

Rate-adaptive turbo-syndrome scheme for Slepian-Wolf Coding

Aline Roumy, Khaled Lajnef, Christine Guillemot¹
INRIA- Rennes, Campus de Beaulieu, 35042 Rennes Cedex, France
Email: aroumy@irisa.fr

Abstract— We consider asymmetric distributed source coding (DSC) and propose a new rate-adaptive DSC code that is needed when the coder does not know the correlation between the sources. The proposed scheme is based on the puncturing of the syndromes of a convolutional code. Optimal decoding is derived with complexity that grows only linearly with the number of punctured positions. Finally, the idea is generalized to the case of turbo-codes.

Keywords— Distributed source coding, distributed video coding, wireless sensor network, turbo codes.

I. INTRODUCTION

Distributed source coding (DSC) refers to the separate compression of (many) correlated sources (separate in the sense that no cooperation/communication between the sources is allowed but joint decoding is performed). The source coding theorem establishes that to reliably compress a source X , the compression rate R_x (in bits per source symbol) must be greater than the entropy of the source $H(X)$. Therefore to jointly compress the sources X and Y (at rate R_x and R_y), the sum rate $R_x + R_y$ must satisfy $R_x + R_y \geq H(X, Y)$. On the other hand, if no cooperation is allowed between the encoders, it is clear that a sum rate of $H(X) + H(Y)$ is enough. A very surprising result due to Slepian and Wolf (SW) [1] shows that the joint entropy $H(X, Y)$ is a sufficient rate for separate encoding (but joint decoding) of two correlated sources. In other words adding the constraint of encoding separation does not incur any loss in terms of compression rate. This promising theoretical result can be applied in sensor networks, where communication between the nodes is not possible or not desired due to its high energy cost (network video camera, sensor network...). The problem also occurs in video compression, where the correlation between successive images is exploited at the decoder only in order to reduce the complexity at the encoder side.

For this video application, the compression scheme operates at a corner point of the SW region. One source is compressed while the other one, called the side information (SI), is sent directly to the decoder. This problem is referred to as the *asymmetric* DSC problem or the lossless source coding problem with side information at the decoder. A way to achieve these compression rates is to use channel codes. One source, say X , is compressed by sending its syndrome S_x

with respect to a channel code, whereas the other source Y is sent at its entropy rate $H(Y)$. The ambiguity about the source X is alleviated through channel decoding, by retrieving the closest sequence to Y in the coset indexed by the syndrome S_x . This solution is shown to be optimal in [2] in the sense that a capacity achieving channel code can be turned into an optimal DSC code. This scheme is known as the syndrome approach or the binning approach or also the Wyner scheme. Practical solutions based on this idea first appeared in [3]. In this contribution, we also consider the syndrome approach since it is a way to design optimal DSC codes.

Note that the only information that needs to be known at the transmitter is the correlation between the sources. However, for many practical scenarios, this correlation may not be known and *rate adaptive* schemes have to be designed. The rate of the code is then controlled via a feedback channel. More precisely if the Bit error rate (BER) at the output of the decoder exceeds a given value, more syndromes are requested from the encoder. In this context, the code designed should be *incremental* s. t. the encoder does not need to re-encode the data. More precisely, the first bits are kept, and only additional bits are sent upon request.

Due to the optimality of the syndrome approach [2], natural design of a rate-adaptive scheme consists in puncturing the syndrome. Previous papers have investigated the influence of puncturing or corrupting the syndromes [5], [6], [7]. In [5], it is shown that the syndrome approach is very sensitive to errors or erasures, such that the puncturing of syndromes will lead to performance degradation. Therefore previous contributions consisted in avoiding or compensating for this performance degradation. For instance, the authors in [6] puncture the parity bits (rather than the syndrome) and then apply a syndrome former and inverse syndrome former for the new punctured code. This method can be applied to any code, in particular to convolutional and turbo codes. However, the resulting code is not incremental. The authors in [7] investigate the puncturing of LDPC (Low density parity check code) encoded syndromes and they propose to protect the punctured syndromes with an accumulator code. In [10], an incremental and rate adaptive code is designed with decoding complexity of $O(n \log n)$, where n is the blocklength of the code.

