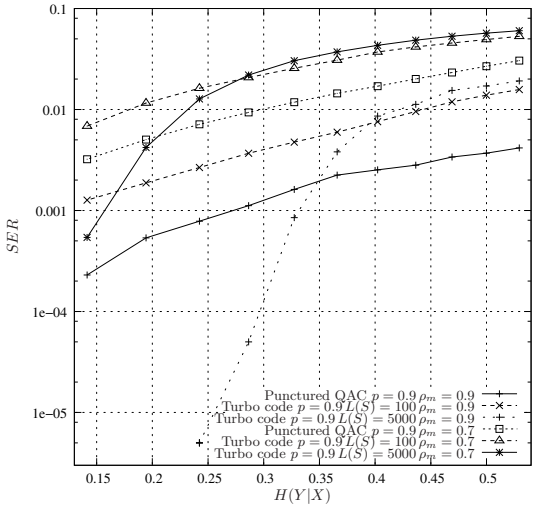

Fig. 4. Performance comparison of DSC schemes at an overall rate of 0.09 bps for first order memory sources

At the overall rate of 0.4 bps, punctured QA codes offer better performance than punctured turbo codes for the two considered input sources and for the same sequence lengths. This can be explained by the fact that turbo codes are less efficient for short sequences. The same punctured turbo codes have also been used for sequences of $L(\mathbf{S}) = 5000$ symbols. In that case, as it can be seen in Fig. 3, the performance of these codes is significantly improved. Indeed, the performance of turbo codes are improved when the constituent interleaver of the code is longer.

In the second experiment, we have considered first order memory sources. The sources are defined for $p = 0.9$ and a memory correlation factor $\rho_m = 0.9$ and $\rho_m = 0.7$. The entropy of these sources are 0.1164 and 0.2591 respectively. The QA codes adapted to these two sources lead to an average compression factor of 0.12 and 0.27 respectively. The proposed DSC scheme has been applied for sequences of $L(\mathbf{S}) = 100$ symbols at an overall rate of 0.09 bits per symbol. The decoding performance in terms of SER of the proposed DSC scheme is depicted in Fig. 4 for the two different sources described above. These results are compared with a turbo code for the same parameters. The convolutional codes of this turbo code are the same as in the first experiment (i.e., (21,37) octal convolutional code). The decoder of the turbo code consists in two BCJR algorithms which exchange their extrinsic information. The first BCJR takes into account the memory of the source, whereas the second decoder can only take into account the source probability (p) because this decoder takes as input an interleaved version of the original message (that contains memory). Hence, the memory of the source is broken due to the interleaver in the second branch of the turbo code.

It can be seen in Fig. 4 that for short sequences, the proposed DSC scheme performs better than the turbo code based DSC scheme. However, for longer sequences, turbo codes are more efficient in terms of SER.

IV. Concluding remarks

We have proposed in this paper a DSC scheme based on punctured quasi-arithmetic codes. These codes can be defined with finite state machines for both memoryless and first-order memory sources, hence allowing to decode them using an optimal BCJR algorithm. The compression capability of these codes together with their capability to integrate extra information on the source probabilities are exploited by the proposed DSC scheme. Simulations reveal that this scheme is able to provide efficient decoding performance for short sequences compared to well-known DSC schemes based on channel codes, particularly when the source probability is very asymmetric and when the memory correlation factor of the source is relatively high. This reveals that QA codes are better suited than channel codes to deal with asymmetric sources. Note that the proposed scheme is a preliminary exploration into the possibility of using punctured QA code for the Slepian-Wolf problem. Iterative structures involving punctured QA codes are being designed in order to reduce the gap with channel codes.

References

[1] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, and M. Ouaret, "The discover codec: Architecture, techniques and evaluation," in *Picture Coding Symposium*, Lisbonne, Portugal,, Nov. 2007.

[2] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inform. Theory*, vol. 19, pp. 471–480, 1973.

[3] A. Wyner and J. Ziv, "The rate distorsion function for source coding with side information at the decoder," *IEEE Trans. Inform. Theory*, vol. 22, pp. 1–10, Jan. 1976.

[4] J. Bajcsy and P. Mitran, "Coding for the slepian-wolf problem with turbo codes," in *Proc. IEEE Globecom*, 2001, pp. 1400–1404.

[5] A. Aaron and B. Girod, "Compression with side information using turbo codes," in *Proc. Data Compression Conf., DCC*, 2002, pp. 252–261.

[6] J. Garcia-Frias and Y. Zhao, "Compression of correlated binary sources using turbo codes," *IEEE Commun. Lett.*, vol. 5, pp. 417–419, Oct. 2001.

[7] A. Liveris, Z. Xiong, and C. Georghiades, "Compression of binary sources with side information at the decoder using ldpc codes," *IEEE Commun. Lett.*, vol. 6, pp. 440–442, Oct. 2002.

[8] D. V. Renterghem, X. Jaspar, B. Macq, and L. Vanderdorpe, "Distributed coding with optimized irregular turbo codes," in *Proc. Intl. Conf. Commun., ICC*, Glasgow, Scotland, June 2007, pp. 963–968.

[9] Q. Zhao and M. Effros, "Lossless and near lossless source coding for multile access network," *IEEE Trans. Inform. Theory*, vol. 49, no. 1, pp. 112–128, Jan. 2003.

[10] X. Artigas, S. Malinowski, C. Guillemot, and L. Torres, "Overlapped quasi-arithmetic codes for distributed video coding," in *Proc. Intl. Conf. Image Processing, ICIP*, San Antonio, USA, 2007.

[11] M. Grangetto, E. Magli, and G. Olmo, "Distributed arithmetic coding," *IEEE Commun. Lett.*, vol. 11, pp. 883–885, Nov. 2007.

[12] S. Ben-Jamaa, C. Weidmann, and M. Kieffer, "Asymptotic error-correcting performance of joint source-channel schemes based on arithmetic coding," in *Proc. MMSP*, Victoria, Canada, October 2006, pp. 262–266.

[13] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, pp. 284–287, Mar. 1974.

[14] T. Guionnet and C. Guillemot, "Soft and joint source-channel decoding of quasi-arithmetic codes," *EURASIP Journal on applied signal processing*, vol. 3, pp. 394–411, Mar. 2004.

[15] J. Hagenauer, J. Barros, and A. Schaefer, "Lossless turbo source coding with decremental redundancy," in *Proc. Intl. Conf. Source and Channel Coding, SCC*, Erlangen, Germany, January 2004.