Main contribution. In this paper, we design a single code for the asymmetric DSC setup that is both rate-adaptive and incremental. We add the constraint of linear (in the code blocklength) encoding and decoding complexity. For

¹This research was partly funded by French National Research Agency (ANR) under the Essor project and by the European Commission IST FP6 program under the DISCOVER project (<http://www.discoverdvc.org>).

optimality, the code is based on the syndrome approach. We target distributed video application with small blocklength, and consider therefore turbo-code. We derive the optimal decoder for convolutional codes under syndrome puncturing and show that performance degradation due to syndrome puncturing can be avoided. Moreover the complexity of the proposed algorithm grows only linearly with the code blocklength and also linearly with the number of punctured positions. Finally, the idea is generalized to the case of turbo-codes.

... add all what is done and shown in this paper ...

II. SLEPIAN WOLF CODING

A. Theoretical results

Let X and Y be two binary correlated memoryless sources with uniform distribution. The statistical dependence between the two sources is modeled as a virtual channel analogous to a binary symmetric channel (BSC) defined by a cross-over probability $p = \mathbb{P}(X \neq Y)$ (BSC(p)). Consider the asymmetric lossless DSC setup for these sources, where X is to be compressed and Y is the side information (SI) available at the decoder. This setup corresponds to one corner point of the SW region [1] and the minimum achievable rate pair for lossless compression is $(R_x, R_y) = (H(X|Y), H(Y))$. When the correlation between X and Y decreases, the conditional entropy $H(X|Y)$ increases, achieving $H(X)$, when X and Y become independent.

B. Syndrome-based coding

The use of parity-check codes for approaching the corner points of the Slepian-Wolf rate region was first proposed in 1974 [2]. In order to present this idea, we first review some notations on binary linear codes. A binary (n, k) code \mathcal{C} is defined by a $(n - k) \times n$ parity-check matrix H , and contains all the n -length vector \mathbf{x} s. t. $H\mathbf{x} = \mathbf{0}$: $\mathcal{C} = \{\mathbf{x} : H\mathbf{x} = \mathbf{0}\}$. If H has full row rank, then the rate of the channel code is k/n . Moreover, for a good code, all the vectors of the code (called words or codewords) have maximum Hamming distance. The code partitions the space of 2^n sequences into 2^{n-k} cosets containing 2^k words with maximum Hamming distance. Each coset is indexed by an $(n-k)$ -length syndrome, defined as $\mathbf{s}_x = H\mathbf{x}$. In other words, all sequences in a coset share the same syndrome: $\mathcal{C}_s = \{\mathbf{x} : H\mathbf{x} = \mathbf{s}\}$. Moreover, as a consequence of the linearity of the code, a coset results from the translation of the code by any representative of the coset:

$$\forall \mathbf{v} \in \mathcal{C}_s, \mathcal{C}_s = \mathcal{C} + \mathbf{v}. \quad (1)$$

Depending on the variables handled, $+$ denotes either the addition over the finite field of order 2 or the addition over the reals (same definition for the multiplication). If a codeword \mathbf{x} is sent over a BSC(p) with error sequence \mathbf{n} , the received sequence is $\mathbf{y} = \mathbf{x} + \mathbf{n}$. Maximum likelihood (ML) decoding over the BSC, searches the coset codeword to \mathbf{y} with respect to the Hamming distance $d_H(\cdot, \cdot)$:

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x} \in \mathcal{C}} d_H(\mathbf{x}, \mathbf{y}).$$

This can be implemented with *syndrome decoding*. This decoding rely on a function $f : \{0, 1\}^{n-k} \rightarrow \{0, 1\}^n$, that computes for each syndrome, a representative called the *coset leader* that has minimal Hamming weight. Thus, a syndrome decoder first computes the syndrome of the received sequence: $\mathbf{s} = H\mathbf{y}$, then the coset leader $f(\mathbf{s})$. This coset leader is the ML estimate of the error pattern \mathbf{n} . Finally, the ML estimate of \mathbf{x} is given by $\hat{\mathbf{x}} = \mathbf{y} + f(H\mathbf{y})$.

In order to use such a code for the asymmetric SW problem, [2] suggests to construct bins as cosets of a capacity-achieving parity-check code. Let \mathbf{x} and \mathbf{y} be two correlated binary sequences of length n . These sequences are the realizations of the sources X and Y . The coder computes and transmits the syndrome of \mathbf{x} , $\mathbf{s}_x = H\mathbf{x}$. The sequence \mathbf{x} of n input bits is thus mapped into its corresponding $(n - k)$ syndrome bits, leading to a compression ratio of $n : (n - k)$. The decoder, knowing the correlation between the sources X and Y and the coset index \mathbf{s}_x , searches for the sequence in the coset that is closest to \mathbf{y} . In other words, maximum likelihood decoding is performed in order to retrieve the original \mathbf{x} sequence:

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x} \in \mathcal{C}_s} d_H(\mathbf{x}, \mathbf{y}). \quad (2)$$

Note that the maximization is performed in a set of vectors with syndrome that may not be $\mathbf{0}$. Therefore, the classical ML channel decoder has to be adapted in order to be able to enumerate all vectors in a given coset \mathcal{C}_s .

Syndrome decoding is rather used here as a mental representation than an efficient decoding algorithm, since it has complexity $O(n2^{n-k})$. There exists codes with linear ML complexity, such as the convolutional codes and their Viterbi decoder. In the following, we will consider convolutional codes (due to their linear ML decoding complexity) and also their extension to turbo codes, that are known to achieve among the best performance at small to moderate code blocklength.

Practical solutions based on the syndrome approach first appeared in [3], with trellis based codes. More precisely, the decoder is modified s.t. it can enumerate all the codes in a given coset. The method uses the linear property (1). For systematic convolutional codes, a representative of the coset is the concatenation of the k -length zero vector and the $n - k$ -length syndrome \mathbf{s} . Note that the resulting $[0\mathbf{s}]$, may not necessarily be the coset leader. This representative is then added to all the codewords labeling the edges of the trellis. A variation is proposed in [4], where this translation by a coset representative is performed outside the decoder. This allows to use off-the-shelf decoders, without a need to modify them.

C. Rate adaptation through syndrome puncturing

We now consider the problem of adapting the rate of a syndrome based DSC system to varying correlation. A natural design of a rate-adaptive scheme consists in puncturing the syndrome. However, using puncturing mechanisms lead performance degradation as reported in [5], [6], [7]. Therefore previous contributions for convolutional codes consisted in avoiding this step. For instance, the authors in [6] puncture the parity bits (rather than the syndrome) and then apply a

usual DSC codec for the new punctured code. However, the resulting code does not meet the incremental constraint, since the data have to be re-encoded.

In the following, we derive the optimal ML decoding algorithm for convolutional codes under syndrome puncturing.

III. PROPOSED DECODING ALGORITHM

A. Syndrome trellis

In order to present our ML decoder for syndrome punctured convolutional code, we will first present a ML decoder for convolutional code, with a search performed in any possible coset. This decoder implements the criterion (2) and holds for any convolutional code and not only systematic codes as in [3] and [4].

The classical trellis is a compact representation of all possible codewords in a convolutional code and is based on the generator polynomials of the code. In order to enumerate all possible sequences in a given coset, one can build a *syndrome trellis*, based on the parity check matrix (or polynomials). This trellis has first been proposed for binary linear block codes [8], and was then generalized to convolutional codes [9]. We present now a novel way to construct the syndrome trellis based on the parity check polynomials. This construction is efficient in the sense that there is no need to expand the parity check polynomial matrix into a matrix of an equivalent block code of large dimension.

For clarity, we present the syndrome trellis for a rate 1/2 convolutional code with generator matrix [5, 7] (in octal form). One possible parity check matrix is $H = [7, 5]$. This can be rewritten as a polynomial parity check matrix $H(D) = [1 + D + D^2, 1 + D^2]$, where D is a dummy variable representing a time offset. Let $x_1(D)$ and $x_2(D)$ be the power series (in the indeterminate D) of the sequence \mathbf{x} at even and odd time instant respectively:

$$x_1(D) = \sum_{l \in \mathbb{N}} x_{2l} D^l, \quad x_2(D) = \sum_{l \in \mathbb{N}} x_{2l+1} D^l$$

where x_l is the l -th component of the sequence \mathbf{x} . By definition, the syndrome (in this rate 1/2 example, there is one single syndrome polynomial) is:

$$s(D) = (1 + D + D^2)x_1(D) + (1 + D^2)x_2(D).$$

The trick to efficiently construct the syndrome trellis is to factorize this polynomial into:

$$s(D) = x_1(D) + x_2(D) + D \underbrace{\left(x_1 + D(x_1 + x_2) \right)}_{\triangleq \sigma_2(D)} \quad (3)$$

which defines the two states of the syndrome trellis as: $\sigma_2(D) = x_1(D) + x_2(D)$ and $\sigma_1(D) = x_1(D) + D\sigma_2(D)$. The computation of the syndrome is shown in Fig. 1. Finally, for each state and each sequence at a given time, the syndrome can be computed. If we collect all the transitions s.t. the syndrome is 0, we get the trellis in the left of Fig. 2. The right side of Fig. 2 shows the trellis section when the syndrome is 1.

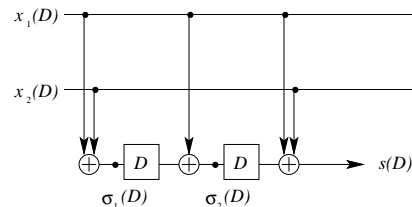


Fig. 1. Block diagram for the computation of the syndrome. The rate 1/2 convolutional code is determined by its polynomial parity check matrix $H(D) = [1 + D + D^2, 1 + D^2]$. The diagram shows the states $\sigma_1(D)$ $\sigma_2(D)$ of the syndrome trellis.

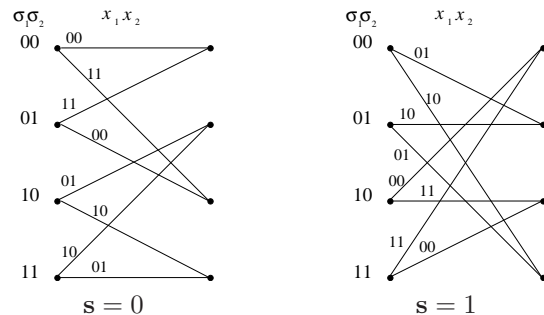


Fig. 2. Syndrome trellis for the rate 1/2 convolutional code defined by its polynomial parity check matrix $H(D) = [1 + D + D^2, 1 + D^2]$. On the left (right), the trellis section corresponds to the syndrome value $\mathbf{s} = 0$ ($\mathbf{s} = 1$, respectively).

Remark 1: The syndrome trellis with syndrome 0 enumerates all the codewords (in \mathcal{C}). However, the syndrome trellis is not the same as the classical trellis since the states of the trellis have different definitions. In the classical trellis, the states are functions of the input bits (of the channel encoder), whereas in the syndrome trellis the states depend on the output bits (of the channel encoder).

Remark 2: The syndrome trellis can be built for any convolutional code, not only systematic codes as in [3] and [4].

Following the same line, the method can be generalized to any convolutional code with rate k/n . More precisely, n power series are defined for the input sequence:

$$\forall j \in \{1, \dots, n\}, x_j(D) = \sum_{l \in \mathbb{N}} x_{nl+j-1} D^l$$

and $n - k$ for the syndrome:

$$\forall i \in \{1, \dots, n - k\}, s_i(D) = \sum_{l \in \mathbb{N}} s_{(n-k)l+i-1} D^l$$

The $(n - k) \times n$ parity-check matrix H has (i, j) -th entry denoted $h_{ij}(D)$. Thus, we get the syndrome as

$$\forall i, s_i(D) = \sum_{j=1}^n h_{ij}(D)x_j(D).$$

Factorizing the syndrome as in (3), we get $n - k$ diagrams similar to Fig. 1.

B. Decoding with the syndrome trellis

The decoder, knowing the syndrome \mathbf{s} , searches for the sequence with that particular syndrome that is closest to \mathbf{y} . First it builds the trellis. Each section of the trellis is determined by the syndrome. More precisely, a section corresponds to the reception of n bits of \mathbf{x} and $n-k$ syndrome bits. The decoder chooses the trellis section defined by the received syndrome value. Once the whole trellis is built, the Viterbi algorithm is performed and chooses the closest sequence to the received \mathbf{y} . From the construction of the trellis, it is clear that the search is performed in the set of sequences with syndrome \mathbf{s} .

C. Rate adaptation through syndrome puncturing: the super trellis

When syndrome bits are punctured, the optimal decoder should in principle search for the closest sequence in a union of cosets. This union corresponds to all possible syndromes that equal the punctured syndrome in the unpunctured positions. Therefore, the number of cosets grows exponentially with the number of punctured bits. A brute force decoder consists in performing a ML algorithm in each coset, but then the decoding complexity of such a method would grow exponentially with the number of punctured bits. On the other hand, if the decoder is not modified, it leads to systematic errors, since the search may be performed in the wrong coset.

However, the syndrome trellis presented in sections III-A and III-B can be modified in order to perform a search in a union of cosets. Take our example of the rate 1/2 convolutional code determined by its polynomial parity check matrix $H(D) = [1+D+D^2, 1+D^2]$. Whenever the syndrome bit is punctured, we build a trellis section that is the union of the two trellis sections in Fig. 2. The resulting trellis is called the *super trellis* and the Viterbi algorithm is performed on this new trellis.

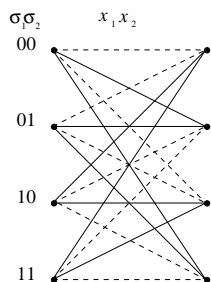


Fig. 3. Super trellis for the rate 1/2 convolutional code defined by its polynomial parity check matrix $H(D) = [1+D+D^2, 1+D^2]$. This trellis is the union of the two trellis sections of Fig. 2.

More generally, assume that a subset of the $n-k$ syndrome bits have been punctured. Then the super trellis section is the union of all the trellises corresponding to syndrome values that are equal to the received syndrome (in the unpunctured positions).

Property 1: Let \mathcal{C} be a convolutional (channel) code of rate k/n . Let nN be the code blocklength. The code partitions the space of the 2^{nN} sequences into $2^{(n-k)N}$ cosets. Each coset

is indexed by an $((n-k)N)$ -length *syndrome*. Let us consider the union of cosets obtained by puncturing the syndrome. The complexity of ML decoding in this coset union grows linearly with the number of punctured position.

Proof: The cosets of the code are indexed by a syndrome. The syndrome trellis (see section III-A) enumerates all the sequences for a given syndrome. Then, the trellises corresponding to different syndromes are superposed. This builds the super trellis (see section III-C). The complexity increases in the trellis sections, where some bits have been punctured. Consider one trellis section and assume that l out of the $n-k$ syndromes bits have been erased. Then, the complexity grows with 2^l , since 2^l trellises have to be superposed. Let us denote L the number of trellis sections containing at least one punctured syndrome bit. Then, at each section the number of path is increased. Therefore, the complexity grows linearly with L . Since, in convolutional code $n \ll N$ and $l \ll L$, the increased complexity is dominated by the number of punctured trellis sections. ■

Remark 3: The above result state a surprising result. Let us assume that at most one bit per trellis section is punctured. Then, property 1 states that the complexity grows logarithmically with the number of cosets.

D. Comparison with the parity approach

methode equivalente SSI on a $G=[I \ H]$

mais le problme important pour l'approche parit EST: bien construire les cosets. Clair pour syndrome cf Wyner mais pas vident pour approche parit.

autre avantage : ne pas avoir de codes systmatiques, et donc pouvoir poinconner tous les syndromes

E. Wyner Ziv

F. Noise resilient

When the syndrome is perfectly known at the receiver, a state transition defines a unique possible sequence \mathbf{x} and the metric associated with this transition is $p(\mathbf{y}|\mathbf{x})$, where \mathbf{x} depends on the syndrome \mathbf{s} . Therefore the transition metric can be rewritten as $p(\mathbf{y}|\mathbf{x}(\mathbf{s}))$. We now consider the case of imperfect syndrome knowledge at the receiver. Let \mathbf{u} denote the noisy version of the syndrome \mathbf{x} . Now the ML decoder seeks for the sequence that maximizes the criteria $p(\mathbf{x}|\mathbf{y}, \mathbf{u}) \propto p(\mathbf{y}, \mathbf{u}|\mathbf{x}) = p(\mathbf{u}|\mathbf{x})p(\mathbf{y}|\mathbf{u}, \mathbf{x})$. For a given transition of the trellis, the syndrome value is fixed and the transition metric reads $p(\mathbf{u}|\mathbf{x}, \mathbf{s})p(\mathbf{y}|\mathbf{u}, \mathbf{x}, \mathbf{s}) = p(\mathbf{u}|\mathbf{s})p(\mathbf{y}|\mathbf{x}, \mathbf{s}) = p(\mathbf{u}|\mathbf{s})p(\mathbf{y}|\mathbf{x}(\mathbf{s}))$. Therefore, the ML decoder under imperfect syndrome knowledge can be performed on the super-trellis, where the transition metrics scale now with factor $p(\mathbf{u}|\mathbf{s})$ that takes into account the noisy syndrome \mathbf{u} .

IV. SIMULATION RESULTS

We first consider a convolutional code. The mother code is a rate-1/3 non-recursive code defined by its parity check matrix

$$H = \begin{pmatrix} 11 & 15 & 6 \\ 15 & 12 & 17 \end{pmatrix}$$

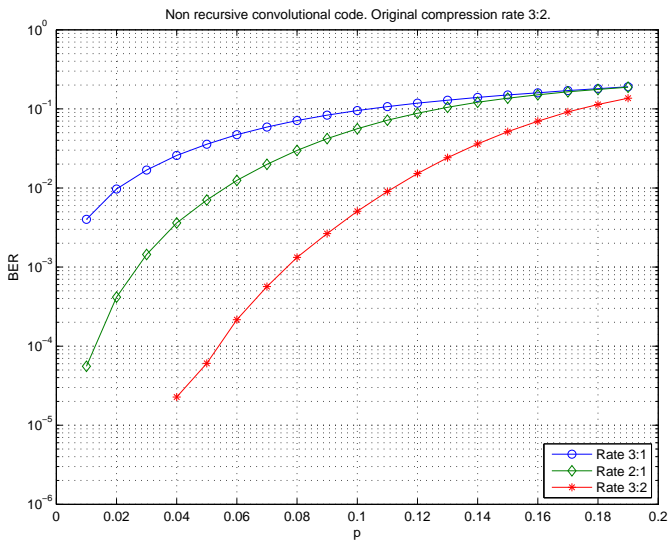


Fig. 4. Syndrome puncturing of a convolutional code: a rate adaptive scheme

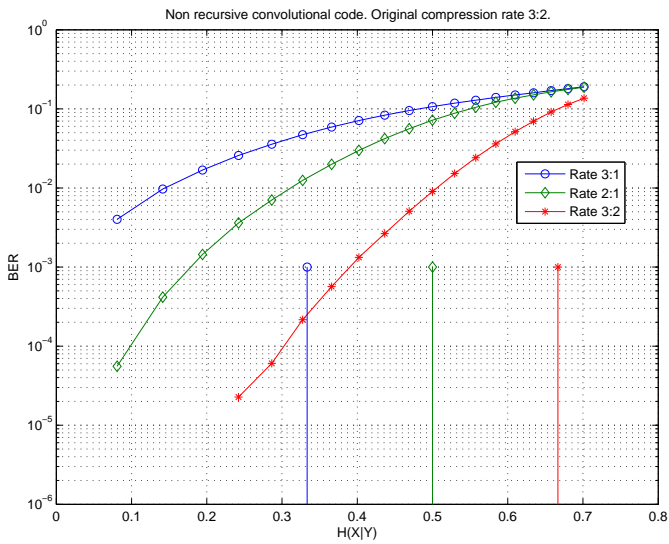


Fig. 5. Syndrome puncturing of a convolutional code: a rate adaptive scheme

The syndrome trellis has 2^6 states. This code leads to a compression rate of 3 : 2. Puncturing the syndrome bits yields compression rates 2 : 1 and 3 : 1. Performance are shown on Fig. 4. Note that we get the same performance as in [6] with the same complexity (2^6 states in both cases), but our scheme can be applied to any convolutional code (not only recursive) and is incremental. Fig. 5 shows the gap between the proposed scheme and the optimal SW bound.

The decoder is then applied to a turbo code. The $1/2$ -rate constituent code of the turbo code is defined by its parity check matrix $H = (23/3335/33)$. The overall compression rate is 1 : 1, and puncturing leads to various compression rates. Performance is shown on Fig. 6 and 8. Simulations have been carried for an interleaver of size 10^5 .

Finally, the syndrome is sent over a BSC channel. The optimal decoder is used in order to perform ML decoding

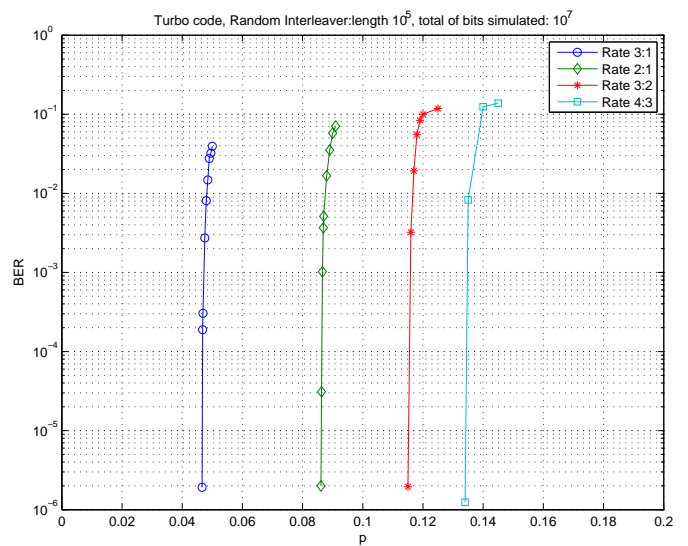


Fig. 6. Syndrome puncturing of a turbo code: a rate adaptive scheme

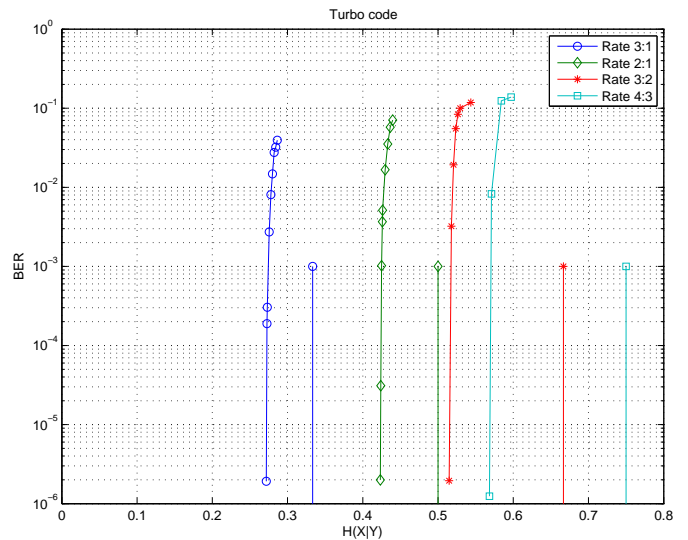


Fig. 7. Syndrome puncturing of a turbo code: a rate adaptive scheme. Performance versus the entropy rate $H(X|Y)$ and comparison with the SW bounds.

under imperfect syndrome knowledge (see section III-F).

V. CONCLUSION

We claim that performance degradation due to syndrome puncturing can be avoided if a well suited decoder is used. When the syndrome is punctured, the optimal decoder consists in searching for a closest sequence in a union of cosets. A brute force optimal decoding algorithm has then complexity that grows exponentially with the number of punctured positions. However, for convolutional codes, we derive an optimal algorithm with linear complexity. The decoder is based on a super trellis that stems from the syndrome trellis proposed in [8]. In the full paper, we will show that, in the case of systematic convolutional codes, puncturing the syndrome or the parity bits

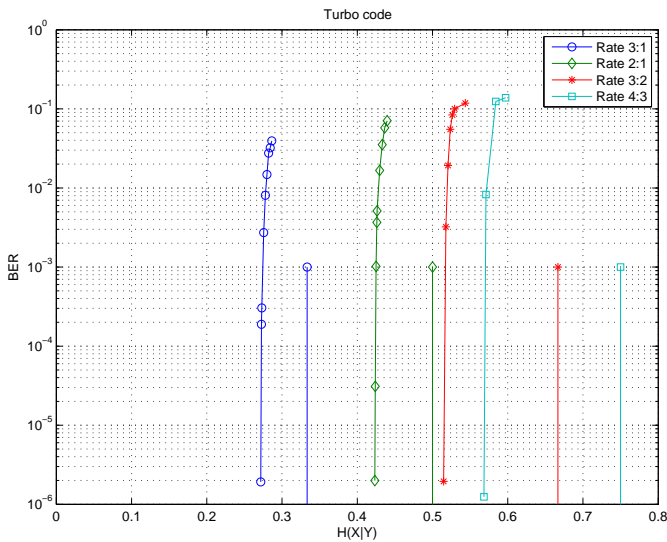


Fig. 8. Syndrome puncturing of a convolutional code: a rate adaptive scheme

is equivalent. Therefore we get exactly the same results as in [6] as illustrated in figure 4. This also shows that there is no performance degradation due to syndrome puncturing when the optimal decoder is used. Moreover our scheme is more general than the one proposed in [6] since it can be applied to any convolutional code (not only systematic). Finally, our scheme is generalized to the turbo codes as shown in figure 6.

REFERENCES

- [1] D. Slepian and J.K. Wolf, "Noiseless Coding of Correlated Information Sources," *IEEE Transactions on Information Theory*, vol. IT-19, pp. 471-480, July 1973.
- [2] A. Wyner, "Recent results in the Shannon theory," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 2-10, Jan. 1974.
- [3] S. S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (DISCUS): Design and construction," in *Proceedings of the 1999 IEEE Data Compression Conference, (Snowbird, Utah)*, March 1999.
- [4] P. Tan and J. Li, "A practical and optimal symmetric slepian-wolf compression strategy using syndrome formers and inverse syndrome formers," in *Proceedings of 43rd Annual Allerton Conference on Communication, Control and Computing*, September 2005.
- [5] P. Tan, and J. Li, "Enhancing the Robustness of Distributed Compression Using Ideas from Channel Coding," *Proceeding of IEEE Global Communications Conference (GLOBECOM)*, St. Louis, MO, Nov. 2005.
- [6] J. Li and H. Alqamzi, "An Optimal Distributed and Adaptive Source Coding Strategy Using Rate-Compatible Punctured Convolutional Codes," *Proceeding of IEEE International Conference on Accoustic, Speech, and Signal Processing (ICASSP)*, Philadelphia, PA, March 2005.
- [7] D. Varodayan, A. Aaron and B. Girod, "Rate-adaptive distributed source coding using low-density parity-check codes," *Proc. Asilomar Conference on Signals, Systems, and Computers, 2005*, Pacific Grove, California, November 2005.
- [8] J. K. Wolf, "Efficient maximum likelihood decoding of linear block codes using a trellis," *IEEE Transaction on Information Theory*, vol. IT-24, pp. 76-80, Jan. 1978.
- [9] V. Sidorenko, and V. Zyablov, "Decoding of convolutional codes using a syndrome trellis," *IEEE Transaction on Information Theory*, vol. IT-40, pp. 1663-1666, Sept. 1994.
- [10] J. Chen, A. Khisti, D. M. Malioutov, and J. S. Yedidia, "Distributed source coding using serially-concatenated-accumulate codes," *IEEE Information Theory Workshop*, San Antonio, TX, 2004.